THE UNIVERSITY OF NEW SOUTH WALES
Thesis/Dissertation Sheet

Surname or Family name: Sarlej

First name: Margaret Other name/s: Krystyna

Abbreviation for degree as given in the University calendar: PhD

School: School of Computer Science and Engineering Faculty: Faculty of Engineering

Title: A lesson learned: using emotions to generate stories with morals

Abstract 350 words maximum:

Storytelling has been central to human communication since before written history, serving as an indispensable mechanism for conveying important information. Consequently, most traditional stories and fables were imbued with a message or lesson: the story's moral. This aspect of storytelling can be leveraged as a framework for story structure, enriching the every-growing toolset employed by computational storytelling systems.

In this thesis, we explore the automatic generation of stories with morals. We focus on six common morals identified in Aesop's fables: retribution, greed, pride, realistic expectations, recklessness, and reward. We posit a relationship between the moral of a story and the patterns of character emotions arising in that story, and therefore adopt character emotions as our foundation for representing morals. To this end, we implement a model of an existing theory of emotion (the OCC theory) using answer set programming, and develop rules associating morals with temporal sequences of emotion. The stories generated using these models are evaluated by human readers to demonstrate that morals are effectively conveyed.

The rules linking morals to emotions are derived using inductive logic programming, based on a corpus of stories from Aesop's fables which have been encoded in terms of our emotion model. Due to the sparsity of training data, many of the resulting rules have low coverage and accuracy. We further develop them by hand to produce a more general set of moral definitions, which we incorporate into our Moral Storytelling System (MOSS).

Our evaluation is modelled on the philosophy underpinning a simple Turing test. Participants are presented with a random selection of stories and asked to identify the moral of each, without being told which stories are MOSS-generated, which are human-authored (within the constraints of MOSS's domains and mode of expression), and which are deliberately moral-free sequences of events. As predicted, the results show that MOSS-generated stories convey morals significantly better than moral-free event sequences, and for certain morals even rival the performance of human-authored stories. This supports our premise that morals and emotions are closely related, and establishes emotions as a useful technique for representing story morals.

A Lesson Learned: Using Emotions to Generate Stories with Morals

Margaret Sarlej

A thesis submitted as a requirement for the degree of

Doctor of Philosphy (PhD)

School of Computer Science and Engineering

Faculty of Engineering

March 2014

Originality Statement

'I hereby declare that this submission is my own work and to the best of my knowledge it contains no materials previously published or written by another person, or substantial proportions of material which have been accepted for the award of any other degree or diploma at UNSW or any other educational institution, except where due acknowledgement is made in the thesis. Any contribution made to the research by others, with whom I have worked at UNSW or elsewhere, is explicitly acknowledged in the thesis. I also declare that the intellectual content of this thesis is the product of my own work, except to the extent that assistance from others in the project's design and conception or in style, presentation and linguistic expression is acknowledged.'

Signed .....

Date .....

Abstract

Storytelling has been central to human communication since before written history, serving as an indispensable mechanism for conveying important information. Consequently, most traditional stories and fables were imbued with a message or lesson: the story's moral. This aspect of storytelling can be leveraged as a framework for story structure, enriching the ever-growing toolset employed by computational storytelling systems.

In this thesis, we explore the automatic generation of stories with morals. We focus on six common morals identified in Aesop's fables: retribution, greed, pride, realistic expectations, recklessness, and reward. We posit a relationship between the moral of a story and the patterns of character emotions arising in that story, and therefore adopt character emotions as our foundation for representing morals. To this end, we implement a model of an existing theory of emotion (the OCC theory) using answer set programming, and develop rules associating morals with temporal sequences of emotion. The stories generated using these models are evaluated by human readers to demonstrate that morals are effectively conveyed.

The rules linking morals to emotions are derived using inductive logic programming, based on a corpus of stories from Aesop's fables which have been encoded in terms of our emotion model. Due to the sparsity of training data, many of the resulting rules have low coverage and accuracy. We further develop them by hand to produce a more general set of moral definitions, which we incorporate into our Moral Storytelling System (MOSS).

Our evaluation is modelled on the philosophy underpinning a simple Turing test. Participants are presented with a random selection of stories and asked to identify the moral of each, without being told which stories are MOSS-generated, which are human-authored (within the constraints of MOSS's domains and mode of expression), and which are deliberately moral-free sequences of events. As predicted, the results show that MOSS-generated stories convey morals significantly better than moral-free event sequences, and for certain morals even rival the performance of human-authored stories. This supports our premise that morals and emotions are closely related, and establishes emotions as a useful technique for representing story morals.

Contents

List of Figures

List of Tables

Listings

Acknowledgements

Although a PhD thesis is, to a large extent, a solitary pursuit, it's too long a road to walk alone. I would certainly not have reached the end without the help, both direct and indirect, of many people.

First and foremost, thank you to my supervisor Malcolm, for taking me on as his first PhD student and providing guidance, feedback, and advice over the last four years. Thank you also to Morri, for although his co-supervisor role was passive much of the time, he provided valuable advice when it was needed.

I am also grateful for the willingness of academics outside of UNSW to provide assistance. Thank you to Ashwin Srinivasan for looking into a problem I encountered with Aleph's classification tree learning, and sending me a development version of Aleph which corrected this. Without that, a good portion of Chapter 5 would be missing. I would also like to thank Martin Gebser, for taking the time to clarify my understanding of ASP during his visit to UNSW, and sending me useful resources.

I want to thank my parents for allowing me to delay my entry into the real world by another half-decade. You always encouraged my love of learning and academic pursuits. Without your support, I would not have embarked on a PhD in the first place. Yes, I can go get a real job now.

Thank you to Marie, for being brave enough to house a PhD student for the final few months leading up to the deadline, and to Francis and the cats for going along with the idea without complaining (at least to me). I'm grateful not only for the time saved in commuting, but also for your patience in putting up with my erratic behaviour, listening to my dilemmas, and offering advice and solutions more often than I had any right to expect. Without that, I probably still wouldn't have a title.

Thank you to everyone who helped proof-read this thesis at the eleventh hour: Marie, Ann, Sunny, and Ee-lin. A PhD thesis is not exactly light reading, so I appreciate the time you all spent to help make mine better.

At the tail end of a PhD, there isn't really time (or cognitive capacity) for anything else. Thank you to Sunny for stepping in and covering my share of the volleyball admin, especially during the final month. Sheltering me from that stress allowed me to find the focus to finish.

Finally, thank you to everyone who put up with me over the last four years, especially as I got closer to the end and my sanity started to deteriorate. After being chained to this roller-coaster for the last four years, the ride is finally over.

Publications

The research presented in this thesis is supported by the following publications:

- Margaret Sarlej and Malcolm Ryan. A Discrete Event Calculus Implementation of the OCC Theory of Emotion. In *The 4th Workshop on Intelligent Narrative Technologies*, pages 57–64, Stanford University, Palo Alto, California, 2011. AAAI Press.

- Margaret Sarlej and Malcolm Ryan. Representing Morals in Terms of Emotion. In *The 8th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE2012)*, pages 69–74, Stanford University, Palo Alto, California, 2012. AAAI Press.

- Margaret Sarlej and Malcolm Ryan. Generating Stories with Morals. In Koenitz, H.; Sezen, T.I.; Ferri, G.; Haahr, M.; Sezen, D.; Çatak, G. (Eds.), *The 6th International Conference on Interactive Digital Storytelling (ICIDS2013)*, pages 217–222, Istanbul, Turkey, 2013. Springer.

Chapter 1

Introduction

If history were taught in the form of stories,
it would never be forgotten.

Rudyard Kipling

Narrative ability, or more simply the ability to tell and understand stories, is central to being human [65]. In most people, storytelling ability is manifested from a very early age, and our society, culture, and communication are founded on this seemingly primitive skill. Researchers in cognitive science [150, 166] have also recognised that stories play an important role in intelligence and the structure of memory. Information presented in the form of a story is easier to remember, and thus, particularly prior to the advent of written communication, stories served as a very effective mechanism for passing on information, even across several generations. This is one of the key reasons storytelling evolved [141, 158], and consequently stories were often constructed specifically to convey a particular message. Although we now experience stories through a range of different media, made possible by rapid advances in technology, they are still a ubiquitous part of our lives, whether it be for entertainment, education, or communication [65].

Given the importance of storytelling in human society, it is little wonder that researchers in artificial intelligence have been striving to build systems capable of replicating this ability. The most obvious applications for storytelling systems are in the realm of entertainment: for example, computer games and interactive fiction. Generating engaging stories on the fly yields a more flexible and more satisfying user experience in these media, which rely heavily on interactivity. Just as significant are the potential

benefits for education. Traditionally, this was the most important function of story. Automating the process paves the way for structuring content in the form of narratives that are not only personalised, but also interactive. The fact that storytelling is the basis of human communication [150] also means that imparting this ability to computers could make interactions with them more natural, a goal that becomes more important as technology invades every aspect of modern life.

Most existing storytelling systems can be divided into two main categories: character-centric systems, which focus on character agents and rely on emergent narrative, and author-centric systems, which control story generation at the plot level, based on authorial goals. Authors can have many different goals in crafting a story: for example, interest, suspense, drama, or humour. Some stories are intended to make the reader think, or question their values. Others aim to teach a lesson, by conveying a moral. The traditional role of stories in communication and education makes this a particularly important author-level goal, which we aim to address with our research. The focus of this thesis is moral-based story generation. In this chapter, we introduce and explain the motivation for our work. We present our research objectives in Section 1.1, and explain their value in Section 1.2. We outline our approach to achieving these objectives in Section 1.3. Section 1.4 summarises our main contributions, and Section 1.5 provides an overview of the structure of the remainder of the thesis.

1.1 Objective

The moral of a story is a valuable author-centric device for guiding plot structure in storytelling systems. It holds tremendous potential for applications in both education and entertainment. However, this area has received only limited attention from researchers in computational storytelling [48, 170]. Existing representations are complex, highly structured, and specific to narrow classes of morals. There is room for the development of more general representational techniques, which are more broadly applicable.

One of the critical factors in understanding stories is emotion. Although we are far from a complete understanding of this complex phenomenon, there has been considerable work on modelling and representing emotions in the field of psychology [14, 50, 58, 76, 83, 84, 118, 123, 137, 140, 153, 155, 174]. Some of this has, in turn, moved into the sphere of computing, in particular for modelling characters [20, 43, 44, 52, 129, 133, 154]. We propose character emotions can be leveraged as a framework for representing story

morals. To investigate the veracity of this hypothesis, we aim to build a storytelling system which represents morals in terms of emotions, and thereby generates stories which convey morals. In pursuit of this goal, we address two core research questions:

1. Is there a relationship between character emotions and the moral of a story?
2. Can sequences of emotion be used to represent story morals, in order to automatically generate stories with morals?

1.2 Rationale

Imagine a classroom: thirty children hunched over copies of Cinderella. Katie turns the page, eager to learn what will happen when the coach turns back into a pumpkin. On another table, Mike has the book open somewhere near the middle, while he daydreams about brave knights and fierce dragons; fairy godmothers and handsome princes are for girls. Level of engagement is a significant factor in how much anyone, but children in particular, benefit from an educational activity [75]. Now, imagine the same classroom, only this time the text in front of each child is different. Each story still carries the same lesson, but while Katie reads about princesses and fairies, Mike is immersed in a world of knights and dragons. Every child is interested and engaged; every child is learning. We all know it is not feasible for a teacher to individually tailor a story to each of their students. But what if it could be done automatically?

This is the extraordinary opportunity technology provides. Stories are not just a form of entertainment. They are, and have always been, an important mechanism for learning [141]. However, to hold educational value, stories must not just be interesting, but also teach us something; they must have a moral. A system capable of generating stories with selected morals across multiple domains has the potential to reinvent the way classroom content is produced. And why stop there? The same stories could be made interactive, with the outcomes experienced by the protagonist dependent on whether the decisions made by the child were right or wrong, in effect allowing them to learn by experience.

However, education is not the only area where a moral-based storytelling system could have an impact. In the entertainment sector, both computer games and interactive fiction could benefit from this approach. Many of the most popular computer games have an intricately crafted story behind them, which forms a vital part of the

gaming experience. A work of interactive fiction is a story, albeit presented in a game-like way. Both media are limited in the degree of control they can grant to users by the underlying authorial burden of constructing a large number of plot lines and endings—a burden which would be alleviated if these storylines could be generated automatically based on the choices users make. Stories with a point are far more interesting and memorable than those without [105], and thus incorporating morals into these media could yield a more engaging and enjoyable user experience.

The less obvious application of a moral-based storytelling system lies in furthering our knowledge of the human cognitive processes associated with storytelling. The ability to automatically generate stories based on particular emotion patterns, and to tweak these patterns at will, gives us the opportunity to empirically investigate the nature of morals and their relationship with emotions. Given the way storytelling pervades all aspects of human culture, this in itself is a worthwhile goal.

1.3 Approach

Our approach is to leverage the relationship between two central aspects of story: morals and character emotions. Emotions have been widely studied, in the context of both psychology and artificial intelligence. Logical formalisms have been devised for deriving emotions from characters' beliefs, desires, and ideals, with respect to events that occur [8]. Using emotions to represent morals provides a good degree of abstraction from events; many different sequences of events can cause the same emotions, just like many different sequences of events can convey the same moral. This promises a simple yet powerful representation.

We begin by examining morals at their source: in stories written by people. Fables provide an ideal corpus, due to their simplicity; there are no superficial events to distract from those which are relevant to the fable's moral. We use Aesop's fables [9] as the foundation of our work, grouping them by moral and selecting six of the most common morals as our focus: retribution, greed, realistic expectations, recklessness, pride, and reward. In order to isolate patterns of emotions which characterise our chosen morals, we produce emotion data from Aesop's fables. To achieve this, we implement a model of an existing theory of emotion, the OCC theory [118], using answer set programming (ASP), and encode the relevant fables in terms of this model. We use inductive logic programming (ILP) to derive temporal sequences of emotion corresponding to each of

the chosen morals, and refine these by hand to produce a final set of emotion-based moral rules.

To determine whether our emotion-based rules successfully capture the essence of the relevant morals, we build a moral-based storytelling system, MOSS (Moral Storytelling System), which uses these rules to control plot trajectory. To gauge the effectiveness of MOSS-generated stories, we conduct a survey-based evaluation, modelled on a simple Turing test. For this, we produce three distinct sets of stories:

1. MOSS-generated stories, produced using our emotion-based moral rules.
2. Human-authored stories, written by people within the confines of the MOSS story worlds, but with the text automatically generated.
3. Automatically generated event sequences which are deliberately moral-free.

Survey participants are asked to identify the moral of a random selection of these stories, without being told which are MOSS-generated, which are human-authored, and which are moral-free. The results show that MOSS-generated stories convey morals significantly better than moral-free event sequences, which validates the use of emotions for representing morals.

1.4 Contributions

The focus of this thesis is the use of character emotions for representing story morals, for the purpose of story generation. The key contributions we make with this work are summarised as follows:

- A moral-based categorisation of Aesop's fables, identifying the most common morals arising within the collection.
- An ASP implementation of the OCC theory of emotion [118], based on Adam, Herzig, and Longin's logical formalism [8].
- Recognition of a relationship between story morals and character emotions, supported by an ILP analysis of emotion data produced from a corpus of Aesop's fables.

- The development of emotion-based ASP rules corresponding to six common morals arising in Aesop's fable collection: retribution, greed, realistic expectations, recklessness, pride, and reward.
- A storytelling system, MOSS, capable of generating stories with morals in multiple story worlds, using the emotion-based rules described above.
- An empirical evaluation of MOSS-generated stories, to demonstrate the effectiveness of emotions as a representation for morals.

1.5 Thesis Structure

The remainder of this thesis is structured as follows:

Chapter 2: We begin with a holistic background about storytelling. We introduce storytelling as a critical element of human culture, and review its many functions. After defining story, we break it down from both a narratological and computing perspective. Finally, we place our work in context through a discussion of existing storytelling systems.

Chapter 3: We introduce and define story morals, explaining their role and significance. We position morals in a computing context by reviewing moral-focussed work in computational storytelling and agent design. We present our moral-based categorisation of Aesop's fables, and select six common morals as the focus of our work: retribution, greed, realistic expectations, recklessness, pride, and reward. We discuss possible techniques for representing morals, and adopt emotions as our approach.

Chapter 4: We shift our focus to emotions, reviewing the dominant psychological theories of emotion and their computational applications. We adopt the OCC theory of emotion for our work, and present its logical formalisation. We describe our implementation of the OCC theory using ASP, and evaluate the performance of our model.

Chapter 5: We tie morals and emotions together. We use ILP to expose relationships between these concepts, and proceed to develop emotion-based ASP rules for the six morals selected in Chapter 3.

Chapter 6: We describe MOSS, which combines an ASP Story Planner with a Text Generator scripted in Perl. We describe the three story worlds we implement, and explain how the resulting stories match the moral rules defined in Chapter 5, with reference to specific examples.

Chapter 7: We explain our evaluation process: using an online survey to compare how effectively MOSS-generated stories convey morals relative to human-authored stories and moral-free event sequences. We present the results, which show that MOSS-generated stories perform significantly better than moral-free stories, and discuss qualitative feedback from survey participants.

Chapter 8: We present our conclusions, asserting a correlation between morals and emotions in stories, and reflecting on the effectiveness of emotions as a representation for morals with a view towards story generation.

Chapter 9: We suggest areas for future work in emotion-driven, moral-based story generation, including extensions to MOSS specifically, as well as broader research directions.

Following the main content of the thesis, we provide six appendices with supporting materials:

Appendix A: A sample excerpt from the OCC emotion survey used to evaluate our ASP emotion model, as described in Chapter 4.

Appendix B: Mode declarations for the emotion predicates provided to Aleph, the ILP system we use for learning moral rules.

Appendix C: The rules produced by Aleph during tree learning, corresponding to the tree diagrams provided in Chapter 5.

Appendix D: The domain descriptions provided to participants for producing human-authored stories.

Appendix E: A comprehensive set of sample stories, including MOSS-generated stories, human-authored stories, and moral-free stories. Three examples of each type are provided per moral, one corresponding to each story world.

Chapter 2

Background

> *Humans are creatures of story, so story touches nearly every aspect of our lives.*
>
> Jonathan Gottschall
> *The Storytelling Animal*

It was a cold winter's evening. The group huddled tightly around a crackling fire, their only defence against the biting chill. Their attention was focussed on an older man, whose gestures cast dancing shadows on the cave walls. The rhythm of his tale held their rapt attention; they laughed when the boy slipped in the mud, gasped in unison at the boar giving chase, and sighed with relief when the hero escaped. When the telling concluded they remained, murmuring quietly about the tale they had just heard, not only what was, but what could have been. It brought them together, a common thread linking the men and the women, the young and the old, mismatched individuals united through the power of story.

Storytelling has been a vital part of human life since communication, in any form, first developed. Remarkably, stories arising from cultures separated by both time and geography are strikingly similar, indicating that storytelling was an important part of our early evolution, and one of the key features setting us apart from other animals [65]. In light of the central role stories play in our lives, it is little wonder that scholars have been trying to analyse and understand them for centuries. We have evidence of this as far back as Aristotle's *Poetics* [12], the earliest surviving work of literary analysis, dating back to the 4th century BC. However, as with most phenomena we as humans strive to understand, we rarely stop at analysis. Next we try to model, and finally

replicate, the phenomena we study. It is thus that storytelling has moved into the realm of computing and artificial intelligence.

This chapter provides an overview of the field of storytelling, both human and computer-generated. Section 2.1 sets the scene by outlining the importance of storytelling in human culture throughout history. We take a step back in Section 2.2 to define story. As befitting an area that sits on the boundary of art and science, we consider story from two perspectives, beginning with a narratological view, then presenting a breakdown more suited to computational modelling. In Section 2.3, we move on to storytelling systems, distinguishing two main approaches and reviewing key examples of each, as well as positioning our system in this context. Section 2.4 summarises the material presented in this chapter.

2.1 Storytelling in Human Culture

As far back as communication and language, we have story: "Storytelling is older than history and is not bounded by one civilization, one continent, or one race." [15] In this section, we focus on the role of storytelling throughout history. Section 2.1.1 provides a brief historical context, and Section 2.1.2 outlines the key reasons people tell stories. The many different functions of storytelling emphasise its continued relevance to modern society, and justify the time invested by artificial intelligence researchers into not only understanding but replicating the process.

2.1.1 Storytelling Through the Ages

All known cultures, both past and present, practice storytelling in some form [29]. It is difficult to say exactly when storytelling began, but there is general agreement that it was long before the advent of written communication [15, 65, 119]. Since then, the media through which people experience story has continued to evolve, paralleling advances in technology. McLuhan [106] divides history into four periods, denoted by advances in communication techniques: the Tribal Age, the Literary Age, the Print Age, and the Electronic Age. During the Tribal Age, communication was primarily oral, and storytelling was very much a social activity, fostering a sense of community through gatherings around a teller or bard. With the development of the phonetic alphabet, societies entered the Literary Age, and stories moved from being exclusively oral to a visual medium. However, it was not until the Print Age, heralded by the development

of the printing press, that there was a true shift in the way stories were consumed, spurred by mass distribution of newspapers and printed stories. The Electronic Age began with the telegraph, but encompasses the present overabundance of technology: radio, television, computers, and mobile phones. These media provide new ways of experiencing an age-old pastime. Movies, interactive fiction, and many genres of computer games are nothing more than stories, presented in a different form. The way people live and interact has changed dramatically since our hunter-gatherer days, but our need for stories remains.

2.1.2 Why Do We Tell Stories?

It is clear that storytelling has been an integral part of human life for a long time. We are born with narrative ability; children do not need to be taught to tell stories, engaging in games of make-believe naturally from a very early age [65]. But why do we, as a species, tell stories? The answer is not as biologically obvious as the answer to questions such as "why do we eat?" Nevertheless, stories perform a number of important functions in our lives. In this section, we outline some of the key reasons people tell stories, including entertainment, education, communication, preservation of culture, making sense of our lives, and facilitating memory. We also reflect on the function of storytelling from a biological and evolutionary perspective.

Entertainment

More often than not, entertainment is the first thing that comes to mind when we think of stories. Looking back across the centuries, storytelling for entertainment provided a way to pass the time by the fire of an evening, or to make mundane, routine work tasks (anything from field work to carting wood, fishing, spinning, or weaving) more pleasant: "The tedium of work was often relieved by a background of tale telling." [119] Today, it is so much more than that. Studies have shown that, on average, the number of hours spent in paid work has gradually decreased since 1970 [117]. A corollary of this is an overall increase in leisure time, fuelling a multi-billion dollar global entertainment industry. Modern society is spoilt by a myriad of entertainment options. Consider some of the most popular: books, movies, television, and video games. On the surface, they are all very different, yet they have a fundamental common thread: story.

Books are the stereotypical form stories take. However, movies and television shows simply use a different medium to achieve the same thing: tell stories. The relationship

of video games to story is less overt. Some genres (for example, puzzle games like *Tetris*) will not necessarily tell a story. However, a vast number of video games do, ranging from first-person shooters, to role-playing games, and adventure games. Even simple platform games often have a story behind them. Think of Mario's never-ending quest to rescue Princess Peach; it may be simple, but it is still a story. In addition to these media, which comprise the present-day mainstream, continual improvements in digital technologies make newer approaches to storytelling, such as interactive fiction, more viable. The constant debate about the death of the book [49] has no reflection on the life of story. We can expect the role of story in entertainment to live on, through new and different media [65].

Communication

Although entertainment might be the function of storytelling we think of most readily, it is not the reason storytelling developed. Historically, storytelling developed to facilitate communication. Schank [150] asserts that conversation, the foundation of human communication, is actually nothing more than mutual storytelling. From this viewpoint, people engage in storytelling constantly, as an integral part of their interactions with others. Stories can have several specific goals within the context of communication, such as conveying a piece of information to a listener, or making the listener feel a particular way. This also applies more broadly than just direct communication between individuals and within small groups. The goals of mass media are exactly the same: to convey information. News, whether it be presented in a newspaper, on the radio, on television, or online, is more often than not structured as a story.

Education

"For most of recorded history, stories were the dominant means of education." [141] This is closely tied to the communicative function of storytelling, because, in essence, education is a special case of communication, where the goal is to inform the target audience about something considered to be useful. This allows people to acquire knowledge without the associated risks and costs of experiencing everything first-hand [65, 148]. Thus, in an educational context, stories serve as "a simulation of experience." [148] The knowledge imparted through stories can be roughly segmented into conveying concrete facts and information, and teaching human social skills.

The most obvious application of stories to education is to convey facts and information. Various societies throughout history used stories to convey facts related to survival, such as information about local geography, animals, and natural phenomena [148, 158]. In some cases, stories may be specifically designed to communicate this information. However, studies have shown that information about the world contained in works of fiction, even just as part of the setting or context, is also internalised by readers and integrated with the rest of their world knowledge [102]. This process occurs despite readers' awareness that not everything contained in a work of fiction is necessarily true or accurate, and that facts are often invented to support the plot [101, 102].

Perhaps the most important function of stories in education, however, is the way they can help develop social skills. Maintaining social groups became important early in human evolution, when it had a very direct impact on individuals' chances of survival [96]. Stories allow people to simulate social situations, and experience the associated emotions through their empathy with the characters [65, 96]. This gives people the opportunity to evaluate their reactions in a controlled environment, and better equips them to deal with similar situations if they arise in their own lives [96]. Empirical studies have shown that readers of fiction display better social ability compared to readers of non-fiction [97]. Child psychologist Bruno Bettelheim [23] makes a compelling argument about the importance of stories, in particular folk and fairytales, in helping children deal with the psychological pressures of growing up. In this context, children are not learning facts or concrete information, but rather the soft skills relating to personal relationships, social situations, and appropriate social behaviours. These skills are just as important for a child's development, but are so much harder to teach. Stories have the power to do it.

Preservation of Culture

Preservation of culture is, in a sense, related to education, although it is not the same thing. People often need to understand the stories implicit in a culture to function effectively in that culture [150]. For this reason, many countries around the world have placed a strong emphasis on storytelling, both in the past and today: Africa, Switzerland, India, the Pacific Islands, the Bahamas, Hungary, and the natives of America and Australia to name a few [119]. In his essay about the universal similarities between cultures, Murdock [113] makes the following point:

> A culture cannot persist unless it is transmitted from generation to gen-

> eration, and a society cannot survive without culture, which embodies in the form of collective habits the successful experience of past generations in meeting the problems of living.

Before written history, orally told stories were the only way in which a society or civilisation could preserve its culture [164], and therein lies their ubiquity in this regard.

Making Sense of Our Lives

Stein and Policastro [164] identify two primary reasons for telling stories. The first is to convey some kind of social message, which ties closely to the goals of communication, education, and preservation of culture, as discussed in the preceding sections. The second is about reorganising personal experiences. Some researchers suggest that people talk in order to analyse their own experiences, structuring them as a story and, by telling that story, selecting the parts of the experience to remember [65, 150]. Stories serve a similar function in psychotherapy; organising the events causing distress into a coherent story can make their emotional effects more manageable [120]. Narrative psychology specifically targets this approach. Stories give us a mechanism for coping with the flood of experiences, both good and bad, that assail our senses every day.

Memory

According to Schank, "human memory is story-based." [150] The implication is that information is better remembered when structured as a story, and in fact, we remember information *by* structuring it as a story. Schank and Abelson go so far as to say, "when it comes to interaction in language, all of our knowledge is contained in stories and the mechanisms to construct them and retrieve them." [151] This is a strong assertion, and one not all researchers agree with [28]. Nevertheless, irrespective of its extent, there is support for a link between narrative and memory. This is evidenced by studies investigating the impact of story structure on recall, which show that stories with a better plot structure are more likely to be remembered, especially long term [95, 166]. This connection to memory explains the usefulness of storytelling in areas like education.

An Evolutionary Perspective

The preceding discussion of the many functions of storytelling clearly shows its importance to every human culture across time and space. In this section, we consider stories

from an evolutionary perspective, and how the biology of the human brain enables us to use them as a rehearsal for real life. This is achieved through what are known as mirror neurons. Mirror neurons fire not only when a person performs a particular action, but also when they observe someone else performing the same action [134]. This extends to the way we experience stories:

> Why do we give ourselves over to emotion during the carefully crafted, heartrending scenes in certain movies? Because mirror neurons in our brains re-create for us the distress we see on the screen. We have empathy for the fictional characters—we know how they're feeling—because we literally experience the same feelings ourselves. And when we watch the movie stars kiss on-screen? Some of the cells firing in our brain are the same ones that fire when we kiss our lovers. "Vicarious" is not a strong enough word to describe the effect of these mirror neurons. When we see someone else suffering or in pain, mirror neurons help us to read her or his facial expression and actually make us feel the suffering or the pain of the other person. These moments, I will argue, are the foundation of empathy and possibly of morality, a morality that is deeply rooted in our biology. [74]

More than merely helping us learn by facilitating the imitation of physical actions, the suggestion is that "we understand the mental states of others by simulating them in our brain." [74] Empathy is crucial for maintaining effective social relationships [100], which in turn are an important part of what makes us human. Stories, because of the role mirror neurons play in our biological make-up, give us the opportunity to practice social interactions.

2.2 What is a Story?

Most of us have an intuitive sense of what a story, or narrative, is. The Oxford Dictionary provides a simple definition for narrative [3]:

> 1. a spoken or written account of connected events; a story:
>   *a gripping narrative*

There are four key points we can take from this definition. Firstly, story and narrative are synonymous; we will use these terms interchangeably throughout this work to refer

to the same kind of artefact. Secondly, a narrative consists of events. Thirdly, these events must be connected. This highlights the importance of plot; an arbitrary sequence of unrelated events does not comprise a narrative. Finally, medium is irrelevant—a story is a story no matter how it is presented. Despite its simplicity, this definition provides a clear idea of what a narrative is.

One common element of narrative, however, is not mentioned in this definition: characters. There is some disagreement among narratologists as to how important characters are to narrative. Barthes presents Aristotle's view that characters are not a requirement [19] :

> There can be fables without characters, according to Aristotle, but there cannot be characters without fables.

Fludernik, on the other hand, provides a very precise definition of narrative which identifies characters (protagonists) as indispensable [55]:

> A narrative (Fr. *recit*, Ger. *Erzählung*) is a representation of a possible world in a linguistic and/or visual medium, at whose centre there are one or several protagonists of an anthropomorphic nature [...] It is the experience of these protagonists that narratives focus on, allowing readers to immerse themselves in a different world and in the life of the protagonists.

We will not take sides on this matter, leaving it to narratological debate, and adopt the simple definition of narrative as an account of connected events. We will remark, however, that all examples of narrative and storytelling (human-authored or automatically generated) discussed in this work do involve characters in some way.

2.2.1 A Narratological Perspective

Narratology, in simple terms, is the study of narrative structure. To facilitate analysis, a distinction is generally made between the content of a story and how this is presented to the reader (or viewer). In light of this, we begin by discussing the tiers of narrative, and defining the relevant terminology. Following this, we move to a discussion of specific narrative theories. Cavazza and Pizzi [35] provide a useful summary of the narrative theories most often cited by research in interactive storytelling. We use their work to focus our discussion on those narratologists most relevant to the computational modelling of narrative: Aristotle, Propp, Greimas, Bremond, and Barthes.

The Tiers of Narrative

Although the division of narrative into tiers[1] separating content from presentation is generally accepted, the number of tiers used varies between narratologists. Chatman [39] proposes a simple two-tier structure consisting of story and discourse: "In simple terms, the story is the *what* in a narrative that is depicted, discourse the *how*." Bal [16] argues that three tiers should be distinguished: fabula (the content of the narrative), story (the ordering and selection of events from the fabula that are presented to the reader), and text (the physical presentation, equivalent to what Chatman calls discourse). In this breakdown, the fabula is equivalent to what Chatman refers to as story, whereas the term story is used to describe what the Russian formalists call sjuzhet.[2] Schmid [156] subdivides the central tier further, proposing a four-tier structure consisting of happenings (equivalent to the fabula), story (a selection from the happenings), narrative (composition), and presentation (the actual text, or other medium). In this case, story and narrative together would correspond to the notion of sjuzhet.

In our work, we assume Bal's three-tier structure and definitions, but adopt the terms fabula, sjuzhet, and discourse. This avoids confusion between Bal and Chatman's differing uses of the word story, and the implication that stories must be written, which is connoted by using text rather than discourse. We define these terms as follows:

Fabula: the content of the narrative (i.e. the characters, setting, and complete chronological sequence of events).

Sjuzhet: the specific selection of events from the fabula that are presented to the reader, along with their ordering.

Discourse: the physical form the story takes (e.g. the text in written narrative, the video footage in a movie, etc.).

The focus of our research is on generating the fabula. Sjuzhet and discourse are still relevant, in that any fabula we generate needs to be presented to readers, but our treatment of those tiers is cursory.

[1]The term tier is used by Schmid [156]; Bal [16] uses the term layer to describe the same concept. Our adoption of Schmid's terminology is arbitrary.

[2]Several different spellings of sjuzhet can be found in the literature (e.g. sujet, sjužet, syuzhet, suzet, etc.). The spelling "sjuzhet" has been adopted in this work.

Key Narrative Theories

Narratology generally focusses on the tiers of fabula and sjuzhet. The goal is to identify and formalise universal elements of structure. Given the diversity of stories in existence, this is a formidable task, and one which has invited a variety of approaches throughout history. The formal study of narrative can be traced back to Aristotle. In *Poetics* [12], he describes tragedy in terms of complication followed by dénouement, or unravelling. This forms the basis for the often referenced dramatic arc, which begins by building conflict, before reaching a climax that is terminated by resolution. This traditional plot structure is still taught to creative writing students today. However, despite its continued relevance, Aristotle's narratological treatment of tragedy lacks the formal depth of more recent approaches, which limits its usefulness.

The first real attempt to formalise narrative was Propp's *Morphology of the Folktale* [126]. Although he focussed on a very specific subset of narrative, the Russian folk tale, his work was groundbreaking in the way it decomposed stories into simpler units, and used them to characterise plot structure. Propp defined the Russian folktale as an ordered combination of 31 narrative functions, one or more of which could be omitted in any given tale. The main disadvatange of Propp's formalism, particularly from an interactive storytelling perspective, is that it "inherently prohibits any kind of 'branching functions' that could alter the course of the folktale to provide alternative paths." [35] Nevertheless, Propp's work can serve as a useful starting point for storytelling systems, and there has been some work in extending his formalism to non-linear interactive dramas [71].

Greimas' theory of narrative [68] builds on Propp's, but focusses on roles; Cavazza and Pizzi [35] describe it as "the first role-based analysis of narratives." Greimas breaks narrative down into a number of actants with opposing roles, such as the subject and the object, the sender and the receiver, and the helper and the opponent [72]. Characters are defined based on what they do (i.e. their role) in a narrative. Bremond [27] similarly focusses on opposing roles, defining two: the agent, who is able to initiate actions and changes in the narrative, and the patient, who is influenced by those changes. The main difference in his approach is that a character's role can change during the narrative, which provides greater flexibility. Bremond also takes into account characters' beliefs, motivations, and goals, making his analysis particularly relevant for character-centric storytelling.

Like Propp, Barthes, through his analysis of *Sarrasine* in *S/Z* [18], focusses on

action, but with a more flexible approach. At the end of *S/Z*, he provides a list of 48 actions appearing in *Sarrasine*. Cavazza and Pizzi [35] liken these to Propp's narrative functions, except they are not restricted to occurring in a fixed order. Barthes also decomposes narrative into five codes, which guide the interpretation of any particular section of text:

1. *Hermeneutic*: elements of a story that are mysterious or unexplained.
2. *Proairetic*: actions that imply further narrative action.
3. *Semantic*: elements suggesting additional meanings.
4. *Symbolic*: help to organise meanings.
5. *Cultural*: elements that refer to science or knowledge.

Cavazza and Pizzi identify the hermeneutic and proairetic codes as particularly important for interactive storytelling, in that they are key determinants of suspense.

The choice of narrative theory to underpin a storytelling system will depend largely on the system's goals. For instance, Greimas or Bremond's theories would suit a character-oriented approach far better than Propp's formalism would. However, a narratological approach to structural analysis does not always match up with the priorities of computer scientists, who focus more on practical implementation considerations than on characterising narrative from an academic standpoint. In acknowledgement of this distinction, we devote the following section to a discussion of the computing view of narrative.

2.2.2 The Computing Approach

As outlined in the previous section, different narratological theories focus on different aspects of narrative, and break it down according to a range of characteristics. When approaching story from a computational perspective, the goal is a breakdown which facilitates implementation. While there can be many different approaches, there are three key aspects of story that are generally distinguishable within computational models of narrative: action, character, and plot [142]. Action and character are the building blocks of fabula, while plot is the structuring element that corresponds to sjuzhet. In this section, we outline each of these elements, along with some of the tools and techniques available to model them.

Action

Action refers to what physically happens in a story. This includes the events that take place, and how they affect the state of the world. As defined in Section 2.2, a particular sequence of events forms a story. Thus, especially at the action level, narrative aligns closely with the artificial intelligence notion of planning: "coming up with a sequence of actions that will achieve a goal." [139] In the most basic sense, planning involves two concepts: a state describes the world at a particular point in time, and an action provides a transition between states. The aim of a planning system is to find a sequence of actions that begins at a given initial state and terminates at a specified goal state, as shown in Figure 2.1. Comparing this to Figure 2.2, which shows the basic structure of a story as a sequence of events, the likeness is clear. This allows for existing planning tools and techniques to be applied to narrative. Consequently, planning approaches have been a popular choice for storytelling systems [38].

Figure 2.1: Planning as a sequence of actions

Figure 2.2: Story as a sequence of events

Alongside classical planning approaches, there are a number of logical formalisms designed for reasoning about action and change. Two well-established formalisms are the situation calculus [130] and the event calculus [110]. The situation calculus is a second-order language designed to represent dynamically changing worlds [130]. It is based on situations, where a situation is represented as a sequence of actions. There are two axioms associated with each action: a possibility axiom, $Poss(\alpha, s)$, which says that the action α can be performed in situation s, and an effect axiom, which describes what happens after an action is executed. The language GOLOG [89] is designed for programming complex actions using the situation calculus. The event calculus explicitly incorporates the notion of time, so actions occur, and properties of the world hold, at specific time-points rather than in particular situations. Unlike in the situation calculus,

where time only exists as an ordering between actions, its time structure is absolute and modelled numerically [66]. Techniques such as constraint satisfaction have also been applied to planning problems [45]. Provided a problem can be reformulated using a standard representation (in this case, as a constraint satisfaction problem, or CSP), efficient off-the-shelf solvers can be used to find solutions. Answer set programming (ASP) provides similar benefits: a standard formulation facilitating the development of broadly applicable, optimised solvers.

However, both planning and the related action logics were developed with a view towards physical systems, where optimality (whether it be measured in terms of time, resources, etc.) is the key goal. As Porteous and Cavazza identify, this is not necessarily the case with narrative [125]:

> [...] the characteristics of a "good" plan, such as optimality, aren't necessarily the same as those of a "good" narrative, where errors and convoluted sequences may offer more reader interest, so some narrative structuring is required.

Although modern planning approaches facilitate placing additional conditions on plan generation (for example, preferences in PDDL3 [63]), the requirements of a good story are rarely defined in terms of specific actions (or events). The relevant features—tension, suspense, humour, emotion—are far more abstract. For this reason, planning at the action level alone, while sufficient to produce sensible and coherent stories, falls short in generating engaging narratives.

Character

Characters are a vital part of narratives. They are the instigators of the actions that move the story forward. More often than not, it is the characters that engage readers and hold their attention. Their relationships and interactions are the key element of some genres, such as drama. However, for characters to be believable, readers must understand the motivations behind their actions. This requires modelling characters' mental states: their beliefs, desires, and goals.

The area of artificial intelligence most relevant to representing character is that of intelligent agents, which are designed to act autonomously in response to their environment. This could be a physical environment, or a virtual story world. The most common framework for developing intelligent agents, the belief-desire-intention (BDI)

model, provides a standard architecture for modelling agents. This approach has been formalised logically [41], and programming languages have been designed to facilitate its implementation [26, 128, 144]. In addition to goal-directed behaviour, research has also been carried out on modelling other facets of character, including emotion [20, 43, 52, 129] and personality [30].

Plot

While action and character are the core constituents of fabula, a story composed by a human author is more than just characters performing actions in a goal-directed fashion. There is an overarching structure, which defines a trajectory for individual actions in order to produce a coherent and meaningful whole. This is plot. It is an aspect of story which is not as well-defined as action and character, because how a plot is structured will depend on the author's goals in telling the story, and the intended effect on a reader. This makes it difficult to define a generic framework. This is the challenge undertaken by author-centric storytelling systems, which will be discussed in more detail in Section 2.3.3. Most existing systems in this category have focussed on implementing a narrow range of well-defined plot features.

2.3 Storytelling Systems

There has been substantial work in developing computational storytelling systems, using a variety of different approaches. Riedl [132] divides storytelling systems into two main categories: character-centric and author-centric. Character-centric systems model characters as autonomous or semi-autonomous agents, with specific beliefs, desires, and goals. These agents are situated in a virtual world, and stories emerge from their interactions. Author-centric systems, on the other hand, focus on the overall structure of a story's plot, aiming to replicate the process of a human author. In Section 2.3.1, we present a rationale for developing storytelling systems. We then move on to discuss select examples of each approach, beginning with character-centric systems in Section 2.3.2, followed by author-centric systems in Section 2.3.3. Section 2.3.4 describes attempts to balance character-centric and author-centric goals. Finally, Section 2.3.5 places our system, MOSS, in context, identifying it as an author-centric storytelling system.

2.3.1 Why Develop Storytelling Systems?

The benefits of developing storytelling systems stem directly from the value of human storytelling. In Section 2.1.2, we outlined the key functions of story in human society. Of these areas, automatic storytelling holds the most notable benefits for entertainment, education, human-computer interaction, and furthering our understanding of human cognitive processes.

Entertainment is perhaps the area where the potential of storytelling systems is easiest to envisage. Story is integral to many modern forms of entertainment. For example, story is a key element of many computer games, in particular role-playing games [173]. The less pervasive medium of interactive fiction is even more strongly hinged on story. In essence, a work of interactive fiction allows the user to participate in the creation of a narrative:

> By offering the opportunity to participate in, or even co-narrate a story, interactive storytelling applications promise radically new modes of user experiences. [138]

The difficulty in granting users a high degree of control over the story in either medium is the underlying need to construct a large number of alternative plot lines and endings. Even if the player only has a handful of options at each decision point, having more than a few decision points quickly leads to a combinatorial explosion that is often infeasible to cater for. A system capable of automatically generating interesting and coherent plot lines based on the choices users make would be invaluable in this domain, granting users real influence over a story's plot, rather than just choosing between a small number of predetermined endings.

As explained in Section 2.1.2, although the term automatically brings to mind fiction and entertainment, storytelling evolved to facilitate communication and learning. Narrative structure is closely linked to memory and understanding [150]; presenting information in the form of a story improves the likelihood of it being learned. Given the vast quantities of knowledge people need to function effectively in modern society, education is arguably more important now than ever before. Storytelling systems give us a way to harness an age-old educational technique through technology:

> One of the most intriguing possibilities raised by the emergence of narrative intelligence is the potential to create narrative-centred learning environments. [109]

Formulating knowledge as a story is time-consuming. A system that could generate such stories automatically would prove extremely useful in education, particularly for children. Stories could be tailored to the interests and abilities of a particular audience, and even made interactive, motivating children to learn by providing an experience that is both interesting as well as educational [109].

The rapid spread of computing technology into every aspect of people's personal and professional lives has spurred a great deal of research in human-computer interaction. The primary aim of this field is to improve the way people interact with computers, or technology in general: "it is about creating user experiences that enhance and augment the way people work, communicate, and interact." [136] Given the central role of storytelling in human communication [150], it stands to reason that imparting aspects of this skill to computers would facilitate a more natural interaction.

From a more theoretical standpoint, building storytelling systems may shed light on human intelligence and cognitive function. Schank [150] identifies the study of storytelling processes as an avenue to better understanding the human mind. Being able to replicate the storytelling ability that is so uniquely human can bring us closer to understanding our own cognitive processes. In turn, breakthroughs in cognitive science could trigger advances in computational storytelling. In this way, the two fields are inextricably linked.

Computational storytelling systems can make a positive impact in a wide a range of areas. However, like many of the skills people take for granted, storytelling is extremely complex. Implementing an artificial storyteller capable of rivalling a human—even a child—is beyond the reach of current techniques. As such, rather than tackling the entire problem, individual systems focus on modelling specific aspects of story, tailored to particular goals or applications. Every system takes a small step forward—often in a slightly different direction, but nevertheless gradually enriching our understanding of the storytelling phenomenon.

2.3.2 Character-Centric Systems

The premise behind character-centric systems is to allow stories to emerge from characters' interactions. Characters are implemented as autonomous or semi-autonomous agents, each with their own set of beliefs, desires, and goals which govern their behaviour. In essence, a character-centric storytelling system is a simulation of these agents acting in a virtual world.

A number of storytelling systems have adopted this approach. One of the earliest examples, not only of this approach but of storytelling systems in general, was *TaleSpin* [107]. The system models animal characters, each of which has knowledge about the world and goals to achieve. Stories emerge from characters acting in pursuit of their goals, and adopting new goals based on the changing state of the world. However, the lack of high-level structure results in stories that, while consistent, "have no point or reason." [170] The more recent *Virtual Storyteller* [165] similarly represents characters as autonomous agents with goals. The main distinction is that the system incorporates a drama manager, which models plot and attempts to influence the character agents to achieve its requirements.

The *Oz Project* [20] and Cavazza, Charles, and Mead's interactive storytelling system [34] are character-centric systems designed for user interaction. In the *Oz Project*, autonomous agents are presented as animated characters, with which the user interacts. Agents have individual goals, and perform behaviours to fulfil those goals. In Cavazza, Charles, and Mead's work [34], characters' goals and plans are represented as hierarchical task networks (HTNs), in which goals are decomposed into sub-goals. The final level (i.e. the terminal nodes of the HTN tree) corresponds to actions that are actually performed. There are clear advantages to using a character-centric approach for storytelling systems that are interactive; the autonomous nature of the character agents allows them to respond to unpredicted user inputs in a robust fashion. However, there is no guarantee these interactions will produce quality narrative.

The main benefit of the character-centric approach, regardless of whether the resulting system is intended to be interactive, is that the characters are believable. Their actions make sense, in that they are consistent with their personalities and goals. The main disadvantage, however, is that there is no guarantee the resulting narrative will be coherent or interesting. To address this, many character-focussed systems do incorporate plot-level techniques to a limited extent, as exemplified by *Virtual Storyteller*'s drama manager. However, the more control given to a higher-level planner, the less true autonomy is possessed by the characters, and thus the greater the likelihood that they will engage in behaviours inconsistent with their traits or personality.

2.3.3 Author-Centric Systems

Author-centric storytelling systems focus on an author's goals in composing a story. This is analogous to the way human authors write. Generally, an author-centric system

will plan a story at a more abstract level than specific events or actions, to achieve plot-level goals. In such systems, characters are nothing more than "the effectors through which the hypothetical author changes the world." [132] Some author-centric systems attempt to formalise a particular plot-level feature (or features), and use this to structure stories. This corresponds roughly to the way in which formalists like Propp [126] break stories down into simpler units which can be manipulated. Other systems are more concerned with modelling the process of a human author. In this section, we discuss examples of both approaches.

One of the early examples of author-centric story generation is *Universe* [85], a storytelling system designed to generate continual, episodic stories, analogous to a soap opera. The system generates stories by selecting author-level goals, and progressively expanding them into sub-goals until reaching the level of specific actions. *Universe* does model personality traits to a limited extent, in an attempt to make characters behave consistently, but this does not drive the plot. The key difference between *Universe* and other author-centric systems is the open-ended nature of the stories produced. In Lebowitz's words [85]:

> An entire UNIVERSE story cannot be planned out, since the kinds of stories we envision theoretically do not end.

Consequently, the system does not plan a story based on a single, pre-defined plot trajectory, but rather based on smaller plot fragments (e.g. forced-marriage [86]), which satisfy particular constraints.

Another approach to modelling plot is to try and induce a particular effect on a reader. The effect a human author aims to achieve depends largely on genre. A thriller aims to create feelings of suspense, a comedy to amuse, whereas a melodrama appeals to readers' emotions. *Suspenser* [40] is a story generation system which attempts to create suspense by controlling what information is presented to the reader based on a given fabula; i.e. it works at the level of sjuzhet. The underlying premise is that the degree of suspense experienced by a reader depends on how many solutions they perceive as available to a problem faced by the protagonist. This manner of correlating a story's intended effect with tangible elements of its structure is useful in that it provides a way to implement what would otherwise be a very abstract notion.

Mexica [122] is a system which aims to model the cognitive process of writing. It incorporates a database of user-defined previous stories, which the system analyses to produce schemas that include knowledge about both content and author-level

constraints. *Mexica* produces stories using an engagement-reflection cycle, and avoids using explicit goals or information about story structure. During the engagement stage, the system searches through the schemas to find a set of possible next actions based on the current story world context. An action is selected at random from those available. The reflection process verifies the coherence of the story; if any actions have unfulfilled preconditions, additional events are inserted to address this. This is different to most planning-based approaches, in which actions are not selected for inclusion in a story unless their preconditions are satisfied. The lack of explicit structural elements also sets *Mexica* apart from the majority of author-centric systems.

Turner's *Minstrel* [170] generates stories about King Arthur and the Knights of the Round Table. Stories are constructed around themes, which are the system's primary structuring element. The system's focus is a particular subset of themes which provide planning advice for certain situations. *Minstrel* implements six such themes, and they are represented using schemas referred to as Planning Advice Themes (PATs), which are an extension of Dyer's Thematic Abstraction Units (TAUs) [48]. In addition to thematic goals, *Minstrel* also takes into account dramatic goals (suspense, tragedy, foreshadowing, and characterisation), consistency goals (avoiding inconsistencies arising in the story world), and presentation goals (selecting and ordering the events to present to readers, and generating the text). What makes *Minstrel* interesting is its attempt to model an author's creative process. Turner describes the system as a case-based reasoner, which treats storytelling as problem-solving, and uses knowledge from existing stories to find solutions. To achieve creativity, *Minstrel* uses special constructs called TRAMs (Transform-Recall-Adapt Methods). TRAMs transform the given problem into a new problem, and retrieve possible solutions based on this new problem. This facilitates searching areas of knowledge not directly related to the current problem. The retrieved solution is adapted to apply to the original problem, resulting in a creative solution.

In general, an author-centric system's main strength corresponds to a character-centric system's weakness. Due to their plot-level focus, author-centric systems usually generate stories which are coherent. Conversely, their characters often lack the depth and consistency produced using the character-centric approach. Many author-centric systems do attempt to take character into account, but in most cases the character modelling they employ is very simplistic.

2.3.4 A Balancing Act

As Riedl [132] points out, balancing plot coherence with character believability is critical for effective story generation:

> The author-centric approach uses planning to solve for goals that represent an outcome or premise that the author wishes a story to have. However, the author's goals are meaningless if the story world characters do not achieve those goals in believable way.

Unfortunately, the two aspects can be difficult to reconcile, which is why the bulk of the work in story generation has prioritised one over the other. In this section, we discuss systems that explicitly set out to balance plot and character. Although the authors of these systems classify them as author-centric, we discuss them separately due to the concerted effort made to incorporate character-centric techniques.

Rield's PhD dissertation [132] specifically sets out to address the problem of balancing plot coherence and character believability. Riedl's system, *Fabulist*, achieves this by extending partial order causal link planning [176] to incorporate both author and character goals. The resulting planning algortithm, referred to as intent-driven partial order causal link planning, "simultaneously searches the space of plans and the space of agent intentions." [132] The algorithm uses backtracking to ensure characters' actions are consistent and intentional. After finding a sequence of actions which produces the desired story outcome (i.e. satisfying the desired author-level goal), *Fabulist* checks to ensure characters' actions are believable. If they are not, the planner backtracks and either selects a different action, a different character, or a different formulation of the original character.

Riedl and Young's *Actor Conference* [131] also uses partial order planning to determine the sequence of actions for a story. The key difference is that individual characters are implemented as expert systems, referred to as Actors. The system implements a blackboard architecture. The overall planning problem is broken up into smaller subproblems, which are allocated to the Actors to fulfil. Each Actor is limited to planning actions for the character it represents, and does so in a believable fashion because it embodies that character's attributes. The resulting plans are integrated into a hypothesised partial narrative by the blackboard.

Façade [104] is an interactive system focussed on the user's interaction with a married couple, Grace and Trip, on the brink of divorce. It takes a different design approach,

loosely based on game mechanics, to achieve its primary goal of granting users agency. The story is organised around "social games" [104]; the player's score impacts how the drama unfolds. Characters are implemented using the reactive planning language ABL [103], but the main structuring elements are units called beats, which consist of particular actions or exchanges between characters. A drama manager, or beat sequencer, determines which beat to activate next to achieve the desired (author-specified) tension arc. This tight integration of plot trajectory with character behaviour implicitly balances the requirements of story coherence and character believability. The main disadvantage of *Façade*'s approach is the extensive manual authoring of beats required.

2.3.5 Placing MOSS in Context

The goal of our work is to generate stories which convey morals. Our Moral Storytelling System (MOSS) is an author-centric system: story generation is controlled at the plot level, with no explicit attempt to model characters. The use of morals as the structuring element of plot suggests a resemblance to *Minstrel*, with its focus on themes. Turner [170] considers themes to be equivalent to morals:

> The story theme is the point or moral of the story.

We believe there are subtle distinctions between these three concepts—theme, point, and moral—and discuss them in Chapter 3. Nevertheless, the specific subset of themes implemented in *Minstrel* do align closely with our notion of morals. In particular, we note the similarity between three of Turner's themes and the morals we implement: PAT:Good-Deeds-Rewarded (Reward), PAT:Hasty-Impulse-Regretted (Recklessness), and PAT:Pride-Fall (Pride). We perform a more in depth comparison of *Minstrel*'s PATs to MOSS's morals in Chapter 3, but use this section to highlight the ways in which our work differs from Turner's.

The main distinction between MOSS and *Minstrel* is the approach to representing morals. *Minstrel*'s PATs are highly structured and complex, representing themes explicitly as advice. The information each PAT contains is grouped based on three categories [170]:

1. *Type of Advice*: the type and value (positive or negative) of the advice.
2. *Advice*: the advice itself, including the decision it relates to, its consequence, the causal link between the decision and consequence, the object to which the decision applies, and the planner to whom the advice applies.

3. *Context*: the planner's active goals, current goal, current plan, and facts about the world which must be true to apply the advice.

While there are certainly benefits to such a detailed representation, reflected in the sophistication of *Minstrel*'s stories, the downside is its limited applicability. PATs can only be used to represent the narrow range of themes which provide advice about handling particular planning situations, each of which must be defined in terms of this schema.

MOSS, on the other hand, does not use structured schemas for representing morals. Instead, we leverage the relationship between morals and another inherent element of story: the emotions characters experience. Every story that involves characters features emotions, whether they are explicitly conveyed to the reader, or only implied through character behaviour. To truly capture the essence of a story, storytelling systems need to address emotion in some way. Thus, our approach offers a way to represent morals in terms of a concept many systems already model. Emotions are a powerful foundation because they embody complex relationships, resulting in a very simple representation at the moral level: a temporal sequence of emotions. Representational complexity is confined to modelling emotion itself, which has been more widely studied [14, 50, 58, 76, 83, 84, 118, 123, 137, 140, 153, 155, 174], and more frequently applied in the sphere of computing [20, 37, 43, 44, 52, 99, 124, 129, 133, 160, 179], than morals have.

Our approach has several advantages. The most obvious is its simplicity (at the moral level; we acknowledge that emotions are a complex phenomenon, but appeal to the fact that formal descriptions are available). It is also more generally applicable. Any story that involves emotion—which is virtually any story—is open to being modelled in terms of its emotional trajectory. The trade-off is less specificity than *Minstrel* is able to achieve. Emotions exist at a relatively high level of abstraction, limiting fine-grained control over the plot. Closely related morals may also share the same (or very similar) emotion patterns. However, our choice of implementation technology, ASP, makes it straightforward to impose further constraints without requiring significant system changes. This also opens the door to using emotions as a more general structuring element for plot trajectory, not necessarily restricted to morals, though morals are the focus of our work.

2.4 Summary

In this chapter, we provided a background to place our work in context. We began by outlining the role of storytelling in human culture, identifying its many important functions, and thereby providing motivation for research in computational storytelling. We proceeded to define story, and presented two different perspectives on its structure: the narratological view, in which stories are analysed in terms of various structural elements, and the computing approach, in which the focus is facilitating implementation.

Shifting our focus to storytelling systems, we started with a rationale for their development, by outlining possible areas of application. Next, we summarised the two main approaches to story generation, the character-centric approach and the author-centric approach, reviewing key examples of systems adopting each. Finally, we identified our system, MOSS, as author-centric, with a focus on morals as the structuring element of plot, and compared it to *Minstrel*, which we identified as closely related due to its analogous goal of generating stories with specific themes. Having set the scene, in the following chapter we explore the underlying element of our work: the moral of a story.

Chapter 3

The Moral of the Story

> *Why do people tell stories? The stories that tend to stick to our bones are those that teach us something. This, I believe, is the primary reason we tell stories — to teach.*
>
> Brian McDonald
> *Invisible Ink*

Some stories stay with us. We can recall them years, even decades later, despite having heard them only once. We do not remember everything, of course. The exact wording will certainly escape us, as will many of the details. What we do remember, however, is the message. A mention of *The Tortoise and the Hare* immediately brings to mind the adage "slow and steady wins the race," just as *The Boy Who Cried Wolf* reminds us why we should not lie. Why is it that we remember these stories, which we may not have heard since childhood, yet so easily forget things we read only last week? It is because they have a moral: a lesson we can apply to our lives, other people's lives, or the world around us. Our brains commit them to memory because they are useful.

Following on from our introduction to story in the previous chapter, here we delve deeper into the aspect of stories that makes them meaningful: their moral. We begin by defining morals in Section 3.1, and move on to discuss their importance in Section 3.2. Section 3.3 reviews related work in computational storytelling and agent design with a focus on morals. In Section 3.4, we return to human storytelling, identifying fables as a useful source of morals. Section 3.5 goes on to describe our categorisation of a selection of Aesop's fables by moral, to produce a list of those that appear most

often. In Section 3.6, we introduce our approach to representing morals using character emotions, and identify the morals we will focus on in our work. Section 3.7 provides a summary of this chapter.

3.1 What is a Moral?

Before entering into a discussion about the importance of morals and how they can be utilised in storytelling systems, it is important to define what we mean by the term moral. The Oxford Dictionary defines a moral (noun) as follows [2]:

> 1. a lesson that can be derived from a story or experience:
>
> *the moral of this story was that one must see the beauty in what one has*

The moral of a story is therefore the lesson that the story conveys to its readers. This can be viewed from two different perspectives: that of the author, and that of the reader. Although ideally the lesson the reader learns from the story will be the same as the lesson the author intends to convey, this is not always the case, because story interpretation is subjective, particularly for sophisticated stories. In their discussion of story morals, Dorfman and Brewer focus on the message the reader believes the author intends to convey [47]:

> For the purposes of this article, we define the moral or point of a fable to be the didactic message or lesson embodied in the text that the reader believes the author intends to convey.

We take the opposite viewpoint; we will be using morals for story generation, and thus it is more important to focus on the lesson the author (or system user) intends, because that is the message the resulting story needs to communicate. The message the reader interprets from the story is still important, in that it allows us to evaluate the system's effectiveness, but in this sense it is secondary.

There are two closely related concepts also worth mentioning here: the theme and the point of a story. Some researchers [47, 48, 170] use one or both of these terms interchangeably with the word moral. This warrants a brief discussion to explain how we believe they are related to, and more importantly differ from, the moral of a story. A theme (noun) is defined as [7]:

1. the subject of a talk, piece of writing, exhibition, etc.; a topic:
   the theme of the sermon was reverence

Themes are the focus of Dyer's work in story comprehension, and he links them directly to morals [48]:

> Understanding a narrative "in depth" involves recognizing the moral or point of a narrative. This is analogous to being able to characterize the theme of a narrative in some appropriate way, as in selecting an apt title or adage for it.

The themes Dyer handles are common adages, which he also refers to as morals. In our view, a theme is a broader concept than a moral. Although a story's moral can be its theme, a theme does not necessarily have to be a moral, as we will demonstrate with reference to some of the adages used in Dyer's work.

Some adages do convey a clear lesson, and thus can be called morals: for example, "two heads are better than one," or "don't count your chickens before they're hatched." However, there are others which do not necessarily teach a lesson. For instance, we would not consider "the pot calling the kettle black" a moral. This common idiom describes someone accusing somebody else of something they are themselves guilty of. The example story Dyer provides for this adage is as follows [48]:

> MINISTER'S COMPLAINT
>
> In a lengthy interview, Reverent R severely criticized President Carter for having "denigrated the office of president" and "legitimized pornography" by agreeing to be interviewed in Playboy magazine. The interview with Reverent R appeared in Penthouse magazine.

Most readers would view this in a negative light, because hypocritical behaviour is usually considered inappropriate. However, such an interpretation has more to do with a reader's pre-existing values and social conditioning than actually being conveyed by the story. Although this adage certainly does characterise the essence of the story, which was Dyer's aim, we would not call it a moral. However, it certainly is a theme, and also a point.

The most pertinent definition of a point (noun) for this context is [4]:

(usually **the point**)
the significant or essential element of something being planned or discussed:
it took her a long time to ***come to the point***

Wilensky argues the importance of a story's point to its representation, and his definition is similar [177]:

> Points are structures that define those things that a story can be about.

In our view, this is more akin to a theme than it is to a moral. As with themes, the moral contained within a story can also be its point (particularly in the case of stories specifically constructed to communicate a moral), but not all stories with a point necessarily convey a moral. As Schank et al. [152] explain, there are many different kinds of points. They identify seven, and only one of these, prescriptive points, come close to the notion of a moral as defined earlier in this section. Throughout this work, when we use the term moral we refer specifically to the lesson a story is intended to convey to its readers.

3.2 The Importance of Morals

Looking back at our discussion of the reasons for storytelling in Section 2.1.2, most of these purposes (most notably communication, education, and preservation of culture) require a story to have a point: to convey information in some way. As discussed in the previous section, a moral is a specific kind of point that teaches the reader a useful lesson. Researchers in fields as diverse as psychology [23, 172], education [77, 82, 141, 158], neuroscience [74, 115], and cognitive science [152] recognise the importance of morals in stories. Given their value in human storytelling, morals deserve comprehensive treatment in computational storytelling as well; this has been acknowledged by a number of researchers [22, 47, 48, 170]. Although some work has been carried out in this area, there has been less of a focus on morals in this field as compared to other aspects of storytelling [47]. In the following section, we review existing work in computational storytelling with a focus on morals, as well as the related concepts of themes and points, which will provide a context for our contribution.

3.3 Morals in Computational Storytelling

The work we discuss in this section concerns not only morals in storytelling systems, but also morals in the related areas of story comprehension and agent design. Although our contribution is geared towards story generation, much of the existing work dealing with morals has been carried out in story understanding and the design of moral or emotional agents, making it important to compare this to our work. In Section 3.3.1, we discuss existing work on morals in story generation. In Section 3.3.2, we review work in story understanding which attempts to leverage a story's moral. Finally, in Section 3.3.3 we discuss the use of morals in building artificial agents.

3.3.1 Story Generation Systems

Work in directly applying morals to story generation has been relatively scant. The only storytelling system which explicitly attempts to model and convey morals, in the form of common adages, is Turner's *Minstrel* [170], which was discussed at a high level in Chapter 2. Turner refers to these adages as themes, which we defined in Section 3.1 as a broader concept than morals. However, he addresses a very specific subset of themes which convey planning advice; i.e. they teach a lesson. This matches up with our definition of a moral, and warrants closer scrutiny. Themes in *Minstrel* are represented using structures called Planning Advice Themes, or PATs. PATs are based on Dyer's Thematic Abstraction Units (TAUs) [48], but modified to represent themes explicitly as planning advice. As explained in Section 2.3.5, PATs are complex structures incorporating information about the type of advice, the advice itself, and the context required to apply the advice. *Minstrel* implements six PATs, summarised in Table 3.1.

Some of *Minstrel*'s PATs are very similar to morals we implement in our own work, which we define in Section 3.5.2 (refer to Table 3.3). The strongest resemblance is between PAT:Good-Deeds-Rewarded and Reward. The main distinction is Turner's requirement that the recipient of the good deed is the one who returns the favour. Our definition is broader; we allow the reward to originate elsewhere, in a more karmic sense.[1] Similarly, PAT:Pride-Fall is subsumed by our more general treatment of Pride. Turner's definition only relates to pride demonstrated by ignoring warnings, whereas

[1] We provide two separate rules for Reward in our system (refer to Chapter 5). The more specific of these, Reward Type 2, captures Turner's required reciprocation.

Planning Advice Theme (PAT)	Advice
PAT:Good-Deeds-Rewarded	Help others, and someday the favour will be returned.
PAT:Spite-Face	Do not plan to cause a goal failure for someone else using an action that will also cause a goal failure for yourself: "cutting off one's nose to spite one's face."
PAT:Bird-In-Hand	Avoid plans which can cause the failure of a previously achieved goal: "a bird in the hand is worth two in the bush."
PAT:Juliet	Do not use deception plans, as they may fool an unintended party and lead to failure.
PAT:Hasty-Impulse-Regretted	Do not implement irreversible plans based on a hasty observation, as it may prove to be incorrect.
PAT:Pride-Fall	Being too proud to heed warnings may bring grief.

Table 3.1: Summary of *Minstrel*'s PATs

in our case there are no restrictions on how pride is exhibited. PAT:Hasty-Impulse-Regretted is comparable to the moral we call Recklessness, though the likeness is more superficial in this case. Turner's theme references an action with irreversible outcomes performed based on a hasty deduction about the world, which later proves to be incorrect. Our interpretation of Recklessness focusses on unanticipated consequences resulting from actions performed in haste. Thus, while both definitions have lack of forethought in common, along with negative consequences arising from a hasty action, the motivation for that action is different. PAT:Juliet could possibly be compared to Lies and Superficiality (which we do not implement), though again Turner's theme is far more specific than the way we define those morals.

A more recent example of a thematic approach is that of Hargood, Millard, and Weal [69]. They adopt Tomashevsky's [168] notion of a theme, a broad idea composed of subthemes and motifs. Hargood et al. describe their model as "built of *natoms* (narrative atoms) which contain *features* that denote *motifs* which in turn connote *themes*." They provide an example: the natom daffodil would relate to the motif flower, which would imply the theme of Spring. This conception of a theme is extremely general, more akin to the definition we provided in Section 3.1 than to Turner's. In principle, a theme can be very similar in nature to a moral, but this is not the case in Hargood et al.'s work; Spring certainly cannot be construed as a moral. Although they did construct a

prototype of their model, the *Thematic Model Builder*, it uses photos rather than story elements as a source of natoms, attempting to select images based on themes rather than directly using their tags. Hargood et al.'s discussion of integrating these ideas into narrative generation systems is more speculative than practical.

3.3.2 Story Comprehension Systems

Although they have not featured prominently in story generation, morals, themes, and points have certainly been investigated in the context of story understanding. For instance, Wilensky's story-understanding system, *PAM* (*Plan Applier Mechanism*), can summarise and answer questions about an input story. *PAM* attempts to detect the points in a story through the use of point prototypes, which define the requirements a story must fulfil to be "pointful." [177] These are essentially frames which are matched against what happens in a story to identify its point. *FAUSTUS* (*Frame Activated Unified STory Understanding System*) [116] works in a similar way, but uses a single processing scheme to make inferences from a range of different kinds of frames, which represent different aspects of a story, including objects, plans, settings, and Wilensky's points.

Significant work on representing ideas similar to our definition of morals was carried out by Dyer [48]. His system, *BORIS*, aims to understand stories based on their themes, and demonstrates this understanding by answering questions about them. In Dyer's case, themes (encapsulated by common adages) are often also what we call morals. Dyer developed Thematic Abstraction Units (TAUs) to represent these adages. TAUs are based on expectation failures which arise from characters' planning failures in a story, which Dyer posits can capture the essence of the adage contained within that story. Dolan [46] extends Dyer's work on TAUs and incorporates them into his system *CRAM* (*Causal Reasoning in Associative Memory*), which recognises planning errors within stories in order to draw analogies between them.

Although Dyer's themes are closely related to our idea of morals, Dyer makes the following comment regarding *BORIS*'s scope [48]:

> BORIS attempts to understand just a few very complicated stories as deeply as possible in contrast to the approach of skimming a great number of stories.

As a result, his Thematic Abstraction Units are closely tailored to specific kinds of stories. He also asserts that TAUs focus only on a particular subset of themes:

> We have stated that TAUs capture one class of themes – i.e. those involving errors in planning.

There are many morals for which it is easy to imagine some examples of stories that involve planning errors, and others that do not. In these cases, TAUs would not be capable of comprehensively characterising stories with those morals, which limits their usefulness as a general representational construct. These observations about TAUs also apply to PATs, described in Section 3.3.1, because they are based on TAUs.

3.3.3 Agents

While there has been some work in agent design referencing morals, this has more to do with agents' moral values than the higher-level concept of a story moral. For example, Shaheed and Cunningham [157] take steps towards implementing agents capable of making moral decisions, as defined based on Kohlberg's theory of moral development [78]. In their system, agents use their emotions, their beliefs about other agents' emotions, and their level of morality to make decisions about how to behave. Although they focus on agents in this work, Shaheed and Cunningham do propose eventually using these moral agents to generate stories with morals. This is analogous to our goal, albeit using a more character-oriented approach.

Battaglino, Damiano, and Lesmo's [22] treatment of morals is less direct. Although they mention story morals in their introduction, their focus is on building emotional agents, and handling what they refer to as "the moral dimension of emotions": namely, pride, self-reproach (or shame), admiration, and reproach. In other work, Battaglino and Damiano [21] consider the moral values conveyed by stories, and how they relate to the moral emotions on which characters base their decisions. They provide a detailed example of how their agent model applies to a particular greek tragedy, *Iphigenia in Aulis*, by Euripides. While it is not the goal of their work, agents capable of making effective moral decisions could certainly be applied to the generation of stories with morals, resulting in more realistic characters with whom readers could empathise.

3.4 Sources of Morals

The number of morals that can be conveyed through story is bounded only by the limits of the human imagination. A story can be crafted to convey any message its author

wishes to communicate. However, some morals arise more frequently than others, not only through time, but also across cultures. Their very prevalence is a strong indicator of their importance. Bettelheim makes this point during his discussion of fairy tales and their value to children [23]:

> The fairy tale, on the other hand, is very much the result of common conscious and unconscious content having been shaped by the conscious mind, not of one particular person, but the consensus of many in regard to what they view as universal human problems, and what they accept as desirable solutions. If all these elements were not present in a fairy tale, it would not be retold by generation after generation. Only if a fairy tale met the conscious and unconscious requirements of many people was it repeatedly retold, and listened to with great interest.

In line with Bettelheim's conclusions, we chose to focus our work on those morals which occur most often in human storytelling. Our first step is to identify these morals, so we can then use examples of stories which convey them to identify common features that can be leveraged for their representation.

The difficulty is that even stories which appear simple to people are actually extremely complex. For one thing, many contain multiple morals, interleaved with one another. This can make it difficult to classify stories by their moral; it depends on interpretation, and the reader's specific focus. Furthermore, stories contain much more than just what is necessary to convey their moral. Authors imbue them with additional features in order to achieve specific goals, such as increasing interest, encouraging empathy, or generating suspense. Thus, even if the primary moral of a story could be reliably identified, there is no way to determine which features of the story are directly relevant to conveying that moral.

In light of this, the most suitable stories for our work would be those that are as simple as possible, but still convey a moral. As it turns out, there is a type of story that fits these criteria perfectly: fables. Fables stem from oral storytelling tradition, and so they are simple by necessity—they had to be short enough to be remembered and retold. More importantly, "fables' visceral plots always generate explicit morals – interpretations that turn the preceding stories into convincing figures for home truths." [90] Mandler and Johnson agree, defining a fable as follows [95]:

$$\text{FABLE} \rightarrow \text{STORY AND MORAL} \tag{3.1}$$

Fables provide a useful source of morals because of their simplicity; there are no (or at least very few) extraneous events or story features to get in the way. Many fables are only a few sentences long, the minimal detail required to communicate the intended message. Close analysis of a fable can thus yield useful information about the minimum requirements to convey its particular moral.

There are numerous collections of traditional fables, from various cultures and different periods in time. One of the best-known collections is Aesop's fables [9], attributed to Aesop, an Ancient Greek slave thought to have lived in the 6th century BC. Other early examples originated in India, including the *Jātaka Tales* [56] and *Panchatantra* [159], which date back to the 4th and 3rd centuries BC respectively. There are also many more recent collections. Krasicki's *Bajki i Przypowieści* (translated as *Fables and Parables*) [79] were written in Poland in 1779. Krylov's fables [81] are Russia's best-known collection, dating back to the early 18th century, whereas *Uncle Remus* [70] is a collection of African-American tales that were first compiled and published in 1881. Hans Christian Andersen's fairy tales [10] similarly date back to the mid-18th century. Collections of fables have also been compiled in the 20th century, such as those by March [98] and Thurber [167].

Although many of the above fable collections were compiled in times long past, they continue to be referenced, reprinted, and retold today, which is a testament to their enduring relevance to people's lives. We chose to base our work on Aesop's fables, not only in light of their place in history, but also their familiarity to many modern readers.

3.5 Morals in Aesop's Fables

For our work, we refer to *Aesop: The Complete Fables* [9], translated by Olivia and Robert Temple, and published by Penguin Classics in 1998. This translation is based on Professor Émile Chambry's 1927 French text, *Ésope Fables, Texte Établi et Traduit par Émile Chambry* [36]. This version contains 358 fables in total, numbered based on their alphabetical ordering by Greek title. The translators acknowledge that there is no consensus among researchers as to which fables would comprise a complete set of Aesop's work, or in fact which fables, if any, were actually written by Aesop. As such, other collections exist which contain varying numbers of fables. For our work, the exact number of fables is not important, provided that the collection contains multiple examples of the same moral for comparison, which this collection does. We present

our moral-based categorisation of Aesop's fables in Section 3.5.1, and identify the most prevalent morals in the collection in Section 3.5.2.

3.5.1 Grouping Aesop's Fables by Moral

The first step in our analysis of Aesop's fables is to group them by moral, allowing us not only to determine which morals are the most prevalent, but also to compare fables with the same moral to identify common features. On the surface, this may seem like a trivial exercise, because the intended message or moral of each fable is explicitly stated following that fable's text. However, these morals are not always directly useful in the form in which they are written, requiring us to perform our own categorisation. There are two main reasons for this, discussed below.

The first issue we encountered with the supplied morals is that they are not generalised into any kind of fixed categories. As a result, two fables with the same broad moral can have that moral described very differently; in fact, no two fables share the same description. To illustrate, consider the morals provided for three of Aesop's fables, as follows:

Fable 3: The Eagle and the Fox *This story shows that if you betray friendship, you may evade the vengeance of those whom you wrong if they are weak, but ultimately you cannot escape the vengeance of heaven.*

Fable 16: The Goat and the Donkey *Whosoever schemes against others owes his own misfortune to himself.*

Fable 244: The Mouse and the Frog *Even death can avenge itself; for divine justice observes everything, and restores the equal proportions of her balance beam.*

Each is worded differently, very specifically referencing the associated fable. However, in essence, all three encapsulate the same basic moral: retribution. In each of these fables, a character does something to harm to another character, and is punished in some way at the end of the story. The congruence between different fables is not always immediately obvious from their moral descriptions, and thus, to group fables into a useful set of categories, we could not simply take the supplied morals as written, but had to carefully consider each along with its associated fable, and assign our own category names.

The second issue is that although a moral is stated at the end of each fable, it is not always meaningful—and it is not always a moral. For instance, the moral stated for Fable 155, *The Gardener and the Dog*, is as follows:

> This fable is addressed to those who are ungrateful and unjust.

We would argue this is not really a moral at all; it only specifies the target audience that should learn from this fable, not the lesson they should absorb. Although it does hint at a moral—one would guess something to do with gratitude or justice—it is not entirely clear from this statement alone what that moral is without making significant assumptions. Reading the fable elucidates that it is in fact another example of retribution, allowing us to classify it as such.

During our categorisation, there were also many fables that had to be excluded, for three reasons. The first is that some fables rely on readers having certain preconceived ideas about the nature of particular animals to convey their morals. For instance, it is taken for granted that snakes and wolves are inherently wicked. Fable 122, *Zeus and the Snake*, illustrates this (the suggested moral is in italics):

> **Fable 122: Zeus and the Snake**
> When Zeus got married, all the animals brought him presents, each according to his means. The snake crawled up to him, a rose in his mouth. Upon seeing him, Zeus said:
> 'From all the others I accept these gifts, but from your mouth I absolutely refuse one.'
> *This fable shows that you should fear the favours of the wicked.*

Without the prior knowledge that snakes are wicked, a reader would not be able to correctly decipher the moral of this fable. While notions such as these may have been generally accepted by readers at the time when these fables were written, today's readers have a different frame of reference. Thus the morals in fables such as this might not be understood without further explanation.

The second reason for exclusion is that some fables have no actual events taking place. Instead, they are comprised entirely of conversations, characters reflecting on abstract concepts, or symbolism relating to particular objects or animals. For example, Fable 259, *The Traveller and Truth*, reads as follows:

Fable 259: The Traveller and Truth

Travelling through a desert, a man came upon a solitary woman who kept her eyes lowered.

'Who are you?' he asked.

'I am Truth [*Alēthia*],' she replied.

'And why have you left the town and come to live in the desert?'

She replied:

'Because, in days gone by, lies were only known to a few men, but now they are everywhere – with everyone you speak to.'

Life will be evil and painful for men while lies prevail over truth.

This fable is a conversation, with no physical events, and its link to the supplied moral is not made explicit (though it is implied). The ultimate goal of our work is story generation, where events will play an important role in modelling a story, as discussed in Section 2.2.2. As such, from a practical perspective, it is only useful to consider fables in which the moral is conveyed through what happens in the story, because these are the kinds of stories we are aiming to generate.[2] As defined in Section 2.2, our definition of a story requires events, and thus a fable without any concrete events is not, for our intents and purposes, a story.

Finally, some fables simply do not have a clear moral. The story alone does not convey a moral in an obvious way, and the link between the story and the supplied moral is tenuous at best. For example, consider Fable 213, *The Lion Who Was Afraid of a Mouse, and the Fox*:

Fable 213: The Lion Who Was Afraid of a Mouse, and the Fox

There was a lion lying asleep one day when a mouse ran all the way up his body. The lion awoke with a start and rolled over and over, trying to find out what or who it was that had attacked him. A fox, who had seen all this, rebuked him for being afraid – he, a lion, afraid of a mouse! To which the lion replied:

'It isn't that I was afraid of the mouse, but I was most surprised that there was anyone at all who could be so bold as to run along the body of a sleeping lion.'

[2]Speech acts could be modelled as events, which would make fables consisting of conversations applicable. However, our system, as described in Chapters 4–6, does not model speech acts, and thus these fables are not useful in our case, and have been excluded.

This fable shows that wise men don't ignore even the little things.

No moral is evident from the text of this fable. Even when considered alongside its supplied moral, the connection is not obvious; there is certainly nothing in the fable to imply wisdom, or demonstrate any benefit to the lion of noticing the mouse. Any categorisation attempted for fables such as this will be inherently contentious, and thus we exclude them from our analysis.

After making the necessary exclusions, we are left with 85 fables, in which we identify 28 different morals, as summarised in Table 3.2.[3] The fable numbers listed in the table correspond to the numbering system used in *Aesop: The Complete Fables*. The morals are ordered based on the number of fables in which they appear (from most to least), or, when this is the same, alphabetically.

3.5.2 The Most Common Morals

Our decision to focus on the most commonly occurring morals was explained in Section 3.4: they are likely to have been the most important to people throughout history, which makes them particularly valuable for story generation. Based on Table 3.2, seven morals stand out, each occurring at least four times in Aesop's collection: retribution, greed, lies, realistic expectations, recklessness, pride, and superficiality. In addition to these morals, we include reward in our analysis, due to the close parallel between reward stories and retribution stories. One is a direct reversal of the other, and we expect this to extend to their representations.

In Table 3.3, we provide a brief description of the lesson corresponding to each of these eight morals, based on their manifestation in Aesop's fables, so their definitions are clear when we refer to them throughout the remainder of this work. Henceforth, when we refer to a particular moral and assume the definition from Table 3.3, we use a capital letter (e.g. Retribution). When the same word is used in another context, assuming its normal English meaning, we do not capitalise (e.g. retribution).

[3]We originally grouped the Retribution and Reward stories together, under Retribution. Despite the word retribution often having negative connotations, it is nevertheless considered a synonym for reward [6]. As such, the version of Table 3.2 appearing in preliminary work [145] distinguishes only 27 morals. However, the clear fundamental differences between stories conveying positive retribution (i.e. reward) and negative retribution drove us to separate the two.

Moral	Fables	Total
Retribution	3, 16, 45, 103, 142, 155, 166, 168, 244, 270	10
Greed	30, 90, 140, 146, 163, 185, 204, 239, 287	9
Lies	15, 35, 51, 55, 59, 305, 318	7
Realistic Expectations	5, 23, 33, 128, 150, 193	6
Recklessness	38, 40, 68, 181, 301	5
Pride	20, 29, 161, 237	4
Superficiality	37, 76, 102, 119	4
Appreciation	257, 271, 273	3
Hard Work	24, 53, 352	3
Honesty	230, 253, 279	3
Hypocrisy	48, 63, 151	3
Laziness	89, 96, 265	3
Learn from Mistakes	184, 209, 308	3
Reward	6, 206, 242	3
Unity	71, 86, 215	3
Loyalty	100, 300	2
Planning	241, 336	2
Quality	194, 342	2
Culpability	245	1
Flexibility	143	1
Jealousy	124	1
Overreaction	294	1
Sacrifice	153	1
Selfishness	41	1
Slander	205	1
Temper	58	1
Violence	73	1
Weakness	291	1
TOTAL		85

Table 3.2: Aesop's fables grouped by moral

Moral	Description
Retribution	If a character performs a reprehensible action, they will be punished.
Greed	If a character behaves in a greedy fashion (e.g. attempting to obtain excessive wealth or possessions), they will suffer as a result.
Lies	If a character lies, harm will come to them as a result.
Realistic Expectations	Characters should have realistic expectations for themselves. If they attempt something unrealistic, they will not succeed, and will often come to harm.
Recklessness	A character should not behave recklessly (i.e. without considering the potential consequences of their actions). Reckless actions lead to harm.
Pride	A character should not be excessively proud. Feeling excessive pride leads to harm.
Superficiality	Surface appearance means nothing. Characters should not pretend to be something they are not, or they will be discovered, and will suffer as a result.
Reward	If a character performs a good deed, they will be rewarded.

Table 3.3: Descriptions of eight prevalent morals identified in Aesop's fables

3.6 Representing Morals

Having identified some common morals, our next step is to choose an approach for representing these morals that will facilitate story generation. As identified in Chapter 2 (refer to Section 2.2.2), from a computational perspective, two key components of story are events and characters. The story's plot emerges from the way the events and characters are assembled into a cohesive whole. The most intuitive way to represent a story is to model the events that take place. However, the specific events that comprise a story are not directly linked to its moral, because many different stories can convey the same moral. For instance, consider *The Boy Who Cried Wolf* and *Pinocchio*; although both convey the moral that lying is bad, they achieve this through very different sequences of events.

To represent morals effectively, we require a feature that is common to a diverse range of stories sharing the same moral. In Section 3.6.1, we discuss possible rep-

resentation options, and select character emotions. Section 3.6.2 acknowledges that additional elements may be necessary to adequately represent certain classes of morals, and explains how this applies to the eight morals identified as most prevalent in Section 3.5.2. Finally, in Section 3.6.3 we list the morals selected for implementation.

3.6.1 Choosing a Representation

A few different representations have been developed for moral-like concepts. In his discussion of the structure of memory, Schank [149] proposes mental structures for organising abstract, domain-independent information, which he calls Thematic Organization Points (TOPs). The purpose of these hypothetical structures is to help people draw parallels between events at a more abstract level. This hints at a relevance to morals, which are a more abstract interpretation of events. However, TOPs are constructed around goals which must involve the relationship of two parties with respect to those goals; this significantly limits their applicability. In Section 3.3, we described Dyer's Thematic Abstraction Units (TAUs) [48] and Turner's Planning Advice Themes (PATs) [170], both of which were developed to model the theme of a story. We identified that although Dyer and Turner's notions of themes are similar to morals, TAUs and PATs are complex and specific structures, based on characters' planning failures and planning advice respectively. Wilensky's point prototypes [177] are also complex, taking into account various types of information depending on the specific point being modelled, such as goals, emotional reactions, state information, and so on.

We suggest morals can be represented in a more general way, using simpler constructs. We seek a representation more in line with Lehnert's plot units [87], which consist of affect states that can be combined to represent particular types of events and plot configurations. However, Lehnert's affect states only distinguish positive, negative, and neutral emotions. To represent many different plot units, they must be combined using complex arrangements of links, which, particularly when stories become more sophisticated, can begin to undermine the simplicity of affect states as a representational choice.

Rather than limiting ourselves to three affect states, we propose using a broader set of emotions to define morals. To explain our approach, we present the basic structure of a story in Figure 3.1. It consists of a sequence of events, which involve one or more characters. Usually, the events that take place in a story will cause the characters involved to experience a variety of emotions. The important point to note here is that

very different sequences of events can result in extremely similar (or even identical) patterns of emotion; this is what we intend to leverage. We suggest that particular morals can be distinguished, and hence represented, by the patterns of character emotions that occur during the course of a story. Thus, by modelling character emotions in our corpus of fables, and identifying sequences of these emotions common to stories with a specific moral, we can then generate stories with that moral by selecting events that make characters feel the required emotions.

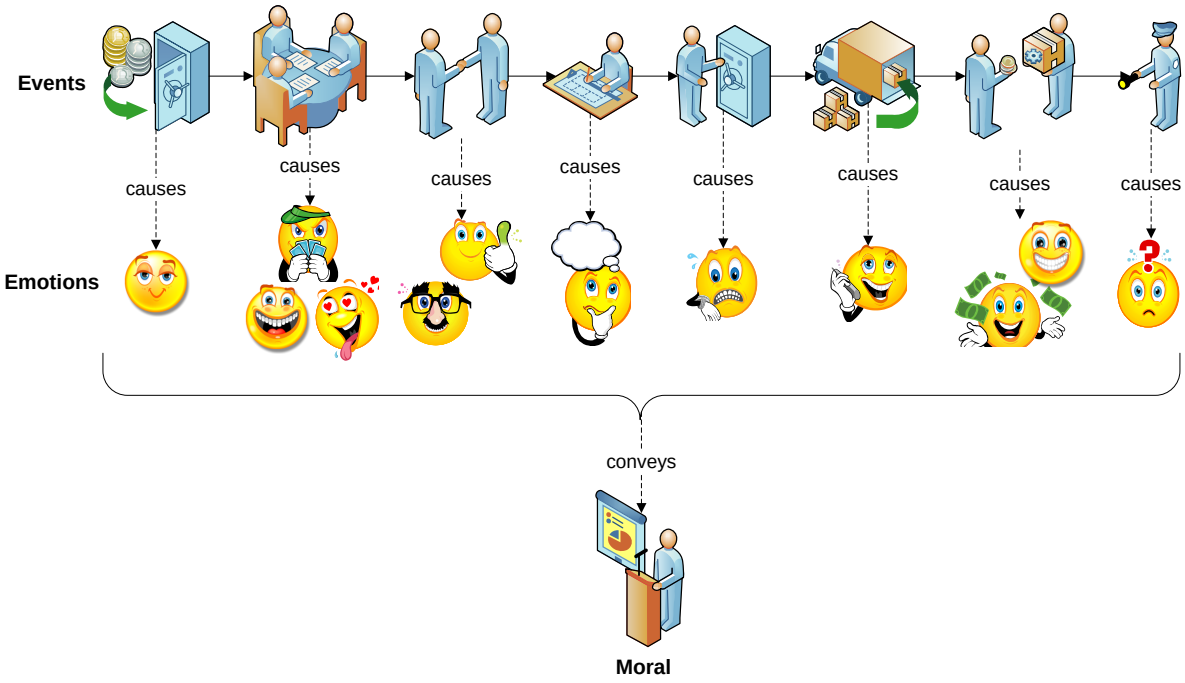

Figure 3.1: The basic structure of a story

There has been considerable work in computationally modelling emotion, in the context of emotional agents [22, 52, 99, 129], character-driven storytelling [43, 44, 165], and even driving narrative plot [121]. Some of these systems were discussed in Chapter 2 (refer to Section 2.3). In our work, we utilise character emotions to produce stories with morals. Although a relationship between morals and emotions has been suggested in other work [22, 157], in these cases the focus was on designing virtual agents and managing their emotional responses to events, not planning story trajectories. Rather than focussing on character, we utilise emotions as an author-centric device for manipulating plot to convey selected morals. The details of our approach to modelling emotions are provided in Chapter 4.

3.6.2 Additional Requirements

Although emotions will be our foundation for representing morals, we certainly do not mean to imply that emotions alone are sufficient for definitively characterising all possible morals. Stories, and consequently morals, can be as complex or specific as an author wishes to make them. Furthermore, some morals are very closely related to one another. In these instances, additional features apart from emotions may be required to distinguish between them. Nevertheless, we consider emotions to be sufficient for generating stories with a core subset of morals. They can also serve as a key element of representing more sophisticated morals.

Upon closer analysis, differences in representational complexity emerge even among the eight morals listed in Table 3.3. Table 3.4 presents these morals, along with the requirements we suggest they place on a storytelling system to effectively produce stories which convey them. All these morals require emotions, but three of them have additional requirements. Realistic Expectations requires an action model that permits actions to fail, because this is what happens when a character attempts an action that is beyond their capabilities (and hence unrealistic). This is a relatively minor requirement, because an action model where all actions always succeed is very limited. Lies and Superficiality, on the other hand, have a more significant prerequisite: characters must be able to have incorrect beliefs. This is non-trivial, because it requires a system to manage how beliefs are acquired and revised, and track each character's beliefs independently of what is actually true in the story world. Appropriately handling beliefs is a substantial artificial intelligence problem in its own right, and there has been extensive work in this area [11, 17, 33, 59, 114, 171].

Moral	Emotions	Action Failure	Incorrect Beliefs
Retribution	×		
Greed	×		
Lies	×		×
Realistic Expectations	×	×	
Recklessness	×		
Pride	×		
Superficiality	×		×
Reward	×		

Table 3.4: Minimum story world requirements to represent different morals

3.6.3 Morals Selected for Implementation

Following the closer inspection of the representational requirements of the eight most prevalent morals in Aesop's fables in Section 3.6.2, we make a considered decision about which to implement in our story generation system. The focus of our work will be those morals that can be represented by sequences of character emotions, using a straightforward story world and simple belief model. We will not delve into the intricacies of belief management, and assume omniscience for our characters (refer to Section 4.5.5 for details of our belief model). For this reason, we implement only six of the eight morals defined in Table 3.3: Retribution, Greed, Realistic Expectations, Recklessness, Pride, and Reward. We isolate these six morals in Table 3.5, showing the fables from Aesop's collection corresponding to each.

Moral	Fables	Total
Retribution	3, 16, 45, 103, 142, 155, 166, 168, 244, 270	10
Greed	30, 90, 140, 146, 163, 185, 204, 239, 287	9
Realistic Expectations	5, 23, 33, 128, 150, 193	6
Recklessness	38, 40, 68, 181, 301	5
Pride	20, 29, 161, 237	4
Reward	6, 206, 242	3
TOTAL		37

Table 3.5: Six morals selected for implementation

3.7 Summary

In this chapter, we introduced morals, and explained their important role in storytelling. Following this introduction, we reviewed existing work in artificial intelligence with a moral focus, concentrating on the areas of story generation, story understanding, and intelligent agents. We identified fables as a useful source of morals, and proceeded to categorise Aesop's fables by moral, which allowed us to identify the most common morals in the collection. We proposed character emotions as an approach for representing morals, with the goal of story generation, and selected six morals for implementation: Retribution, Greed, Realistic Expectations, Recklessness, Pride, and Reward. In the following chapter, we describe our implementation of an established

theory of emotion, the OCC theory, which we will then use in Chapter 5 to develop a representation for the selected morals.

Chapter 4

Modelling Emotion

Emotion is one of the most central and pervasive aspects of human experience.

Andrew Ortony, Gerald Clore, Allan Collins
The Cognitive Structure of Emotions

Whether they be in the form of a book, a movie, or orally told, stories have the power to make us laugh or cry, happy or sad, angry or frightened. We often rate the quality of a story based on how effectively it makes us feel these emotions. But why do we feel them, despite knowing what we read is fiction? More often than not, it is the empathy we feel towards the characters in the story. We identify with them, put ourselves in their shoes, and imagine what it would be like to have those same experiences. When the protagonist overcomes a seemingly insurmountable obstacle, we feel their elation; when they are in the depths of despair following a terrible loss, we feel it too. The emotional roller-coaster characters ride is vital to our interpretation and understanding of a story, and thus also a critical tool in crafting a story's moral.

In this chapter, we explore the psychology of human emotion, with the goal of applying it to the morals identified in Chapter 3. We begin with an overview of the dominant psychological theories of emotion in Section 4.1. Section 4.2 reviews how these emotion theories have been applied thus far in a computing context. In Section 4.3, we dive deeper into the most popular emotion theory for computational applications, the OCC theory, before providing its logical formalisation in Section 4.4. Section 4.5 describes our implementation of the OCC theory, which we will use in Chapter 5 to represent morals, and Section 4.6 presents the evaluation we performed on this model.

We summarise the contributions of this chapter in Section 4.7.

4.1 Theories of Emotion

Many psychological theories of emotion have been proposed over the years, identifying varying sets of emotions, and describing different mechanisms for their formation. Battaglino, Damiano, and Lesmo [22] discern three main categories into which most emotion theories can be grouped: physiological theories, dimensional theories, and appraisal theories. In this section, we introduce these three approaches and identify the key differences between them.

Physiological emotion theories [50, 76, 83] are founded on the premise that emotions result from physiological reactions to external stimuli. Emotions are effectively our interpretations of these physical reactions, as explained by James [76]:

> Our natural way of thinking about these standard emotions is that the mental perception of some fact excites the mental affection called the emotion, and that this latter state of mind gives rise to the bodily expression. My thesis on the contrary is that *the bodily changes follow directly the* PERCEPTION *of the exciting fact, and that our feeling of the same changes as they occur* IS *the emotion*.

To illustrate this perspective with a simple example, the premise is that we feel fear because we tremble. This is in direct opposition to other types of emotion theories, in which we tremble because we feel fear; the sequence has been reversed. We can also include in this category theories in which physiological reactions and emotions occur simultaneously [32], as distinct from the dimensional and appraisal theories, where emotions result from cognitive processes which then govern physical responses.

Dimensional theories [123, 140, 155, 174] define emotions using one or more dimensions. The number of dimensions, what they are, and how they are arranged depend on the specific theory. For example, Russell's circumplex model [140] uses two orthogonal axes: the horizontal axis represents valence (positive-negative), the vertical axis arousal. Emotions are placed along a circle in this two-dimensional space. On the other hand, the PANA model [174] places positive affect (low-high) and negative affect (low-high) on separate axes (vertical and horizontal respectively), and adds valence and arousal as second set of orthogonal axes at 45 degrees to those.

Appraisal theories [14, 58, 84, 118, 137, 153] assume that emotions result from a person's cognitive appraisal of a situation, regardless of physiological arousal. Appraisal can be based on a range of different factors, usually dependent on how a person evaluates a particular event or situation with respect to their goals [118]. Given the absence of any physiological elements to a computer system, it is hardly surprising that appraisal theories have generally been the most popular choice for computational applications. We discuss some of these applications in the following section.

4.2 Applications of Emotion in Computing

Computers are based on logic. Their programming, at least in the traditional sense, is inherently rational. Nevertheless, emotion has found a place in a range of computational applications. Given how crucial emotions are in governing people's behaviour, it follows naturally that imbuing computers with some sense of this phenomenon can facilitate improved interaction. Emotion can be utilised by computational systems in two distinct ways, which can also be combined within a given system:

1. The creation of artificial emotional agents, which maintain an emotional state that governs the agent's behaviour.

2. Modelling the emotional state of the system's user, and using this information to adapt and improve the interaction.

In this section, we briefly outline applications that adopt each of the above approaches. We do not aim to provide a comprehensive review of all the work that has been done in the field of affective computing, but rather to give a taste of the key areas where emotion has had an impact on computing and artificial intelligence.

The first approach, building emotional agents, is generally geared towards applications within entertainment domains. Emotion can be incorporated into non-player characters in computer games as a mechanism for guiding their decisions and behaviour, bringing them closer to the goal of realistically simulating human opponents [37, 51, 160]. Realistic and believable agents are also a key ingredient in interactive storytelling and drama [20, 43, 44, 52, 129]; in fact, Damiano and Pizzo argue that it is emotion which establishes "a distinction between autonomous agents and drama characters." [43] Emotional agents can even be used as virtual companions, not just for entertainment (for example, virtual pets [57]), but also for serious applications, such as

companions for elderly people [80], or to support patients during their stay in hospital [24]. Moving further away from entertainment, incorporating intelligent emotional agents in user interfaces can make interactions with software systems more pleasant [133]. Furthermore, given the importance of emotion in human decision making, some researchers suggest emotions could be utilised in agent architectures more broadly, for agent control in situations where they compete for survival with other agents [154].

The alternative focus is modelling the user's emotions. The premise is simple: if a program can somehow monitor the emotional state of its user, it can adapt appropriately to provide a better experience. Automated tutoring systems are a good example [67, 124, 179]. Emotions can be applied in such systems to better manage content presentation to reflect the user's emotional state, and help promote motivation:

> An intelligent tutoring agent should be capable of responding to the learners [*sic*] activities in an emotionally appropriate way, that is in such a way that it stimulates the learning process and avoids that the student becomes uninterested or frustrated. [124]

The same principle can be applied more generally to user interface design. Once a user's emotional state is identified (which can be achieved through various means, ranging from facial recognition [108], to physiological signs gathered through touch [13] or wearable sensors [93]), it becomes possible to "adapt the system functionality to this state." [73] What exactly this means will depend on the nature of the application, and of course the nature of the emotion. Continued improvements in reliably identifying people's emotions, both physiologically and through facial expressions, will surely spur an even broader range of applications.

4.3 The OCC Theory

Of the many psychological theories of emotion described in Section 4.1, few are amenable to computational modelling due to a general informality of the psychological descriptions. An important exception is the OCC theory of emotion [118], which fits into the category of appraisal theories. The OCC theory was specifically designed to be computationally modelled, making it a popular choice for computational applications, ranging from emotional agents [52, 99, 129] to story generation [121, 165]. We adopt it as the foundation for our emotion model.

The OCC theory identifies three distinct aspects of the world with regards to which emotions can be experienced: events, agents, and objects. This gives rise to three main categories of emotions: event-based emotions, agent-based emotions, and object-based emotions. Additionally, there is a group of emotions that are simultaneously agent and event-based, referred to as compound emotions. The overall structure of the emotions defined by the OCC theory is shown in Figure 4.1. Ortony, Clore, and Collins [118] take care to emphasise that the language of emotions "comes replete with ambiguity," and thus the emotion words they use are intended to denote general emotion types, rather than specific emotional states. Many different emotion words, describing particular emotional states of varying character and intensity, correspond to each of these emotion types.

In the following sections, we describe the OCC emotion categories and their subsumed emotion types in more detail. We begin with event-based emotions (Section 4.3.1), followed by agent-based emotions (Section 4.3.2), object-based emotions (Section 4.3.3), and finally compound emotions (Section 4.3.4). All definitions and examples of emotion words are transcribed from Ortony et al.'s descriptions. Henceforth in this thesis, when we refer to an emotion type, and assume the OCC meaning, we will use a capital letter (e.g. Joy). When the same word is used in another context, assuming its normal English meaning, we do not capitalise (e.g. joy).

4.3.1 Event-Based Emotions

Event-based emotions arise from agents' reactions to events. Ortony et al. identify three kinds of event-based emotions: well-being emotions, prospect-based emotions, and fortunes-of-others emotions. Although all three groups are dependent on the desirability of events, each fixates on this desirability from a different angle, and is thus affected by a different set of intensity variables. We discuss each type of event-based emotions separately in the ensuing sections.

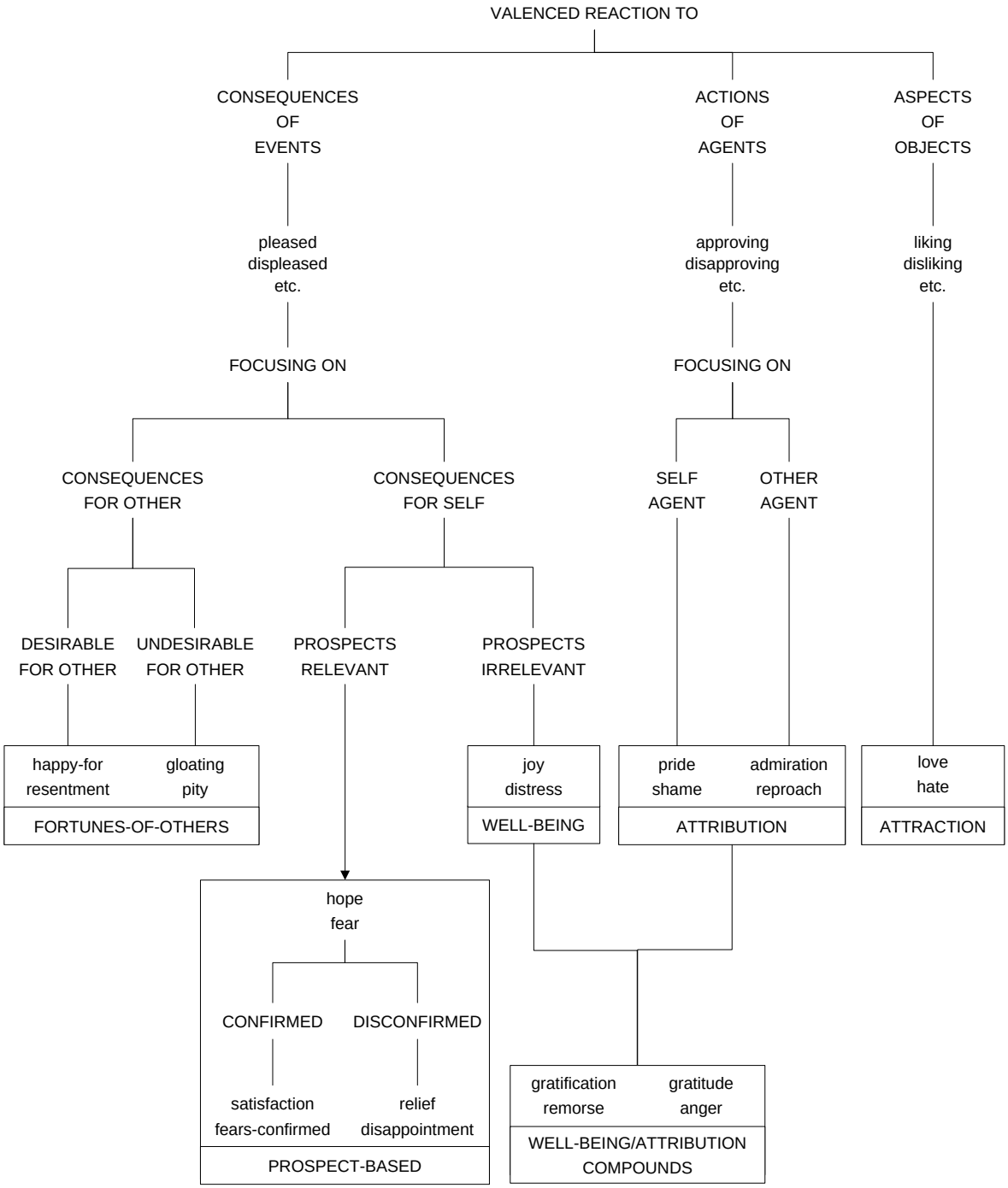

Figure 4.1: Structure of the OCC emotions

Well-Being Emotions

The well-being emotions (Joy and Distress) arise purely from an agent's appraisal of an event as desirable or undesirable, as summarised in Table 4.1. Ortony et al. define them as follows:

Joy: being pleased about a desirable event; e.g. contented, cheerful, delighted, ecstatic, elated, glad, happy, joyful, jubilant, pleased, etc.

Distress: being displeased about an undesirable event; e.g. depressed, distressed, displeased, distraught, grief, miserable, sad, unhappy, upset, etc.

The only variable affecting the intensity of the well-being emotions is the degree to which the event is desirable (for Joy) or undesirable (for Distress). It is also worth noting that "an event can have both desirable and undesirable aspects" [118]; the OCC theory allows both Joy and Distress to be experienced with respect to the same event. To illustrate when such a situation might arise, Ortony et al. use the example of learning about the death of a beloved relative, from whom one has inherited a great fortune. This event has both desirable and undesirable aspects.

Appraisal of Event	
Desirable	Undesirable
Joy	Distress

Table 4.1: Well-being emotions

Prospect-Based Emotions

The key distinction between prospect-based emotions and the well-being emotions described above is that prospect-based emotions take into account an agent's expectations. Prospect-based emotions can be experienced with regards to the prospect of an event (i.e. when that event is unconfirmed), and also with regards to the confirmation or disconfirmation of that prospect. The six prospect-based emotions are summarised in Table 4.2, and defined as follows:

Hope: being pleased about the prospect of a desirable event; e.g. hope, anticipation, expectancy, looking forward to, etc.

Fear: being displeased about the prospect of an undesirable event; e.g. fear, apprehensive, anxious, dread, fright, nervous, scared, terrified, worried, etc.

Satisfaction: being pleased about the confirmation of the prospect of a desirable event; e.g. satisfaction, hopes realised, gratification, etc.

Fears-Confirmed: being displeased about the confirmation of the prospect of an undesirable event; e.g. fears confirmed, worst fears realised, etc.

Disappointment: being displeased about the disconfirmation of the prospect of a desirable event; e.g. disappointment, dashed hopes, despair, frustration, etc.

Relief: being pleased about the disconfirmation of the prospect of an undesirable event; e.g. relief.

There are two variables which affect the intensity of the unconfirmed prospect-based emotions:

1. The degree to which the event is desirable (for Hope) or undesirable (for Fear).
2. The likelihood of the event.

There are three variables which affect the intensity of the confirmed and disconfirmed prospect-based emotions:

1. The intensity of the attendant Hope emotion (for Satisfaction and Disappointment) or Fear emotion (for Fears-Confirmed and Relief).
2. The effort expended in trying to attain the event (for Satisfaction and Disappointment) or prevent the event (for Fears-Confirmed and Relief).
3. The degree to which the event is realised.

Event Status	Appraisal of Prospective Event	
	Desirable	Undesirable
Unconfirmed	Hope	Fear
Confirmed	Satisfaction	Fears-Confirmed
Disconfirmed	Disappointment	Relief

Table 4.2: Prospect-based emotions

Fortunes-of-Others Emotions

Fortunes-of-others emotions are felt with regards to events concerning others. Thus, unlike the other two categories of event-based emotions, they also take into account the presumed desirability of the event for the other person, as well as how the appraising agent feels about that person (i.e. whether they like them or not). These emotions are summarised in Table 4.3, and defined thus:

Happy-For: being pleased about an event desirable for someone else; e.g. happy for, delighted for, pleased for, etc.

Pity:[1] being displeased about an event undesirable for someone else; e.g. pity, sorry for, compassion, sad for, sympathy, etc.

Resentment: being displeased about an event desirable for someone else; e.g. resentment, envy, jealousy, etc.

Gloating: being pleased about an event undesirable for someone else; e.g. gloating, Schadenfreude, etc.

The four variables affecting the intensity of the fortunes-of-others emotions include:

1. The degree to which the event is desirable (for Happy-For and Gloating) or undesirable (for Pity and Resentment) for oneself.
2. The degree to which the event is presumed to be desirable (for Happy-For and Resentment) or undesirable (for Pity and Gloating) for the other person.
3. The degree to which the other person deserved (for Happy-For and Gloating) or did not deserve (for Pity and Resentment) the event.
4. The degree to which the other person is liked (for Happy-For and Pity) or not liked (for Resentment and Gloating).

[1] In the OCC theory, Ortony et al. use the term Pity for this emotion type in their diagram of the theory's structure (reproduced in Figure 4.1), but later refer to it as Sorry-For in their definitions. We use the term Pity to refer to this emotion type throughout this work.

Reaction of Self	Presumed Value for Other	
	Desirable	Undesirable
Pleased	Happy-For	Gloating
Displeased	Resentment	Pity

Table 4.3: Fortunes-of-others emotions

4.3.2 Agent-Based Emotions

Agent-based emotions (also referred to as attribution emotions) are directed towards the agent responsible for the event concerned, and depend on whether their action is judged to be praiseworthy or blameworthy, as shown in Table 4.4. Praiseworthiness and blameworthiness are evaluated with respect to the appraising agent's standards. The attribution emotions are divided into those that are directed at the self (Pride and Shame), and those that are directed towards other agents (Admiration and Reproach).

Identity of Agent	Appraisal of Agent's Action	
	Praiseworthy	Blameworthy
Self	Pride	Shame
Other	Admiration	Reproach

Table 4.4: Attribution emotions

Self-Directed Attribution Emotions

The self-directed attribution emotions are defined as follows:

Pride: approving of one's own praiseworthy action; e.g. pride.

Shame: disapproving of one's own blameworthy action; e.g. shame, embarrassment, feeling guilty, mortified, self-blame, self-reproach, etc.

The intensity of self-directed attribution emotions depends on the following variables:

1. The degree of judged praiseworthiness (for Pride) or blameworthiness (for Shame).
2. The strength of the cognitive unit with the actual agent.
3. Deviations of the agent's action from person/role-based expectations (i.e. unexpectedness).

The second variable in the preceding list, cognitive unit strength, represents the extent to which the appraising agent feels connected to the agent carrying out the action. This extended notion of the self allows emotions such as Pride and Shame to be experienced with regards to actions performed by others, provided that the appraising agent identifies strongly with that other agent. The example Ortony et al. provide is Mary feeling Pride that her daughter saved someone's life; although Mary did not perform the action herself, the strength of her cognitive unit with her daughter is sufficient for the emotion to arise.

Other-Directed Attribution Emotions

The other-directed attribution emotions are analogous to the self-directed attribution emotions, but directed towards others. They are defined as follows:

Admiration: approving of someone else's praiseworthy action; e.g. admiration, appreciation, awe, esteem, respect, etc.

Reproach: disapproving of someone else's blameworthy action; e.g. reproach, appalled, contempt, disdain, indignation, etc.

This time, there are only two relevant intensity variables, because the strength of the cognitive unit with the self is not a factor:

1. The degree of judged praiseworthiness (for Admiration) or blameworthiness (for Reproach).

2. Deviations of the agent's action from person/role-based expectations (i.e. unexpectedness).

4.3.3 Object-Based Emotions

The object-based emotions (also referred to as attraction emotions) are based on the appealingness of objects, which can include other agents. There are only two such emotions, Love and Hate, as shown in Table 4.5. Their definitions are as follows:

Love: liking an appealing object; e.g. love, adore, affection, attracted to, like, etc.

Hate: disliking an unappealing object; e.g. hate, aversion, detest, disgust, dislike, loathe, repelled by, revulsion, etc.

Ortony et al. identify two variables affecting their intensity:

1. The degree to which the object is appealing (for Love) or unappealing (for Hate).
2. The degree of familiarity with the object.

Appraisal of Object	
Appealing	Unappealing
Love	Hate

Table 4.5: Attraction emotions

4.3.4 Compound Emotions

Compound emotions result from simultaneously focussing on the praiseworthiness and desirability of a particular situation. There are four compound emotions, and each results from a specific pairing of an attribution emotion with an event-based emotion, as shown in Table 4.6. It is worth highlighting Ortony et al.'s assertion that both the relevant event-based and attribution emotions must be experienced for a compound emotion to arise. For example, a praiseworthy action that has no desirable consequences for the appraising agent cannot result in Gratitude, even if desirable consequences were intended by the agent performing the action. Due to the fact that attribution emotions can be self or other-directed, the compound emotions derived from them are partitioned the same way.

Attribution Emotion	Event-Based Emotion	
	Joy	Distress
Pride	Gratification	–
Shame	–	Remorse
Admiration	Gratitude	–
Reproach	–	Anger

Table 4.6: Compound (event/attribution) emotions

Self-Directed Compound Emotions

Self-directed compound emotions are derived from self-directed attribution emotions when combined with event-based emotions, and are defined as follows:

Gratification: approving of one's own praiseworthy action and being pleased about the related desirable event; e.g. gratification, pleased with oneself, self-satisfaction, smug, etc.

Remorse: disapproving of one's own blameworthy action and being displeased about the related undesirable event; e.g. remorse, penitent, self-anger, etc.

There are four intensity variables affecting the self-directed compound emotions:

1. The degree of judged praiseworthiness (for Gratification) or blameworthiness (for Remorse).
2. The strength of the cognitive unit with the agent.
3. Deviations of the agent's action from person/role-based expectations.
4. The degree to which the event is desirable (for Gratification) or undesirable (for Remorse).

As with the self-directed attribution emotions, cognitive unit strength is a factor here.

Other-Directed Compound Emotions

Other-directed compound emotions result from other-directed attribution emotions in combination with event-based emotions, and are defined thus:

Gratitude: approving of someone else's praiseworthy action and being pleased about the related desirable event; e.g. gratitude, appreciation, feeling indebted, thankful, etc.

Anger: disapproving of someone else's blameworthy action and being displeased about the related undesirable event; e.g. anger, annoyance, exasperation, fury, incensed, indignation, irritation, livid, offended, outrage, rage, etc.

In this case, there are only three relevant intensity variables, as the strength of the cognitive unit with the appraising agent is irrelevant:

1. The degree of judged praiseworthiness (for Gratitude) or blameworthiness (for Anger).

2. Deviations of the agent's action from person/role-based expectations.

3. The degree to which the event is desirable (for Gratitude) or undesirable (for Anger).

4.4 Logical Formalisation of the OCC Theory

There is significant ambiguity inherent in any psychological theory, even one as well-defined as the OCC theory. The informal descriptions need to be formalised logically if they are to be implemented computationally. Adam, Herzig, and Longin [8] provide such a formalisation, using modal logic. They handle 20 of the 22 emotions described by the OCC theory, excluding only the object-based emotions, Love and Hate. In this section, we present the key definitions from their formalism, and explain the differences between their logic and our implementation. We begin with an overview of Adam et al.'s notation in Section 4.4.1, before moving on to the different categories of emotions: event-based emotions in Section 4.4.2, agent-based emotions in Section 4.4.3, object-based emotions in Section 4.4.4, and compound emotions in Section 4.4.5.

4.4.1 Notation

We will not delve too deeply into the logical semantics of Adam et al.'s formalism, but before presenting the emotion definitions we introduce some key aspects of their notation. In general, i, j, k... are agents, α, β, γ... are events, p, q... are atomic propositions, and $i{:}\alpha$ means agent i intentionally caused event α. The proposition φ is defined using BNF (Backus Naur Form) as:

$$\begin{aligned}\varphi ::= \bot \mid p \mid \neg\varphi \mid \varphi \vee \varphi \mid \mathit{Bel}_i\varphi \mid \mathit{Prob}_i\varphi \mid \mathit{Des}_i\varphi \\ \mid \mathit{Idl}\varphi \mid \mathit{After}_{i:\alpha}\varphi \mid \mathit{Before}_{i:\alpha}\varphi \mid G\varphi \mid H\varphi\end{aligned} \tag{4.1}$$

All logical symbols above have their normal meaning, and the remaining terms are defined as follows:

$\mathit{Bel}_i\varphi$: agent i believes φ is true.

$Prob_i\varphi$: agent i believes φ is more probable than $\neg\varphi$.

$Des_i\varphi$: φ is desirable for agent i.

$Idl\varphi$: ideally, φ is true.

$After_{i:\alpha}\varphi$: φ is true after i performs action α.

$Before_{i:\alpha}\varphi$: φ was true before i performed α.

$G\varphi$: henceforth, φ will be true.

$H\varphi$: φ has always been true in the past.

Adam et al. also define several useful abbreviations, to simplify their emotion definitions:

$Happens_{i:\alpha}\varphi \stackrel{def}{=} \neg After_{i:\alpha}\neg\varphi$: α is about to be performed by agent i, after which φ will be true.

$Done_{i:\alpha}\varphi \stackrel{def}{=} \neg Before_{i:\alpha}\neg\varphi$: agent i has just performed α, and φ was true before.

$F\varphi \stackrel{def}{=} \neg G\neg\varphi$: φ is true or will be true at some future instant.

$P\varphi \stackrel{def}{=} \neg H\neg\varphi$: φ is or was true.

$Idl_i\varphi \stackrel{def}{=} Bel_i Idl\varphi$: from the point of view of agent i, it is ideal that φ be true.

$Expect_i\varphi \stackrel{def}{=} Prob_i\varphi \wedge \neg Bel_i\varphi$: agent i expects φ to be true, but envisages that it could be false.

All emotion definitions in the following sections are presented using this notation. For detailed information about the semantics of the logic, proofs, and further derivations, refer to Adam et al.'s work [8].

4.4.2 Event-Based Emotions

The OCC theory describes event-based emotions as reactions to events. However, as explained in Section 4.3.1, the theory does allow for the same event to have both desirable and undesirable aspects. In order to formalise this notion, Adam et al. introduce consequences, which equate to the outcomes of events. Any given event can have multiple consequences, some of which may be desirable, others undesirable. However, any

particular consequence cannot be both desirable and undesirable at the same time. Applying this to the example from Section 4.3.1, the death of the beloved relative would have two consequences: the death of the relative (undesirable), and inheriting a great fortune (desirable). Thus, in Adam et al.'s formalism, event-based emotions are felt with respect to the consequences of events, rather than the events themselves. This allows a consistent logical formalisation of desires (i.e. $Des_i\varphi$ and $Des_i\neg\varphi$ never have to be true simultaneously), while still allowing mixed emotions (e.g. both Joy and Distress) to arise as a result of a single event. We present the logical definitions for the event-based emotions in the following sections.

Well-Being Emotions

Well-being emotions depend only on the desirability of a particular consequence for a particular agent. If an agent believes a desirable consequence is true, they feel Joy; if they believe an undesirable consequence is true, they feel Distress. These are defined logically as follows:

$$Joy_i\varphi \stackrel{def}{=} Bel_i\varphi \wedge Des_i\varphi \tag{4.2}$$

$$Distress_i\varphi \stackrel{def}{=} Bel_i\varphi \wedge Des_i\neg\varphi \tag{4.3}$$

Prospect-Based Emotions

Prospect-based emotions focus on the desirability of consequences that are not currently true (that is, the agent does not currently believe they are true), but are considered likely. If an agent considers a desirable consequence to be likely, they Hope that it is true; if they consider an undesirable consequence to be likely, they Fear that it is true. The definitions are as follows:

$$Hope_i\varphi \stackrel{def}{=} Expect_i\varphi \wedge Des_i\varphi \tag{4.4}$$

$$Fear_i\varphi \stackrel{def}{=} Expect_i\varphi \wedge Des_i\neg\varphi \tag{4.5}$$

It is worth noting that consequences which are hoped for or feared do not necessarily have to relate to the future. These emotions can arise in relation to the present or the past, provided that the agent experiencing the emotions is uncertain about the truth value of the relevant consequences (but considers them probable).

Additional emotions arise if these consequences are either confirmed or disconfirmed. If the prospect of a desirable event is confirmed, an agent will feel Satisfaction; in the case of an undesirable event, they will feel FearsConfirmed:[2]

$$Satisfaction_i\varphi \stackrel{def}{=} Bel_i PExpect_i\varphi \wedge Des_i\varphi \wedge Bel_i\varphi \tag{4.6}$$

$$FearsConfirmed_i\varphi \stackrel{def}{=} Bel_i PExpect_i\varphi \wedge Des_i\neg\varphi \wedge Bel_i\varphi \tag{4.7}$$

When desirable and undesirable events are disconfirmed, agents feel Relief and Disappointment respectively:

$$Relief_i\varphi \stackrel{def}{=} Bel_i PExpect_i\neg\varphi \wedge Des_i\varphi \wedge Bel_i\varphi \tag{4.8}$$

$$Disappointment_i\varphi \stackrel{def}{=} Bel_i PExpect_i\neg\varphi \wedge Des_i\neg\varphi \wedge Bel_i\varphi \tag{4.9}$$

Although Adam et al. present the above definitions in terms of their primitives, we find it useful to note that Equations 4.6–4.9 can be rewritten as below, by substituting Joy and Distress for their respective definitions from Equations 4.2 and 4.3:

$$Satisfaction_i\varphi \stackrel{def}{=} Bel_i PExpect_i\varphi \wedge Joy_i\varphi \tag{4.10}$$

$$FearsConfirmed_i\varphi \stackrel{def}{=} Bel_i PExpect_i\varphi \wedge Distress_i\varphi \tag{4.11}$$

$$Relief_i\varphi \stackrel{def}{=} Bel_i PExpect_i\neg\varphi \wedge Joy_i\varphi \tag{4.12}$$

$$Disappointment_i\varphi \stackrel{def}{=} Bel_i PExpect_i\neg\varphi \wedge Distress_i\varphi \tag{4.13}$$

Equations 4.10–4.13 will be the form used in our implementation (refer to Section 4.5.6 for details).

Fortunes-of-Others Emotions

Fortunes-of-others emotions are based on how desirable an outcome is presumed to be for another agent. To borrow the example presented by Ortony et al., if Fred believes Mary has won a thousand dollars, and also believes that Mary desires to have that money, then provided Fred likes Mary, he will feel happy for her. On the other hand, if Fred dislikes Mary, he would be more likely to feel resentment about her good fortune. However, because Adam et al. do not model the object-based emotions (Love and Hate),

[2]In Adam et al.'s work, they call this emotion FearConfirmed. However, the OCC theory calls it fears-confirmed. We stay faithful to the naming in the original theory, and use FearsConfirmed.

they have no direct way to represent whether two agents like one another. Instead, they define what they call "non-logical axioms" [8] to handle this. To explain the idea with reference to the example above, if Fred likes Mary and believes that Mary desires a thousand dollars, then that means Fred desires that Mary believes she has a thousand dollars. This is captured using the following definitions:

$$HappyFor_{i,j}\varphi \stackrel{def}{=} Bel_i\varphi \wedge Bel_i Des_j\varphi \wedge Des_i Bel_j\varphi \tag{4.14}$$

$$Pity_{i,j}\varphi \stackrel{def}{=} Bel_i\varphi \wedge Bel_i Des_j\neg\varphi \wedge Des_i\neg Bel_j\varphi \tag{4.15}$$

$$Resentment_{i,j}\varphi \stackrel{def}{=} Bel_i\varphi \wedge Bel_i Des_j\varphi \wedge Des_i\neg Bel_j\varphi \tag{4.16}$$

$$Gloating_{i,j}\varphi \stackrel{def}{=} Bel_i\varphi \wedge Bel_i Des_j\neg\varphi \wedge Des_i Bel_j\varphi \tag{4.17}$$

In our case, because we do implement Love and Hate, we are able to use them to represent the idea of liking and disliking which Adam et al. describe. As such, we do not implement Equations 4.14–4.17 as written above, but replace the non-logical axioms with the appropriate Love or Hate relationship. In doing so, we assume that an agent must Love another agent in order to feel HappyFor towards them, but the requirement for Pity is weaker; an agent will feel Pity provided they do not Hate the other agent. We make this distinction because it is easy to imagine circumstances where people pity others they do not even know (for example, the victims of natural disasters in other countries). As such, a Love relationship cannot be a prerequisite for this emotion. However, people do not generally feel happy for others unless they care about what happens to them, which requires a positive relationship (i.e. Love[3]). The definitions corresponding to our implementation are as follows:

$$HappyFor_{i,j}\varphi \stackrel{def}{=} Bel_i\varphi \wedge Bel_i Des_j\varphi \wedge Love_{i,j} \tag{4.18}$$

$$Pity_{i,j}\varphi \stackrel{def}{=} Bel_i\varphi \wedge Bel_i Des_j\neg\varphi \wedge \neg Hate_{i,j} \tag{4.19}$$

$$Resentment_{i,j}\varphi \stackrel{def}{=} Bel_i\varphi \wedge Bel_i Des_j\varphi \wedge Hate_{i,j} \tag{4.20}$$

$$Gloating_{i,j}\varphi \stackrel{def}{=} Bel_i\varphi \wedge Bel_i Des_j\neg\varphi \wedge Hate_{i,j} \tag{4.21}$$

[3]Although the word love often has strong connotations, recall that in the OCC theory this represents an emotion type, which includes many emotions, all of positive valence but varying intensity, such as like, adore, affection, etc. As we do not model emotion intensity, our use of Love takes into account all of these emotions.

4.4.3 Agent-Based Emotions

Agent-based emotions are felt towards agents with respect to actions they carry out, depending on whether those actions are judged to be praiseworthy or blameworthy. Adam et al. distinguish between an action, which is assumed to be intentional, and an event, which is not performed deliberately. Thus, agent-based emotions can only be felt about actions, not events. Whether an action is deemed praiseworthy or blameworthy depends on how it relates to the standards (or ideals) of the evaluating agent. These standards are expressed using the Idl_i operator presented in Section 4.4.1.

Self-Directed Attribution Emotions

Pride and Shame are the self-directed agent-based emotions, which an agent can feel about their own actions. Adam et al. define these emotions as follows:

$$\begin{aligned} &Pride_i(i{:}\alpha, \varphi) \stackrel{def}{=} \\ &\quad Bel_i Done_{i:\alpha}(Idl_i Happens_{i:\alpha}\varphi \wedge Prob_i After_{i:\alpha}\neg\varphi) \wedge Bel_i\varphi \qquad (4.22) \\ &Shame_i(i{:}\alpha, \varphi) \stackrel{def}{=} \\ &\quad Bel_i Done_{i:\alpha}(Idl_i \neg Happens_{i:\alpha}\varphi \wedge Prob_i After_{i:\alpha}\neg\varphi) \wedge Bel_i\varphi \qquad (4.23) \end{aligned}$$

These definitions are considerably more complicated than those for the event-based emotions, and thus we will explain their interpretation. In the definition of Pride, before action α is carried out, agent i considers it ideal to perform α resulting in the outcome φ, but at the same time believes it is probable that after α the result will actually be $\neg\varphi$. However, after α is carried out, i believes φ is true. In other words, in order for an agent to feel Pride, the outcome of their action must have been thought unlikely; this captures the notion of actions that are difficult, or not likely to succeed. The definition for Shame has the same interpretation, except in this instance the action is considered unideal.

The above definitions are based not only on an action's outcome, but also on the likelihood of that outcome when the action is performed (i.e. how difficult the action is). Thus, an agent would not feel Pride about an action that they considered ideal if the outcome was probable. Adam et al. also describe an example where an agent can feel Shame regardless of the outcome. In this instance, merely carrying out the action α is unideal, which can be expressed as $Idl_i Happens_{i:\alpha}\top$ (i.e. setting $\varphi = \top$ in Equation

4.23), and it is improbable for α to be performed by that particular agent due to their social role ($Prob_i After_{i:\alpha}\bot$). Comparing this to the OCC definitions, the theory does mention the "deviations of the agent's action from person/role-based expectations (i.e., unexpectedness)" [118] as a variable affecting the intensity of Pride and Shame, but does not say they are a prerequisite for these emotions to occur.

In our implementation, we leave expectedness out of the definition. We consider it to be a factor affecting intensity only, and, as will be discussed in Section 4.5, we do not implement the intensity variables described by the OCC theory. However, we make an important distinction between the definitions of Pride and Shame. In order to feel Pride, an agent's action must be successful. We choose to implement it this way because usually an agent will consider an action to be ideal because of the consequences of that action, for themselves or others. If the action fails, those consequences will not occur, and thus the agent would not be expected to feel Pride. In fact, they will actually feel Shame that they failed. However, an agent can feel Shame about an action they attempt whether they succeed or not. This is because, even if they fail, the intention to perform the action was there, which conflicts with the agent's standards. Ortony et al. identify a similar asymmetry between praiseworthy and blameworthy actions [118]:

> However, there is an interesting asymmetry vis à vis positive and negative emotions with respect to the role of intention. While praiseworthy acts clearly have to be considered to have been intended, blameworthy acts do not. Failing to do what is expected of one can be blameworthy even if the failure was unintentional.

The definitions we implement can be better characterised as:

$$Pride_i(i{:}\alpha, \varphi) \stackrel{def}{=} Bel_i Done_{i:\alpha} Idl_i Happens_{i:\alpha}\varphi \wedge Bel_i\varphi \tag{4.24}$$

$$\begin{aligned} Shame_i(i{:}\alpha, \varphi) \stackrel{def}{=} & Bel_i Done_{i:\alpha} Idl_i \neg Happens_{i:\alpha}\top \vee \\ & (Bel_i Done_{i:\alpha} Idl_i Happens_{i:\alpha}\varphi \wedge Bel_i\neg\varphi) \end{aligned} \tag{4.25}$$

In the case of Pride, the emotion will arise if an agent successfully performs a praiseworthy action. On the other hand, Shame will arise in one of two cases: if an agent attempts an action they consider unideal (regardless of outcome), or if the agent attempts an ideal action, but fails.

Other-Directed Attribution Emotions

The other-directed agent-based emotions, Admiration and Reproach, are analogous to Pride and Shame, but felt towards other agents rather than the self. Admiration occurs if agent i approves of agent j's praiseworthy action, and Reproach if i disapproves of j's blameworthy action. As with the self-directed agent-based emotions, Adam et al.'s definitions of Admiration and Reproach take into account the expectedness of the action and its outcome. Their definitions of these emotions are shown below (note these exactly match the definitions of Pride and Shame if $i = j$):

$$\begin{aligned} &Admiration_{i,j}(j{:}\alpha, \varphi) \stackrel{def}{=} \\ &\qquad Bel_i Done_{j:\alpha}(Idl_i Happens_{j:\alpha}\varphi \wedge Prob_i After_{j:\alpha}\neg\varphi) \wedge Bel_i\varphi \qquad (4.26) \\ &Reproach_{i,j}(j{:}\alpha, \varphi) \stackrel{def}{=} \\ &\qquad Bel_i Done_{j:\alpha}(Idl_i \neg Happens_{j:\alpha}\varphi \wedge Prob_i After_{j:\alpha}\neg\varphi) \wedge Bel_i\varphi \qquad (4.27) \end{aligned}$$

In our implementation, we handle Admiration and Reproach in a similar fashion to the way we handle Pride and Shame. i feels Admiration towards j if j performs an action i considers praiseworthy, and succeeds (i.e. the action has the intended outcome). i feels Reproach towards j if j performs an action i considers blameworthy, regardless of whether the action succeeds. Our definitions are shown below:

$$Admiration_{i,j}(j{:}\alpha, \varphi) \stackrel{def}{=} Bel_i Done_{j:\alpha} Idl_i Happens_{j:\alpha}\varphi \wedge Bel_i\varphi \qquad (4.28)$$

$$Reproach_{i,j}(j{:}\alpha, \varphi) \stackrel{def}{=} Bel_i Done_{j:\alpha} Idl_i Happens_{j:\alpha}\top \qquad (4.29)$$

Note, however, that in this case we do not have an alternative condition for Reproach as we did for Shame, resulting from a praiseworthy action that fails. This is because in the case of the self-directed emotions an agent is aware of the intended outcomes of their own actions. This cannot be assumed for a different agent's actions, so a praiseworthy action performed by another agent which fails results in neither Admiration nor Reproach.

4.4.4 Object-Based Emotions

Adam et al. do not handle the object-based emotions (Love and Hate), and thus do not provide logical definitions for them. However, we consider these emotions to be important for story generation, as they define the most basic relationships between

characters, which in turn play an important role in determining other emotions (in particular, the fortunes-of-others emotions). In light of this, our model does incorporate these emotions, albeit only directed towards other agents, not inanimate objects. We define Love and Hate based on Admiration and Reproach. If agent i feels Admiration towards agent j, they will feel Love towards j, provided that they do not feel Reproach towards j at the same time. Conversely, if i feels Reproach towards j (and no Admiration), they will feel Hate towards j. This can be represented logically as follows (α and β are different actions; φ and ψ are different outcomes of those actions):

$$Love_{i,j} \stackrel{def}{=} Admiration_{i,j}(j{:}\alpha, \varphi) \wedge \neg Reproach_{i,j}(j{:}\beta, \psi) \tag{4.30}$$

$$Hate_{i,j} \stackrel{def}{=} Reproach_{i,j}(j{:}\alpha, \varphi) \wedge \neg Admiration_{i,j}(j{:}\beta, \psi) \tag{4.31}$$

If both Admiration and Reproach are felt at the same time, neither Love nor Hate will arise.[4]

We acknowledge that this is a very simplistic view of these emotions. In reality, not every praiseworthy or blameworthy action will cause people to completely change their feelings about one another; this will depend, among other things, on the relative intensity of the different emotions. Not modelling intensity, and allowing the object-based emotions to fluctuate instantaneously between two extremes, will undoubtedly produce very simplistic characters, whose relationships will be somewhat childlike in their changeability. However, this is sufficient for the simple stories we aim to model. In our view, modelling Love and Hate, even in a basic way, is valuable because it ensures characters' emotions are consistent with the way their relationships evolve during the course of a story, even if those relationships are volatile.

4.4.5 Compound Emotions

Compound emotions are those which are simultaneously event and agent-based. They are directed towards an agent, with respect to a consequence of an action that agent performed. Compound emotions are defined as specific pairings of event-based emotions

[4]In our implementation, we assume that only one event can happen at a time (refer to Section 4.5.3). As such, it is impossible for an agent to feel Admiration and Reproach simultaneously, because they cannot judge the same action to be both ideal and unideal. In light of this, it would be sufficient to use simpler definitions: $Love_{i,j} \stackrel{def}{=} Admiration_{i,j}(j{:}\alpha, \varphi)$ and $Hate_{i,j} \stackrel{def}{=} Reproach_{i,j}(j{:}\alpha, \varphi)$. However, we nonetheless implement these emotions as defined in Equations 4.30 and 4.31, to facilitate future extension of the system to allow simultaneous events.

with agent-based emotions, as follows:

$$Gratification_i(i{:}\alpha, \varphi) \stackrel{def}{=} Pride_i(i{:}\alpha, \varphi) \wedge Joy_i\varphi \tag{4.32}$$

$$Remorse_i(i{:}\alpha, \varphi) \stackrel{def}{=} Shame_i(i{:}\alpha, \varphi) \wedge Distress_i\varphi \tag{4.33}$$

$$Gratitude_{i,j}(j{:}\alpha, \varphi) \stackrel{def}{=} Admiration_{i,j}(j{:}\alpha, \varphi) \wedge Joy_i\varphi \tag{4.34}$$

$$Anger_{i,j}(j{:}\alpha, \varphi) \stackrel{def}{=} Reproach_{i,j}(j{:}\alpha, \varphi) \wedge Distress_i\varphi \tag{4.35}$$

Thus, for example, if an agent feels Pride about performing an action, and also feels Joy as a result of one of its consequences, the agent will feel Gratification. Gratification and Remorse are directed towards the self; Gratitude and Anger are directed towards other agents.

4.5 Implementing the OCC Theory

Having provided formal definitions of the OCC emotions in Section 4.4, here we move on to implementation, with the goal of modelling character emotions in stories, and ultimately using them to control story trajectories. In Section 4.5.1, we introduce the technologies used for our implementation. Section 4.5.2 goes on to explain the simplifications we made with respect to the logical descriptions in the previous section, to facilitate implementation. We outline the key assumptions of our model in Section 4.5.3, and explain the underlying action and belief frameworks in Sections 4.5.4 and 4.5.5 respectively. Finally, Section 4.5.6 presents ASP rules for each of the OCC emotions.

4.5.1 Implementation Technologies

As identified in Chapter 2, there are a variety of implementation technologies available which can be applied to story generation. They range from classical planning, to logical formalisms (e.g. the situation and event calculi), and mathematically founded techniques such as constraint satisfaction and answer set programming. Each has its benefits and drawbacks, and the appropriate choice will depend very much on the nature of the system being developed.

In preliminary work [145], we built a basic action and emotion model using the *Discrete Event Calculus Reasoner* (*decreasoner*) [111], a system which implements the discrete form of the event calculus [110]. This implementation was not directly based

on Adam et al.'s logical framework; rather than modelling desires and ideals, we allocated consequences to emotion classes based on the most basic emotions, Joy and Distress. The prospect-based emotions (Hope, Fear, Satisfaction, FearsConfirmed, Relief, and Disappointment) were excluded. In testing the system, we encoded a selection of Aesop's fables, and used the emotion model to generate all the characters' emotions. The main issue we encountered with *decreasoner* was that programs quickly became intractable as the complexity of the storytelling domains increased. Although fables are very simple stories, we were sometimes unable to generate solutions without partitioning them into shorter segments and running them through the system in parts, to reduce the number of time-steps and other variables for grounding. Even fables with only two characters and three or four time-steps could take hours to terminate, which is not promising for generating stories in a reasonable time-frame. This prompted us to look for an alternative approach; we chose to investigate answer set programming.

Answer set programming (ASP) is an approach to declarative problem-solving based on stable model semantics [62]. Lifschitz sums up the premise behind this approach [91]:

> The idea of answer set programming is to represent a given computational problem by a program whose answer sets correspond to solutions, and then use an answer set solver to find a solution.

An answer set (or stable model) is a minimal set of atoms that satisfies a particular formula [53], or, in the case of a logic program, its set of rules. The problem description is written in a Prolog-like syntax, and there are a number of solvers readily available [60, 64, 88, 92, 161]. Improvements in the efficiency of solvers have made ASP a very useful technique for complex problem-solving domains.

In our case, we found two main benefits of ASP as compared with our original *decreasoner* program. Most importantly, there were significant runtime improvements, making the resulting story generation system far more usable. In addition, it was easier to implement the logical definitions of the OCC emotions (described in Section 4.4) using ASP syntax, compared to translating them into the event calculus. Our discussion in this chapter will focus on the ASP implementation, as this was the version used for story generation.

4.5.2 Restricting the Logic

The drawback of logical models is that although they can provide an unambiguous and comprehensive description of a theory, they are not always straightforward to implement, or computationally tractable. We place some important restrictions on the interpretation of the logical definitions provided in Section 4.4, to facilitate implementation.

Limiting the Language

One of our most important simplifications is to limit the scope of the proposition φ, which was defined in Equation 4.1. Adam et al.'s definition allows for infinite nesting of beliefs, desires, ideals, and so on, which places significant demands on any computational implementation. To avoid this issue, we limit φ by disallowing recursive nesting. More specifically, in our implementation φ, as used in $Bel_i\varphi$, $Prob_i\varphi$, and $Des_i\varphi$, represents only fluents or the consequences of events (as will be described in Section 4.5.4). In the context of $Idl_i\varphi$, φ represents events (or their negations). In order to handle beliefs about other agents' desires without a recursive definition, we introduce a separate `bdes` predicate in our implementation, which directly represents beliefs about desires (refer to Section 4.5.5). This restriction also means disjunctions cannot appear in *Happens*, which is how Adam et al. represent non-deterministic events (e.g. $Happens_{i_1:tossCoin}(heads \vee tails)$). We handle non-determinism by introducing a `succeeded` predicate, which will be described in Section 4.5.4.

Limiting the Scope of Prospect-Based Emotions

In both the OCC theory and Adam et al.'s logic, the prospect-based emotions, Hope and Fear, can be felt not only with respect to things that may happen in the future, but also about the past, and even the present in cases where the state is unknown (Adam et al. provide the example of hoping that an email has been delivered to its addressee). This introduces considerable complexity; an agent could Hope (or Fear) for a particular outcome to be true in the present, the future, or the past. In our implementation, any Hope or Fear experienced by an agent in the present is always with reference to an outcome that may or may not be true in the future. In fact, due to our omniscience assumption for agents, which will be described in Section 4.5.3, it is impossible for an agent to feel Hope or Fear about something in the present or the past.

The definitions of Hope and Fear require uncertainty, whereas agents in our system have perfect knowledge about the true state of the world.

4.5.3 Key Assumptions

In this section, we describe the key assumptions made in our ASP implementation of the OCC theory. Some of these assumptions are already inherent in the OCC theory itself, or in Adam et al.'s logic, in which case we identify them as such. Others are assumptions we introduce to simplify our implementation.

Emotions Are Momentary

In general, emotions that arise in our model endure only for a single time-step; that is, they are momentary, not persistent. This is consistent with Adam et al.'s assertion that "an affective state having a long duration is not so much emotion as mood." [8] We make no attempt to model mood. The only exceptions are the object-based emotions, Love and Hate. These emotions express relationships between agents, which are crucial in defining the fortunes-of-others emotions: HappyFor, Pity, Resentment, and Gloating. To avoid erratic fluctuations in these emotions, Love and Hate persist until something occurs to change them; i.e. an action that results in Admiration or Reproach, which form the basis of our definitions of Love and Hate (refer to Equations 4.30 and 4.31).

Events Are Instantaneous and Their Consequences Are Immediate

All events and actions are instantaneous; they take one time-point to execute. If an event happens at time T, its consequences will become true at T+1 (i.e. the following time-point). All emotions (agent, event, and object-based) which arise as a result of that event will also hold at T+1. As explained in the preceding section, apart from Love and Hate, emotions persist only for a single time-step, so they will cease to hold at T+2.

No Emotion Intensity

In real life, emotions can arise in various intensities. The OCC theory defines three central intensity variables which determine the intensity of particular emotions: desirability, praiseworthiness, and appealingness. Although there may be other factors

involved depending on the specific emotion, these three variables are the most important, because they apply to all emotions in the relevant category. That is, desirability influences the intensity of all event-based emotions, praiseworthiness influences the intensity of all agent-based emotions, and appealingness influences the intensity of all object-based emotions. The OCC theory provides an in-depth discussion of these and many other variables affecting emotion intensity.

In our implementation, which deals with very simple stories, we consider it sufficient to model absolutes rather than degrees of emotion. For example, at any given timepoint an agent can feel Joy or not feel Joy, but there are no varying degrees of Joy. As such, we do not model numeric intensity variables, and only consider emotions in a discrete sense. A particular outcome can be either desirable or undesirable (or neither, in which case no emotions will arise).

No Simultaneous Events

Adam et al.'s logic "does not impede the parallel execution of several actions." [8] However, in our implementation, we only permit one event to happen at a time; that is, we disallow simultaneous events. This is sufficient for modelling simple stories such as fables, in which events occur in sequence. It also avoids the need for encoding additional restrictions about whether agents can be involved in two events simultaneously, which would need to be handled case-by-case depending on the specific nature of the events.

Omniscient Agents

As identified in Section 3.6.2, belief management is a significant problem in artificial intelligence, which we consider to be outside the scope of this work. To keep our belief model simple, we assume all agents have perfect knowledge about what is true in the world. More importantly, agents have correct beliefs about not only their own but also other agents' desires, which is important for triggering the fortunes-of-others emotions (defined in Section 4.4.2).

However, in certain stories it does not always make sense for all agents to be aware of everything. For example, if a story introduces a new character halfway through, it does not make sense for that character to experience emotions about the events which take place before they appear in the story. Similarly, if a character is in a different location, it does not always make sense for them to be aware of events occurring elsewhere. To handle this, we model location. To avoid a significant explosion in grounding from

having a large number of locations, we only model two: onstage and offstage. If a character is located onstage, they are aware of all events and consequences taking place in the story. Conversely, if the character is offstage, they are not aware of the events or consequences occurring at that time-point. In fact, they do not have beliefs at all, and thus will not experience any emotions. The details of how we handle location are discussed in Section 4.5.4.

Actions Are Intentional, Events Are Not

Intentionality plays an important role in agent-based emotions (Pride, Shame, Admiration, and Reproach). These emotions require attribution of responsibility for an action. However, not all events are intentional, or even performed by agents at all. For example, accidents are actions which are not intentional, whereas naturally occurring events, such as a storm, cannot be blamed on any agent. One would not expect agent-based emotions to arise in these cases. As such, we make the same distinction between actions and events as Adam et al. [8]:

> We highlight here that what we call action is assumed to be intentional, that is the agent always intend [*sic*] to perform actions that he is about to perform. This is the difference between actions and events. Thus if an agents [*sic*] does something unintentionally (like sneezing) it is an event, and it can only trigger event-based emotions.

All actions are assumed to be intentional, and agent-based emotions can only arise with respect to actions, not events. All actions are events, but not all events are necessarily actions.

Desires Change, Ideals Do Not

All the emotions defined in Section 4.4 rely on agents' desires or ideals (or both). In principle, an agent's desires and ideals could change over time (for example, as they mature, or their interests change, etc.). In their definitions, Adam et al. assume desires are fixed: "For the sake of simplicity, we make the hypothesis that preferences are stable: what is desirable for an agent persists." [8] However, in our model we found it useful to allow desires to change under certain circumstances:

1. Love or Hate relationships between agents can affect the desirability of certain outcomes. For example, if Bob loves Alice, one would expect Bob to feel Distress

if Alice dies (i.e. Bob desires Alice to be alive). Conversely, if Bob hates Alice, he might feel Joy about her death (i.e. he desires Alice not to be alive). If there is no object-based emotion between Bob and Alice, Bob would not be expected to react emotionally to Alice's death. Love and Hate between agents can vary during the course of a story. If all desires remain fixed, this can lead to emotional reactions that do not make sense. For example, say Bob hates Alice at the beginning of a story, and wants her to die. Later in the story, Bob's feelings change, and he begins to love Alice. If his desires do not also change to reflect this, Bob will feel Joy if Alice dies, even though he loves her. To avoid this inconsistency, desires which depend on how agents feel towards one another must be able to vary.

2. Fixed desires can cause inappropriate emotions to arise from intentional actions. For example, if an agent desires money, they should feel Joy if they receive money, and Distress if they lose money. However, if all desires were fixed, that agent would feel Distress if they intentionally gave money to another agent (e.g. as a present), which does not make sense in that context. The action of giving therefore needs to modify an agent's desire for the item in question for the following time-step.

Thus, while some desires remain fixed through time, others are permitted to vary. We define domain-specific rules governing when changes occur with respect to actions or Love/Hate relationships.

As with desires, Adam et al. treat ideals as fixed: "As we do not deal with the dynamics of ideals, it is quite reasonable to consider that ideals are stable, at least for a given time interval." [8] In this case, we adopt their approach. For any given character, their ideals are pre-defined and fixed across time (in fact, our `idl` predicate, presented in Section 4.5.5, does not have a time parameter). This could certainly be extended in the future to create more complex characters, but for the simple stories we aim to generate it is sufficient.

Closed World Assumption

In our implementation, we make the closed world assumption. We assume all fluents are defined at all times (i.e. for every fluent, either it or its negation must be true at every time-point). To achieve this, we provide a rule which generates the negation of any fluent which does not have a value specified in the domain description for the initial state. For all ensuing time-points, fluent values persist unless they are explicitly

changed by an event. Events can only reverse fluent values (i.e. between true and false), and thus fluents are never undefined.

4.5.4 The Action Layer

Before presenting the ASP emotion rules used in our implementation, we introduce the underlying action and belief frameworks which we use to model stories. We separate the Action Layer, which defines the physical aspects of the story world, from the Belief Layer, which models agents' mental structures. In this section we describe the Action Layer, presenting key predicates, along with an explanation of their use and justification of specific design choices. We do not provide all ASP rules and constraints comprising the Action Layer for any particular domain here, but focus on those that are important for understanding how our model functions. The complete source code for all three story worlds we implement, annotated with explanatory comments, is available online.[5]

Time

The events in a story form a temporal sequence. Thus, any system aiming to model stories needs to model time. We represent time as a sequence of integers. In ASP, time must be declared as a fixed range in the domain description, so that it can be grounded. Listing 4.1 shows an example time declaration which allows stories of length 3; i.e. events can occur at $T = 0$, $T = 1$, and $T = 2$. To generate stories of a different length, the time declaration must be modified.

```
1 time(0..2).
```

Listing 4.1: ASP time declaration

It is worth noting that the outcomes of events, and the associated emotions, occur in the time-step following the inciting event. Thus, if an event occurs at time T, its consequences, and the relevant emotions, will hold at $T + 1$. The corollary of this is that consequences and emotions may arise one time-step later than the last allowable event. With reference to the time declaration in Listing 4.1, although no events can occur after $T = 2$, consequences and emotions can arise at $T = 3$. Agents must also be able to have beliefs in the time-point following the last allowable event, in order to perceive

[5]The complete ASP source code is available at: `www.cse.unsw.edu.au/~msarlej/moss/system/storyplanner`.

its consequences, and experience the associated emotions. To handle this, our belief and emotion rules (which will be described in Sections 4.5.5 and 4.5.6 respectively) are defined with reference to $T+1$, where T is a valid time value.

Agents and Objects

All agents and objects that can appear in a story must be declared. Listing 4.2 shows examples of ASP code for declaring agents, objects, and their properties. Agents can possess objects. However, we do not track quantity, and thus an agent can only possess one of any given object at a time.

```
1  % Declaring an agent
2  agent(wizard).
3
4  % Declaring an object
5  object(sword).
6
7  % Declaring that all agents are also objects
8  object(Agent) :- agent(Agent).
9
10 % Declaring that all agents are edible
11 edible(Object) :- agent(Object).
```

Listing 4.2: ASP declarations of agents, objects, and their properties

Fluents

Fluents are properties of the world which can be true or false. Their values can be modified by events that take place, and collectively they represent the current state of the story world. Fluent values for the initial state can be specified in the domain description. As mentioned in Section 4.5.3, we make the closed world assumption, and as such any fluent not declared to be true in the initial state has its negation set to true. A fluent and its negation cannot be true at the same time. Fluent values persist unless changed by an event; Listing 4.3 shows the frame axiom used to achieve this.

```
% Frame axiom for fluents
true(Fluent,T+1) :-
    negate(Fluent,Negation),
    time(T),
    true(Fluent,T),
    not true(Negation,T+1).
```

Listing 4.3: Frame axiom for fluents

Location

As mentioned briefly in Section 4.5.3, we model agents' locations. Location is treated as a fluent. Our purpose for modelling location is to handle characters that only feature in part of a story, and thus should not experience emotions during the remainder of the story. To achieve this, it suffices to model two locations: onstage and offstage. We require that every agent is in exactly one location at every time-point; the ASP rule which enforces this condition is shown in Listing 4.4. It is usually also important for an agent to remain onstage for the time-point immediately following an event they are involved in, so they can experience the relevant emotions. We achieve this using domain-specific rules for such events. Our model allows agents' locations to change instantaneously between time-points. An alternative approach would be to implement a move action, and treat changing location as an event. However, moving in itself generally does not illicit emotions, so such an extension would not provide benefits in our case.

```
% Each agent must be in exactly one location at every timepoint.
1 { true(located(A,L),T) : location(L) } 1 :-
    agent(A),
    time(T).
```

Listing 4.4: Every agent must be in exactly one location at every time-point

Events and Actions

Events and actions need to be defined for each story domain. Listing 4.5 provides an example of an action definition for stealing an object from another agent. We use this example to explain the structure of events and actions in our model.

```
1  % Declaring an action for stealing an object
2  action(steals_from(Target,Object)) :-
3      object(Object),
4      agent(Target).
5
6  % Stealing an object (if successful) causes the thief to have it
7  causes(Agent,steals_from(Target,Object),has(Agent,Object),T) :-
8      agent(Agent),
9      agent(Target),
10     object(Object),
11     time(T).
12
13 % Stealing an object causes the target not to have it
14 causes(Agent,steals_from(Target,Obj),neg(has(Target,Obj)),T) :-
15     agent(Agent),
16     agent(Target),
17     object(Obj),
18     time(T).
19
20 % Preconditions for stealing to be possible
21 possible(Agent,steals_from(Target,Object),T) :-
22     agent(Agent),
23     agent(Target),
24     object(Object),
25     time(T),
26     Agent != Target,
27     Agent != Object,
28     true(has(Target,Object),T),
29     not true(has(Agent,Object),T),
30     true(located(Target,onstage),T).
```

Listing 4.5: Example ASP action definition: steals_from

Declaration: Actions must be declared in terms of predicates that can be grounded. In this case, lines 2–4 declare the `steals_from(Target,Object)` action, where `Object` is the object being stolen, and `Target` is the agent it is stolen from.

Effects: We use the predicate `causes(Agent,Action,Fluent)` to define each action's effects on the world. In this example, the action (if successful) has two effects: the agent performing the action will have the object (lines 7–11), and the agent the object is stolen from will no longer have the object (lines 14–18). If the action does not succeed, these fluents will not become true. It is also possible for a failed

action to have different outcomes. Where applicable, we represent this using the analogous predicate, `causesf(Agent,Action,Fluent)`.

Preconditions: The predicate `possible(Agent,Action,T)` defines the preconditions for an action. This predicate must hold for the action to be executable. In this case, an agent cannot steal from themselves (line 26) or steal themselves (line 27), the agent they are stealing from must have the object (line 28), the thief must not already have the object (line 29),[6] and the agent they are stealing from must be onstage (line 30).[7] In domains where agents can be killed, we add a precondition that the agent must be alive to perform an action.

All actions are also events, and thus the rules for managing actions are defined in terms of the `event` type. Listing 4.6 shows the most important rules for managing events in our model. We explain each below.

Onstage requirement: An agent cannot cause an event (or action) if they are not located onstage (lines 2–6).

Non-determinism for events: We model non-determinism for events using the rule on lines 9–10. At every time-point, zero or one `succeeds(Agent,Event,T)` predicates can hold. If `succeeds` holds for a particular event, that event will succeed if it happens at that time-point; if not, the event will fail.

No simultaneous events: The rule on lines 13–14 enforces that exactly one event must occur at each time-point.

Enforcing preconditions: An action cannot occur if its prerequisites are not satisfied (i.e. if the corresponding `possible(Agent,Event,T)` predicate does not hold). This is enforced using the constraint on lines 17–18.

Effects on fluents: The rule on lines 21–28 updates fluent values based on events that occur. If an event causes a fluent (line 27), and the event takes place at time T (line 26) and succeeds (line 28), the fluent will become true at $T+1$ (line 21).

[6]This is a requirement because in our storytelling domains agents can only possess one of any particular object at a time. This restriction could be removed if agents were permitted to possess more than one of the same item.

[7]The agent performing the action also needs to be onstage, but this requirement is enforced elsewhere, as shown in Listing 4.6.

```
1  % For an agent to cause an event, they must be onstage
2  :- happens(Agent,Event,T),
3      not true(located(Agent,onstage),T),
4      agent(Agent),
5      event(Event),
6      time(T).
7
8  % 'succeeds' can hold or not for every T
9  0 { succeeds(Agent,Event,T) : agent(Agent) : event(Event) } 1 :-
10     time(T).
11
12 % Exactly one event happens at each timepoint
13 1 { happens(Agent,Event,T) : agent(Agent) : event(Event) } 1 :-
14     time(T).
15
16 % For an action to happen, it must be possible
17 :- happens(Agent,Event,T),
18     not possible(Agent,Event,T).
19
20 % Fluent caused by an event becomes true if the event succeeds
21 true(Fluent,T+1) :-
22     time(T),
23     negate(Fluent,Negation),
24     agent(Instigator),
25     event(Event),
26     happens(Instigator,Event,T),
27     causes(Instigator,Event,Fluent,T),
28     succeeds(Instigator,Event,T).
```

Listing 4.6: ASP rules for handling events

Consequences

In our model, events cause fluents. It may seem reasonable, then, for event-based emotions to be experienced with respect to fluents. If Bob gives Alice some flowers at $T=0$, one would expect Alice to feel Joy about having the flowers at $T=1$. However, fluents (such as having flowers) persist; Alice will still have the flowers at $T=2$, $T=3$, and so on. If emotions are defined with respect to fluents, Alice will continue to feel Joy as long as she has the flowers, which conflicts with the idea of emotions as momentary (refer to Section 4.5.3). What is important for causing the emotion is not the fluent itself, but rather the change in value of the fluent (i.e. going from not having flowers to having flowers). Thus, the consequences of events are not actually fluents, but changes in

value of fluents. To model this, we define a `consequence` predicate, which represents the change in value of a fluent. The basic structure of a consequence is `becametrue(Fluent)`. If this predicate holds at time T (i.e. `true(becametrue(Fluent),T)`) it means `Fluent` changed in value to true at T (i.e. it was false at T − 1).

However, this construct alone is insufficient for defining certain emotions. To illustrate why, consider Relief and Disappointment, which arise because an expected outcome did not occur. While it is straightforward to model things that happen, it is harder to model things that do not happen. For example, if an agent hopes for a particular outcome, Satisfaction can be defined based on when the desired outcome becomes true in the world. Defining Disappointment, on the other hand, requires explicitly identifying when that outcome fails to occur; this does not correspond to a change in value of a fluent. To overcome this hurdle, we define two different types of consequences. The first represents changes in fluent values resulting from events, and corresponds to the `becametrue(Fluent)` predicate described above. The second type of consequence represents fluent values that failed to change, but would have if the preceding event had succeeded. In this case, the structure is `failedtobecometrue(Fluent)`.

Listing 4.7 shows how both types of consequences are defined. We also define a relationship between `becametrue(Fluent)` and `failedtobecometrue(Fluent)` (lines 28–29). This is important for emotions like Relief and Disappointment. If an agent feels Hope for some consequence `becametrue(c1)`, the failure of that consequence to occur is represented by `failedtobecometrue(c1)`. Thus, to generate Disappointment, our model must recognise that `becametrue(c1)` and `failedtobecometrue(c1)` refer to the same consequence, but distinguish success and failure.

```
1  % Defining a consequence as a change in value of a fluent
2  % A fluent becoming TRUE
3  true(becametrue(Fluent),T+1) :-
4      true(neg(Fluent),T),
5      true(Fluent,T+1),
6      fluent(Fluent),
7      time(T).
8  % A fluent becoming FALSE
9  true(becametrue(neg(Fluent)),T+1) :-
10     true(Fluent,T),
11     true(neg(Fluent),T+1),
12     fluent(Fluent),
13     time(T).
14
15 % Defining failed consequences
16 true(failedtobecometrue(Fluent),T+1) :-
17     time(T),
18     negate(Fluent,Negation),
19     agent(Instigator),
20     event(Event),
21     happens(Instigator,Event,T),
22     causes(Instigator,Event,Fluent,T),
23     not succeeds(Instigator,Event,T),
24     not true(Fluent,T).
25
26 % becametrue(F) and failedtobecometrue(F) relate to the same
27 % consequence, just distinguish success and faliure
28 reverse_consequence(becametrue(F),failedtobecometrue(F)) :-
29     negate(F,Negation).
```

Listing 4.7: Defining consequences

4.5.5 The Belief Layer

In essence, the Belief Layer models character. In this layer, we define agents' beliefs, desires, ideals, and expectations, with reference to the available events, objects, and fluents defined in the Action Layer (Section 4.5.4). As with our discussion of the Action Layer, we do not present the complete set of ASP rules comprising the Belief Layer in this section, but focus on a representative selection which elucidates our approach.

Beliefs

As discussed in Section 4.5.3, all agents in our model are omniscient, in the sense that their beliefs about what is true in the world (including other agents' desires) are always correct. This allows for very simple rules governing beliefs. Beliefs about consequences are entailed directly from the consequences that are true in the world at any given time-point, as shown on lines 3–8 of Listing 4.8. It is worth noting the requirement that an agent must be onstage in order to have beliefs (line 8); this is to avoid agents who are not currently part of the story experiencing emotions. We also define constraints to ensure an agent cannot believe a consequence and its negation at the same time.

In addition to beliefs about consequences, we also need to model beliefs about fluents, because they are required to define `expect` (refer to *Expectations* later in this section). These beliefs are handled the same way; they are defined directly based on which fluents are true at any given time-point, as shown on lines 11–16 of Listing 4.8. Beliefs about other agents' desires are represented using a separate predicate, `bdes(Agent1,Agent2,Consequence,T)`, defined on lines 19–25. This is interpreted as: `Agent1` believes `Agent2` desires `Consequence` at time `T`. It is derived directly from `Agent2`'s extant desires (line 25).

For domains in which agents can die, an additional requirement that the agent must be alive is included in the rules. However, we permit agents to have beliefs in the time-point immediately following the event that caused their death. This is so that the agent can also experience emotions at that time-point: specifically, Distress about dying. This is required so that when we develop rules for morals based on patters of emotions (refer to Chapter 5), the solver is able to recognise that death causes Distress.

```
1  % Agents' beliefs correctly reflect what is true in the world
2  % (agents must be onstage to have beliefs)
3  bel(Agent,Consequence,T+1) :-
4      agent(Agent),
5      consequence(Consequence),
6      time(T),
7      true(Consequence,T+1),
8      true(located(Agent,onstage),T+1).
9
10 % Agents' beliefs w.r.t. fluents (to define 'expect')
11 bel(Agent,Fluent,T+1) :-
12     agent(Agent),
13     negate(Fluent,Negation),
14     time(T),
15     true(Fluent,T+1),
16     true(located(Agent,onstage),T+1).
17
18 % Agents have correct beliefs about each others' desires
19 bdes(Agent1,Agent2,Consequence,T+1) :-
20     agent(Agent1),
21     agent(Agent2),
22     consequence(Consequence),
23     time(T),
24     Agent1 != Agent2,
25     des(Agent2,Consequence,T+1).
```

Listing 4.8: Implementing beliefs

Ideals

In their discussion of agents' ideals (or standards) Adam et al. adopt two different notations:

1. $Idl\varphi$, which denotes generally accepted social or moral rules.

2. $Idl_i\varphi$, which refers to standards agent i has internalised.

We agree it is useful to distinguish between these two kinds of ideals. For instance, agents with different character traits may internalise different subsets of the general social rules. As such, we define both types in our implementation. The example provided in Listing 4.9 uses the idea of alignment from the popular table-top game *Dungeons and Dragons* [42], in which characters are categorised along two axes: ethical (lawful,

neutral, or chaotic) and moral (good, neutral, or evil). The ideals individual agents adopt from the generic social ideals depend on their specific alignment. Generic social ideals take the form `idl(Action)`, with no `Agent` parameter. Listing 4.9 provides an example on lines 2–3: killing agents is unideal (an unideal action is modelled in terms of its negation being ideal). The rules on lines 6–11 state that lawful agents adopt all the generic social ideals. On the other hand, evil and chaotic agents consider it ideal to kill others (lines 14–18). Agents cannot have conflicting ideals, as enforced by the constraint on lines 21–22. As discussed in Section 4.5.3, ideals do not change over time, and thus neither `idl` predicate has a time parameter.

```
1  % Example generic ideal: it is not ideal to kill agents
2  idl(neg(kills(Agent))) :-
3      agent(Agent).
4
5  % Lawful agents follow the generic ideals
6  idl(Agent,Action) :-
7      idl(Action),
8      lawful(Agent).
9  idl(Agent,neg(Action)) :-
10     idl(neg(Action)),
11     lawful(Agent).
12
13 % An evil chaotic agent considers it ideal to kill others
14 idl(Agent,kills(Target)) :-
15     evil(Agent),
16     chaotic(Agent),
17     agent(Target),
18     Agent != Target.
19
20 % Agents cannot have conflicting ideals
21 :- idl(Agent,Action),
22     idl(Agent,neg(Action)).
```

Listing 4.9: Managing ideals

Desires

Unlike ideals, agents' desires can change under certain circumstances (explained in Section 4.5.3). We therefore implement desires with a time parameter; the relevant predicate takes the form `des(Agent,Consequence,T)`. Desires are domain-specific, defined separately for each story world. We provide examples of different ways of defining

desires in Listing 4.10. Lines 2–4 show a general desire which applies to all agents. The desire on lines 7–11 is conditional on a particular state of the world: an agent desires to have an edible object if they are hungry. Common sense also tells us that if an agent desires some fluent to become true, they will desire the negation of that fluent to remain false (i.e. fail to become true); this is expressed on lines 15–19. We use constraints to ensure agents cannot have conflicting desires (i.e. an agent cannot desire a consequence and its negation at the same time).

```
1  % General desire: all agents desire to be alive
2  des(Agent,becametrue(is_alive(Agent)),T+1) :-
3      agent(Agent),
4      time(T).
5
6  % Conditional desire: hungry agents desire an edible object
7  des(Agent,becametrue(has(Agent,Object)),T+1) :-
8      agent(Agent),
9      edible(Object),
10     time(T),
11     true(is_hungry(Agent),T+1).
12
13 % Desiring becametrue(F) is equivalent to desiring
14 % failedtobecometrue(neg(F))
15 des(Agent,failedtobecometrue(Fluent),T+1) :-
16     negate(Fluent,Negation),
17     agent(Agent),
18     time(T),
19     des(Agent,becametrue(Negation),T+1).
```

Listing 4.10: Defining desires

Expectations

The prospect-based emotions require us to model agents' expectations, which in turn requires agents to have a sense of which outcomes they consider probable (or at least more probable than their negation, which is how $Prob_i\varphi$ is defined in Section 4.4.1). As with desires, the `prob(Agent,Consequence,T)` predicate is often defined in a domain-specific way. We demonstrate this using the example on lines 3–7 of Listing 4.11. In this case, the agent `Target` will consider it probable that they will die if another agent, `Killer`, attempts to kill them. In addition to specific definitions like this one, we also make the general assumption that agents consider the intended outcomes of

their own actions to be probable (lines 11–17). This is not necessarily warranted in all circumstances, because characters do sometimes attempt actions even if they consider themselves unlikely to succeed, but implementing a general rule like this simplifies our domain definitions.

Expect$_i\varphi$ was defined in Section 4.4.1 in terms of *Prob*$_i\varphi$ and *Bel*$_i\varphi$; we mirror that logical definition in ASP on lines 20–25 of Listing 4.11. An agent expects a particular consequence if they consider it probable (line 24), but do not currently believe the corresponding fluent is true (line 25). Note that the time parameter in `expect(Agent,Consequence,T)` represents the time at which the agent has the expectation, not the time at which they expect `Consequence` to be true; expectations are always interpreted relative to the present. Expectations must be consistent: an agent cannot expect a consequence and its negation at the same time (this is enforced using constraints).

```
1  % Example of specific prob definition: if an agent tries to kill
2  % you, you consider it probable that you will die
3  prob(Target,becametrue(neg(is_alive(Target))),T) :-
4      agent(Target),
5      agent(Killer),
6      time(T),
7      happens(Killer,kills(Target),T).
8
9  % Assume agents consider the intended outcomes of their own
10 % actions to be probable
11 prob(Agent,becametrue(Fluent),T) :-
12     agent(Agent),
13     action(Action),
14     negate(Fluent,Negation),
15     time(T),
16     happens(Agent,Action,T),
17     causes(Agent,Action,Fluent,T).
18
19 % Defining expect (as per Adam et al.'s definition)
20 expect(Agent,becametrue(Fluent),T) :-
21     agent(Agent),
22     negate(Fluent,Negation),
23     time(T),
24     prob(Agent,becametrue(Fluent),T),
25     not bel(Agent,Fluent,T).
```

Listing 4.11: Defining expectations

4.5.6 The Emotion Layer

This section describes the Emotion Layer of our ASP system. It is not domain-specific, and can thus be decoupled from the Action and Belief Layers. The same ASP emotion file can be used with multiple story worlds, provided they adhere to the same general structure (i.e. define the required predicates introduced in Sections 4.5.4 and 4.5.5: `action`, `consequence`, `idl`, `bel`, `des`, etc.). In the ensuing sections, we provide ASP rules for each of the 22 OCC emotions and explain our implementation choices, including any deviations from the logical formalism presented in Section 4.4. As explained in Section 4.5.4, emotions arise in the time-point following the event that caused them, and thus the rules presented in this section produce emotions at $T+1$, where T is a valid time at which events can occur. However, in our explanations we refer to T, not $T+1$, for simplicity.

Joy

As explained in Section 4.5.4, we define the event-based emotions in terms of consequences rather than fluents. Our implementation of Joy (Listing 4.12) is faithful to the logical definition in Equation 4.2, under the assumption that φ is a consequence (this will be assumed for all ensuing definitions). An agent feels Joy at time T if they desire a particular consequence at T (line 6), and believe it is true at T (line 5). In this context, a consequence can be either a successful or failed consequence.

```
1 joy(Agent,Consequence,T+1) :-
2     agent(Agent),
3     negate_consequence(Consequence,Negation),
4     time(T),
5     bel(Agent,Consequence,T+1),
6     des(Agent,Consequence,T+1).
```

Listing 4.12: ASP rule for Joy

Distress

As with Joy, our ASP definition of Distress (Listing 4.13) corresponds to Equation 4.3, provided that φ is a consequence. An agent feels Distress at time T if they believe a particular consequence is true at T (line 5), but desire its negation (line 6).

```
1 distress(Agent,Consequence,T+1) :-
2     agent(Agent),
3     negate_consequence(Consequence,Negation),
4     time(T),
5     bel(Agent,Consequence,T+1),
6     des(Agent,Negation,T+1).
```

Listing 4.13: ASP rule for Distress

Hope

In Equation 4.4, Hope occurs when an agent expects and desires some consequence φ. This is the basis for our ASP definition (Listing 4.14), but it is worth noting that we require the agent to desire the consequence not only at time T (line 6), but also at T + 1 (line 7). This is to avoid the confusing emotional reactions that could occur if an agent's desire for a consequence suddenly changed. For example, consider an agent `a` who desires consequence `c` at time `t`, and thus feels `hope(a,c,t)`. Say an event occurs at time `t` to cause `c`, but at the same time `a`'s desire for `c` changes, so they desire `neg(c)` at time `t+1`. This will cause the agent to feel Distress rather than Joy, which in turn will result in FearsConfirmed (defined in Listing 4.17). It does not make sense for an agent to experience FearsConfirmed without first experiencing Fear.[8]

```
1 hope(Agent,Consequence,T) :-
2     agent(Agent),
3     consequence(Consequence),
4     time(T),
5     expect(Agent,Consequence,T),
6     des(Agent,Consequence,T),
7     des(Agent,Consequence,T+1).
```

Listing 4.14: ASP rule for Hope

Unlike all the other emotions, which arise in the time-point following the inciting event, the unconfirmed prospect-based emotions (Hope and Fear) arise in the same time-point as the event, or earlier. They can only exist before the consequences resulting from the event hold true, because at that point they will be confirmed (causing Satisfaction or

[8]It is possible for FearsConfirmed to arise without a preceding Fear because FearsConfirmed is defined (both in Adam et al.'s logic and our implementation) in terms of `expect` and `distress`, rather than `fear` (refer to Listing 4.17).

FearsConfirmed) or disconfirmed (causing Relief or Disappointment). For this reason, the rules defining Hope and Fear reference T, not $T+1$ like the other emotion rules.

Fear

Our ASP rule for Fear (Listing 4.15) matches Equation 4.5, with the additional requirement that the agent desires the negation of the consequence at both T (line 6) and $T+1$ (line 7). The reasons for this are analogous to those explained for Hope above.

```
1 fear(Agent,Consequence,T) :-
2     agent(Agent),
3     negate_consequence(Consequence,Negation),
4     time(T),
5     expect(Agent,Consequence,T),
6     des(Agent,Negation,T),
7     des(Agent,Negation,T+1).
```

Listing 4.15: ASP rule for Fear

Satisfaction

We implement Satisfaction (Listing 4.16) as described in Equation 4.10. It is based on an agent's earlier expectation of a desirable consequence occurring (line 6), and the resulting Joy when it does (line 8). The `TPrev < T+1` relation (line 7) ensures that Hope and Satisfaction cannot be felt at the same time.

```
1 satisfaction(Agent,Consequence,T+1) :-
2     agent(Agent),
3     consequence(Consequence),
4     time(T),
5     time(TPrev),
6     expect(Agent,Consequence,TPrev),
7     TPrev < T+1,
8     joy(Agent,Consequence,T+1).
```

Listing 4.16: ASP rule for Satisfaction

FearsConfirmed

FearsConfirmed (Listing 4.17) is implemented according to Equation 4.11. The definition is the same as that for Satisfaction, except that the agent must feel Distress as a result of the previously expected consequence (line 8), rather than Joy.

```
1 fearsconfirmed(Agent,Consequence,T+1) :-
2     agent(Agent),
3     consequence(Consequence),
4     time(T),
5     time(TPrev),
6     expect(Agent,Consequence,TPrev),
7     TPrev < T+1,
8     distress(Agent,Consequence,T+1).
```

Listing 4.17: ASP rule for FearsConfirmed

Relief

Relief arises when an undesirable consequence is disconfirmed. Although the logical definition in Equation 4.12 is almost identical to that for Satisfaction (the only difference is that one occurrence of φ is negated), the implementation (Listing 4.18) has some differences due to the way we represent failed consequences. If an agent expects a consequence to occur (specifically, a `succeeded_consequence`, which corresponds to `becametrue(Fluent)`, as defined in Section 4.5.4), they will feel relief when that consequence fails to occur (i.e. the corresponding `failed_consequence`, structured as `failedtobecometrue(Fluent)`, becomes true). We use `reverse_consequence` (line 3) to equate a `succeeded_consequence` with the corresponding `failed_consequence`; their differing structure means they cannot be matched to the same variable.

```
1 relief(Agent,FailedConsequence,T+1) :-
2     agent(Agent),
3     reverse_consequence(SucceededConsequence,FailedConsequence),
4     succeeded_consequence(SucceededConsequence),
5     time(T),
6     time(TPrev),
7     expect(Agent,SucceededConsequence,TPrev),
8     TPrev < T+1,
9     joy(Agent,FailedConsequence,T+1).
```

Listing 4.18: ASP rule for Relief

Disappointment

Disappointment (Listing 4.19) is handled in the same manner as Relief, with respect to Equation 4.13. In this case, an agent experiences Disappointment when they feel Distress about the previously expected consequence failing.

```
1 disappointment(Agent,FailedConsequence,T+1) :-
2     agent(Agent),
3     reverse_consequence(SucceededConsequence,FailedConsequence),
4     succeeded_consequence(SucceededConsequence),
5     time(T),
6     time(TPrev),
7     expect(Agent,SucceededConsequence,TPrev),
8     TPrev < T+1,
9     distress(Agent,FailedConsequence,T+1).
```

Listing 4.19: ASP rule for Disappointment

HappyFor

Fortunes-of-others emotions are based not only on beliefs about the world and other agents' desires, but also on how agents feel about one another. The interpretation of `happyfor(Agent1,Agent2,Consequence,T)` is: `Agent1` feels HappyFor towards `Agent2` because of `Consequence` at time `T`. Our ASP rule for HappyFor (Listing 4.20) matches the definition in Equation 4.18. If `Agent1` believes some consequence is true (line 7), and also believes `Agent2` desires that consequence (line 8), they will feel HappyFor towards `Agent2`, provided they Love `Agent2` (line 9). Agents cannot feel fortunes-of-others emotions towards themselves (line 4).

```
1 happyfor(Agent1,Agent2,Consequence,T+1) :-
2     agent(Agent1),
3     agent(Agent2),
4     Agent1 != Agent2,
5     negate_consequence(Consequence,Negation),
6     time(T),
7     bel(Agent1,Consequence,T+1),
8     bdes(Agent1,Agent2,Consequence,T+1),
9     love(Agent2,Agent2,T+1).
```

Listing 4.20: ASP rule for HappyFor

Pity

The rule for Pity (Listing 4.21) corresponds to Equation 4.19. For the most part, the ASP rule matches the logical definition: if `Agent1` believes some consequence is true (line 7), and also believes `Agent2` desires the negation of that consequence (line 8), they will feel Pity towards `Agent2`, provided they do not Hate `Agent2` (line 10). However, there is one additional requirement, on line 9: `Agent1` must not desire that consequence. This prevents an agent from pitying another agent even though they intentionally caused the negative consequence for that agent. For example, if Bob steals money from Alice, one would usually not expect Bob to feel Pity towards Alice as a result.

```
1  pity(Agent1,Agent2,Consequence,T+1) :-
2      agent(Agent1),
3      agent(Agent2),
4      Agent1 != Agent2,
5      negate_consequence(Consequence,Negation),
6      time(T),
7      bel(Agent1,Consequence,T+1),
8      bdes(Agent1,Agent2,Negation,T+1),
9      not des(Agent1,Consequence,T+1),
10     not hate(Agent1,Agent2,T+1).
```

Listing 4.21: ASP rule for Pity

Resentment

The rule for Resentment (Listing 4.22) is based on Equation 4.20. If `Agent1` believes some consequence is true (line 7), and also believes `Agent2` desires that consequence (line 8), they will feel Resentment towards `Agent2` if they Hate `Agent2` (line 9).

```
1  resentment(Agent1,Agent2,Consequence,T+1) :-
2      agent(Agent1),
3      agent(Agent2),
4      Agent1 != Agent2,
5      negate_consequence(Consequence,Negation),
6      time(T),
7      bel(Agent1,Consequence,T+1),
8      bdes(Agent1,Agent2,Consequence,T+1),
9      hate(Agent1,Agent2,T+1).
```

Listing 4.22: ASP rule for Resentment

Gloating

The rule for Gloating (Listing 4.23) is based on Equation 4.21. If `Agent1` believes some consequence is true (line 7), and also believes `Agent2` desires the negation of that consequence (line 8), they will feel Gloating towards `Agent2` if they Hate `Agent2` (line 9).

```
1 gloating(Agent1,Agent2,Consequence,T+1) :-
2     agent(Agent1),
3     agent(Agent2),
4     Agent1 != Agent2,
5     negate_consequence(Consequence,Negation),
6     time(T),
7     bel(Agent1,Consequence,T+1),
8     bdes(Agent1,Agent2,Negation,T+1),
9     hate(Agent1,Agent2,T+1).
```

Listing 4.23: ASP rule for Gloating

Pride

The rule for Pride (Listing 4.24) is based on Equation 4.24. An agent feels Pride about an action they perform only if they consider the action to be ideal (line 7), and it is successful (line 6).

```
1 pride(Agent,Action,T+1) :-
2     agent(Agent),
3     action(Action),
4     time(T),
5     happens(Agent,Action,T),
6     succeeds(Agent,Action,T),
7     idl(Agent,Action).
```

Listing 4.24: ASP rule for Pride

Shame

As explained in Section 4.4.3, there is an asymmetry between the definitions of Pride and Shame. The disjunction in the definition of Shame presented in Equation 4.25 translates into two separate ASP rules. In Listing 4.25, lines 3–9 represent Shame resulting from an agent performing an action which they consider unideal. This corresponds to the

first part of Equation 4.25: $Bel_i Done_{i:\alpha} Idl_i \neg Happens_{i:\alpha} \top$. Note that `succeeds` does not feature in this rule, because Shame should arise regardless of whether the action is successful. The second rule (lines 12–18) models the Shame that results from an agent attempting an action which is normally considered ideal, but failing. This corresponds to the second part of Equation 4.25: $Bel_i Done_{i:\alpha} Idl_i Happens_{i:\alpha} \varphi \wedge Bel_i \neg \varphi$.

```
1  % Shame resulting from attempting an unideal action
2  % (regardless of success)
3  shame(Agent,Action,T+1) :-
4      agent(Agent),
5      action(Action),
6      negate_action(Action,Negation),
7      time(T),
8      happens(Agent,Action,T),
9      idl(Agent,Negation).
10
11 % Shame resulting from failing an ideal action
12 shame(Agent,Action,T+1) :-
13     agent(Agent),
14     action(Action),
15     time(T),
16     happens(Agent,Action,T),
17     not succeeds(Agent,Action,T),
18     idl(Agent,Action).
```

Listing 4.25: ASP rules for Shame

Admiration

We implement the definition for Admiration from Equation 4.28. In Listing 4.26, `Agent` is the agent experiencing the emotion, and `Actor` is the agent whose action is being admired. As with the definition of Pride, an action must not only be considered ideal (line 8), but must also be successful (line 7) to cause Admiration. Note the additional restriction on line 10 that the agent feeling Admiration must be onstage when the action is performed, so that they see the action happen (i.e. they must be aware of it). This is not a requirement for event-based emotions, because in those cases the action itself is irrelevant, and need not be observed; only the outcome is important. Agents cannot feel Admiration towards themselves (line 9).

```
1  admiration(Agent,Actor,Action,T+1) :-
2      agent(Agent),
3      agent(Actor),
4      action(Action),
5      time(T),
6      happens(Actor,Action,T),
7      succeeds(Actor,Action,T),
8      idl(Agent,Action),
9      Agent != Actor,
10     true(located(Agent,onstage),T).
```

Listing 4.26: ASP rule for Admiration

Reproach

Our implementation of Reproach (Listing 4.27) is based on Equation 4.29. The main difference to Admiration (other than the action being unideal this time, which is defined on line 8 in terms of its negation being ideal) is that it is irrelevant whether the action succeeds or fails. A reprehensible action that is attempted still attracts Reproach, regardless of its outcome, because of the intent behind it. As with Admiration, an agent cannot feel Reproach towards themselves (line 9).

```
1  reproach(Agent,Actor,Action,T+1) :-
2      agent(Agent),
3      agent(Actor),
4      action(Action),
5      negate_action(Action,Negation),
6      time(T),
7      happens(Actor,Action,T),
8      idl(Agent,Negation),
9      Agent != Actor,
10     true(located(Agent,onstage),T).
```

Listing 4.27: ASP rule for Reproach

Love

In Equation 4.30, we define Love in terms of Admiration and Reproach. If an agent feels Admiration towards another agent, but no Reproach, they will feel Love towards that agent. The specific action causing these emotions is irrelevant to the definition of Love, so we define helper predicates to simplify our rules: `some_admiration(Agent,Actor,T)` captures whether there is any Admiration felt by `Agent` towards `Actor` at time `T`, and `some_reproach(Agent,Actor,T)` achieves the same thing for Reproach. The rules defining these predicates are shown in Listing 4.28.

```
1  some_admiration(Agent,Actor,T+1) :-
2      agent(Agent),
3      agent(Actor),
4      Agent != Actor,
5      action(Action),
6      time(T),
7      admiration(Agent,Actor,Action,T+1).
8
9  some_reproach(Agent,Actor,T+1) :-
10     agent(Agent),
11     agent(Actor),
12     Agent != Actor,
13     action(Action),
14     time(T),
15     reproach(Agent,Actor,Action,T+1).
```

Listing 4.28: ASP rules for some_admiration and some_reproach

Our ASP rules for Love are provided in Listing 4.29. The first rule (lines 2–8) implements Equation 4.30: `Agent` will feel Love towards `Actor` if they feel Admiration towards `Actor` (line 7), but no Reproach (line 8). However, as discussed in Section 4.5.3, the object-based emotions persist through time. This requires a frame axiom for maintaining Love (lines 11–17), provided no Reproach is experienced in the meantime (line 17). We also allow Love between any pair of agents to be set arbitrarily at the beginning of a story, representing relationships between characters. This is achieved using the rule on line 20.

```
1  % Defining Love in terms of Admiration and Reproach
2  love(Agent,Actor,T+1) :-
3      agent(Agent),
4      agent(Actor),
5      Agent != Actor,
6      time(T),
7      some_admiration(Agent,Actor,T+1),
8      not some_reproach(Agent,Actor,T+1).
9
10 % Frame axiom for Love persisting through time
11 love(Agent,Actor,T+1) :-
12     agent(Agent),
13     agent(Actor),
14     Agent != Actor,
15     time(T),
16     love(Agent,Actor,T),
17     not some_reproach(Agent,Actor,T+1).
18
19 % Initially, there may be Love between any pair of characters
20 0 { love(Agent1,Agent2,0) : agent(Agent1) : agent(Agent2) } 1.
```

Listing 4.29: ASP rules for Love

Hate

Hate (Listing 4.30) is based on Equation 4.31, and also uses the helper predicates defined in Listing 4.28. Lines 2–8 define Hate as the opposite of Love: `Agent` will feel Hate towards `Actor` if they feel Reproach towards `Actor` (line 7), but no Admiration (line 8). The frame axiom for maintaining Hate through time, provided no Admiration occurs, is defined on lines 11–17. As with Love, we allow Hate to be set arbitrarily between any pair of agents at the beginning of a story (line 20). To ensure these generators do not produce an inconsistency, the constraint on lines 23–24 ensures that an agent cannot feel Love and Hate simultaneously towards the same agent.

```
1  % Defining Hate in terms of Admiration and Reproach
2  hate(Agent,Actor,T+1) :-
3      agent(Agent),
4      agent(Actor),
5      Agent != Actor,
6      time(T),
7      some_reproach(Agent,Actor,T+1),
8      not some_admiration(Agent,Actor,T+1).
9
10 % Frame axiom for Hate persisting through time
11 hate(Agent,Actor,T+1) :-
12     agent(Agent),
13     agent(Actor),
14     Agent != Actor,
15     time(T),
16     hate(Agent,Actor,T),
17     not some_admiration(Agent,Actor,T+1).
18
19 % Initially, there may be Hate between any pair of characters
20 0 { hate(Agent1,Agent2,0) : agent(Agent1) : agent(Agent2) } 1.
21
22 % You cannot Love and Hate the same agent at the same time
23 :- love(Agent1,Agent2,T),
24     hate(Agent1,Agent2,T).
```

Listing 4.30: ASP rules for Hate

Gratification

Compound emotions are defined based on specific pairwise combinations of event and agent-based emotions. Gratification (Listing 4.31) reflects Equation 4.32: it arises if an agent experiences Pride about their own action (line 6), and Joy about one of its consequences (line 7).

```
1  gratification(Agent,Action,Consequence,T+1) :-
2      agent(Agent),
3      action(Action),
4      succeeded_consequence(Consequence),
5      time(T),
6      pride(Agent,Action,T+1),
7      joy(Agent,Consequence,T+1).
```

Listing 4.31: ASP rule for Gratification

Remorse

The ASP rule for Remorse (Listing 4.32), based on Equation 4.33, requires that an agent feels Shame about their own action (line 6), and Distress about one of its consequences (line 7).

```
1 remorse(Agent,Action,Consequence,T+1) :-
2     agent(Agent),
3     action(Action),
4     succeeded_consequence(Consequence),
5     time(T),
6     shame(Agent,Action,T+1),
7     distress(Agent,Consequence,T+1).
```

Listing 4.32: ASP rule for Remorse

Gratitude

Gratitude (Listing 4.33) is analogous to Gratification, except directed at an agent other than the self. As per Equation 4.34, to feel Gratitude an agent must feel Admiration towards another agent for an action they performed (line 8), and experience Joy about one of its consequences (line 9).

```
1 gratitude(Agent,Actor,Action,Consequence,T+1) :-
2     agent(Agent),
3     agent(Actor),
4     Agent != Actor,
5     action(Action),
6     succeeded_consequence(Consequence),
7     time(T),
8     admiration(Agent,Actor,Action,T+1),
9     joy(Agent,Consequence,T+1).
```

Listing 4.33: ASP rule for Gratitude

Anger

Anger (Listing 4.34) is the other-directed version of Remorse, and is defined in Equation 4.35. To feel Anger, an agent must feel Reproach towards another agent for an action they performed (line 8), and experience Distress about one of its consequences (line 9).

```
anger(Agent,Actor,Action,Consequence,T+1) :-
    agent(Agent),
    agent(Actor),
    Agent != Actor,
    action(Action),
    succeeded_consequence(Consequence),
    time(T),
    reproach(Agent,Actor,Action,T+1),
    distress(Agent,Consequence,T+1).
```

Listing 4.34: ASP rule for Anger

4.6 Evaluation

Having defined ASP rules for each of the OCC emotions, it is important we evaluate our implementation before applying it to story generation. In this case, the measure of performance is whether our emotion model generates the appropriate combinations of emotions for particular circumstances within a story, as compared to the emotions readers infer when they read that story. In Section 4.6.1, we explain how we produce emotion data from a selection of Aesop's fables using our model. Section 4.6.2 describes the survey we carry out based on the same set of fables, to determine which emotions people expect characters to feel in those situations. We explain our approach to data analysis in Section 4.6.3, before presenting the results in Section 4.6.4. We identify the limitations of our model in Section 4.6.5, and present our conclusions in Section 4.6.6.

4.6.1 Producing Emotion Data

Initially, we performed this evaluation on our *decreasoner* implementation of the OCC theory [145]. To produce emotion data for evaluation, we arbitrarily selected six of Aesop's fables, as follows:

1. Fable 3: *The Eagle and the Fox*
2. Fable 15: *The Goat and the Goatherd*
3. Fable 40: *The Fox and the Billy-Goat*
4. Fable 58: *The Man and the Fox*

5. Fable 82: *The Ploughman and the Frozen Snake*

6. Fable 331: *The Wasp and the Snake*

We decomposed each fable into its constituent events and their consequences, which were then encoded in *decreasoner* syntax, along with information about how characters appraise particular events and consequences. Each consequence was allocated to an emotion class [145], based on the combination of Joy and Distress it was expected to cause for the agents involved.

Running each fable through the system produced all the emotions experienced by the characters at every time-point, as predicted by our emotion model (except the prospect-based emotions, Hope, Fear, Satisfaction, FearsConfirmed, Relief, and Disappointment, which, as mentioned in Section 4.5.1, were excluded from this implementation). Data was produced for three different versions of our *decreasoner* emotion model:

1. **Version 1:** In this version, Love and Hate[9] are fixed at $T = 0$, and we use our own assignment of consequences to emotion classes.

2. **Version 2:** In this version, Love and Hate are also fixed, but consequences are assigned to emotion classes based on the majority responses to the Joy/Distress questions in our survey (described in Section 4.6.2). This gives us an indication of whether the other emotions are entailed from Joy and Distress in line with readers' expectations.

3. **Version 3:** In this version, Love and Hate are permitted to vary based on admirable and reproachable actions. This is a significant change, because Love and Hate play an important role in determining the fortunes-of-others emotions. Emotion classes are assigned as in Version 2.

Apart from the differences described above, all emotion definitions and fable encodings are identical between the three versions of the *decreasoner* model.

Following the development of the ASP implementation of our emotion model (described in detail in Section 4.5), we wished to compare its performance to the original *decreasoner* implementation. To do so, we encoded the same six fables (listed above)

[9]In our *decreasoner* model, Love and Hate are renamed to Like and Dislike. However, because their interpretations are identical to that of Love and Hate in the ASP version, we use those terms to avoid confusion.

in ASP, using the same breakdown of events and consequences.[10] We used the *Potsdam Answer Set Solving Collection* (*Potassco*) [60] to generate solutions, and compared the resulting emotion data to the *decreasoner* output. Note that our ASP model does implement the prospect-based emotions. They were disregarded for the purposes of this evaluation, as they are not included in the *decreasoner* emotion data or the survey results.

4.6.2 Survey Design

To determine whether the emotions generated by our OCC model are consistent with readers' expectations of what characters should feel, we conducted an online survey based on the same six fables listed in Section 4.6.1.[11] Participants were provided with the complete text of each fable, followed by a series of questions asking them to select which emotions they expected characters to feel following each event. Care was taken to ensure the decomposition of fables into events and consequences as presented in the survey matched the way they were encoded in our system. As the survey was originally designed to evaluate our *decreasoner* implementation, which did not handle the prospect-based emotions, the survey questions do not address Hope, Fear, Satisfaction, FearsConfirmed, Relief, or Disappointment. The survey format was multiple-choice, with a separate question for each emotion type, per agent, per event. To ensure a consistent interpretation of emotion words by different respondents, we included their definitions in the list of options, as shown in Figure 4.2. The full set of questions pertaining to one fable is provided in Appendix A.

[10] Some of the fables used for this evaluation are also used in Chapter 5 to contribute emotion data for certain morals. In some cases, the final decomposition of the fables into events and consequences differs from that used in the *decreasoner* version and the emotion survey (refer to Section 4.6.2). In these cases, the emotion data from the final encoding (i.e. that used for inductive logic programming in Chapter 5) was not used for this evaluation. The fables concerned were re-implemented in ASP to reflect the same event/consequence breakdown as used in the *decreasoner* system and the survey, to ensure a valid comparison.

[11] This study was approved by UNSW Human Research Ethics Advisory Panel 'H,' with approval number 08/2011/29.

1.1.6. Which of the following (if any) would you expect the FOX to feel about losing her cubs?

☐ Pleased about this outcome (Joy)

☐ Displeased about this outcome (Distress)

☐ None of the above

Figure 4.2: Sample question from the OCC emotion survey

4.6.3 Approach to Analysis

To evaluate our model, we require a baseline for comparison; we define this based on the survey results. We decompose each question into the available emotion options, such that each can be treated as a binary classification (i.e. true if the emotion was produced in that circumstance, false if not). In each case, the majority response from the survey is considered to be the prediction of an ideal classifier, represented as a binary vector, which will serve as our baseline. This is the best possible result given the noise in the data due to participants' differing opinions.

In comparing the performance of our system to the baseline classifier, we consider two types of error:

1. **False positive rate:** the percentage of instances in which the system predicts an emotion which was not reported by a user. This is calculated as shown in Equation 4.36, where FP is the number of false positives, and TN is the number of true negatives. The denominator sums to the total number of negative instances (i.e. instances where an emotion was not predicted by a participant).

$$\textit{False Positive Rate} = \frac{FP}{FP + TN} \tag{4.36}$$

2. **False negative rate:** the percentage of instances where the system fails to predict an emotion which was predicted by a user. This is calculated as shown in Equation 4.37, where FN is the number of false negatives, and TP is the number of true positives. In this case, the denominator is the total number of positive instances (i.e. instances where an emotion was predicted by a participant).

$$\textit{False Negative Rate} = \frac{FN}{FN + TP} \tag{4.37}$$

To calculate the error rates for the ideal classifier, we compare its binary vector to the vector obtained from each participant's survey response, summing the false positives, false negatives, true positives, and true negatives across all events and fables. The percentages are calculated as described in Equations 4.36 and 4.37. Error rates for each of our emotion model implementations were calculated in the same manner.

4.6.4 Results

A total of 25 participants took part in the survey. All participants responded to the first fable, 23 responded to the second, and 17 completed the entire survey. Although such a small sample size will not yield statistically significant results, the responses, particularly those where there is a strong agreement between participants, provide a useful indication of the emotions readers would expect the characters to feel.

Across the six fables, there were a total of 462 emotions users could select. Treating each as a binary value (true if the emotion occurs, false if it does not) yields a binary vector of length 462 for each respondent. If all 25 participants answered every question, this would provide 11,550 values overall. However, not all participants completed the entire survey. Our data consists of 8,964 meaningful values, where we take meaningful to mean they are defined (i.e. the user submitted an answer to the corresponding question). Each data source (the ideal classifier reflecting the majority survey responses, the three *decreasoner* implementations, and the ASP model) also yields a binary vector of length 462, which represents the emotions generated by that implementation. We compare these vectors to every individual survey response vector (ignoring undefined values), and sum the total number of false positives and false negatives. Across all participant responses, there were 2,831 positive instances (where a participant predicted a particular emotion would occur), and 6,133 negative instances (where a participant indicated an emotion would not occur). The percentages shown in Table 4.7 are calculated by dividing the total false positives and false negatives by 2,831 and 6,133 respectively.[12]

[12]The percentages presented in Table 4.7 for the baseline classifier and the *decreasoner* implementations differ from those presented in our earlier work [145]. The values were misreported in the previous publication, and have been corrected here.

Version	Error Rates (FP = False Positive Rate, FN = False Negative Rate)							
	Overall		Joy/Distress		Love/Hate		Other Emotions	
	FP	FN	FP	FN	FP	FN	FP	FN
Baseline	6.72%	13.32%	8.86%	7.22%	6.80%	20.68%	6.61%	14.12%
DEC v1	10.75%	58.35%	2.82%	29.87%	16.27%	92.67%	11.03%	60.71%
DEC v2	15.65%	51.29%	10.88%	12.41%	16.27%	92.67%	16.06%	56.21%
DEC v3	15.42%	37.83%	10.88%	12.41%	24.93%	37.96%	15.54%	49.60%
ASP	8.23%	33.27%	5.44%	20.49%	16.00%	23.04%	7.54%	40.55%

Table 4.7: Error rate comparison to baseline classifier

The results presented in Table 4.7 allow us to compare the error rates of the different emotion implementations. In addition to overall error rates, which take into account all modelled emotions, we present the error rates for three different subsets of emotions:

1. **Joy/Distress:** error rates for Joy and Distress only.
2. **Love/Hate:** error rates for Love and Hate only.
3. **Other Emotions:** error rates for all emotions apart from Joy, Distress, Love, and Hate.

The intention behind isolating Joy/Distress and Love/Hate is to assess the impact of the differences between the three *decreasoner* implementations, as described in Section 4.6.1. Such partitioning is not meaningful for evaluating the ASP version because, unlike the *decreasoner* implementations, it does not use emotion classes based on Joy and Distress.

There are also a few important points to note when interpreting the results presented in Table 4.7:

- The extremely high false negative error rates for Love/Hate in the first two *decreasoner* implementations (DEC v1 and DEC v2) result from the fact that Love and Hate are fixed in both. Thus, for example, no matter what happens during the fable, if two characters Love each other at the beginning, they will Love each other at the end. This is not always realistic, particularly after they eat each other's offspring (as in Fable 3), so high error rates are to be expected.

- Love and Hate are prerequisites for the generation of other emotions (i.e. the fortunes-of-others emotions). As such, it is not surprising that higher false negative rates for Love/Hate correspond to higher false negative rates overall. If Love and Hate are not produced as expected, the appropriate fortunes-of-others emotions will not be produced either.

- The Joy/Distress error rates for DEC v2 and DEC v3 are identical because the same assignment of consequences to emotion classes was used. Joy and Distress are the only emotions that result directly from emotion class assignment [145].

- The only difference between DEC v2 and DEC v3 is that Love and Hate are permitted to vary. Therefore any differences in error rates between these two implementations stem from the impact of changes in Love and Hate on the generation of other emotions.

- The error rates for the baseline classifier are a reflection of the disagreement between participants' responses (i.e. how many responses differed from the majority).

In most cases, the error rates for the ASP emotion model are lower than those for the *decreasoner* implementations. We suggest it is most useful to compare the performance of the ASP model to DEC v3, as both allow Love and Hate to vary during the course of a story, making them markedly different from DEC v1 and DEC v2. The ASP model outperforms DEC v3 in all error rates bar one: the Joy/Distress false negative error rate. However, given that emotion classes in DEC v3 were assigned to consequences based on the majority survey responses (as explained in Section 4.6.1), this result is hardly surprising.

To help visualise the differences in error rates for the various implementations, we graph the overall results from Table 4.7, showing error bars for 95% confidence intervals calculated using the Wilson score method [178]. Figure 4.3 shows the false positive rates, and Figure 4.4 the false negative rates. It is clear from the graphs that, overall, the ASP implementation has significantly lower false positive and false negative rates as compared to all three *decreasoner* versions.

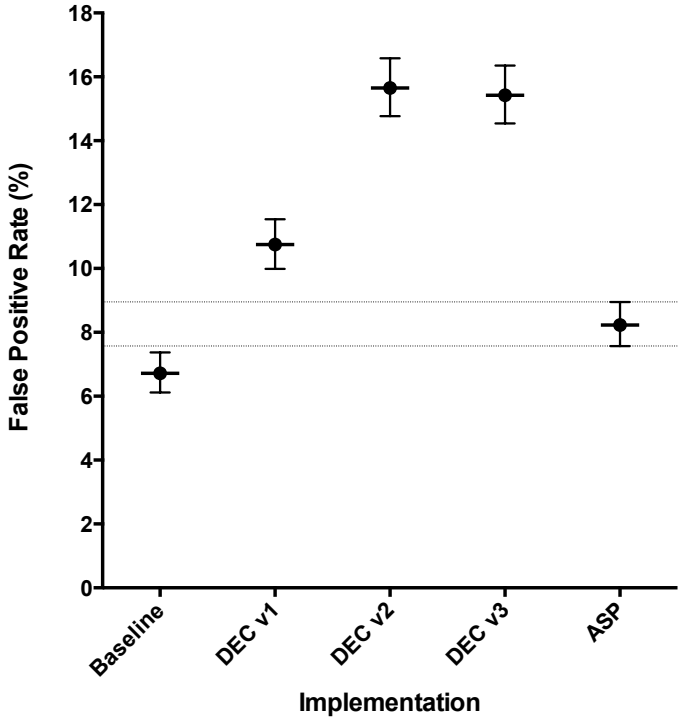

Figure 4.3: False positive rates showing 95% confidence intervals

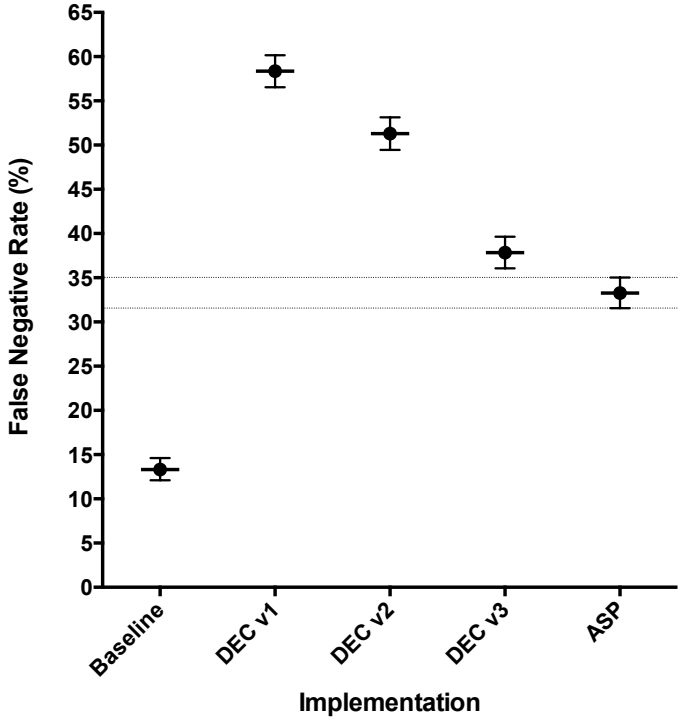

Figure 4.4: False negative rates showing 95% confidence intervals

Closer analysis of the survey results led us to one other observation worthy of note. Our emotion model aims to implement the OCC theory, and it is with respect to this goal that we should judge its performance. However, our survey results showed that people's interpretations of emotions are not always consistent with the OCC definitions. For example, in Fable 3, only 50% of survey respondents believed the eagle would feel Distress about the fox no longer being hungry; the rest believed the eagle should experience no emotions about this consequence. Despite this, 85% of respondents indicated that the eagle should feel Anger about the fox no longer being hungry, even though, according to the OCC theory, Distress is a prerequisite for Anger. This highlights the ambiguity inherent in interpreting emotions, even when clear definitions are provided, and hence the difficulty in performing an objective evaluation. A computational model based on a particular psychological theory can, at best, reflect people's views only as well as that theory does.

4.6.5 Limitations

As with any model of a complex phenomenon, our OCC implementation has a number of limitations. This does not preclude the model from being useful (in our case, for story generation), but it is important that these limitations are acknowledged. Most relate directly to the assumptions we made in Section 4.5.3. We observed the remainder during our efforts in implementing Aesop's fables, upon finding that certain story features could not be represented satisfactorily. Our discussion here focusses on how these limitations affect our system's story generation capabilities, as this is its intended application. We identify the following limitations of our ASP implementation of the OCC theory of emotion:

- The main limitation arising from our action model is that we do not allow simultaneous events (refer to Section 4.5.3). This limits the complexity of the stories we are able to represent, and thus the complexity of the resulting emotions. In principle, our emotion definitions should work without modification in a domain that allows parallel events, but this has not been tested.

- Our simplistic omniscient belief model (refer to Section 4.5.3) prevents agents from experiencing incorrect (or perhaps better termed inappropriate) emotions resulting from incorrect beliefs, since agents can never have incorrect beliefs. It is easy to imagine a situation in which Bob is happy for Alice after she wins

a chocolate hamper in a raffle, without realising that Alice is actually allergic to chocolate. She may be upset when Bob enthusiastically congratulates her; a more appropriate reaction would have been pity. Agents are also unable to attribute blame for an event to an agent who is not actually responsible. Such misattribution is often the source of interesting conflicts in stories. Similarly, incorrect assessments of others' emotions are a key ingredient in drama. For example, Alice loves Bob, but marries Carl because she thinks Bob does not love her, only to discover too late that he does. Many interesting plots can result from characters having incorrect beliefs; these cannot be modelled by our system.

- Our implementation does not provide a mechanism for modelling emotion intensity (refer to Section 4.5.3). Although we deem this unnecessary for very simple stories such as fables, in more sophisticated stories, characters' behaviours (or at least whether those behaviours make sense to a reader) can depend very much on the intensity of certain emotions. For example, the intense anger a character feels after his best friend is brutally attacked might prompt him to shoot the attacker. The same reaction would not be justified in response to the comparatively mild anger he feels when his younger brother eats the last slice of cake without offering to share.

- Our definitions of Love and Hate, based on Admiration and Reproach, are extremely simplistic (refer to Section 4.4.4). This is a significant limitation, in that it also affects the generation of the fortunes-of-others emotions (refer to Section 4.4.2). The difference in error rates shown in Table 4.7 between DEC v2 and DEC v3 highlight the impact that changing the behaviour of these two emotions can have. A more realistic definition could significantly improve the system's overall performance.

- Prospect-based emotions can only be felt with regards to possible future outcomes (refer to Section 4.5.2). Again, while this is sufficient for simple stories, it restricts the nature of the emotions characters can experience. This also relates to our omniscient belief model, because for agents to feel Hope or Fear about the past or the present there must be some degree of uncertainty in their beliefs.

- Agent-based emotions can only be experienced immediately following the inciting event. As a result, emotions such as Gratitude and Anger cannot be felt retrospectively about earlier events. For example, in Fable 79, an eagle saves a ploughman

from a collapsing wall by stealing his headband so that the ploughman chases him. It is only later, upon returning to the wall, that the ploughman realises what the eagle did, and feels Joy that he was not crushed by the wall. One would expect the ploughman to feel Gratitude towards the eagle in this situation. However, our implementation does not provide a way to connect the eagle's action of stealing the ploughman's headband with saving him from the collapsing wall, because this outcome is only perceived several time-steps later.

4.6.6 Conclusion

As identified in Section 4.6.5, there are a number of limitations to our emotion model, arising not only from the simplifying assumptions we make, but also from the ambiguity inherent in the emotional interpretation of events. However, our aim was not to produce a perfect psychological model of human emotion. We only require a reasonable and consistent approximation, which we can apply to story generation. Although a more sophisticated emotion model would undoubtedly result in more realistic characters and finer-grained control over story trajectories, we consider the implementation described in this chapter sufficient for modelling emotions in the simple kinds of stories we aim to represent, and ultimately generate. Our choice of the ASP model over the *decreasoner* implementations is justified not only by marked improvements in runtime, but also the lower error rates when compared to the emotions people expect characters to feel in the relevant situations.

4.7 Summary

In this chapter, we described our implementation of the OCC theory of emotion. We began with an overview of existing psychological theories of emotion, followed by a summary of the main computational applications for emotion. After justifying our choice of the OCC theory for implementation, we described the theory and its logical formalisation. We explained our ASP implementation of the OCC theory, and evaluated the performance of our model by comparing the emotions it generates to those that human readers expect to arise in the same stories. In the following chapter, we use this emotion model to develop a representation for the morals that were selected for implementation in Chapter 3.

Chapter 5

Correlating Morals with Emotions

So far, about morals, I know only that what is moral is what you feel good after and what is immoral is what you feel bad after.

Ernest Hemingway
Death in the Afternoon

Morals give stories meaning. Emotions help us derive meaning from stories. It is thus not unreasonable to propose a relationship between morals and emotions. Such a relationship could be leveraged to yield a representation for morals in terms of emotions, a phenomenon which, while far from completely understood, is at least more frequently modelled. This would pave the way for generating stories with particular morals in arbitrary domains, by selecting sequences of events that cause characters to experience the necessary emotions.

In Chapters 3 and 4, we introduced these two crucial aspects of story: morals and emotions. Here we bring them together, aiming to use character emotions as a foundation for representing morals. We adopt inductive logic programming (ILP) to automatically derive rules which define particular morals in terms of OCC emotions, using Aesop's fables as a corpus. Due to the limited size of the data set, the performance of the resulting rules is inadequate for them to be used directly to generate stories. Nevertheless, they provide a useful starting point, which we further develop by hand to produce more effective definitions suitable for story generation. Section 5.1 begins by describing how we produce emotion data from Aesop's fables, using the emotion model described in Chapter 4. We introduce inductive logic programming in Section

5.2, and describe our experiments in tree-learning and rule-learning in Sections 5.3 and 5.4 respectively. In Section 5.5, we extend the rules obtained using ILP to establish ASP rules for each of the six morals that were selected for implementation in Chapter 3. These rules form the basis of our Moral Storytelling System, which will be described in Chapter 6. We summarise the contributions of this chapter in Section 5.6.

5.1 Obtaining Emotion Data

To examine the relationship between morals and emotions, we require emotion data from stories corresponding to particular morals. In Chapter 3, we identified fables as a useful source of morals, and selected six of these to implement: Retribution, Greed, Realistic Expectations, Recklessness, Pride, and Reward. We presented a list of Aesop's fables corresponding to each of these morals in Section 3.6.3 (refer to Table 3.5). We use these 37 fables as a source of emotion data.

Although we could annotate the fables with emotions by hand, we opt instead to encode them in ASP and run them through the emotion model described in Chapter 4. There are two benefits to this approach. The first is that it is less error-prone. Emotions will always be generated according to the ASP rules in a consistent fashion. The second stems from the fact that the same emotion model will ultimately be used for story generation. Thus, regardless of how accurately the emotions describe what characters feel in any given story, when we later use the same emotion model to generate stories with those emotions, they should convey the appropriate moral.

To produce emotion data, we follow the same process as that described in Section 4.6.1: we decompose each fable into events and their consequences, and encode this in our ASP action model (described in Section 4.5.4). We define each character's desires, ideals, and expectations (as per Section 4.5.5) in the context of the available actions and consequences, and run the solver on the resulting encoding to generate the corresponding emotions. It is worth noting here that we initially generated emotion data for these fables using our *decreasoner* implementation, and used this to identify relationships between emotions and morals [146]. However, upon moving to the ASP implementation, all fables were re-encoded and the data re-generated with the new emotion model, which is based on Adam et al.'s logical framework [8]. This chapter describes the analysis we perform on the emotion data generated using ASP.

5.2 Inductive Logic Programming

Even stories as simple as fables can involve a large number of emotions. This makes it difficult to draw out common emotion sequences by hand, particularly from among many examples. We use inductive logic programming (ILP) [112] to help facilitate this task, and identify relationships between emotions and morals that we can ultimately leverage for story generation. The idea behind ILP is to automatically construct predicate-based rules from supplied background knowledge and positive and negative examples. In Section 5.2.1 we introduce Aleph, the ILP system we use for our research. Sections 5.2.2 and 5.2.3 describe the background knowledge and examples respectively that we supply to Aleph in order to learn rules corresponding to morals.

5.2.1 Aleph

Aleph [163] is an ILP system based on Progol, which evolved from the earlier system P-Progol [162]. Aleph is written in Prolog, and designed for use with the YAP Prolog compiler [143]. Aleph's basic learning algorithm, as described in *The Aleph Manual* [163], works as follows:

1. Select an example to be generalised.
2. Build the most-specific-clause (also called the bottom clause) which entails the selected example.
3. Find a more general clause by searching for a subset of the literals in the bottom clause with the best score.
4. Add the best clause from step 3 to the current theory, and remove all examples that are made redundant.

Steps 1–4 are repeated until no examples remain. The system provides a wide range of options for modifying the search parameters, including search type, evaluation function, and so on. We will not describe all the available options here; a comprehensive list is provided in *The Aleph Manual*. The options we selected in our use of the system are shown in Sections 5.3 and 5.4.

Aleph requires three data files in order to construct theories: a background knowledge file (with file extension .b), a file containing positive examples (with file extension

.f), and a file containing negative examples (with file extension .n). We describe the format of each, including illustrative examples, in Sections 5.2.2 and 5.2.3.

5.2.2 Background Knowledge

The background knowledge file (filename.b) supplied to Aleph needs to contain Prolog clauses that encode any information relevant to the domain. In our case, this includes the emotion data for all the relevant fables (as described in Section 5.1), as well as additional predicates we define to represent useful relationships between emotions (in particular, temporal relations). It also needs to include mode declarations, which declare the mode of call for predicates, types, which need to be specified for every predicate argument, and determinations, which specify the predicates that can be used to construct a hypothesis.

Mode Declarations

The general form of a mode declaration is as follows:

```
:- mode(RecallNumber,PredicateMode).
```

`RecallNumber` bounds the number of instantiations of the predicate (if this is known in advance); this can be numeric, or `*` if there is no fixed limit. We use `*` in all of our mode declarations. `PredicateMode` is a template showing a predicate's structure. Each argument must be declared to be one of three types: input variables (e.g. `+T`), output variables (e.g. `-T`), or constants (e.g. `#T`). Input variables are those for which the values are expected to be provided when querying the rule-base, whereas output variables are supplied as variables, for which Aleph should return solutions [135]. Constants must be ground terms.

Listing 5.1 provides some examples of mode declarations used in our work, based on Retribution (the only difference for the other morals is the head clause, which matches the name of the moral). Line 2 declares the predicate which is to appear in the head of the learned clauses, in this case `retribution`. The remaining declarations are for predicates which are to appear in the body of these clauses. Lines 5–7 show examples of mode declarations for emotions; lines 10–12 declare their negations. Finally, line 15 provides an example of a mode declaration for an auxiliary predicate we define to capture temporal relationships between emotions in a story (this is explained in the *Additional Predicates* section).

```
1  % Mode declaration for the head of the clause
2  :- modeh(1,retribution(+story)).
3
4  % Mode declarations for emotions (for clause body)
5  :- modeb(*,joy(+story,-agent,-consequence,-t)).
6  :- modeb(*,pity(+story,-agent,-agent,-consequence,-t)).
7  :- modeb(*,remorse(+story,-agent,-event,-consequence,-t)).
8
9  % Negation mode declarations for emotions (for clause body)
10 :- modeb(*,not(joy(+story,-agent,-consequence,-t))).
11 :- modeb(*,not(pity(+story,-agent,-agent,-consequence,-t))).
12 :- modeb(*,not(remorse(+story,-agent,-event,-consequence,-t))).
13
14 % Mode declarations for additional predicates (for clause body)
15 :- modeb(*,after(-t,-t)).
```

Listing 5.1: Examples of Aleph mode declarations

Types

Each of the types referenced in the mode declarations in Listing 5.1 needs to be defined. Listing 5.2 provides an example of each: `t` (representing time, which has a maximum value of 7 in the fable data), `agent` (defined for each agent appearing the emotion data), `consequence` (defined for each consequence appearing in the emotion data), `event` (defined for each event appearing in the emotion data), and `story` (defined for each fable used to provide data).

```
1  % Define time ('t')
2  t(0).
3
4  % Define 'agent'
5  agent(a1).
6
7  % Define 'consequence'
8  consequence(becametrue(is_trapped(a1))).
9
10 % Define 'event'
11 event(bites(a1)).
12
13 % Define 'story'
14 story(fable142).
```

Listing 5.2: Examples of Aleph type definitions

Determinations

Determination statements are used to specify which predicates Aleph can use in constructing a hypothesis. The general form of a determination statement is as follows:

```
:- determination(TargetName/Arity,BackgroundName/Arity).
```

`TargetName` is the predicate that is to appear in the head of the hypothesised clauses, whereas `BackgroundName` specifies a predicate that can appear in the body. A determination statement needs to be provided for each predicate Aleph is to use in searching for a hypothesis.

In Listing 5.3, we show examples of determinations defined for the moral Retribution. Those for the other morals are identical, except with `TargetName` replaced by the name of the appropriate moral. The first three determinations (lines 2–4) correspond to the mode declarations on lines 5–7 of Listing 5.1. Similar determinations are provided for each of the emotions except Love and Hate. These emotions persist, so their value at any specific time-point is not significant. In any case, they are indirectly taken into account via the fortunes-of-others emotions. Line 7 shows the determination for `not`, which allows hypothesised clauses to include negations of emotions. Finally, line 10 shows the determination for the additional predicate `after`, which we describe in the following section.

```
1  % Example determination statements for emotions
2  :- determination(retribution/1,joy/4).
3  :- determination(retribution/1,pity/5).
4  :- determination(retribution/1,remorse/5).
5
6  % Determination statement for 'not'
7  :- determination(retribution/1,not/1).
8
9  % Determination statement for additional predicate 'after'
10 :- determination(retribution/1,after/2).
```

Listing 5.3: Examples of determination statements

Additional Predicates

Apart from the predicates comprising the emotion data, we define three additional predicates for Aleph to use in constructing a theory, shown in Listing 5.4. The `after` and `after_or_equal` predicates allow Aleph to learn temporal relationships between

emotions (for example, if Joy must occur before Distress in the story). It is not necessary to define a predicate for simultaneity, because in this case Aleph will simply match the same variable for the time parameter in both emotion predicates. The `reverse_consequence` predicate allows Aleph to recognise the equivalence between a consequence that was successful (for example, `becametrue(is_trapped(a1))`) and the corresponding failed consequence (in this case, `failedtobecometrue(is_trapped(a1))`). We provide determinations for each of these predicates, so they are all included in Aleph's search for rules.

```
1  % Predicate to define temporal > relation between emotions
2  after(T2,T1) :-
3      t(T1),
4      t(T2),
5      T2 > T1.
6
7  % Predicate to define temporal >= relation between emotions
8  after_or_equal(T2,T1) :-
9      t(T1),
10     t(T2),
11     T2 >= T1.
12
13 % Consequence equivalence except for success/failure
14 reverse_consequence(becametrue(X),failedtobecometrue(Y)) :-
15     consequence(becametrue(X)),
16     consequence(failedtobecometrue(Y)),
17     X = Y.
```

Listing 5.4: Additional predicates provided as background knowledge

Emotion Data

Across the emotion data obtained from all 37 relevant fables, there are a large number of different characters, events, and consequences, even though any given fable will usually only have two or three characters, and three or four events. In running preliminary learning experiments on this data, we found the search took a prohibitively long time. We were able to achieve significant time improvements by reducing the number of unique ground values for characters. Thus, we define only three agents; this is the largest number of characters appearing in any of the fables we deal with. For each fable, we replace the character names with `a1`, `a2`, or `a3`. Although we could have performed a similar replacement for consequences and events as well, the speed improvement from

replacing character names was sufficient to perform the learning in a practical timeframe. Listing 5.5 shows an example of the structure of the emotion data supplied to Aleph. The data in this example corresponds to Fable 204.

```
1 joy(fable204,a1,becametrue(has_hare(a1)),1).
2 hope(fable204,a1,becametrue(has_deer(a1)),2).
3 disappointment(fable204,a1,failedtobecometrue(has_deer(a1)),3).
4 distress(fable204,a1,becametrue(neg(has_hare(a1))),3).
5 distress(fable204,a1,failedtobecometrue(has_deer(a1)),3).
6 remorse(fable204,a1,chases_deer,becametrue(neg(has_hare(a1))),3).
7 shame(fable204,a1,chases_deer,3).
```

Listing 5.5: Emotion data corresponding to Fable 204

5.2.3 Examples

In addition to background knowledge, Aleph requires positive and negative examples of the concept to be learned. These are supplied as separate files, filename.f for positive examples, and filename.n for negative examples. The examples take the form of ground facts, as shown in Listing 5.6. For learning a particular moral, the fables corresponding to that moral would be listed in the .f file, and all other fables would be listed in the .n file. Note, however, that Aleph's tree-learning approach does not require a negative examples file; this is discussed further in Section 5.3.1.

```
1 pride(fable20).
2 pride(fable29).
3 pride(fable161).
4 pride(fable237).
```

Listing 5.6: Format of Aleph examples files

5.3 A Tree-Based Approach

As a first step in determining whether there is a relationship between morals and emotions, we use Aleph's tree-learning approach to produce a classification tree for morals, based on the emotions arising in the relevant fables. Despite the relatively low accuracy of the resulting classifiers, they reveal similarities between certain morals, which may impact how easily they are distinguished from one another in stories.

We initially performed the experiments described in this section using data from our decreasoner implementation, and a slightly different set of morals [145]. Although the process described here is the same, all results are based on more recent experiments using the data produced by our ASP system. In Section 5.3.1, we explain the modifications that had to be made to Aleph's background knowledge file to use the tree-learning approach. Section 5.3.2 presents the search parameters we set for this experiment, and Section 5.3.3 explains how we evaluate the results. Sections 5.3.4 and 5.3.5 present the results of two separate tree-learning experiments we performed: classifying morals individually, and in groups. We draw conclusions from our results in Section 5.3.6.

5.3.1 Adjustments to Background Knowledge

There are a few important differences in the way background knowledge needs to be provided for tree-learning. In this case, the predicate defined as the head of the hypothesised clause must be of arity 2, where the first parameter represents the example, and the second parameter is the class to be predicted (in our case, the moral). Thus, the declaration for the head of the clause is the predicate `moral/2`, as shown on line 2 of Listing 5.7, instead of moral-specific predicates such as the one exemplified on line 2 of Listing 5.1. This also requires us to define which parameter of the `moral/2` predicate is the class to be predicted (i.e. the dependent variable). In our case, this is the second parameter, as specified on line 5 of Listing 5.7. The mode declarations for the body predicates are identical to those provided in Listing 5.1, but the determinations need to be modified to reference the `moral/2` predicate, as shown on line 8 of Listing 5.7. It is also necessary to provide the list of classes to be learned, which in our case are the morals (refer to search settings in Section 5.3.2).

```
% Declare predicate for head of clause for tree-learning
:- modeh(1,moral(+story,-moral)).

% Specify the 2nd parameter of moral/2 is to be predicated
:- set(dependent,2).

% Example determination referencing moral/2
:- determination(moral/2,joy/4).
```

Listing 5.7: Aleph mode declarations and determinations for tree-learning

The other significant difference is that tree-learning does not require a negative examples file (i.e. a .n file). This is because the structure of the positive examples file (with extension .f) is such that it provides both positive and negative examples. Unlike the format shown in Listing 5.6, this time we include ground instances of the `moral/2` predicate, which specifies the class each example belongs to. An excerpt of the file contents is shown in Listing 5.8. Both the background knowledge and examples files used in the tree-learning experiment are available online.[1]

```
moral(fable30,greed).
moral(fable90,greed).
...
moral(fable20,pride).
moral(fable29,pride).
...
moral(fable5,realistic).
moral(fable23,realistic).
...
moral(fable38,recklessness).
moral(fable40,recklessness).
...
moral(fable3,retribution).
moral(fable16,retribution).
...
moral(fable6,reward).
moral(fable206,reward).
...
```

Listing 5.8: Excerpt of Aleph examples file for tree-learning

[1]The Aleph files used for tree-learning are available at: `www.cse.unsw.edu.au/~msarlej/moss/aleph/treelearning`.

5.3.2 Aleph Search Settings

Listing 5.9 shows the search settings we use to learn a classification tree for morals. Line 2 sets the type of tree to a classification tree, and lines 5–6 define the classes to be learned (in our case, the morals). Line 9 specifies that at least 2 examples must be covered by a leaf node for refinement to proceed. Line 12 sets the evaluation function to entropy, and line 15 limits the maximum clause length to 10. Finally, line 19 forces the search to continue until all remaining elements in the search space are definitely worse than the current best element. All other settings are left as the Aleph defaults. Once the data files have been loaded into Aleph, we run the tree learner using the command:

```
induce_tree.
```

The output is a list of rules which describe a binary decision tree for classifying the morals (refer to Appendix C for examples of Aleph's output).

```
1  % Set the tree type to 'classification'
2  :- set(tree_type,classification).
3
4  % Specify the classes to be learned
5  :- set(classes,[greed,realistic,recklessness,
6                  retribution,reward,pride]).
7
8  % Set the minimum examples in a leaf for splitting
9  :- set(minpos,2).
10
11 % Set the evaluation function to 'entropy'
12 :- set(evalfn,entropy).
13
14 % Set the maximum clause length in the theory to 10
15 :- set(clauselength,10).
16
17 % Force the search to continue until all remaining elements in
18 % the search space are definitely worse than the current best
19 :- set(explore,true).
```

Listing 5.9: Aleph settings for learning a classification tree

5.3.3 Approach to Validation

Due to the small number of examples available, it was not feasible to separate the data into disjoint training and test sets. Instead, we use leave-one-out cross-validation to estimate the accuracy of our model, maximising the number of examples that can be used for training each fold. We partitioned the data into a training set of 36 examples, and a test set consisting of the one remaining example. This was repeated 37 times, with a different example left out of the training set each time, allowing us to estimate the overall accuracy of the classifier on unseen examples.

5.3.4 Experiment 1: Classifying Individual Morals

The first experiment we performed was to learn a classification tree for individual morals. In Figure 5.1, we present a diagram of the classification tree Aleph produced when all 37 fables were used for training (the rules output by Aleph, on which this diagram is based, are provided in Appendix C). We are able to prune this tree in two places, where both the YES and NO branches return the same moral.[2] The pruned version is provided in Figure 5.2. The numbers below the leaf nodes show how many fables of the specified class are correctly predicted by that branch of the tree. Note that 7 fables are not classified correctly by this tree (1 Greed, 1 Pride, 1 Realistic Expectations, 3 Recklessness, and 1 Reward), and thus the numbers on the diagram sum to only 30 out of 37 fables.

To estimate the error of the model due to overfitting, we performed leave-one-out cross-validation. The results are presented as a confusion matrix in Table 5.1. Overall, the performance of this classifier was poor. The best performance (in terms of both recall and precision) was for Retribution, which is not surprising given that this was also the moral with the most data available. It is likely that there were insufficient examples to learn useful rules for the other morals. It is also important to note that due to the relatively small number of examples across the board, the numeric performance measures cannot be considered highly reliable. Figure 5.3 shows the recall and precision percentages for each moral, along with the associated 95% confidence intervals, calculated using the Wilson *score* method [178]. It is clear from the graph that the margin of error is high.

[2]Aleph does provide a pruning option, but we found it impractical because the search did not return in a reasonable time-frame when this setting was used.

Figure 5.1: Classification tree for individual morals

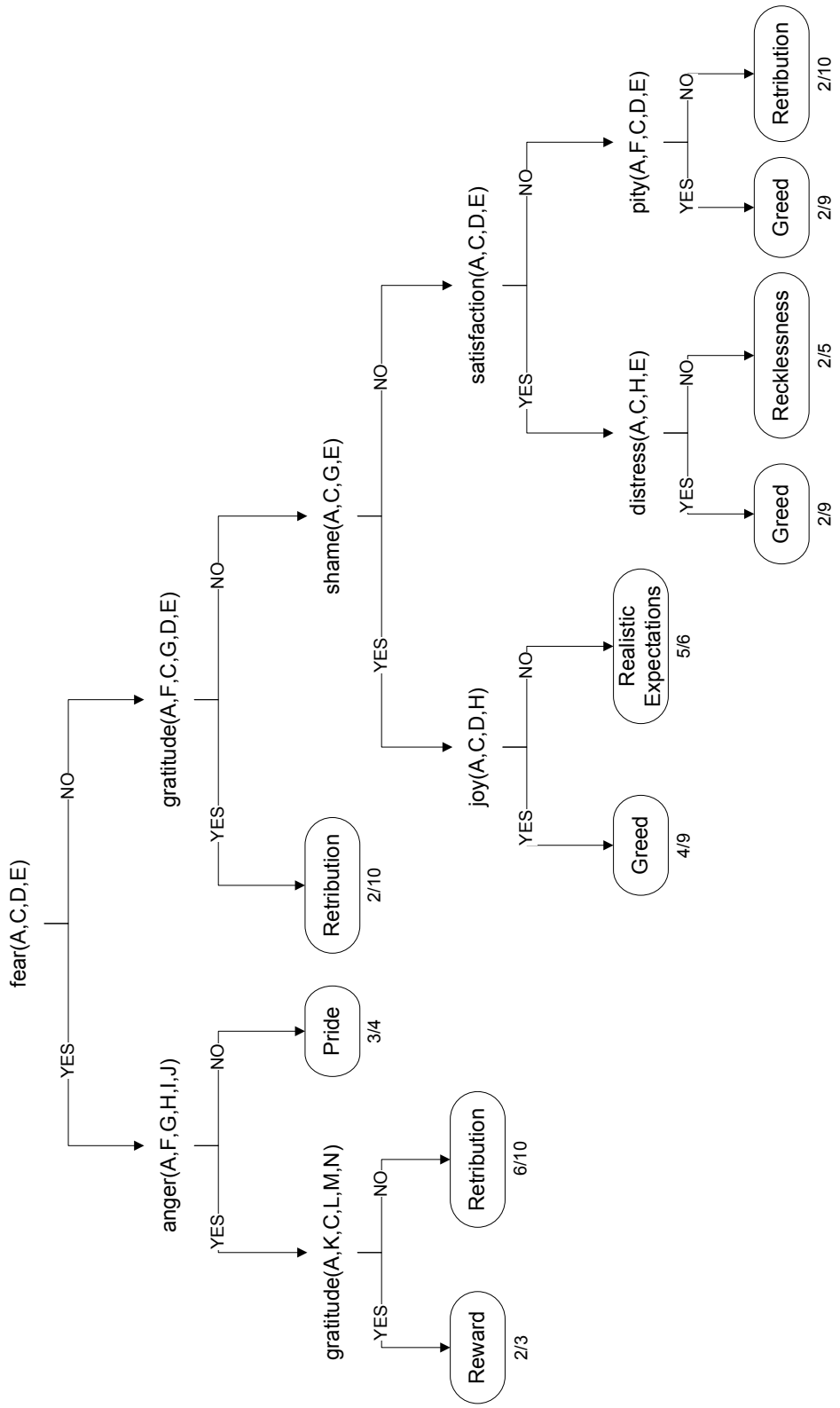

Figure 5.2: Pruned version of the moral classification tree from Figure 5.1

	Retr	Greed	Pride	Real Exp	Reck	Reward	RECALL
Retribution	**6**	2	0	1	1	0	60.0%
Greed	0	**5**	0	1	3	0	55.6%
Pride	1	1	**2**	0	0	0	50.0%
Real Exp	0	1	0	**2**	3	0	33.3%
Recklessness	0	3	0	2	**0**	0	0.0%
Reward	2	0	0	1	0	**0**	0.0%
PRECISION	66.7%	41.7%	100.0%	28.6%	0.0%	0.0%	

Table 5.1: Confusion matrix for individual morals (Experiment 1)

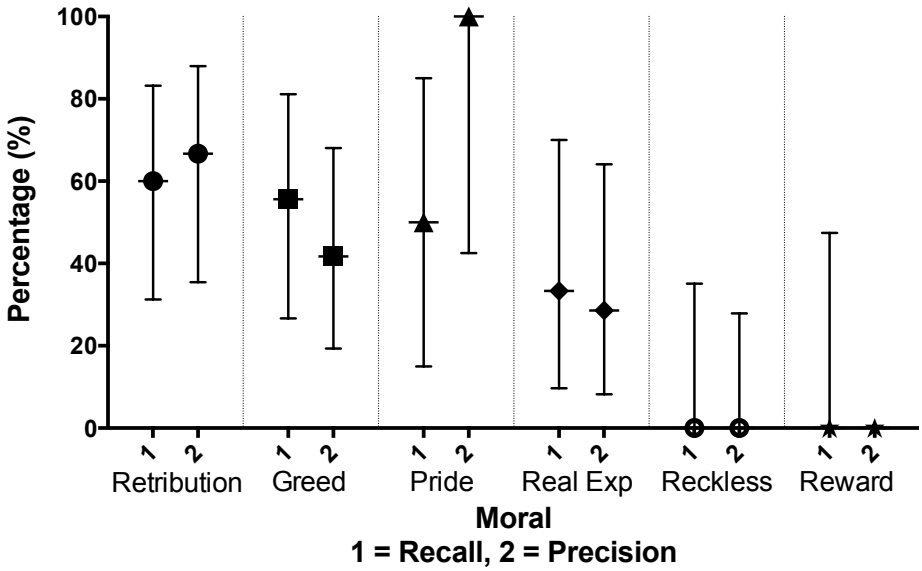

Figure 5.3: Recall and precision for individual morals (Experiment 1)

However, despite the poor accuracy of the classifier, the confusion matrix reveals informative patterns. Some morals were consistently misclassified as certain other morals. In Table 5.1, we have grouped these morals together, to make it easier to see the patterns in the matrix data. Retribution, Greed, and Pride were often confused, and similarly Realistic Expectations with Recklessness. This could indicate an inherent similarity between particular morals, which makes it difficult to separate them on a relatively small amount of data, or perhaps even based exclusively on emotions.

Intuitively, the similarity between Retribution, Greed, and Pride makes sense. To demonstrate that greedy and excessively proud behaviour is reprehensible, fables present the behaviour, followed by negative consequences for the character responsible, as punishment. This is precisely the idea behind Retribution: punishing a character for a reprehensible action. In fact, both Greed and Pride can be considered specific subclasses of Retribution. With regards to Realistic Expectations and Recklessness, both involve a character experiencing negative outcomes from their own action, whether it be failing to achieve a desirable outcome, or suffering some kind of harm. Similarities in emotion patterns would therefore also be expected here, and with the relatively small number of examples of each moral (six and five fables respectively), it is likely there was insufficient data to identify distinguishing features reliably. To investigate the veracity of these groupings, we repeat the tree-learning experiment in Section 5.3.5, this time learning moral groups.

5.3.5 Experiment 2: Learning Moral Groups

In this version of the tree-learning experiment, we use Aleph to learn a classification tree based on only three classes: `ret_greed_pride` (including Retribution, Greed, and Pride), `real_reck` (including Realistic Expectations and Recklessness), and `reward`. As such, we need to replace lines 5–6 from Listing 5.9 with line 2 from Listing 5.10. All other search settings remain the same, as described in Section 5.3.2. We take the same leave-one-out cross-validation approach to evaluation.

```
1 % Specify the classes to be learned
2 :- set(classes,[ret_greed_pride,real_reck,reward]).
```

Listing 5.10: Aleph classes for moral groups

Figure 5.4 shows the classification tree produced when all 37 fables were used for training; the corresponding Aleph rules are supplied in Appendix C. We observe a branch of the tree that can be pruned, and provide the pruned version in Figure 5.5. Again, the numbers below the leaf nodes show how many fables of the specified moral group were predicted correctly by that branch of the tree. However, in this case the numbers for Retribution/Greed/Pride add up to 24 out of 23 fables. This is because one fable, Fable 163, was predicted by two branches of the tree. This anomaly does not appear in Figure 5.5, as those two branches were merged during pruning. One example of Realistic Expectations/Recklessness is not predicted correctly, and thus the numbers in Figure 5.5 sum to 36 out of 37.

The cross-validation performance of this classifier was considerably better than the one based on individual morals. We present the relevant confusion matrix in Figure 5.2. This indicates that the proposed moral groupings are reasonable. The larger number of examples available for each class (with the exception of Reward) is also likely to have impacted the quality of the learning. It also reduced the predicted margin of error in the results, as evident from the confidence interval graph presented in Figure 5.6. As before, the 95% confidence intervals shown are calculated using the Wilson score method.

Figure 5.4: Classification tree for moral groups

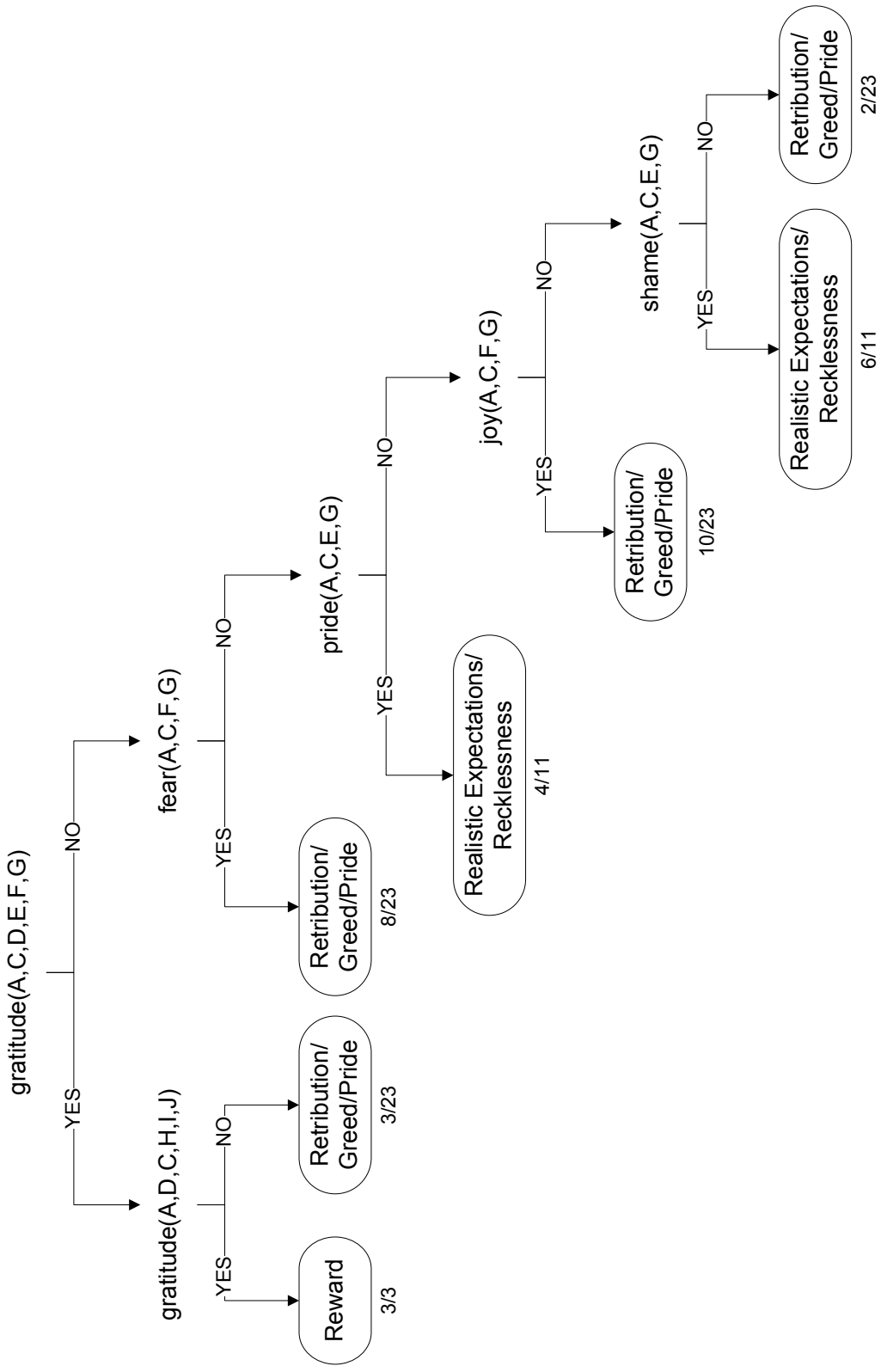

Figure 5.5: Pruned version of the moral group classification tree from Figure 5.4

	Ret/Greed/Pride	Real Exp/Reck	Reward	RECALL
Ret/Greed/Pride	**24**	2	0	91.3%
Real Exp/Recklessness	4	**7**	0	63.6%
Reward	0	1	**2**	66.7%
PRECISION	84.0%	70.0%	100.0%	

Table 5.2: Confusion matrix for moral groups (Experiment 2)

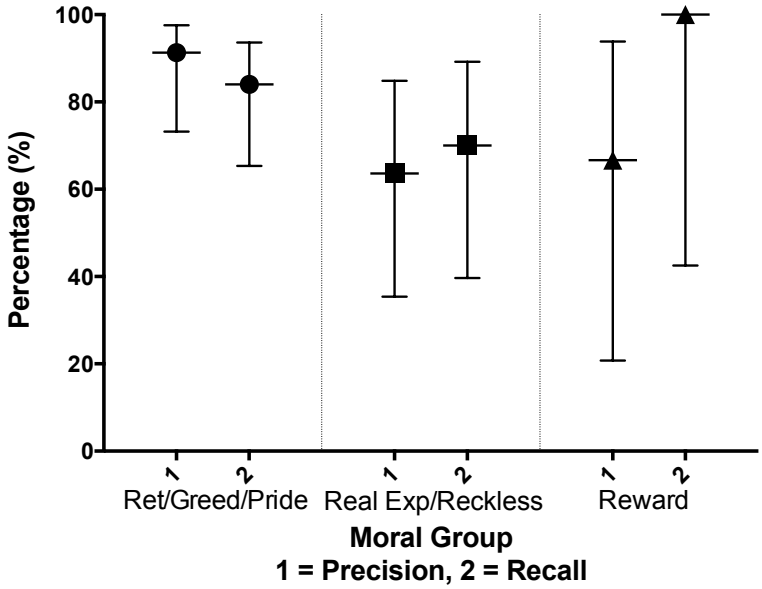

Figure 5.6: Recall and precision for moral groups (Experiment 2)

5.3.6 Conclusion

Our goal for the experiments described in this section was to establish whether there is a relationship between morals and character emotions, so we can exploit this for story generation. Due to the small number of examples available, Aleph was unable to reliably identify patterns of emotions corresponding to individual morals. We expect that a larger pool of examples would lead to significantly better results and more specific correlations. Nevertheless, a relationship between emotions and morals is evident from the results when similar morals are grouped. This also indicates an inherent similarity between certain morals, which should be taken into account when generating stories.

5.4 Learning Moral Rules

After using a tree-learning approach to substantiate the notion of a relationship between morals and emotions, we now apply Aleph's standard rule-based theory construction to learn rules for individual morals. The resulting rules cannot be utilised directly for story generation, due to inadequate performance, but many identify useful emotion patterns which we build on to develop the rules that will be presented in Section 5.5.

We use the same emotion data as for the experiments described in Section 5.3. The background knowledge is formatted as described in Section 5.2.2, without the modifications that were necessary for tree-learning. We do not, however, apply this approach to learning rules for moral groups. Our objective is to produce rules that can be used to generate stories with specific morals; learning rules for moral groups will not help achieve this. The search settings we use are listed in Section 5.4.1. We explain how we evaluate the moral rules produced by Aleph in Section 5.4.2, and present the results of the learning, including the rules that were generated for each moral, in Section 5.4.3. We assess the ILP approach in Section 5.4.4, and present our conclusions in Section 5.4.5.

5.4.1 Aleph Search Settings

We provide the search settings used for this approach in Listing 5.11. Setting `minpos` to 2 (line 2) ensures that any acceptable clause covers at least 2 examples, and thus prevents Aleph from adding ground unit clauses to the final theory. Line 5 sets the maximum clause length to 6, and line 8 allows up to 3 negative examples to be covered by an acceptable clause. We also increase the node limit for the search to 120,000 (line 11). Aleph's default is 5,000, but we found that no acceptable clauses were returned for some subsets of examples until this limit was significantly increased. Apart from the settings listed below, the rest are left as their defaults. Note in particular that the default evaluation function is coverage. This is calculated by P − N, where P is the number of positive examples covered, and N the number of negative examples covered. Once all files have been loaded into Aleph, we run the learner using the following command:

```
induce_cover.
```

```
1  % Avoid adding ground unit clauses to theory
2  :- set(minpos,2).
3
4  % Set maximum clause length to 6
5  :- set(clauselength,6).
6
7  % Max no. of negative examples that can be covered by a clause
8  :- set(noise,3).
9
10 % Increase node limit for search to 120,000
11 :- set(nodes,120000).
```

Listing 5.11: Aleph settings used for rule-learning

5.4.2 Approach to Validation

Our approach to validation is the same as for tree-learning; due to the small number of examples available, leave-one-out cross-validation is the only feasible method. Again, we repeat the learning 37 times, leaving out a different example as the test set each time. The main difference is that unlike the tree-learner, where a single classifier predicted all the morals, this time a separate classifier needs to be learned for each individual moral. Consequently, the cross-validation process is also performed separately for each moral classifier. For each example tested, the result will be one of only two possible outcomes: either the example is predicted as having that moral, or not.

5.4.3 Results

In this section, we summarise the results of the rule-learning experiment. First, we present the Aleph output for each moral when all 37 examples were used in the learning process. This includes the rules selected for the final theory, as well as a summary of the training set performance. Aleph's accuracy measurement is calculated as the sum of true positives (TP) and true negatives (TN), divided by the total number of examples (in our case, 37):

$$Accuracy = \frac{TP + TN}{37} \tag{5.1}$$

The caveat for these accuracy measurements is that Aleph runs the tests on the training set, which is not a reliable performance measure due to the risk of overfitting. We provide a summary of the cross-validation results, which are a better gauge of true

performance, at the end of this section. The rules produced by Aleph give no indication of the type of each variable, and hence we provide the mode declarations for all emotion predicates in Appendix B, to aid interpretation.

Retribution

Aleph's theory for Retribution consists of three rules, shown in Listing 5.12. The first is not very useful, and specific to our narrow data set; just because a story does not contain Hope does not necessarily mean it must convey Retribution. Rules 2 and 3 are more useful; we provide an interpretation for each below.

Rule 2: At time F, agent B feels Anger towards agent C, and agent C feels Joy. That is, C performed an action to cause themselves Joy, and B Distress. At some other time-point H, C feels Distress. Presumably, this is their punishment (note, however, there is no temporal relationship specified between time-points F and H, other than the fact they are different).

Rule 3: Agent B feels Joy, and agent E feels Gloating and Reproach towards B, all at different times. Again, temporal relationships are not specified, but Reproach causes Hate (refer to Section 4.5.6), which is a prerequisite for Gloating. Thus, the likely interpretation is that B performs an action which E disapproves of, making E Hate B. Later, B experiences Distress (their punishment for the reproachable action), about which E feels Gloating.

The performance of these rules on the training set is quite good. Rules 2 and 3 in particular each cover 7 of the 10 available examples of Retribution, and only 1 and 2 negative examples respectively.

```
[Rule 1] [Pos cover = 3 Neg cover = 0]
retribution(A) :-
   not hope(A,B,C,D).

[Rule 2] [Pos cover = 7 Neg cover = 1]
retribution(A) :-
   anger(A,B,C,D,E,F),
   distress(A,C,G,H),
   joy(A,C,E,F).

[Rule 3] [Pos cover = 7, Neg cover = 2]
retribution(A) :-
   joy(A,B,C,D),
   gloating(A,E,B,F,G),
   reproach(A,E,B,H,I).

[Training set performance]
            Actual
          +      -
      + 10       2     12
Pred
      -  0      25     25

         10      27     37

Accuracy = 0.945945945945946
```

Listing 5.12: Rules produced by Aleph for Retribution

Greed

The theory for Greed contains two rules. As evident from Listing 5.13, the accuracy is significantly worse than that for Retribution. The first rule covers 8 of the 9 examples of Greed, but also covers 8 negative examples. The second rule performs even worse, covering only 4 positive examples, but 13 negative examples. Overall, the accuracy on the training set is less than 60%; these rules do not effectively characterise the Greed stories appearing in the data set. Neither rule lends itself to an obvious interpretation in a narrative context.

```
1  [Rule 1] [Pos cover = 8 Neg cover = 8]
2  greed(A) :-
3      distress(A,B,C,D),
4      joy(A,B,E,F),
5      not pride(A,B,G,H),
6      not fear(A,B,I,H).
7
8  [Rule 2] [Pos cover = 4 Neg cover = 13]
9  greed(A) :-
10     pity(A,B,C,D,E),
11     not pride(A,C,F,E),
12     not fear(A,C,G,E).
13
14 [Training set performance]
15             Actual
16         +       -
17     + 9     15      24
18 Pred
19     - 0     13      13
20
21         9     28      37
22
23 Accuracy = 0.594594594594595
```

Listing 5.13: Rules produced by Aleph for Greed

Pride

The theory for Pride (Listing 5.14) appears to have a very high accuracy, but this is misleading, and not reflected at all in the cross-validation results. We discuss the reasons later (refer to *Cross-Validation Results* at the end of this section), but highlight for now that the theory consists of two rules, each of which covers only two examples (there were only four examples of Pride in total). Removing any of these for cross-validation significantly impacts the rules learned.

Rule 1: Agent B feels Hope while agent E feels Disappointment. In addition, B cannot feel Shame at any time during the story. It is not clear how this corresponds to the kinds of stories which convey the moral of Pride. Most likely it is an emotion pattern specific to the two examples covered, but not Pride stories more generally.

Rule 2: This rule is easier to interpret: agent B must feel Distress and Pride simultaneously, meaning they performed an action they were proud of, which caused

them negative consequences. However, this rule is also very specific. In general, the punishment for excessive Pride need not always occur as a direct result of the action evoking Pride.

```
1  [Rule 1]  [Pos cover = 2 Neg cover = 1]
2  pride(A) :-
3      hope(A,B,C,D),
4      disappointment(A,E,F,D),
5      not shame(A,B,G,H).
6
7  [Rule 2] [Pos cover = 2 Neg cover = 0]
8  pride(A) :-
9      distress(A,B,C,D),
10     pride(A,B,E,D).
11
12 [Training set performance]
13             Actual
14         +       -
15     + 4       1       5
16 Pred
17     - 0      32      32
18
19         4      33      37
20
21 Accuracy = 0.972972972972973
```

Listing 5.14: Rules produced by Aleph for Pride

Realistic Expectations

The theory for Realistic Expectations also includes two rules (Listing 5.15). As with Pride, Aleph's reported accuracy is high, but this is not reflected in cross-validation.

Rule 1: Agent B feels Resentment and Disappointment, but no Fear. Disappointment makes sense in this context, because stories conveying Realistic Expectations generally involve a character attempting something unrealistic, and failing. The need for Resentment is less clear; it may be specific to the two examples covered by this rule.

Rule 2: Agent B feels Shame during the story, but no Joy. Presumably, the Shame results from attempting an unrealistic action, but failing to achieve it (this matches the second rule for Shame in our emotion model, shown in Listing 4.25).

```
1  [Rule 1] [Pos cover = 2 Neg cover = 0]
2  realexp(A) :-
3     resentment(A,B,C,D,E),
4     disappointment(A,B,F,G),
5     not fear(A,H,I,J).
6
7  [Rule 2] [Pos cover = 5 Neg cover = 2]
8  realexp(A) :-
9     shame(A,B,C,D),
10    not joy(A,B,E,F).
11
12 [Training set performance]
13             Actual
14          +      -
15      + 6        2       8
16 Pred
17      - 0       29      29
18
19          6      31      37
20
21 Accuracy = 0.945945945945946
```

Listing 5.15: Rules produced by Aleph for Realistic Expectations

Recklessness

Only one acceptable rule is returned for Recklessness, shown in Listing 5.16, and it covers only two of the five examples. This indicates significant variation in the fables corresponding to Recklessness, at least in terms of emotion patterns. The repercussion is that when either of those two examples are removed for cross-validation, no acceptable rules are found, leading to extremely poor performance (refer to *Cross-Validation Results* for a more detailed discussion).

The rule for Recklessness includes a temporal relation, specifying that time B must occur after time C. An agent, D, must feel Gratification and no Fear at C, and later feel Distress at B. Given its extremely low coverage and poor cross-validation performance, this theory is not useful for story generation.

```
1  [Rule 3] [Pos cover = 2 Neg cover = 7]
2  recklessness(A) :-
3     after(B,C),
4     distress(A,D,E,B),
5     gratification(A,D,F,G,C),
6     not fear(A,D,G,C).
7
8  [Training set performance]
9              Actual
10           +      -
11      + 2       7       9
12 Pred
13      - 3      25      28
14
15          5      32      37
16
17 Accuracy = 0.72972972972973
```

Listing 5.16: Rules produced by Aleph for Recklessness

Reward

There is only one rule produced for Reward, shown in Listing 5.17, but this time the performance is extremely good, not only based on Aleph's reported accuracy, but also the cross-validation results. This may seem counterintuitive, given that Reward has the smallest number of examples available in the corpus (only three). The strong performance of the classifier indicates that the examples of Reward have consistent emotion patterns, which are distinctly different from the other morals, and therefore there is no confusion (we discuss this further in *Cross-Validation Results*).

The interpretation of the rule is straightforward: it is essentially reciprocal Gratitude. Agent B feels Gratitude towards agent C at some time-point F, and C feels Gratitude towards B at some other time-point I. It does not matter which occurs first, because the two instances of Gratitude are symmetrical. We base one of our ASP rules for generating Reward stories very closely on this rule (refer to Section 5.5.7).

```
[Rule 1] [Pos cover = 3 Neg cover = 0]
reward(A) :-
    gratitude(A,B,C,D,E,F),
    gratitude(A,C,B,G,H,I).

[Training set performance]
            Actual
          +       -
      + 3         0       3
Pred
      - 0        34      34

          3        34      37

Accuracy = 1.0
```

Listing 5.17: Rules produced by Aleph for Reward

Cross-Validation Results

To evaluate the accuracy of the moral rules produced by Aleph, we perform leave-one-out cross-validation for each moral. In Table 5.3, we present the precision, recall, false positive rate, and false negative rate for each learned classifier.[3] As with the tree-learning results, it is important to be wary of the reliability of these error rates, due to the small number of examples available. We show the 95% confidence intervals (based on the Wilson score interval) for these results graphically in Figures 5.7 and 5.8. Figure 5.7 shows recall and precision for each moral, and Figure 5.8 shows the false positive and false negative rates.

The only moral that performed well was Reward, despite having the smallest number of examples. This indicates that the emotion patterns present in Reward stories are distinctly different from the other five morals. Considering the nature of these morals (as defined in Table 3.3), this is perhaps not surprising. Reward is the only moral for which stories conclude with a positive outcome for the protagonist, and thus they would be expected to feel positive emotions. Conversely, the other morals require a negative outcome for the main character, to serve as a punishment or lesson. This similarity makes it harder to distinguish them reliably based on limited data.

[3]Note that for two folds of Recklessness (when Fables 40 and 68 were left out), Aleph did not find any acceptable clauses, even with the node limit set to 120,000. We treat these as classification errors for the purposes of data analysis.

	Precision	Recall	False Positive Rate	False Negative Rate
Retribution	54.5%	60.0%	18.5%	40.0%
Greed	33.3%	88.9%	57.1%	11.1%
Pride	0.0%	0.0%	15.2%	100.0%
Real Exp	20.0%	16.7%	12.9%	83.3%
Recklessness	0.0%	0.0%	28.1%	100.0%
Reward	100.0%	100.0%	0.0%	0.0%

Table 5.3: Cross-validation results for moral rules

Pride and Recklessness performed extremely poorly, with no examples correctly retrieved by the corresponding classifiers. In the case of Pride, although the theory produced by training on all 37 examples appeared to perform well (100% recall, and covering only one negative example), it consisted of two separate rules, each covering only two examples of Pride. Thus, when one example was removed as part of the cross-validation, the rule which would have covered that example was not added to the theory, because it would only cover one example (as explained in Section 5.4.1, we set the minimum threshold for an acceptable clause to be a coverage of two). This is a very clear demonstration of the problems that can arise from insufficient examples for learning. For Recklessness, the best rule that could be found, even using all examples for training, only had a coverage of two. Removing either of those examples resulted in no acceptable rule being found (these were the folds corresponding to Fables 40 and 68 being left out as the test set).

For most morals (Greed is the exception), the false negative rate was considerably higher than the false positive rate. In the cases of Pride, Realistic Expectations, and Recklessness, the difference is significant based on 95% confidence intervals, despite the size of the error bars (refer to Figure 5.8). This shows that those classifiers rarely retrieve stories with a different moral, but often miss stories with the matching moral. This likely results from a combination of two factors. Firstly, the training set for each classifier contains substantially more negative examples (i.e. fables with all the other morals) than positive examples (i.e. fables with that moral). Consequently, removing one negative example from the training set for testing has minimal impact on the resulting rules, in contrast to removing a positive example, which has considerable impact. Secondly, there may be significant variation within the available examples of particular morals, which makes it difficult to identify robust relationships when the number of examples is so small.

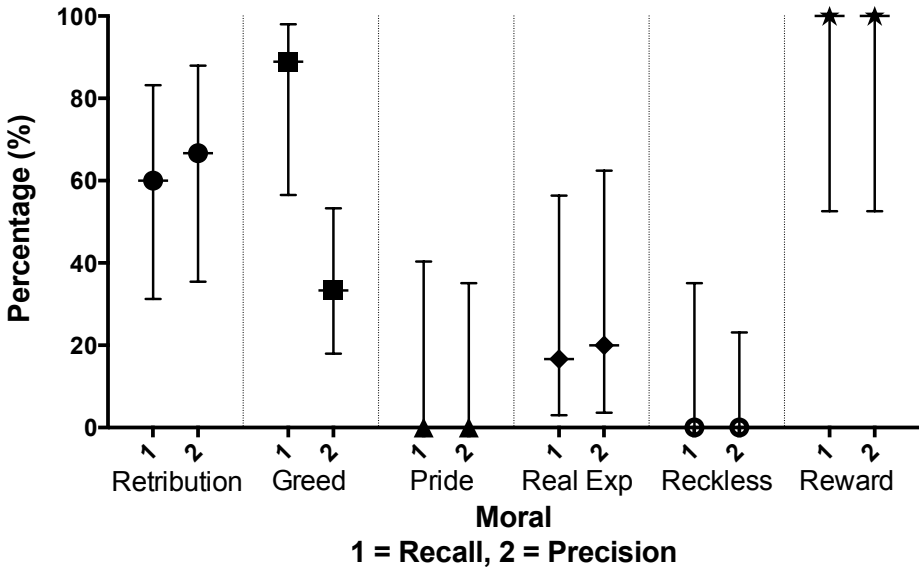

Figure 5.7: Recall and precision by moral (rule-learning)

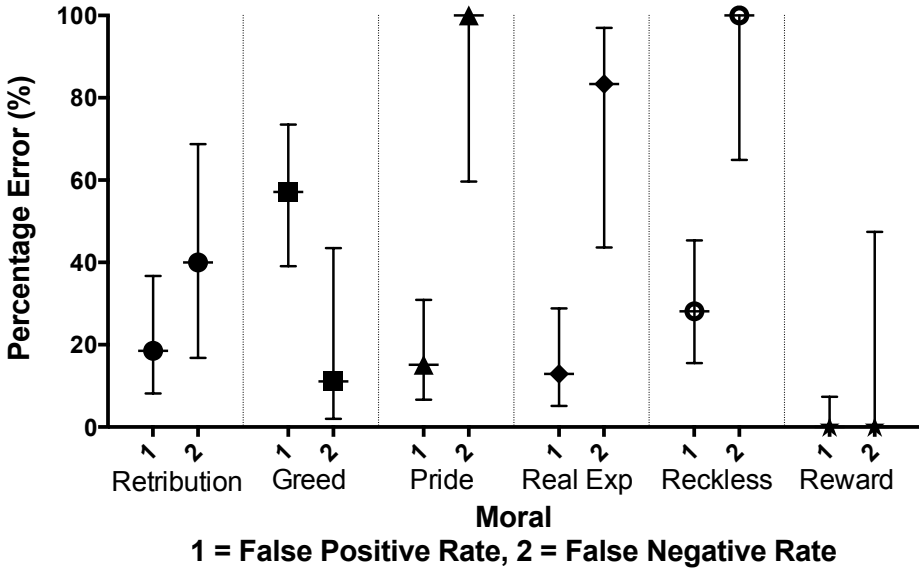

Figure 5.8: False positive and false negative rates by moral (rule-learning)

5.4.4 Discussion

As evidenced by the generally poor cross-validation results of our ILP experiments, there are a number of shortcomings with our use of this approach. Arguably the most important is the size of the data set used for learning. Our corpus consisted of a total of only 37 fables, which, when divided across 6 morals, only provides an average of 6.17 examples per moral. In reality, it was less than this in most cases, due to the uneven distribution of fables amongst the morals. Given the complexity of character emotion, and the large number of emotions appearing in even simple stories, this is insufficient for an ILP system to learn robust relationships. However, there are many other collections of fables available (some were listed in Section 3.4). Building a corpus of emotion data combining several collections would yield a much greater number of examples per moral, and likely lead to far better rules.

The background knowledge provided to Aleph also has a significant impact on the theory that is constructed, not only in terms of the emotion data provided, but the definition of any additional predicates which expose relationships in this data. In our case, we only defined very basic temporal relationships: two predicates, `after(T1,T2)` and `after_or_equal(T1,T2)`. Defining a broader range of additional predicates, whether they be more specific temporal relationships (for example, whether two time-points are consecutive), or other relationships entirely, would give Aleph more expressivity to work with. Of course, this would also increase the search space, so there is no guarantee better solutions would be found. Nevertheless, it is an area worth investigating further.

Finally, Aleph offers a large number of search options and parameters that can be tweaked. Due to the size of the search space (even with the limited data available), in most cases the search will not cover every possibility, and thus the choice of these parameters (in particular, search algorithms) will impact the rules that are constructed. In our case, because ILP was not the focus of our work, we did not undertake a thorough comparison of all the different options available. It is possible that using different search algorithms (for example, randomised search methods) could uncover better rules. This is certainly an area open for future work, not only using our limited corpus, but also with an extended data set incorporating other fable collections.

5.4.5 Conclusion

In this section, we described our use of inductive logic programming to automatically learn useful relationships between selected morals and OCC emotions, based on data produced from Aesop's fables. Our results support the notion that morals and emotions are related, and also reveal similarities between certain morals in terms of the patterns of emotions the corresponding stories contain. However, the performance of the specific rules produced by Aleph was not adequate for them to be applied directly to story generation. To fully utilise the potential of ILP, we would need to provide significantly more data; this would inevitably lead to better rules. The approach itself shows promise, and it is not inconceivable that, with a large enough corpus, rules could be produced of sufficient quality to be fed directly into a story generation system. In our case, ILP was not the primary focus of our work, but rather a stepping stone towards our goal of generating stories with morals. Therefore we do not pursue further improvements in the learning process, but use the rules presented in Section 5.4.3 as a starting point, which we build upon to create ASP rules for each moral that can be used for story generation. We present these rules in Section 5.5.

5.5 The Moral Layer

Having presented our system's underlying framework in Chapter 4 (the Action, Belief, and Emotion Layers), we now move on to describe the Moral Layer. As discussed in Section 5.4, the moral rules obtained using ILP were of varying quality, and in most cases not suitable for direct implementation in a story generation system. We use the rules produced by Aleph as a starting point, and refine them through manual analysis of the corresponding fables. In this section, we provide ASP rules for each of the six morals that were selected in Chapter 3. We begin by describing the helper predicates used in our moral definitions, in Section 5.5.1. We then present the moral rules in Sections 5.5.2 (Retribution), 5.5.3 (Greed), 5.5.4 (Pride), 5.5.5 (Realistic Expectations), 5.5.6 (Recklessness), and 5.5.7 (Reward). Finally, in Section 5.5.8 we explain the parameter-free predicates we define for each of these moral rules, used for specifying moral constraints on stories.

We make two further comments regarding the ASP rules provided in Sections 5.5.2–5.5.7 before we present them. Firstly, each rule includes a predicate that has not been discussed previously: `maxtime(T)`. For any given story domain, this predicate is defined

as the maximum allowable time-point for emotions (not actions). It is defined to be the largest valid time value, plus 1 (since emotions caused by the last action in the story will arise in the subsequent time-point). This allows us to specify that the final emotion (or emotions) in a moral's definition must occur at the end of a story. This is important in helping readers recognise that moral as the story's point. If other superfluous events continue to happen afterwards, the reader's focus changes, and the story's moral can become lost. Secondly, as was the case for the emotion rules defined in Chapter 4, the time-points in the moral rules are defined with reference to $T + 1$, where T is a valid time value, to allow emotions to arise in the time-point following the last event.

5.5.1 Helper Predicates

Most of the moral rules we define in the following sections involve a number of different emotions. The specific events or consequences to which these emotions relate are irrelevant to the definitions, but nevertheless introduce a large number of variables into the rules if emotion predicates are used directly. Each emotion appearing in a rule relates to either an event, a consequence, or both. Generally, the event and/or consequence does not have to be the same between emotions, and thus each requires a separate variable. For instance, if a moral depends on Joy (consequence), Anger (event and consequence), and Gratitude (event and consequence), the corresponding rule would require two event variables and three consequence variables. This impacts the time it takes for the solver to generate solutions, because all the variables must be grounded.

To avoid this variable explosion, we introduce a number of helper predicates, which abstract away superfluous variables. Using them in our moral rules instead of the corresponding emotion predicates significantly reduced the time required by the ASP solver to find a solution. We do not include the ASP rules for the exhaustive set of helper predicates here, but show one example of each type, to illustrate how they are defined and explain their interpretation, so this is clear when we reference them in the moral rules. The full source code of the Moral Layer, including the helper predicates, is available online.[4]

[4]The Moral Layer source code is available at: `www.cse.unsw.edu.au/~msarlej/moss/system/storyplanner/morals.lp`.

feels_emotion(Agent1,Agent2,T)

The `feels_emotion(Agent1,Agent2,T)` predicate is interpreted as: `Agent1` feels `emotion` towards `Agent2` at time `T`. The specific event and consequence the emotion relates to are irrelevant. Listing 5.18 shows the rule for `feels_anger`. We define analogous helper predicates for Gratitude and Reproach, in the same manner.

```
1 feels_anger(Agent1,Agent2,T+1) :-
2     agent(Agent1),
3     agent(Agent2),
4     time(T),
5     action(Action),
6     consequence(Consequence),
7     Agent1 != Agent2,
8     anger(Agent1,Agent2,Action,Consequence,T+1).
```

Listing 5.18: ASP helper predicate feels_anger

some_emotion(Agent,T)

The `some_emotion(Agent,T)` predicate means: `Agent` feels `emotion` at time `T`. Not only are events and consequences irrelevant here, but the agent the emotion is directed towards (in the case of agent-based and compound emotions) is also abstracted away. This predicate is used in situations where an agent needs to feel a particular emotion at a certain time, but it does not matter what it is about or whom it is felt towards. The definition of `some_reproach` is provided in Listing 5.19. We implement analogous predicates for the following emotions: Joy, Distress, Hope, Fear, Satisfaction, FearsConfirmed, Disappointment, Relief, Pride, Shame, Admiration, and Remorse.

```
1 some_reproach(Agent1,T+1) :-
2     agent(Agent1),
3     agent(Agent2),
4     time(T),
5     action(Action),
6     reproach(Agent1,Agent2,Action,T+1).
```

Listing 5.19: ASP helper predicate some_reproach

some_emotion_after(Agent,T)

This helper predicate achieves more than simply abstracting away extraneous variables. `some_emotion_after(Agent,T)` means: `Agent` feels `emotion` at some time-point later than `T`. This predicate features in some of the moral rules in negated form, i.e. `not some_emotion_after(Agent,T)`, which is interpreted as: `Agent` will not feel `emotion` at any time-point in the story after `T`. We implement this predicate for Joy and Distress; Listing 5.20 shows `some_joy_after` as an example.

```
1 some_joy_after(Agent,T) :-
2     agent(Agent),
3     time(T),
4     time(TLater),
5     consequence(Consequence),
6     TLater+1 > T
7     joy(Agent,Consequence,TLater+1).
```

Listing 5.20: ASP helper predicate some_joy_after

emotion_other_than(Agent,T)

The `emotion_other_than` predicate is also defined primarily for achieving its negation. `emotion_other_than(Agent,T)` means: some agent other than (i.e. different from) `Agent` feels `emotion` at time `T`. The negation, `not emotion_other_than(Agent,T)`, is interpreted as: no agent apart from `Agent` feels `emotion` at time `T`. We implement this predicate only for Distress, as shown in Listing 5.21.

```
1 distress_other_than(Agent1,T+1) :-
2     agent(Agent1),
3     agent(Agent2),
4     time(T),
5     consequence(Consequence),
6     Agent1 != Agent2,
7     distress(Agent2,Consequence,T+1).
```

Listing 5.21: ASP helper predicate distress_other_than

any_emotion(Agent)

The predicate `any_emotion(Agent)`, which we implement only for Satisfaction, is interpreted as: `Agent` feels `emotion` at some time during the story. In this case, the specific time-point is irrelevant. This predicate is also defined for use in negated form. The negated version, `not any_emotion(Agent)`, means: `Agent` does not feel `emotion` at any time in the story. The ASP rule for `any_satisfaction` is provided in Listing 5.22.

```
1 any_satisfaction(Agent) :-
2     agent(Agent),
3     time(T),
4     consequence(Consequence),
5     satisfaction(Agent,Consequence,T+1).
```

Listing 5.22: ASP helper predicate any_satisfaction

5.5.2 Retribution

Of the rules produced by Aleph for Retribution, Rule 2 (refer to Listing 5.12) best captures the essence of this moral: if you make someone angry, you will later experience distress yourself. Our careful inspection of the Retribution fables revealed two clearly discernible types of Retribution stories, and although both conformed to this rule, one had a stricter requirement. The two types can be described as follows:

1. **Type 1:** Retribution can come from anywhere (it does not have to be brought about by the original victim).

2. **Type 2:** The original victim carries out the retribution (this is a stricter requirement than the rule produced by Aleph).

Although both types of stories could be generated using a single rule corresponding to Type 1 above, we implement two separate rules for Retribution. This allows us to investigate whether the moral is conveyed better by the more specific type. We describe both rules in this section. Before launching into our definitions, it is also important to clarify that we do not claim all possible Retribution stories can be represented by our rules (or even purely using emotions). We do not aim to devise rules that can generate all possible Retribution stories; we endeavour to define rules such that the stories they generate convey Retribution. This applies to all the other morals as well.

Retribution Type 1

Retribution Type 1 is the more general form of Retribution. In fact, it encompasses within its definition stories generated using the rule for Type 2. We provide the ASP rule in Listing 5.23. The crux of the definition, which corresponds directly to the rule generated by Aleph, is on lines 9–10: `Protagonist` performs an action to make `Victim` feel Anger, and later in the story `Protagonist` experiences Distress (this is their punishment for causing harm to `Victim`). We provide an English description of each component of the rule, along with an explanation for why it is required, in Table 5.4.

```
1  % RETRIBUTION - TYPE 1
2  % (note this includes stories generated by t2)
3
4  retribution_t1(Victim,T1+1,Protagonist,T2+1) :-
5      agent(Protagonist),
6      agent(Victim),
7      time(T1),
8      time(T2),
9      feels_anger(Victim,Protagonist,T1+1),
10     some_distress(Protagonist,T2+1),
11     not some_joy_after(Protagonist,T1+1),
12     not some_disappointment(Protagonist,T2+1),
13     some_reproach(Protagonist,T2+1),
14     not some_distress(Victim,T2+1),
15     Protagonist != Victim,
16     T1 < T2,
17     maxtime(T2+1).
```

Listing 5.23: ASP rule for Retribution Type 1

Time[5]	Line	Description	Explanation
T1	9	Protagonist does something to make Victim feel Anger.	This is the inciting incident, for which Retribution will occur.
	11	Protagonist does not feel any Joy after this point.	This is to ensure the sense of Retribution is not dulled by positive outcomes for Protagonist.
T2	10	Protagonist feels Distress.	This is Protagonist's punishment for making Victim angry.
	12	Protagonist does not feel Disappointment.	This is to prevent the Distress Protagonist experiences (line 10) from resulting from a failed action; the sense of punishment is stronger when it is not accidental or unintended.
	13	Protagonist feels Reproach towards some other agent (not necessarily Victim).	This is to ensure the Distress felt by Protagonist results from another agent's intentional action, as opposed to a natural event (e.g. it rains). This better conveys a sense of punishment.
	14	Victim does not feel any Distress.	Whatever happens to cause Protagonist Distress should be targeted at them; if Victim feels Distress as well, there is no sense of Retribution.

Table 5.4: Explanation of Retribution Type 1

[5]In the table, we refer to the time-points as T1 and T2, whereas in Listing 5.23 they are referenced as T1+1 and T2+1. As explained in Chapter 4, the +1 in the ASP rule is required because emotions arise in the time-step after the event that caused them. The last event in a story occurs at the last valid time-point, and thus any resulting emotions will arise outside the range defined for the `time` predicate. Only the relationship between time-points, not their specific value, is relevant to our explanations, so we discard the +1 in the table (i.e. T1 corresponds to T1+1 in the ASP rule, and T2 corresponds to T2+1). This is the case for all explanatory tables provided in this section.

Retribution Type 2

Retribution Type 2 is more specific than Type 1. This time, reciprocal Anger is a requirement: `Protagonist` performs an action to make `Victim` feel Anger, and then `Victim` performs an action to make `Protagonist` feel Anger in return. In this case, `Protagonist`'s punishment is caused directly by `Victim`. This form of Retribution could more specifically be called payback. The ASP rule is shown in Listing 5.24, and we explain the conditions comprising the rule in Table 5.5.

```
1  % RETRIBUTION - TYPE 2
2  % (note this is a specific subset of t1 stories)
3
4  retribution_t2(Victim,T1+1,Protagonist,T2+1) :-
5      agent(Protagonist),
6      agent(Victim),
7      time(T1),
8      time(T2),
9      feels_anger(Victim,Protagonist,T1+1),
10     feels_anger(Protagonist,Victim,T2+1),
11     not some_distress(Victim,T2+1),
12     Protagonist != Victim,
13     T1 < T2,
14     maxtime(T2+1).
```

Listing 5.24: ASP rule for Retribution Type 2

Time	Line	Description	Explanation
T1	9	Protagonist does something to make Victim feel Anger.	This is the inciting incident, for which Retribution will occur.
T2	10	Victim does something to make Protagonist feel Anger.	This is the payback for Protagonist's earlier action. Unlike in Retribution Type 1, it must be carried out by Victim.
	11	Victim does not feel any Distress.	Whatever Victim does to cause Protagonist Distress should be targeted at Protagonist; if Victim feels Distress as well, there is no sense of Retribution.

Table 5.5: Explanation of Retribution Type 2

5.5.3 Greed

Our analysis of the Greed fables also brought to light two distinct ways in which the moral was manifested:

1. **Type 1:** An agent gets what they desire, but also experiences Distress.
2. **Type 2:** An agent fails to get what they desire, and also experiences additional Distress.

Greed Type 1 has some similarities to the first rule produced by Aleph for Greed (refer to Listing 5.13), in which an agent must feel Joy and Distress simultaneously. However, the poor accuracy of the rule shows this emotion pattern is common to stories with other morals as well. As such, although we do provide two different rules for Greed, we make them more specific, to avoid such significant overlap with other morals.

Greed Type 1

Greed Type 1 is modelled on the kind of story in which a character behaves in a greedy fashion to achieve an outcome, only to later have that outcome reversed as punishment. A concrete example might be a schoolyard bully taking all the other children's lunches, only to have them stolen by a stray dog when he is not looking. The ASP rule we define is shown in Listing 5.25. We provide a line-by-line explanation in Table 5.6.

```
1  % GREED - TYPE 1
2
3  greed_t1(Protagonist,T1+1,T2+1) :-
4      agent(Protagonist),
5      agent(OtherAgent),
6      time(T1),
7      time(T2),
8      negate_consequence(Consequence,Negation),
9      satisfaction(Protagonist,Consequence,T1+1),
10     not some_distress(Protagonist,T1+1),
11     feels_reproach(OtherAgent,Protagonist,T1+1),
12     some_shame(Protagonist,T1+1),
13     distress(Protagonist,Negation,T2+1),
14     not distress_other_than(Protagonist,T2+1),
15     T1 < T2,
16     maxtime(T2+1).
```

Listing 5.25: ASP rule for Greed Type 1

Time	Line	Description	Explanation
T1	9	Protagonist feels Satisfaction about some outcome.	Protagonist achieves something they desired, through a greedy action. Satisfaction is used rather than Joy to highlight this is something Protagonist hoped for previously (Hope does not need to be explicitly included in the rule, as it is a prerequisite for Satisfaction).
	10	Protagonist does not feel any Distress.	This ensures all direct consequences of the preceding event were positive for Protagonist.
	11	Some other agent feels Reproach towards Protagonist.	The action Protagonist performed in the preceding time-step was against the standards of some other agent (generally, a greedy action is considered unideal).
	12	Protagonist feels Shame about the preceding action.	The action Protagonist performed was against their own standards (even though the outcomes were positive for them). Often the agent performing a greedy action will recognise it is unideal, even though they perform the action anyway.
T2	13	Protagonist feels Distress about the reversal of the outcome that satisfied them at T1.	Whatever was achieved through the original greedy action has been undone.
	14	No agent other than Protagonist feels Distress at this time.	Only Protagonist should be punished.

Table 5.6: Explanation of Greed Type 1

Greed Type 2

The alternative definition of Greed, Type 2, shows greedy behaviour through a character performing two shameful actions in sequence, each resulting in Satisfaction. These actions are consecutive, to emphasise the agent's greed. We show the corresponding ASP rule in Listing 5.26. Line 18 specifies that time-points T1 and T2 are consecutive: `T2 = T1 + 1`. Table 5.7 provides a detailed explanation of the ASP rule.

```
1  % GREED - TYPE 2
2
3  greed_t2(Protagonist,T1+1,T2+1,T3+1) :-
4      agent(Protagonist),
5      time(T1),
6      time(T2),
7      time(T3),
8      consequence(Consequence),
9      succeeded_consequence(Consequence2),
10     satisfaction(Protagonist,Consequence,T1+1),
11     some_shame(Protagonist,T1+1),
12     some_satisfaction(Protagonist,T2+1),
13     some_shame(Protagonist,T2+1),
14     distress(Protagonist,Consequence2,T3+1),
15     not distress_other_than(Protagonist,T3+1),
16     not some_joy(Protagonist,T3+1),
17     some_reproach(Protagonist,T3+1),
18     T2 = T1+1,
19     T2 < T3,
20     maxtime(T3+1).
```

Listing 5.26: ASP rule for Greed Type 2

Time	Line	Description	Explanation
T1	10	Protagonist feels Satisfaction about some outcome.	Protagonist achieves something they desired, through a greedy action. Satisfaction is used rather than Joy to highlight this is something Protagonist hoped for previously (Hope does not need to be explicitly included in the rule, as it is a prerequisite for Satisfaction).
	11	Protagonist feels Shame about the preceding action.	The action Protagonist performed to achieve the desirable outcome was against their own standards (a greedy action is generally considered unideal, and the agent performing the action often recognises this).
T2	12	Protagonist feels Satisfaction about another outcome.	Protagonist achieves something else they desired, through a second consecutive greedy action.
	13	Protagonist feels Shame about the preceding action.	The action Protagonist performed to achieve the second desirable outcome was also against their own standards.
T3	14	Protagonist feels Distress.	This is the punishment for Protagonist's greedy behaviour.
	15	No agent other than Protagonist feels Distress.	The punishment should be directed at Protagonist alone. If other agents feel Distress as a result of the same event, it dulls the sense of punishment.
	16	Protagonist does not feel any Joy.	If Protagonist feels Joy as well as Distress, it dulls the sense of punishment.
	17	Protagonist feels Reproach towards some other agent.	Protagonist's Distress should result from another agent's intentional action, not their own action, or unintended events.

Table 5.7: Explanation of Greed Type 2

5.5.4 Pride

As explained in Section 5.4.3, the rules produced by Aleph for Pride were not very useful, and had to be discarded. We base our definition on a manual analysis of the relevant fables. Put simply, the storyline we attempt to capture involves an agent feeling excessively proud of an action they performed, only to have the desirable outcome of

that action reversed as punishment for their pride. We define one rule for Pride, shown in Listing 5.27. Table 5.8 explains the emotion conditions in the rule.

We include an additional restriction, on lines 15–19, that stories generated for Pride cannot contain the emotion sequence defined for Retribution Type 2. Some Retribution Type 2 stories also satisfy the conditions for Pride, and we wish to avoid generating them here. This is not necessarily to say stories with those emotion sequences cannot convey Pride; rather, we include this restriction so when we evaluate stories generated for Pride, we can be sure none of them are also Retribution Type 2 stories. Otherwise, there would be uncertainty about which emotion pattern caused readers to recognise a particular moral. For example, if a story intended to convey Pride was judged by a reader to convey Retribution, there would be no way to determine whether it was due to the Pride emotions, or the fact that the Retribution emotions were also present.

```
1  % PRIDE
2
3  pride(Protagonist,T1+1,T2+1) :-
4      agent(Protagonist),
5      time(T1),
6      time(T2),
7      negate_consequence(Consequence,Negation),
8      some_pride(Protagonist,T1+1),
9      joy(Protagonist,Consequence,T1+1),
10     distress(Protagonist,Negation,T2+1),
11     not distress_other_than(Protagonist,T2+1),
12     some_reproach(Protagonist,T2+1),
13     T1 < T2,
14     maxtime(T2+1),
15     not retribution_t2(A2,Time1+1,A1,Time2+1),
16     agent(A2),
17     agent(A1),
18     time(Time1),
19     time(Time2).
```

Listing 5.27: ASP rule for Pride

Time	Line	Description	Explanation
T1	8	Protagonist feels Pride about an action they performed.	This is the excessive pride for which Protagonist will be punished.
	9	Protagonist feels Joy about an outcome of the action they performed.	Protagonist experiences a positive outcome as a result of the action they are proud of.
T2	10	Protagonist feels Distress about the reversal of the preceding positive outcome.	The positive outcome of Protagonist's initial action is undone; this is their punishment for excessive pride.
	11	No agent other than Protagonist feels Distress.	If another agent also experiences Distress, it would be less clear that this is a punishment for Protagonist.
	12	Protagonist feels Reproach towards some other agent.	This ensures the Distress Protagonist feels results from another agent's action rather than their own, producing a stronger sense of punishment.

Table 5.8: Explanation of Pride

5.5.5 Realistic Expectations

The general idea encapsulated by this moral is that a character wants to achieve something which is unrealistic, and fails in the attempt. Thus, Hope and Disappointment are key (although Hope does not have to feature explicitly in the rule, because it is a prerequisite for Disappointment). Disappointment does feature in one of the rules Aleph produced for Realistic Expectations (refer to Listing 5.15), but the generally poor performance of Aleph's theory when evaluated using cross-validation prompted us to discard those rules and start afresh. We provide a single definition for Realistic Expectations, presented in Listing 5.28, along with an explanation in Table 5.9.

```
1  % REALISTIC EXPECTATIONS
2
3  realistic_expectations(Protagonist,T2+1) :-
4      agent(Protagonist),
5      time(T1),
6      time(T2),
7      reverse_consequence(Consequence,FailedConsequence),
8      not some_joy(Protagonist,T1),
9      disappointment(Protagonist,FailedConsequence,T2+1),
10     not any_satisfaction(Protagonist),
11     T1 < T2+1,
12     maxtime(T2+1).
```

Listing 5.28: ASP rule for Realistic Expectations

Time	Line	Description	Explanation
T1	8	Protagonist does not feel any Joy.	This is to keep the focus on the Disappointment experienced at T2.
T2	9	Protagonist feels Disappointment about their failure to achieve some outcome.	Protagonist hoped for something unrealistic, and thus feels Disappointment when unable to achieve it. Hope does not need to be explicitly included in the rule, as it is a prerequisite for Disappointment.
ALL	10	Protagonist does not feel Satisfaction at any time during the story.	This is to ensure the action which failed (and caused Disappointment) does not succeed at any other time during the story, which would undermine the notion of it being unrealistic.

Table 5.9: Explanation of Realistic Expectations

5.5.6 Recklessness

This moral is conveyed through a reckless action resulting in negative consequences. In general, these consequences do not have to be immediate. However, as identified in Section 4.6.5, one of the limitations of our emotion model is the inability to trace causality back in time; i.e. only the direct consequences of an event can be attributed to that event. This means we can only generate stories in which reckless actions have direct negative outcomes. The rule Aleph produced for Recklessness (presented in Listing 5.16) performed too poorly to be used, and thus we base our rules on a manual inspection of the Recklessness fables. We identified two ways in which Recklessness was conveyed within the corpus:

1. **Type 1:** An agent performs an action with an intended positive outcome, but the action also has negative effects, which the agent did not consider.

2. **Type 2:** An agent anticipates a potential negative outcome of an action, but (recklessly) performs the action anyway.

We develop separate rules corresponding to each of these story types.

Recklessness Type 1

Recklessness Type 1 (Listing 5.29) is based on the idea that a reckless action is carried out with some positive intended outcome, but also has negative effects. Often, the agent involved did not consider the possible negative effects when deciding to perform the action, thus making the action reckless (a reckless action is one that is careless or ill-considered [5]). We explain how this is expressed in terms of emotions in Table 5.10.

```
1  % RECKLESSNESS - TYPE 1
2
3  recklessness_t1(Protagonist,T+1) :-
4      time(T),
5      agent(Protagonist),
6      some_satisfaction(Protagonist,T+1),
7      some_distress(Protagonist,T+1),
8      not some_admiration(Protagonist,T+1),
9      not some_reproach(Protagonist,T+1),
10     maxtime(T+1).
```

Listing 5.29: ASP rule for Recklessness Type 1

Time	Line	Description	Explanation
T	6	Protagonist feels Satisfaction about an outcome of the preceding action.	This relates to the desired positive outcome Protagonist intended to achieve with their action.
	7	Protagonist feels Distress about an outcome of the preceding action.	This is due to the unintended negative outcome resulting from Protagonist's action.
	8	Protagonist does not feel any Admiration.	This ensures the preceding action was performed by Protagonist, not some other agent.
	9	Protagonist does not feel any Reproach.	This ensures the preceding action was performed by Protagonist, not some other agent.

Table 5.10: Explanation of Recklessness Type 1

Recklessness Type 2

Recklessness Type 2 presents a different perspective on the notion of a reckless action. This time, an agent anticipates a possible negative outcome of an action, but performs the action anyway, hence demonstrating recklessness (i.e. carelessness). The ASP rule is shown in Listing 5.30, and the corresponding explanation in Table 5.11.

```
1 % RECKLESSNESS - TYPE 2
2
3 recklessness_t2(Protagonist,T+1) :-
4     time(T),
5     agent(Protagonist),
6     some_fearsconfirmed(Protagonist,T+1),
7     some_remorse(Protagonist,T+1),
8     maxtime(T+1).
```

Listing 5.30: ASP rule for Recklessness Type 2

Time	Line	Description	Explanation
T	6	Protagonist feels FearsConfirmed as a result of their reckless action.	Protagonist performed the preceding action despite anticipating its negative outcome, resulting in their fears being confirmed.
	7	Protagonist feels Remorse about the preceding action.	This ensures the preceding action is performed by Protagonist, not some other agent.

Table 5.11: Explanation of Recklessness Type 2

5.5.7 Reward

Reward is essentially the opposite of Retribution. The rule produced by Aleph for this moral (see Listing 5.17) had extremely good performance, and we use it as the basis for one of our rules. However, as with Retribution, Reward can occur in one of two ways:

1. **Type 1:** The reward is not necessarily administered by the original benefactor.
2. **Type 2:** The original benefactor provides the reward (this corresponds to the Aleph rule generated for this moral: mutual Gratitude).

Although all the Reward stories in Aesop's fable collection are of Type 2, we provide a rule for the more general Type 1 form as well, analogous to Type 1 Retribution.

Reward Type 1

Reward Type 1 is the more general form, which, as is the case for Retribution, also subsumes stories of Type 2. The general idea is `Protagonist` performs an action to make `Beneficiary` grateful, and later in the story `Protagonist` experiences Joy (this is their reward for doing a good deed). We provide the ASP rule in Listing 5.31, and a line-by-line explanation in Table 5.12.

```
1  % REWARD - TYPE 1
2  % (analogous to Retribution Type 1)
3
4  reward_t1(Beneficiary,T1+1,Protagonist,T2+1) :-
5      time(T1),
6      time(T2),
7      agent(Protagonist),
8      agent(Beneficiary),
9      feels_gratitude(Beneficiary,Protagonist,T1+1),
10     some_joy(Protagonist,T2+1),
11     not some_distress_after(Protagonist,T1+1),
12     not some_relief(Protagonist,T2+1),
13     some_admiration(Protagonist,T2+1),
14     not some_joy(Beneficiary,T2+1),
15     Protagonist != Beneficiary,
16     T1 < T2,
17     maxtime(T2+1).
```

Listing 5.31: ASP rule for Reward Type 1

Time	Line	Description	Explanation
T1	9	Protagonist does something to make Beneficiary feel Gratitude.	This is the inciting incident, for which Protagonist will be rewarded.
	11	Protagonist must not feel any Distress after this point.	Any Distress experienced by Protagonist after performing the good deed would undermine the sense of Reward.
T2	10	Protagonist feels Joy.	This is Protagonist's reward for doing a good deed at T1.
	12	Protagonist does not feel Relief.	This is to prevent the Joy experienced by Protagonist from resulting from a failed action. The sense of Reward is stronger when it is not accidental or unintended.
	13	Protagonist feels Admiration towards some other agent.	This is to ensure the event which caused Protagonist Joy was another agent's intentional action, to strengthen the sense of Reward.
	14	Beneficiary does not feel any Joy.	This is to keep the focus on Protagonist's Joy.

Table 5.12: Explanation of Reward Type 1

Reward Type 2

Reward Type 2 is the more specific subtype, requiring reciprocal Gratitude. The ASP rule is provided in Listing 5.32, and corresponds very closely to the rule produced by Aleph, shown in Listing 5.17. The basic interpretation is `Protagonist` does something to make `Beneficiary` grateful, so `Beneficiary` does something to make `Protagonist` grateful in return. Table 5.13 provides a detailed explanation.

```
1  % REWARD - TYPE 2
2  % (analogous to Retribution Type 2)
3
4  reward_t2(Beneficiary,T1+1,Protagonist,T2+1) :-
5      time(T1),
6      time(T2),
7      agent(Protagonist),
8      agent(Beneficiary),
9      feels_gratitude(Beneficiary,Protagonist,T1+1),
10     feels_gratitude(Protagonist,Beneficiary,T2+1),
11     not some_joy(Beneficiary,T2+1),
12     Protagonist != Beneficiary,
13     T1 < T2,
14     maxtime(T2+1).
```

Listing 5.32: ASP rule for Reward Type 2

Time	Line	Description	Explanation
T1	9	Protagonist does something to make Beneficiary feel Gratitude.	This is the inciting incident, for which Protagonist must be rewarded.
T2	10	Beneficiary does something to make Protagonist feel Gratitude.	This the reward for Protagonist's earlier action. Unlike in Reward Type 1, it must be carried out by Beneficiary.
	11	Beneficiary does not feel Joy.	Whatever Beneficiary does to cause Protagonist Joy should be targeted at Protagonist; if Beneficiary feels Joy as well, this undermines the sense of Reward.

Table 5.13: Explanation of Reward Type 2

5.5.8 Parameter-Free Moral Predicates

Each of the moral predicates described in Sections 5.5.2–5.5.7 have parameters, which represent the agent/s involved and the time-points at which relevant emotions arise. The values matched to these variables for a particular story are used in the text generation process, to ensure the emotions relevant to the selected moral are included in the discourse (refer to Chapter 6, Section 6.3.3, for details). However, when specifying constraints to enforce a particular moral rule, it is more straightforward to do so using predicates without parameters. As such, for each moral rule we define a corresponding parameter-free predicate, which is used for specifying moral constraints on a story. Listing 5.33 shows how these predicates are defined, using Reward Type 2 as an example. We demonstrate how these predicates are used for enforcing the moral rules in Chapter 6 (refer to Section 6.2.7).

```
1 % Parameter-free predicate for Reward Type 2
2 reward_t2 :-
3     reward_t2(A1,T1+1,A2,T2+1),
4     agent(A1),
5     agent(A2),
6     time(T1),
7     time(T2).
```

Listing 5.33: Parameter-free predicate for Reward Type 2

5.6 Summary

In this chapter, we used the ASP emotion model described in Chapter 4 to generate emotion data for a select group of Aesop's fables corresponding to the six morals we aimed to represent, as identified in Chapter 3. We applied inductive logic programming to this emotion data to establish relationships between morals and specific patterns of emotions, and refined the resulting rules to produce final ASP definitions for each moral. In the following chapter, we combine the emotion model from Chapter 4 with the moral rules developed in this chapter to assemble a system that generates stories with morals.

Chapter 6

Generating Stories with Morals

Stories are not complicated. They are, in fact, deceptively simple. But like anything simple, they are difficult to create.

Brian McDonald
Invisible Ink

For people, telling stories is natural. We use them not just to amuse or entertain others, but also to bring a point across, or explain a difficult concept by framing it in terms of something to which our audience can relate. Because the skill is so basic to us, we rarely think about all the different elements involved in crafting a story. At the most elementary level, we need characters and events, but there are many other factors that come into play. We need to decide which parts to tell and what to leave out, whether to present events in chronological order, what kind of language to use, and how much detail to go into. Our choices will vary depending on our goals in telling the story. However, most people can tell a reasonable story without having to consciously think about any of these things.

Not so for a computer. Computers do not have the innate talent for storytelling that people do. They need detailed instructions for every step of the process, not easy to provide when, to a large extent, we do not even understand how we do it: "nothing so central to the human condition is so incompletely understood." [65] This is what makes computational storytelling difficult. All we can do is attempt pieces of the puzzle, and hope they all fit together at the end. The piece we address with our research is generating sequences of events which convey a lesson: stories that have a moral.

Following on from the moral rules developed in Chapter 5, here we describe our implementation of a moral-based storytelling system. We begin by presenting the high-level structure of the system in Section 6.1, before delving deeper into its components in Sections 6.2 and 6.3, which describe how we generate the fabula and discourse respectively. Section 6.4 describes the three story worlds we implement, and Section 6.5 shows how to execute the components of the system together to generate stories. In Section 6.6, we provide examples of stories generated using each of the moral rules presented in Chapter 5, and explain how they match those definitions. Section 6.7 identifies the limitations of our system, which will lead into the discussion of areas for future work in Chapter 9. We provide a summary of this chapter in Section 6.8.

6.1 Overall System Structure

Our Moral Storytelling System (MOSS) is comprised of two main components: a Story Planner and a Text Generator. In narrative theory terms, these components produce the fabula and discourse of the story respectively, as defined in Chapter 2. Figure 6.1 shows the high-level structure of the system. A moral is supplied to the Story Planner, which generates a sequence of events and character emotions satisfying that moral. These are fed to the Text Generator, which outputs the story as simple English sentences. The Story Planner is implemented using answer set programming, whereas the Text Generator is a Perl script. The following sections describe each component in more detail.

6.2 Generating the Fabula

In simple terms, the fabula is the sequence of events comprising a story. As identified in Chapter 2, generating the fabula corresponds to the artificial intelligence notion of planning. The required plan is expressed at the event (or action) level. However, in our case, the goal is to achieve a particular sequence of emotions, and thus the primary conditions the plan must satisfy are at the emotion level. This is where implementing the system using ASP yields considerable benefits; constraints specified at any level (action, belief, emotion, or moral) are equivalent for the solver. The MOSS Story Planner, which produces the fabula, is a consolidation of the Action, Belief, Emotion, and Moral Layers described in Chapters 4 and 5. We will not repeat that information

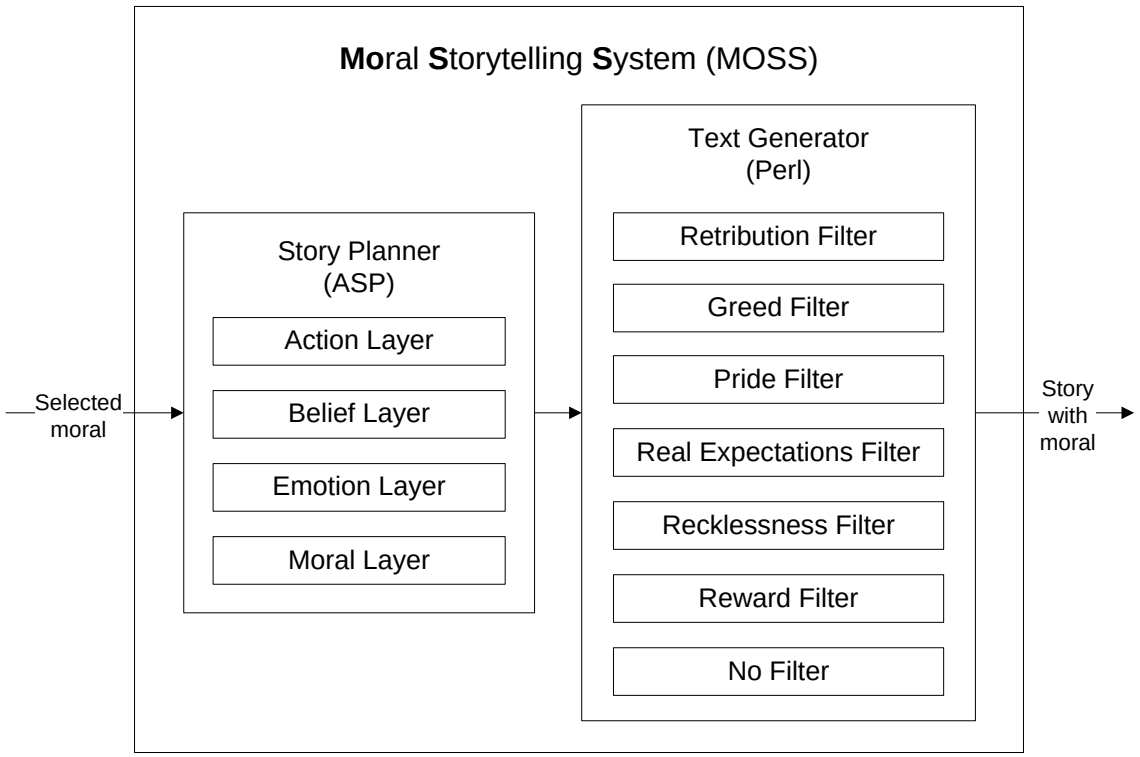

Figure 6.1: High-level structure of the Moral Storytelling System (MOSS)

in this section, but focus on explaining how the layers relate to one another in the context of story generation, using a simple running example. Illustrative ASP rules for each layer were provided in Sections 4.5.4 (Action Layer), 4.5.5 (Belief Layer), 4.5.6 (Emotion Layer), and 5.5 (Moral Layer), and the complete source code is available online.[1]

In Chapter 2, we described story from a computational perspective in terms of three elements: action, character, and plot. In MOSS, the Action Layer defines action, the Belief and Emotion Layers together model character, and the Moral Layer corresponds to plot. Table 6.1 summarises the aspects of story modelled by each layer. The Action Layer alone is enough to produce stories: it can be used to generate sequences of events. Augmenting the world model with beliefs and emotions lets us model characters' reactions to these events, which are in turn used by the Moral Layer to constrain plot trajectories. In Sections 6.2.1–6.2.3, we use a simple running example to illustrate how each layer contributes to story content and structure. We describe MOSS's file structure in Section 6.2.4, and explain the solving process in Section 6.2.5. Section 6.2.6 makes

[1]ASP source code is available at: `www.cse.unsw.edu.au/~msarlej/moss/system/storyplanner`.

a few key remarks about fabula generation using MOSS, and Sections 6.2.7 and 6.2.8 outline the Story Planner's input and output formats respectively.

Layer	Story Element	Concepts Modelled
Action	Action	Agents, objects, fluents, events, actions, preconditions, consequences, location, time
Belief	Character	Beliefs, desires, ideals, expectations
Emotion	Character	Emotions: joy, distress, hope, fear, satisfaction, fears-confirmed, disappointment, relief, happy-for, gloating, resentment, pity, pride, shame, admiration, reproach, love, hate, gratification, remorse, gratitude, anger
Moral	Plot	Morals: retribution, greed, pride, realistic expectations, recklessness, reward

Table 6.1: Summary of Story Planner layers

6.2.1 Action: Describing the Story World

For any given story world, a range of domain-specific information needs to be provided in the Action Layer, including:

- The agents/objects available, along with their properties.
- The events/actions available, along with their prerequisites.
- The consequences available.
- Causality relationships between events and consequences.
- The number of time-points available.
- Initial state information (e.g. fluent values).

In some cases, this is simply a matter of providing ground instances of the corresponding predicates. For example, to specify the agents available, we provide ground instances of `agent`. For other predicates, we define rules which the grounder uses to derive ground instances. For example, actions are generally defined using rules that reference agents as variables, since most actions can be performed by any agent. The grounder uses these rules to generate ground instances for all possible combinations of agents within the domain.

To illustrate, we provide a deliberately simplistic sample domain in Listing 6.1. We do not show all the Action Layer axioms that were discussed in Chapter 4, but only the rules which define the specifics of this domain: agents, objects, fluents, actions, and their effects. To avoid complicating Listing 6.1, we also omit the preconditions of actions. These are defined such that `kicks` and `bakes_cake_for` are always possible, and `gives_to` is possible provided that `Actor` has `Object`.

The domain defined by Listing 6.1 includes two agents, Alice and Bob, and two objects, chocolate and cake. Agents can possess objects, and they can be injured. There are three possible actions: kicking another agent, giving another agent an object, and baking a cake for another agent. Kicking an agent causes them to be injured. Giving an object to another agent causes them to have that object, and for the giver not to have the object anymore. Baking a cake for another agent causes them to have a cake. In the initial state, Bob has chocolate. Stories produced using this domain will have length 2; i.e. events can happen at time 0 and at time 1.[2]

Running the ASP solver (*clasp* [60]) with the Action Layer in isolation produces multiple solutions for this domain, each corresponding to a different event sequence. The raw *clasp* output contains a very large number of predicates, because the values of all fluents at every time-point are displayed, regardless of their relevance to the events taking place. This makes interpretation difficult. In Listing 6.2, we show one possible solution for this domain, but display only the events and their consequences (i.e. changes in fluent values). We have reformatted the predicate list for ease of reading, presenting predicates in temporal order, one per line, along with explanatory comments. The event sequence describes a very simple (though somewhat absurd) story: Bob gives Alice some chocolate, and then Alice kicks Bob. However, without the Belief and Emotion Layers, we cannot describe how the characters react to these events. In the following section, we augment this domain with beliefs and emotions, and show how this affects the output.

[2]As explained in Chapter 5, `maxtime` defines the maximum possible time-point at which emotions, not actions, can occur; it is set to the largest allowable time value, plus 1. Although we are using the Action Layer in isolation in this section, we define `maxtime` here because we consider it to be part of the definition of time.

```
% Time
time(0..1).
maxtime(2).

% Agents
agent(alice).
agent(bob).

% Objects
object(chocolate).
object(cake).

% Fluents
fluent(has(Agent,Object)) :- agent(Agent), object(Object).
fluent(is_injured(Agent)) :- agent(Agent).

% Action 1: Kicking someone
action(kicks(Target)) :- agent(Target).
% Kicking someone (if successful) causes the target to be injured
causes(Actor,kicks(Target),is_injured(Target),T) :-
    agent(Actor),
    agent(Target),
    time(T).

% Action 2: Giving an object to another agent
action(gives_to(Target,Object)) :- object(Object), agent(Target).
% Giving an object causes the Target agent to have it
causes(Actor,gives_to(Target,Object),has(Target,Object),T) :-
    agent(Actor),
    agent(Target),
    object(Object),
    time(T).
% Giving an object causes the Actor not to have it anymore
causes(Actor,gives_to(Target,Object),neg(has(Instigator,Object)),T) :-
    agent(Actor),
    agent(Target),
    object(Object),
    time(T).

% Action 3: Baking a cake for another agent
action(bakes_cake_for(Target)) :- agent(Target).
% Baking a cake for the Target causes them to have a cake
causes(Actor,bakes_cake_for(Target),has(Target,cake),T) :-
    agent(Actor),
    agent(Target),
    time(T).

% Initial state information
true(has(bob,chocolate),-1).
```

Listing 6.1: Action Layer for simple example domain

```
1  % TIME = 0
2  % Event taking place at t=0
3  happens(bob,gives_to(alice,chocolate),0)
4  succeeds(bob,gives_to(alice,chocolate),0)
5
6  % TIME = 1
7  % Consequences resulting from event at t=0
8  true(becametrue(has(alice,chocolate)),1)
9  true(becametrue(neg(has(bob,chocolate))),1)
10 % Event taking place at t=1
11 happens(alice,kicks(bob),1)
12 succeeds(alice,kicks(bob),1)
13
14 % TIME = 2
15 % Consequences resulting from event at t=1
16 true(becametrue(is_injured(bob)),2)
```

Listing 6.2: Example event sequence, produced using the Action Layer only

6.2.2 Character: Modelling Beliefs and Emotions

Characters' emotional reactions to events are based on their beliefs, desires, and ideals. These elements are defined in the Belief Layer, which, like the Action Layer, is domain-specific; characters' beliefs, desires, and ideals relate to the objects and fluents available in the story world. They can be defined as generic rules (if a particular desire or ideal is shared by all agents), or supplied as ground instances (if a desire or ideal only applies to one particular agent).

Continuing with the example domain described in the preceding section, we define desires and ideals for Alice and Bob in Listing 6.3. We define agents' desires as fixed across all time-points: both agents desire not to be injured, Bob desires cake, and Alice desires chocolate. Both agents adopt the same ideals: it is not ideal to kick others, it is ideal to give to others, and it is ideal to bake cakes for others. Beliefs are derived directly from what is true in the world at any given time-point, as per the omniscience assumption described in Chapter 4. As was the case for the Action Layer, we include only domain-specific rules in Listing 6.3. The additional rules required for managing beliefs, desires, ideals, and expectations were described in detail in Chapter 4 (refer to Section 4.5.5).

```
% DESIRES
% All agents desire to not be injured
des(Agent,becametrue(neg(is_injured(Agent))),T+1) :-
    agent(Agent),
    time(T).
% Bob desires to have cake
des(bob,becametrue(has(bob,cake)),T+1) :-
    time(T).
% Alice desires to have chocolate
des(alice,becametrue(has(alice,chocolate)),T+1) :-
    time(T).

% IDEALS
% It is not ideal to kick others
idl(neg(kicks(Agent))) :-
    agent(Agent).
% It is ideal to give to others
idl(gives_to(Agent,Object)) :-
    agent(Agent),
    object(Object).
% It is ideal to bake cakes for others
idl(bakes_cake_for(Agent)) :-
    agent(Agent).
% All agents adopt all ideals
idl(Agent,Action) :-
    idl(Action),
    agent(Agent).
```

Listing 6.3: Belief Layer for simple example domain

Due to our omniscience assumption, there is little value in showing *clasp* output augmented with beliefs, as agents' beliefs simply duplicate the truth values of fluents. More important are the emotional reactions that result from the desires and ideals defined in Listing 6.3. These are derived using the rules comprising the Emotion Layer, which were described in detail in Chapter 4; we will not reproduce them here (no domain-specific rules need to be added to the Emotion Layer). Listing 6.4 shows the same story that was presented in Listing 6.2, but produced by running clasp with the Action, Belief, and Emotion Layers together. As before, we do not include all predicates, focussing on events, their consequences, and characters' emotions. We present the predicates in temporal order, and include explanatory comments.

```
% TIME = 0
% Pre-existing emotions at t=0
love(bob,alice,0)
% Event taking place at t=0
happens(bob,gives_to(alice,chocolate),0)
succeeds(bob,gives_to(alice,chocolate),0)

% TIME = 1
% Emotions persisting from t=0
love(bob,alice,1)
% Consequences resulting from event at t=0
true(becametrue(has(alice,chocolate)),1)
true(becametrue(neg(has(bob,chocolate))),1)
% Emotions resulting from event at t=0
joy(alice,becametrue(has(alice,chocolate)),1)
admiration(alice,bob,gives_to(alice,chocolate),1)
gratitude(alice,bob,gives_to(alice,chocolate),
          becametrue(has(alice,chocolate)),1)
love(alice,bob,1)
happyfor(bob,alice,becametrue(has(alice,chocolate)),1)
pride(bob,gives_to(alice,chocolate),1)
% Event taking place at t=1
happens(alice,kicks(bob),1)
succeeds(alice,kicks(bob),1)

% TIME = 2
% Emotions persisting from t=1
love(alice,bob,2)
% Consequences resulting from event at t=1
true(becametrue(is_injured(bob)),2)
% Emotions resulting from event at t=1
distress(bob,becametrue(is_injured(bob)),2)
reproach(bob,alice,kicks(bob),2)
anger(bob,alice,kicks(bob),becametrue(is_injured(bob)),2)
hate(bob,alice,2)
pity(alice,bob,becametrue(is_injured(bob)),2)
shame(alice,kicks(bob),2)
```

Listing 6.4: Story from Listing 6.2, with characters' emotions

The simple story from Listing 6.2 (Bob gives Alice some chocolate, and then Alice kicks Bob) is now augmented with character emotions, making it considerably richer:

> Bob loves Alice. He gives her chocolate. Alice feels joy about receiving the chocolate, and grateful towards Bob for giving it to her. She admires his action, and it makes her love him. Bob is happy for Alice, and proud of his action. Then, Alice kicks Bob, injuring him. He feels distress about being injured, and reproach and anger towards Alice for kicking him. He starts to hate Alice. Alice pities Bob for being injured, and is ashamed of kicking him.

A set of events which we previously described in one sentence now forms a paragraph when emotions are added. However, even with characters having appropriate emotional reactions, this story lacks meaningful structure. It does not make sense for Alice to kick Bob, especially seeing as she loves him for giving her chocolate. Although emotions arise sensibly from events, they do not place any constraints on the story sequence; that is achieved with the Moral Layer.

6.2.3 Plot: Structuring Stories Using Morals

The Moral Layer uses emotions to place constraints on story structure, thereby defining plot. The moral rules we implement were described in detail in Chapter 5. They are based only on emotions, so no domain-specific information needs to be added. Listing 6.5 shows an event sequence generated using the same domain described in the preceding sections, but enforcing the emotion patterns for Reward Type 2 in the Moral Layer. This time, the storyline is different: Bob gives Alice chocolate, and then Alice bakes a cake for Bob. Alice's more reasonable response to the gift of chocolate results from the restrictions placed on Bob's emotions by the moral rule.

```
% TIME = 0
% Pre-existing emotions at t=0
love(bob,alice,0)
% Event taking place at t=0
happens(bob,gives_to(alice,chocolate),0)
succeeds(bob,gives_to(alice,chocolate),0)

% TIME = 1
% Emotions persisting from t=0
love(bob,alice,1)
% Consequences resulting from event at t=0
true(becametrue(has(alice,chocolate)),1)
true(becametrue(neg(has(bob,chocolate))),1)
% Emotions resulting from event at t=0
joy(alice,becametrue(has(alice,chocolate)),1)
pride(bob,gives_to(alice,chocolate),1)
admiration(alice,bob,gives_to(alice,chocolate),1)
gratitude(alice,bob,gives_to(alice,chocolate),
          becametrue(has(alice,chocolate)),1)
happyfor(bob,alice,becametrue(has(alice,chocolate)),1)
love(alice,bob,1)
% Event taking place at t=1
happens(alice,bakes_cake_for(bob),1)
succeeds(alice,bakes_cake_for(bob),1)

% TIME = 2
% Emotions persisting from t=0
love(bob,alice,2)
love(alice,bob,2)
% Consequences resulting from event at t=1
true(becametrue(has(bob,cake)),2)
% Emotions resulting from event at t=1
joy(bob,becametrue(has(bob,cake)),2)
pride(alice,bakes_cake_for(bob),2)
admiration(bob,alice,bakes_cake_for(bob),2)
gratitude(bob,alice,bakes_cake_for(bob),
          becametrue(has(bob,cake)),2)
happyfor(alice,bob,becametrue(has(bob,cake)),2)
```

Listing 6.5: Enforcing moral structure

We reproduce the ASP rule for Reward Type 2 in Listing 6.6, to illustrate how the emotions in the story satisfy the requirements of the moral. Matching the variables from Listing 6.6 with the story elements from Listing 6.5, `Protagonist` is Bob, `Beneficiary`

is Alice, `T1` is 0 (i.e. `T1+1` is 1), and `T2` is 1 (i.e. `T2+1` is 2). The rule for Reward Type 2 places three emotional constraints on a story. In Table 6.2, we show how the emotions experienced by the characters in Listing 6.5 correspond to each of these constraints. This illustrates how the Moral Layer is used to control plot trajectory. Representing morals in terms of emotions allows many different event sequences to satisfy each moral rule (depending, of course, on the richness of the story world).

```
1  % REWARD - TYPE 2
2
3  reward_t2(Beneficiary,T1+1,Protagonist,T2+1) :-
4      time(T1),
5      time(T2),
6      agent(Protagonist),
7      agent(Beneficiary),
8      feels_gratitude(Beneficiary,Protagonist,T1+1),
9      feels_gratitude(Protagonist,Beneficiary,T2+1),
10     not some_joy(Beneficiary,T2+1),
11     Protagonist != Beneficiary,
12     T1 < T2,
13     maxtime(T2+1).
```

Listing 6.6: ASP rule for Reward Type 2

Line No. (Listing 6.6)	Interpretation of ASP Rule	Line No. (Listing 6.5)	Description of Matching Story Emotion
8	Beneficiary feels Gratitude towards Protagonist.	18–19	Alice feels Gratitude towards Bob for giving her chocolate.
9	Protagonist feels Gratitude towards Beneficiary.	36–37	Bob feels Gratitude towards Alice for baking him a cake.
10	Beneficiary does not feel any Joy at T2+1.	N/A	Alice does not feel any Joy at time 2.

Table 6.2: Emotional constraints matched to story elements

6.2.4 ASP File Structure

Having illustrated how the Action, Belief, Emotion, and Moral Layers each contribute to story generation, in this section we describe the corresponding file structure. We

implement each layer of the Story Planner as a separate ASP file. This modular approach makes it easy to modify or extend the system, by replacing any of these files with an alternative implementation. For example, to use a different belief model, only the file corresponding to the Belief Layer needs to be replaced; this will not affect the rest of the system (provided, of course, that those predicates which are referenced by the other layers retain the same structure).

Most important, however, is that while the Action and Belief Layers are domain-specific (as evidenced by the domain-specific rules that had to be provided in Sections 6.2.1 and 6.2.2), the Emotion and Moral Layers are completely decoupled from the story world. Although these files do reference the predicates defined in the Action and Belief Layers, this is only in an abstract sense, as variables; they do not reference any specific instances of agents, actions, etc. Therefore, while the Action and Belief files must be tailored to each new story world, the Emotion and Moral files can be used without modification across all domains.

In addition to the ASP files corresponding to the four layers of the Story Planner, we provide a separate display file. This file uses *gringo*'s `#hide` and `#show` directives [61] to specify which predicates are to be displayed in the output. In our case, only events and emotions are important to the story; any additional predicates (such as fluent values, beliefs, etc.) would only make the output more difficult to interpret. The display file we use is available online with the rest of the ASP source code.[3]

6.2.5 Finding Solutions

As mentioned briefly in Chapter 4, MOSS uses the *Potassco* suite of answer set solving tools [60] to generate story plans. The ASP files are supplied as input to *gringo*, a grounder. *gringo*'s output is piped to *clasp*, a solver which computes the program's answer sets [61]. Answer set programming is declarative, and thus, unlike in procedural programming, an ASP program does not specify the steps to be taken in searching for a solution. The solving process is domain-independent (the search procedure employed by *clasp* is described in Gebser et al.'s work [60]), which is one of the key benefits of the ASP approach. As such, the steps we list below do not reflect *clasp*'s search procedure. Nevertheless, the premise on which MOSS's Story Planner is based can be better understood when conceptualised as follows:

[3]The display file is available at: `www.cse.unsw.edu.au/~msarlej/moss/system/storyplanner/display.lp`.

1. The user specifies a moral. This is done by adding an ASP constraint to the Moral Layer (refer to Section 6.2.7 for an example).

2. The selected moral corresponds to a particular sequence of OCC emotions, specified in one of the rules in the Moral Layer. These emotions are therefore also a requirement for an acceptable solution.

3. The system finds a sequence of events that will cause the appropriate pattern of emotions to arise.

4. The resulting event sequence, when presented as a story, should convey the selected moral.

This can be compared to traversing in reverse through the breakdown of story presented in Figure 3.1. There we described a story as consisting of events, which cause characters to experience emotions, and in turn cause the reader to interpret a moral. Here, we begin with a desired moral, identify the corresponding emotions, and find a sequence of events which causes them.

6.2.6 Remarks on Fabula Generation using MOSS

We make three further remarks about fabula generation using the MOSS Story Planner in its current form:

1. Although we do model location (refer to Section 4.5.4), and made use of this feature when producing emotion data from Aesop's fables, when generating stories as described in this chapter, we keep all characters located `onstage` at all times. This is to reduce the complexity of the Text Generator (described in Section 6.3). If characters were able to change location during a story, we would need to provide additional text descriptions to convey when they move in and out of the story's focus.

2. MOSS is not a character-centric system, and as such the goals and motivations of the characters (as expressed through their desires and ideals) do not have any direct impact on their behaviour. The control of the story is at the plot level, through achieving the required emotion sequences. As a result, characters will not always behave in a rational or expected fashion. For example, stories in which a knight loves a princess, only to kill her in the following time-step (and feel

distress about it) are quite possible. Although characters' desires and ideals do not control their behaviour, the reader's awareness of them is often what allows a moral to be conveyed. For instance, one might consider the knight's action in this example to be quite reckless; it would not be reckless, on the other hand, if the knight hated the princess to start with. The system could certainly be extended to make characters behave more rationally, and this will be discussed as future work in Chapter 9.

3. All valid time values must be declared as part of the ASP problem definition, so the time variables appearing in rules can be grounded. In effect, this is equivalent to specifying the exact length of the stories to be produced by the solver. To generate stories of a different length, the time declaration needs to be modified. Hence, a single run of the system cannot generate stories of varying length.

6.2.7 Story Planner Input

In addition to the ASP files containing the Action, Belief, Emotion, and Moral Layers and the display file, we need to specify which moral is to be generated. This is done using a simple ASP constraint. In Listing 6.7, we show the constraint used to produce the story presented in Section 6.2.3, corresponding to Reward Type 2 (note that this is the parameter-free Reward Type 2 predicate, as defined in Section 5.5.8). This constraint asserts that the `reward_t2` predicate must appear in any valid solution, which in turn requires that the emotions corresponding to that moral also feature. Such constraints can be provided in a separate file, or added to the Moral Layer (we provide them as separate files, to avoid having multiple versions of the Moral Layer file). It is also possible to specify multiple morals simultaneously, but we do not investigate that possibility in this work.

```
1 :- not reward_t2.
```

Listing 6.7: Example moral constraint (Reward Type 2)

Assuming the files containing the Action, Belief, Emotion, and Moral Layers are called action.lp, belief.lp, emotion.lp, and morals.lp respectively, display.lp contains the display settings, and reward_t2.lp contains the constraint shown in Listing 6.7, we use the following command in a terminal window to run the grounder/solver combination[4]:

```
gringo action.lp belief.lp emotion.lp morals.lp display.lp reward_.lp
    | clasp
```

This command assumes that *gringo* and *clasp* are installed on the machine and in the current path, and that all the *.lp files are in the current directory. Both *gringo* and *clasp* provide a range of options (described in detail in Gebser et al.'s user guide [61]); we use the default settings.

6.2.8 Story Planner Output

The format of *clasp*'s output is shown in Listing 6.8. In keeping with the same running example, this solution corresponds to the story presented in Section 6.2.3. Note that lines 5–22 are actually displayed on a single line as an unordered, space-separated predicate list; we have added line-breaks for ease of reading. Line 23 asserts that the problem was satisfiable (i.e. solutions could be found); if there are no solutions, this will read UNSATISFIABLE. There may be zero, one, or many solutions for any particular set of input files; *clasp* provides an option that allows the user to specify how many solutions should be displayed (the default is 1). Line 25 indicates there are multiple solutions to this problem. However, the exact number was not computed because *clasp* was only asked to find one solution, at which point the search stopped. The time taken in searching for a solution is displayed on lines 26–27.

Most of the predicates included in the answer set shown in Listing 6.8 should be familiar from Chapters 4 and 5. The only exception is the `started_to_love` predicate, which has not been discussed previously. This predicate (and the analogous `started_to_hate` predicate) is not used at all in planning a story; the rule defining it could be removed completely and it would have no impact on the solutions found. It serves an assistive function for the text generation script, allowing easy identification of when Love/Hate relationships between characters change (this will be discussed in Section 6.3). We also display the moral predicate (in this case, `reward_t2`) in the out-

[4]The *Potassco* suite also provides a combined grounder and solver, *clingo*, which we could have used to produce equivalent results instead of piping from *gringo* to *clasp*.

put.[5] Although this does not correspond to any segment of story text, it allows the text generation script to identify which time-points in the story match up to the time variables in the moral rule, which is important for implementing the emotion filters that will be described in Section 6.3.3. The story text corresponding to the answer set in Listing 6.8 will be presented in Section 6.3.

```
1  clasp version 2.1.0
2  Reading from stdin
3  Solving...
4  Answer: 1
5  happens(alice,bakes_cake_for(bob),1)
6  happens(bob,gives_to(alice,chocolate),0)
7  succeeds(bob,gives_to(alice,chocolate),0)
8  succeeds(alice,bakes_cake_for(bob),1)
9  joy(bob,becametrue(has(bob,cake)),2)
10 joy(alice,becametrue(has(alice,chocolate)),1)
11 pride(bob,gives_to(alice,chocolate),1)
12 pride(alice,bakes_cake_for(bob),2)
13 admiration(bob,alice,bakes_cake_for(bob),2)
14 admiration(alice,bob,gives_to(alice,chocolate),1)
15 gratitude(bob,alice,bakes_cake_for(bob),becametrue(has(bob,cake)),2)
16 gratitude(alice,bob,gives_to(alice,chocolate),
17           becametrue(has(alice,chocolate)),1)
18 love(alice,bob,2) love(alice,bob,1) love(bob,alice,2)
19 love(bob,alice,1) love(bob,alice,0) started_to_love(alice,bob,1)
20 happyfor(bob,alice,becametrue(has(alice,chocolate)),1)
21 happyfor(alice,bob,becametrue(has(bob,cake)),2)
22 reward_t1(alice,1,bob,2) reward_t2(alice,1,bob,2)
23 SATISFIABLE
24
25 Models       : 1+
26 Time         : 0.026s (Solving: 0.00s 1st Model: 0.00s Unsat: 0.00s)
27 CPU Time     : 0.000s
```

Listing 6.8: Sample *clasp* output enforcing Reward Type 2

[5]Line 22 actually displays both `reward_t2` and `reward_t1`. This is because, as explained in Chapter 5 (refer to Section 5.5.7), the Reward Type 1 rule is more general, and subsumes the Reward Type 2 rule, so any story satisfying Reward Type 2 will also satisfy Reward Type 1.

6.3 Generating the Text

Although it describes a story, the *clasp* output presented in the previous section is by no means human-readable. To make it so, we require a separate text generation component: this is the Text Generator from Figure 6.1. The Text Generator is a Perl script, which accepts as input the *clasp* output for a single story, and converts it into a simple English description of the events, their outcomes, and the emotions characters feel. The focus of our work is story planning, not text generation. The text we generate uses simple language and repetitive sentence structure, with the goal of being easily understood by a human reader. The Text Generator uses predefined sentences corresponding to particular actions and consequences, along with grammatical substitutions to insert pronouns and the appropriate tense. We begin in Section 6.3.1 by presenting the text generated from the *clasp* output in Listing 6.8. With this in mind as our objective, we move on to describe the details of our implementation in Section 6.3.2. Finally, Section 6.3.3 explains the filters we implement to remove emotions which are not essential for conveying the selected moral, resulting in more concise stories.

6.3.1 Illustrating the Outcome

In Section 6.2.8 (Listing 6.8), we provided an example of raw *clasp* output: a space-separated, unordered list of predicates. The predicates displayed represent the events taking place in the story, along with the emotions characters experience at each time-step. To convert this predicate list into English text, we take the *clasp* output as is, and provide it as input to the Text Generator Perl script. The resulting text is shown in Figure 6.2. It begins with a short introductory paragraph, introducing Alice and Bob, followed by a separate paragraph for each event in the story. All the emotions the characters feel are explicitly expressed in the text. This example demonstrates the typical structure of a MOSS story. In the following section, we explain the algorithm used by the Text Generator to produce it.

In a small town there lived Alice and Bob. Bob loved Alice.

One day Bob gave the chocolate to Alice. As a result, Alice had the chocolate. Alice felt joy that she had the chocolate. Bob felt happy for Alice because Alice had the chocolate. Alice felt gratitude towards Bob about giving the chocolate to her because she had the chocolate. Alice felt admiration towards Bob about giving the chocolate to her. Bob felt pride about giving the chocolate to Alice. Alice started to love Bob.

Later that day Alice baked a cake for Bob. As a result, Bob had the cake. Alice felt happy for Bob because Bob had the cake. Bob felt joy that he had the cake. Alice felt pride about baking a cake for Bob. Bob felt gratitude towards Alice about baking a cake for him because he had the cake. Bob felt admiration towards Alice about baking a cake for him.

Figure 6.2: Text produced from *clasp* output in Listing 6.8

6.3.2 Script Architecture

In this section, we present the high-level algorithm of the Text Generator, such that its behaviour can be understood. We do not go into detail about specific implementation choices, because they are not relevant to the outcome. Each story is generated using the same simple schema: it begins with a short introduction to set the scene, followed by one paragraph for every event. The contents of each paragraph (i.e. the description of the event, its consequences, and characters' emotions) always appear in the same order. We hand-author text fragments corresponding to all the predicates relevant to each domain, and these are inserted into generic sentence templates as appropriate. Due to the pre-scripted nature of the text, a separate text generation script is required for each story world, but they all follow the same basic process; only the included text fragments are different. The Text Generator scripts for all three story worlds we implement are available online.[6]

Figure 6.4 shows the text generation process as a flowchart. The text generated by each process is shown in the box on the right. The elements enclosed in square brackets (e.g. [agent]) are variables, which are replaced by story-specific values. The names of the process boxes are abbreviated, so we provide explanations in Table 6.3, presented

[6]The Text Generator source code is available at: `www.cse.unsw.edu.au/~msarlej/moss/system/textgenerator`.

in the order they appear in the flowchart. The legend for interpreting the symbols is provided in Figure 6.3. Apart from the introduction and the descriptions of events and their consequences, all other sentences are optional: they are only included in the discourse if those particular emotions exist (and are allowed by the selected emotion filter, as will be explained in Section 6.3.3).

The first paragraph of every story introduces all the characters that will feature, as well as any pre-existing object-based emotions (i.e. Love or Hate) between them. This is important in helping readers understand any fortunes-of-others emotions that may arise later in the story. Following the introduction, there is a separate paragraph for each event. The paragraph corresponding to the event which occurs at time T begins begins with a sentence describing any Hopes or Fears felt by the characters. These emotions often relate to the outcomes of the event taking place at T, and can hence provide motivation for characters' actions. A description of the event is presented next. The text fragment used depends on whether the event was successful. The event is followed by a list of its consequences, which are grouped into a single sentence. The remainder of the paragraph focusses on the emotions experienced by the characters at $T+1$ (i.e. resulting from the event at T).

Self-directed event-based emotions are presented first. Those relating to the same consequences are grouped together, to avoid having a separate sentence for every emotion. For example, instead of:

> The troll felt joy that he had the crown. The troll felt satisfaction that he had the crown.

the output would be:

> The troll felt joy and satisfaction that he had the crown.

Other-directed event-based emotions are presented next, before moving on to compound and agent-based emotions. In displaying self-directed compound emotions, consequences are grouped together in a single sentence if the same emotion is felt with regards to multiple consequences. These are followed by self-directed agent-based emotions (i.e. Pride and Shame). In this case, there can only be one emotion at a time, because we do not allow simultaneous events, and a single event cannot be judged both ideal and unideal by the same agent. The other-directed compound and agent-based emotions are handled in the same fashion.

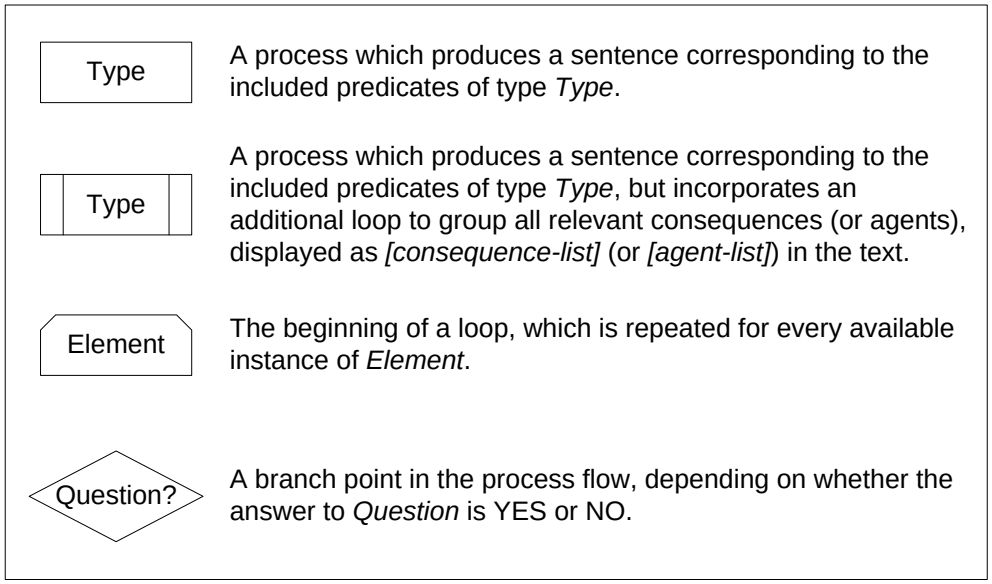

Figure 6.3: Symbol interpretation legend for Figure 6.4

Process Box	Description	Examples
Object-based	Object-based emotions	Love, Hate
Future-directed	Unconfirmed prospect-based emotions	Hope, Fear
Happens	Events that take place	Based on the `happens` predicate
Consequences	The outcomes of events	Based on the `consequence` predicate
Event: self	Self-directed event-based emotions	Joy, Distress, Satisfaction, Relief, FearsConfirmed, Disappointment
Event: other	Other-directed event-based emotions (i.e. fortunes-of-others emotions)	HappyFor, Pity, Resentment, Gloating
Compound: self	Self-directed compound emotions	Gratification, Remorse
Agent: self	Self-directed agent-based (attribution) emotions	Pride, Shame
Compound: other	Other-directed compound emotions	Gratitude, Anger
Agent: other	Other-directed agent-based (attribution) emotions	Admiration, Reproach

Table 6.3: Explanation of process box names from Figure 6.4

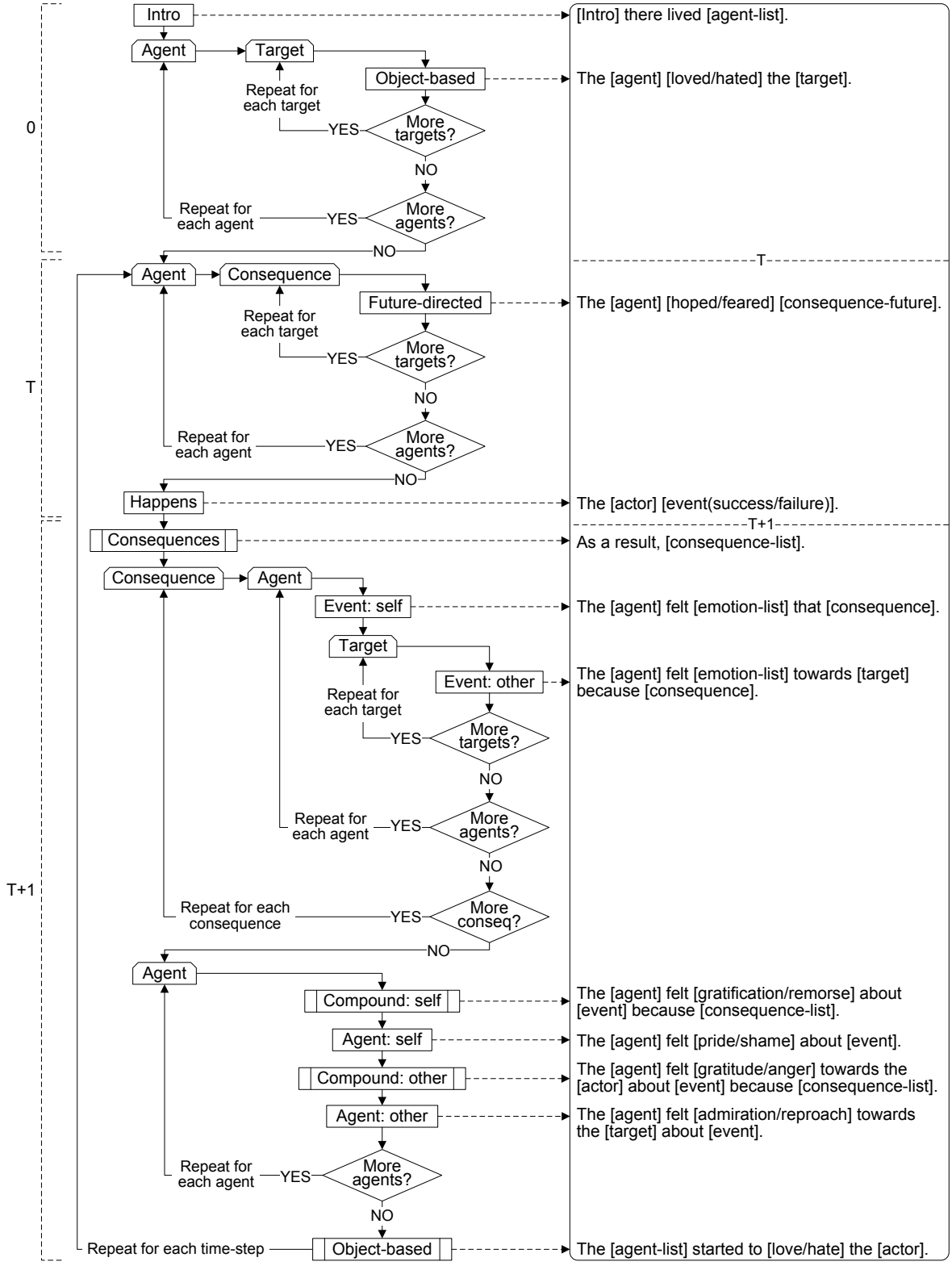

Figure 6.4: Text Generator flowchart

Every paragraph concludes with a description of any changes in object-based emotions. Only changes are displayed because object-based emotions persist through time (refer to Section 4.5.3). For example, if the knight loved the princess at the beginning of the story, unless something happened to change his love, he would continue to love the princess at T = 1, T = 2, etc., and every paragraph in the story would end with: "The knight loved the princess." To avoid this needless repetition, we define helper predicates in the Emotion Layer (`started_to_love` and `started_to_hate`), which hold when Love or Hate become true. They are not referenced by any other predicates in the Story Planner, serving only to identify changes in object-based emotions for the Text Generator. Love and Hate are based on Admiration and Reproach (refer to Section 4.4.4), so changes in these emotions can only arise with respect to the agent who performed the last action. In formulating the sentence, we group together all agents who started to Love/Hate the agent who performed the preceding action. For example:

> The dwarf and the unicorn started to love the princess.

Object-based changes are not displayed in the final paragraph of the story. There are no further events, so they are no longer necessary for aiding the reader's interpretation of characters' reactions. They may actually distract the reader from the other emotions presented in that paragraph, which are important for understanding the story's moral.

Each predicate in the sentences shown in Figure 6.4 is replaced with an appropriate text fragment. We present examples of each type of substitution in Table 6.4. The parts of the text written in CAPITALS are later replaced with another appropriate substitution. Pronoun substitutions are used to make the text seem more natural. If a character's name appears multiple times in the same sentence, the second occurrence is replaced with the appropriate pronoun. The initial character introduction and the sentence describing the event in each paragraph are prepended with a short beginning to introduce variety and flow (see the last five entries in Table 6.4). Five separate sets of these sentence beginnings are provided, because what is appropriate will vary depending on when in the story it appears. For instance, prepending the first event in the story with "later that day" would not make sense. For each new paragraph, a beginning is selected at random from the appropriate set. A number of other simple substitutions are performed throughout the script (for example, to change "the apple" to "an apple" the first time it is referenced, to capitalise sentences, etc.). We will not list them all here.

Substitution Type	Example	Corresponding Text
Event (success)	steals_from	the AGENT stole OBJECT from the TARGET.
Event (failure)	steals_from	the AGENT tried to steal OBJECT from the TARGET, but failed.
Object	treasure	the treasure
Fluent	is_enchanted	was under a spell
Fluent (negation)	neg(is_enchanted)	was not under a spell
Fluent (future)	is_enchanted	would be under a spell
Fluent (future negation)	neg(is_enchanted)	would not be under a spell
Pronoun (3rd person subject)	knight	he
Pronoun (3rd person object)	knight	him
Pronoun (3rd person possessive)	knight	his
Emotion (past tense)	joy	felt joy
Intro	–	Once upon a time
Beginning (T = 0)	–	One summer's morning
Connective (T = 1)	–	The next day
Connective (T = 2)	–	Some time later
Connective (T ≥ 3)	–	Eventually

Table 6.4: Examples of text substitutions

The algorithm described in this section, with no further processing, produces text like that shown in Figure 6.2. However, one of the most noticeable things about that story, apart from the repetitive sentence structure, is that it is extremely wordy, making it tedious to read. Describing all the emotions that occur during a story is not useful for a reader; it only distracts them from the events taking place. In the following section, we describe how we reduce the number of emotions in the text, by displaying only those that are relevant to the selected moral.

6.3.3 Emotion Filters

In even very simple and short stories, characters can experience a large number of emotions. Looking back at the *clasp* output provided in Listing 6.8, this story only involves two events and two characters. Despite this, there are 15 emotions experienced. Including all of them explicitly in the story will result in a copious amount of text that does not say very much. Although it is useful to include some information about emotions in a story, because it helps the reader empathise with the characters and understand their reactions, not every emotion experienced by a character is relevant, or useful in bringing the point across. If anything, too much information about emotions will distract a reader from the story, and make it harder to follow, as evidenced by Figure 6.2.[7] This story is extremely laborious to read.

To combat this overabundance of emotion data, the Text Generator only displays a select subset of character emotions: those that are relevant to the given story. Which emotions are relevant, however, varies by moral. To manage this, the Text Generator incorporates moral-specific emotion filters. These filters are based on the emotions which appear in each moral's definition, as described in Section 5.5. Those emotions were deemed necessary to define the moral, so it is reasonable to assume a reader's awareness of them will help that reader perceive the moral in the story. Figure 6.5 shows the same story as Figure 6.2, only this time the Reward Type 2 emotion filter has been applied; it is much easier to read. The Text Generator also provides the option of generating text which includes all the emotions which arise.[8] Such stories play a role in our evaluation process, which will be described in Chapter 7.

Our use of emotion filters to control which emotions are included in the discourse is the extent to which we address sjuzhet, which was defined in Chapter 2 as the specific selection of events from the fabula that are presented to the reader. Although we include all events in the discourse, we do not present the reader with the complete fabula, because that would include all characters' emotions as well.

[7] A careful reader will note there are only 12 emotions displayed in Figure 6.2, when 15 are shown in Listing 6.8. The other 3 emotions are object-based, and as explained in Section 6.3.2, object-based emotions are only displayed when they change.

[8] The object-based emotions are still only displayed when they change, as described in Section 6.3.2.

In a small town there lived Alice and Bob. Bob loved Alice.

One morning Bob gave the chocolate to Alice. As a result, Alice had the chocolate. Alice felt joy that she had the chocolate. Alice felt gratitude towards Bob about giving the chocolate to her because she had the chocolate. Alice started to love Bob.

The next day Alice baked a cake for Bob. As a result, Bob had the cake. Bob felt joy that he had the cake. Bob felt gratitude towards Alice about baking a cake for him because he had the cake.

Figure 6.5: Text from Figure 6.2 using Reward Type 2 emotion filter

Table 6.5 summarises the emotions which are displayed by the emotion filter for each moral. The time-points listed in the table correspond to the time variables used in the relevant moral rule. This is why we display the parametrised moral predicates in the *clasp* output, as exemplified on line 22 of Listing 6.8. These predicates allow the text generation script to identify which time-points in the story correspond to T1, T2, etc. in the applicable moral's rule,[9] and thus display the relevant emotions at those time-points only. We do not provide each emotion's full parameter structure in the table, but use variables (A, A1, and A2) to show which agents are experiencing the emotion, so this is clear for morals where multiple agents are involved.

It is worth highlighting the differences between the emotions appearing in the emotion filters compared to the corresponding moral rules. There are four main points to note:

1. In the moral rules, some emotions do not need to be specified because they are implicit in the OCC definitions of other emotions. For instance, Hope need not be specified if Satisfaction is, because Hope is a prerequisite for Satisfaction. However, we do want the Hope to be displayed, so it needs to be specified in the emotion filter. This is also the case for Fear, with respect to FearsConfirmed. For this reason, we use $T1-1$, $T2-1$, and $T-1$ in these cases; there is no variable in the applicable moral rules corresponding to the time-point when Hope (or Fear) occurs.

2. The emotions in the table are not necessarily guaranteed to arise in the story;

[9]The ASP moral rules reference the time-points as $T1+1$, $T2+1$, etc., but, as was explained in Section 5.5, this is only to manage the time-values which are permissible for emotions as compared to events. When comparing the time variables in Table 6.5 to the moral rules, the $+1$ can be ignored.

Filter	Time	Emotions Displayed
Retribution Type 1	T1	anger(A2), distress(A2)
	T2	anger(A1), distress(A1)
Retribution Type 2	T1	anger(A2), distress(A2)
	T2	anger(A1), distress(A1)
Greed Type 1	T1 – 1	hope(A)
	T1	satisfaction(A)
	T2	distress(A)
Greed Type 2	T1 – 1	hope(A)
	T1	satisfaction(A), shame(A), hope(A)
	T2	satisfaction(A), shame(A)
	T3	distress(A)
Pride	T1	pride(A), joy(A)
	T2	distress(A)
Realistic Expectations	T2 – 1	hope(A)
	T2	disappointment(A)
Recklessness Type 1	T – 1	hope(A)
	T	satisfaction(A), distress(A)
Recklessness Type 2	T – 1	fear(A)
	T	fearsconfirmed(A)
Reward Type 1	T1	gratitude(A2), joy(A2)
	T2	gratitude(A1), joy(A1)
Reward Type 2	T1	gratitude(A2), joy(A2)
	T2	gratitude(A1), joy(A1)

Table 6.5: Summary of Text Generator emotion filters

it is just that if they do, they should be displayed. For example, we display Distress and Anger for both agents in Retribution Type 1 and Type 2, even though the Type 1 definition (Listing 5.23) does not necessarily require Anger for A1. However, if Anger does arise, it is relevant to the moral, so it should be displayed.

3. In some moral rules (specifically, Retribution Type 1, Greed Type 1 and 2, Pride, and Reward Type 1), Admiration or Reproach are included to ensure that the preceding action is carried out by an agent other than the protagonist. For example, in the case of Retribution Type 1, the protagonist is required to feel Reproach

at the end of the story, to ensure their punishment results from another agent's intentional action. In these cases, the Admiration/Reproach is not useful for interpreting the moral of the story, and as such it is not included in that moral's emotion filter. Remorse and Shame serve analogous functions in Recklessness Type 2 and Greed Type 1 respectively, except to ensure the protagonist *is* the one to carry out the preceding action, and thus they are similarly excluded from the corresponding emotion filters.

4. For Realistic Expectations, we have used T2 − 1 rather than T1 in the emotion filter. This is because T1 (as per the ASP rule in Listing 5.28) is not necessarily equivalent to T2 − 1, as T1 and T2 do not have to be consecutive.

6.4 Story Worlds

To demonstrate that our moral rules can be applied across multiple domains, we implement three distinctly different story worlds: a fairytale world, a world of animals (inspired by the subject matter of many fable collections), and a family. Each story world contains enough agents, objects, and events to produce a large number of unique stories. In this section, we provide a basic description of each story world, including the available characters, objects, fluents, and actions. Note that we only include domain-specific fluents here, and thus exclude location, which is modelled in all domains. We do not list the preconditions and effects of the events and actions in this section, but a full description of each domain is included in Appendix D. All three story worlds are used in our evaluation of the system, which we describe in Chapter 7.

6.4.1 Domain 1: Fairytale

The fairytale domain is modelled on a stereotypical fairytale world, which contains a mix of human and magical characters. Table 6.6 provides a summary of this domain's elements. The properties of characters and objects are defined as predicates that apply to particular character or object instances. For example, the wizard is a magical being, and thus the predicate `magical(wizard)` holds in this domain. The variable names used for the fluent and event/action parameters in the table denote the type of instance to which they can apply. For example, the `eats` action can only be performed if the object in question has the property `edible`.

Element	No. of Instances	List of Instances
Characters	8	dragon, dwarf, fairy, knight, princess, troll, unicorn, wizard
Character Properties	9	canfly, chaotic, evil, good, human, lawful, magical, monster, warrior
Objects	11	apple, axe, crown, fairydust, gold, mushroom, rose, spellbook, sword, treasure, wand
Object Properties	5	edible, magicaid, poisonous, valuable, weapon
Fluents	8	has(Agent,Object), has_magic_aid(Agent), has_pricked_finger(Human), is_alive(Agent), is_enchanted(Agent), is_free(Agent), is_hungry(Agent), is_injured(Agent)
Events/ Actions	14	casts_spell_on(Agent), climbs_tree_to_pick_apple, eats(Edible), frees_from_spell(Agent), gets_hungry, gives_to(Agent,Object), heals(Agent), kidnaps(Agent), kills(Agent), picks_mushroom, picks_rose, releases(Agent), rescues_from(Agent,Agent), steals_from(Agent,Object)

Table 6.6: Summary of Fairytale story world

For the Fairytale domain, we borrow the idea of character alignment from the popular table-top game *Dungeons and Dragons* [42] to define characters' ideals. The two axes available are ethical (lawful–chaotic) and moral (good–evil); various combinations of these properties can be used to define each character's standards. To illustrate, Listing 6.9 shows some excerpts from the Fairytale domain's Belief Layer. Lawful agents internalise all the generic moral laws (corresponding to $Idl\varphi$ from Section 4.5.5), but chaotic agents do not; their personal ideals ($Idl_i\varphi$) are defined separately. For example, chaotic agents who are evil consider it ideal to kill others (lines 10–14), whereas chaotic agents who are good consider it unideal (lines 17–22).

```
1  % Lawful agents consider it ideal to follow general 'moral laws'
2  idl(Agent,Action) :-
3      idl(Action),
4      lawful(Agent).
5  idl(Agent,neg(Action)) :-
6      idl(neg(Action)),
7      lawful(Agent).
8
9  % Chaotic evil agents consider it ideal to kill others
10 idl(Agent,kills(Target)) :-
11     evil(Agent),
12     chaotic(Agent),
13     agent(Target),
14     Agent != Target.
15
16 % Chaotic good agents consider it unideal to kill others
17 idl(Agent,neg(kills(Target))) :-
18     agent(Agent),
19     good(Agent),
20     chaotic(Agent),
21     agent(Target),
22     not evil(Target).
```

Listing 6.9: Ideals for the Fairytale domain based on character alignment

6.4.2 Domain 2: Animals

Many fables are stories about animals rather than people. In light of this, we implement one story world with an animal focus. Table 6.7 provides a summary of this domain; again, the variable names used in fluent and event/action predicates indicate the parameter type. In this story world, the kinds of actions a character can successfully perform often depend on their size (i.e. whether they are `small`) and diet (`herbivore`, `carnivore`, or `omnivore`). Some ideals also depend on an animal's diet. For example, a carnivore considers it ideal to attack other animals, whereas a herbivore does not. All agents in this domain are considered edible, so animals can eat one another.

Element	No. of Instances	List of Instances
Characters	7	bear, bee, bird, lion, squirrel, toad, wolf
Character Properties	6	carnivore, edible, herbivore, omnivore, poisonous, small
Objects	7	berries, fish, honey, meat, nuts, pollen, vine
Object Properties	4	collectable, edible, poisonous, vegetarian
Fluents	5	has(Agent,Object), is_alive(Agent), is_free(Agent), is_hungry(Agent), is_in_snare(Agent)
Events/ Actions	14	attacks(Agent), catches(Agent), catches_fish, collects(Collectable), collects_pollen, eats(Edible), frees_from_snare(Agent), gets_hungry, gives_to(Agent,Object), lets_go(Agent), makes_honey, picks_vine, takes_from(Agent,Object), takes_meat_from_snare

Table 6.7: Summary of Animals story world

6.4.3 Domain 3: Family

The third story world is a family consisting of a mother, father, son, and daughter in a household environment. Table 6.8 provides a summary. As before, variable names represent parameter types; `is_clean(Cleanable/Washable)` means the fluent is defined both for objects that are `cleanable` and objects that are `washable`. Although there are fewer characters in this domain, there are considerably more objects and actions available. The distinguishing property for characters is whether they are an adult or a child; this determines many of their ideals and desires. For example, a child desires to have a toy, whereas an adult does not. To introduce more variety into the possible emotion patterns arising from this story world, we also make some distinctions between the boy's ideals and the girl's ideals. For example, the boy considers it ideal to play in the mud, whereas the girl does not.

Element	No. of Instances	List of Instances
Characters	4	boy, father, girl, mother
Character Properties	2	adult, child
Objects	15	breakfast, clothes_of(boy), clothes_of(girl), dinner, dishes, flower, icecream, lunch, picture, room_of(boy), room_of(girl), stove, toy, vase, window
Object Properties	8	breakable, buyable, cleanable, edible, furniture, meal, movable, washable
Fluents	6	has(Agent,Object), is_broken(Breakable), is_clean(Cleanable/Washable), is_hungry(Agent), is_hurt(Agent), on_high_shelf(Movable)
Events/ Actions	19	breaks(Breakable), buys(Buyable), cleans(Cleanable), cooks_for(Agent,Meal), draws_picture, eats(Edible), fixes(Breakable), gets_hungry, gets_off_high_shelf(Movable), gives_to(Agent,Object), messes_up(Cleanable), picks_flower, plays_in_mud, pushes_over_in_mud, puts_on_high_shelf(Movable), rides_bike, takes_from(Agent,Object), throws_cricket_ball, washes(Washable)

Table 6.8: Summary of Family story world

6.5 Running MOSS

In Section 6.2.7, we provided an example of the command used to run the Story Planner. To run the Story Planner in combination with the Text Generator, it is simply a matter of piping the *clasp* output from the Story Planner into the Text Generator Perl script. Assuming the same files names as those referenced in Section 6.2.7 (i.e. action.lp for the Action Layer, belief.lp for the Belief Layer, emotion.lp for the Emotion Layer, morals.lp for the Moral Layer, display.lp for the display file, and reward_t2.lp for the moral constraint), and given that the Text Generator Perl script is called write_story.pl, we run the complete system as follows:

```
gringo action.lp belief.lp emotion.lp morals.lp display.lp reward_t2.lp
    | clasp | ./write_story.pl
```

Again, this command is run from a terminal window, assuming *gringo*, *clasp*, and Perl

are installed on the machine and in the current path, and that all the *.lp files and the write_story.pl script are in the current directory.

6.6 Story Examples

After devoting the preceding sections to explaining how MOSS functions, in this section we present examples of stories generated by each of its moral rules. The relevant emotion filter has been applied in all cases, to remove superfluous emotions. We include examples of stories from each of the three story worlds described in Sections 6.4.1–6.4.3. We do not reproduce the ASP moral rules here (refer to Section 5.5). However, for each story we provide a table which summarises the moral's emotion requirements, and how they are fulfilled. We also make observations about each story that highlight aspects of the story generation process. We begin with Retribution in Section 6.6.1, followed by Greed in Section 6.6.2, Pride in Section 6.6.3, Realistic Expectations in Section 6.6.4, Recklessness in Section 6.6.5, and Reward in Section 6.6.6.

6.6.1 Retribution

The ASP rules for Retribution were provided in Section 5.5.2. There we identified two types of Retribution, and defined a separate rule for each. We provide an example story for both kinds, Type 1 and Type 2.

Retribution Type 1

Retribution Type 1 is the more general form of this moral; the original victim is not necessarily the one to bring about retribution for the protagonist. The relevant ASP rule was provided in Listing 5.23. Figure 6.6 shows a story generated using this rule, for the Animals domain. In this example, the original victim is the bear, but retribution for the protagonist (the lion) is carried out by the bee. Table 6.9 summarises the requirements of the ASP rule, and identifies which parts of the story correspond to each emotion to satisfy them.

Deep in the jungle there lived a bear, a lion and a bee. The lion hated the bee. The bee loved the bear.

Early one morning the lion attacked and killed the bear. As a result, the bear was dead. The bear felt distress that he was dead. The bear felt anger towards the lion about attacking him because he was dead. The bee started to hate the lion.

Not long afterwards, the bee attacked and killed the lion. As a result, the lion was dead. The lion felt distress that he was dead. The lion felt distress that the bee was alive.

Figure 6.6: Example story for Retribution Type 1

ASP Time	ASP Line	Description of ASP Constraint	Story Time	Matching Part of Story
T1	9	Victim feels Anger towards Protagonist.	1	The bear felt anger towards the lion about attacking him because he was dead.
T1	11	Protagonist does not feel any Joy after this point.	1	The lion does not feel any Joy after $T = 1$.
T2	10	Protagonist feels Distress.	2	The lion felt distress that he was dead.
T2	12	Protagonist does not feel Disappointment.	2	The lion does not feel Disappointment at $T = 2$.
T2	13	Protagonist feels Reproach towards some other agent.	2	The lion feels Reproach towards the bee for killing him, but this is removed by the Retribution Type 1 emotion filter.
T2	14	Victim does not feel any Distress.	2	The bear does not feel Distress at $T = 2$ (he is already dead).

Table 6.9: Retribution Type 1 emotions matched to story elements

This story highlights a few interesting points. What stands out most is the absurdity of a bee attacking and killing a lion. However, this is merely a consequence of our story world, in which we place no restrictions on which animals can kill each other. A more useful observation is the importance of the object-based emotions in story interpretation. If it was not stated at the beginning of the story that the bee loved the bear, readers would have difficulty understanding why the bee would then attack

and kill the lion. The other point, which came up often in reader feedback (as will be discussed in Chapter 7), is that both the bear and the lion continue to feel emotions after they are dead. As explained in Chapter 4, agents are permitted to feel emotions in the time-point immediately following their death, so the ASP solver is able to recognise death as an undesirable outcome (the ASP rules reference Distress, so Distress must arise for a story to match the rule). However, even though they are required by the Story Planner, it would be possible to filter these emotions during text generation, and not display them.

Retribution Type 2

Retribution Type 2 is the more specific definition, akin to payback, which requires reciprocal anger. The ASP rule for Retribution Type 2 was provided in Listing 5.24. We show a story generated by this rule in Figure 6.7, based on the Fairytale domain. There is some subtlety in how this story matches the moral rule. The princess is the protagonist, but it is the knight, not the wizard, who is the victim. Even though the princess kills the wizard, it is the knight who feels Anger about this (because of his love for the wizard), and it is the knight who then kills the princess, causing her to feel Anger in return. We explain how this story matches the ASP rule for Retribution Type 2 in Table 6.10.

> Once upon a time there lived a wizard, a knight and a princess. The knight loved the wizard.
>
> One summer's morning the knight picked a rose, but pricked his finger. As a result, he had a pricked finger and he had the rose.
>
> A short time later the princess killed the wizard. As a result, the wizard was dead. The knight felt distress that the wizard was dead. The knight felt anger towards the princess about killing the wizard because the wizard was dead. The knight started to hate the princess.
>
> Some time later the knight killed the princess. As a result, the princess was dead. The princess felt distress that she was dead. The princess felt anger towards the knight about killing her because she was dead.

Figure 6.7: Example story for Retribution Type 2

ASP Time	ASP Line	Description of ASP Constraint	Story Time	Matching Part of Story
T1	9	Victim feels Anger towards Protagonist.	2	The knight felt anger towards the princess about killing the wizard because the wizard was dead.
T2	10	Protagonist feels Anger towards Victim.	3	The princess felt anger towards the knight about killing her because she was dead.
T2	11	Victim does not feel any Distress.	3	The knight does not feel Distress at $T = 3$.

Table 6.10: Retribution Type 2 emotions matched to story elements

One thing that stands out about this story is the first event that takes place: the knight picking a rose. This event is completely irrelevant to the rest of the story. It has been included because the length of the story (three events) is more than what is necessary to convey the moral (only two distinct time-points are required by the ASP rule for Retribution Type 2). As such, the solver inserts an additional event at random. It does not matter that it has nothing to do with the rest of the story, provided that it does not interfere with the required emotion pattern. For this reason, longer stories can sometimes be less coherent than shorter stories. Also worth noting is that object-based emotions (in this case, Love) are set arbitrarily for the initial state, so stories that are very different from a stereotypical fairytale (such as the knight loving the wizard instead of the princess) are common.

6.6.2 Greed

Greed is another moral for which we provided two separate ASP rules, in Section 5.5.3. Here we similarly provide two example stories, one corresponding to Greed Type 1, and the other to Greed Type 2.

Greed Type 1

The ASP rule for Greed Type 1 was provided in Listing 5.25. The requirement is for an agent to achieve something they desire by performing a reproachable action, only to have that outcome reversed later. Figure 6.8 presents a story generated using this rule for the Family domain. In this example, the mother performs the greedy action,

by taking the father's dinner. She then loses the dinner, however, when her daughter takes it from her in the following time-step. Table 6.11 highlights how the elements in the story correspond to the ASP rule in Listing 5.25.

In a modern inner-city apartment there lived a father, a mother and a girl.

One day the mother cooked dinner for the father. As a result, the dishes were not clean and the father had dinner. The father and the girl started to love the mother.

The mother hoped that she would have dinner. Not long afterwards the mother took dinner from the father. As a result, she had dinner and the father didn't have dinner anymore. The mother felt satisfaction that she had dinner. The father started to hate the mother.

Some time later the girl took dinner from the mother. As a result, the mother didn't have dinner anymore. The mother felt distress that she didn't have dinner anymore.

Figure 6.8: Example story for Greed Type 1

Apart from the mother's somewhat irrational behaviour, in cooking dinner for the father only to take it from him, the most confusing part of this story is the mention of the dirty dishes. They are not referenced again, which is against the basic dramatic principle referred to as Chekhov's gun [94]. This principle states that a loaded gun should not be put on stage unless it is to be fired by the last act; i.e. only the necessary elements should be present in a story. The dishes play no role in the story, and therefore should not be mentioned at all. However, although the Text Generator filters irrelevant emotions, it does not filter consequences. As such, all the consequences defined for a given event, whether they are relevant to the story or not, are displayed if the event occurs. This can sometimes be confusing for readers.

ASP Time	ASP Line	Description of ASP Constraint	Story Time	Matching Part of Story
T1	9	Protagonist feels Satisfaction.	2	The mother felt satisfaction that she had dinner.
T1	10	Protagonist does not feel any Distress.	2	The mother does not feel Distress at $T = 2$.
T1	11	Another agent feels Reproach towards Protagonist.	2	The father feels Reproach towards the mother, but this is removed by the Greed Type 1 emotion filter.
T1	12	Protagonist feels Shame.	2	The mother feels Shame, but this is removed by the Greed Type 1 emotion filter.
T2	14	Protagonist feels Distress about the reversal of the outcome that satisfied them at T1.	3	The mother felt distress that she didn't have dinner anymore.
T2	14	No agent other than Protagonist feels Distress.	3	No-one other than the mother feels Distress at $T = 3$.

Table 6.11: Greed Type 1 emotions matched to story elements

Greed Type 2

The ASP rule for Greed Type 2, as shown in Listing 5.26, requires the protagonist to perform two shameful actions in sequence, each resulting in Satisfaction, and then to experience Distress due to another agent's action. An example story, based on the Fairytale domain, is provided in Figure 6.9. The knight is the protagonist; first he steals an axe from a dwarf, and then he steals treasure from a dragon. As punishment for his greed, he is killed by the dragon. We highlight how the story satisfies the corresponding ASP rule in Table 6.12.

The story in Figure 6.9 illustrates why the future-directed emotions experienced at time T are described before the event. In the second paragraph, the knight hopes for the axe, then steals it. In a reader's mind, the knight's hope for the axe is what prompts him to steal it. The same applies in the third paragraph, when the knight hopes for and then steals the treasure. In this way, Hope and Fear can provide motivation for characters' actions.

Long ago in a far away land there lived a knight, a dragon and a dwarf.

The knight hoped that he would have the axe. One summer's morning the knight stole the axe from the dwarf. As a result, he had the axe and the dwarf didn't have the axe anymore. The knight felt satisfaction that he had the axe. The knight felt shame about stealing the axe. The dwarf started to hate the knight. The dragon started to love the knight.

The knight hoped that he would have the treasure. Later that day the knight stole the treasure from the dragon. As a result, he had the treasure and the dragon didn't have the treasure anymore. The knight felt satisfaction that he had the treasure. The knight felt shame about stealing the treasure. The dragon started to hate the knight.

The day after that, the dragon killed the knight. As a result, the knight was dead. The knight felt distress that he was dead.

Figure 6.9: Example story for Greed Type 2

The emotion that does not make sense in this story is the dragon starting to love the knight, at the end of the second paragraph. This occurs because our emotion model defines Love and Hate based on Admiration and Reproach (refer to Section 4.4.4). The dragon, which has been defined in the Fairytale world as an evil and chaotic agent, considers theft to be ideal. As such, the dragon admires the knight's action of stealing from the dwarf; this Admiration causes Love. However, for readers without an understanding of the underlying emotion model, this is confusing. We discuss this further in Chapter 7, with reference to survey participant feedback (refer to Section 7.6.2).

ASP Time	ASP Line	Description of ASP Constraint	Story Time	Matching Part of Story
T1	10	Protagonist feels Satisfaction.	1	The knight felt satisfaction that he had the axe.
T1	11	Protagonist feels Shame.	1	The knight felt shame about stealing the axe.
T2	12	Protagonist feels Satisfaction.	2	The knight felt satisfaction that he had the treasure.
T2	13	Protagonist feels Shame.	2	The knight felt shame about stealing the treasure.
T3	14	Protagonist feels Distress.	3	The knight felt distress that he was dead.
T3	15	No agent other than Protagonist feels Distress.	3	No-one other than the knight feels Distress at $T = 3$.
T3	16	Protagonist does not feel any Joy.	3	The knight does not feel Joy at $T = 3$.
T3	17	Protagonist feels Reproach towards some other agent.	3	The knight feels Reproach towards the dragon, but this is removed by the Greed Type 2 emotion filter.

Table 6.12: Greed Type 2 emotions matched to story elements

6.6.3 Pride

The ASP rule for Pride is defined in Listing 5.27, in Section 5.5.4. It requires an agent to feel proud of an action they perform, and satisfied about its outcome, only to have that outcome reversed as punishment for their pride. Figure 6.10 presents a story satisfying this rule, based on the Animals domain. The bee is the protagonist, exhibiting Pride about producing honey. She is punished for her Pride when the honey is taken from her by the toad. Table 6.13 shows how this story matches the ASP rule from Listing 5.27.

The story provided in Figure 6.10 demonstrates a side-effect of one of our implementation choices in the Text Generator. As explained in Section 6.2.4, we only include events and emotions in the Story Planner output, to avoid displaying the truth value of every fluent in the domain. Therefore, in order to display the consequences of events, the Text Generator obtains them from emotion predicates. For example, the second parameter of the predicate `joy(bee,becametrue(has(bee,honey)),2)` represents

the relevant consequence, `becametrue(has(bee,honey))`. This is why, in Figure 6.10, the second paragraph does not end with: "As a result, the bee had pollen." The bee did not feel any emotions about obtaining the pollen, and thus the Text Generator was unable to retrieve this consequence from the *clasp* output. However, we consider our approach to be sufficient because the most important consequences in a story (i.e. those that are relevant to its moral) incite emotions. It is these emotions that constitute our moral rules.

In a forest there lived a toad and a bee.

One warm afternoon the bee collected some pollen.

The next day the bee made honey. As a result, she had honey. The bee felt joy that she had honey. The bee felt pride about making honey.

The day after that, the toad took honey from the bee. As a result, he had honey and the bee didn't have honey anymore. The bee felt distress that she didn't have honey anymore.

Figure 6.10: Example story for Pride

ASP Time	ASP Line	Description of ASP Constraint	Story Time	Matching Part of Story
T1	8	Protagonist feels Pride.	2	The bee felt pride about making honey.
T1	9	Protagonist feels Joy.	2	The bee felt joy that she had honey.
T2	10	Protagonist feels Distress about the reversal of the outcome that caused them Joy.	3	The bee felt distress that she didn't have honey anymore.
T2	11	No agent other than Protagonist feels Distress.	3	No-one other than the bee feels Distress at $T = 3$.
T3	12	Protagonist feels Reproach towards some other agent.	3	The bee feels Reproach towards the toad, but this is removed by the Pride emotion filter.

Table 6.13: Pride emotions matched to story elements

6.6.4 Realistic Expectations

Realistic Expectations is expressed through the protagonist attempting an unrealistic action, and failing. The relevant ASP rule was provided in Section 5.5.5 (refer to Listing 5.28). The story in Figure 6.11 was generated using this rule. In this story, a dwarf attempts to kill a unicorn, but is unsuccessful. We relate this story to the ASP rule for Realistic Expectations in Table 6.14.

> In a distant kingdom there lived a unicorn and a dwarf. The dwarf hated the unicorn.
>
> The dwarf hoped that the unicorn would not be alive. One day the dwarf tried to kill the unicorn, but failed. The dwarf felt disappointment that the unicorn was alive.

Figure 6.11: Example story for Realistic Expectations

ASP Time	ASP Line	Description of ASP Constraint	Story Time	Matching Part of Story
T1	8	Protagonist does not feel any Joy.	0	The dwarf does not feel Joy at $T = 0$.
T2	9	Protagonist feels Disappointment.	1	The dwarf felt disappointment that the unicorn was alive.
ALL	10	Protagonist does not feel Satisfaction at any time during the story.	0–1	The dwarf does not feel Satisfaction at any time during the story.

Table 6.14: Realistic Expectations emotions matched to story elements

6.6.5 Recklessness

We defined two ASP rules for Recklessness, in Section 5.5.6. We provide an example story generated by each rule in the following sections.

Recklessness Type 1

The ASP rule for Recklessness Type 1 was provided in Listing 5.29. We present a story generated using this rule, for the Fairytale domain, in Figure 6.12. The rule

is based on the protagonist experiencing both Satisfaction and Distress as a result of their own action. In this story, the troll eats a mushroom to satisfy his hunger, without considering that the mushroom could be poisonous. He dies as a result. We show how this story conforms to the relevant ASP rule in Table 6.15.

Once upon a time there lived a troll.

One day the troll picked a poisonous mushroom.

Later that day the troll became hungry.

The troll hoped that he would not be hungry. Some time later the troll ate the mushroom. The troll felt satisfaction that he was not hungry. The troll felt distress that he was dead.

Figure 6.12: Example story for Recklessness Type 1

ASP Time	ASP Line	Description of ASP Constraint	Story Time	Matching Part of Story
T	6	Protagonist feels Satisfaction.	3	The troll felt satisfaction that he was not hungry.
T	7	Protagonist feels Distress.	3	The troll felt distress that he was dead.
T	8	Protagonist does not feel any Admiration.	3	The troll does not feel Admiration at $T = 3$.
T	9	Protagonist does not feel any Reproach.	3	The troll does not feel Reproach at $T = 3$.

Table 6.15: Recklessness Type 1 emotions matched to story elements

The problem readers identified with this story was that it is not clear whether the troll knew the mushroom was poisonous. This affects a reader's interpretation of the story, and whether they perceive the troll's action as reckless. We discuss this further in Chapter 7 (refer to Section 7.6.13).

Recklessness Type 2

Recklessness Type 2 is based on the protagonist performing an action with a negative consequence, even though they anticipate that consequence. The relevant ASP rule

was provided in Listing 5.30. The example story in Figure 6.13 is set in the Family domain. A boy fears breaking a vase, but despite this he attempts to get it off a high shelf—thereby breaking the vase. Table 6.16 illustrates how this story matches the ASP rule from Listing 5.30.

> In a modern inner-city apartment there lived a boy.
>
> The boy feared that the vase would be broken. One chilly autumn day the boy tried to get the vase off a high shelf, but failed. As a result, he was hurt and the vase was broken. The boy felt his fears were confirmed that the vase was broken.

Figure 6.13: Example story for Recklessness Type 2

ASP Time	ASP Line	Description of ASP Constraint	Story Time	Matching Part of Story
T	6	Protagonist feels FearsConfirmed.	1	The boy felt his fears were confirmed that the vase was broken.
T	7	Protagonist feels Remorse.	1	The boy feels Remorse, but this is removed by the Recklessness Type 2 emotion filter.

Table 6.16: Recklessness Type 2 emotions matched to story elements

6.6.6 Reward

We define two ASP rules for Reward, analogous to those defined for Retribution. They are presented in Section 5.5.7. We provide example stories generated using both of these rules below.

Reward Type 1

Reward Type 1, defined in Listing 5.31, is more general. The protagonist's reward for a good deed does not necessarily have to come from the original beneficiary. We provide an example story conforming to this rule in Figure 6.14, based on the Family domain. In this case, the girl performs a good deed by cooking breakfast for her mother. She

is rewarded by the boy, who then cooks her lunch. Table 6.17 matches the elements of this story to the ASP rule for Reward Type 1.

> In a nice house on a quiet street there lived a boy, a mother and a girl. The boy loved the girl.
>
> One chilly autumn day the girl became hungry.
>
> A short time later the girl cooked breakfast for the mother. As a result, the dishes were not clean and the mother had breakfast. The mother felt joy that she had breakfast. The mother felt gratitude towards the girl about cooking breakfast for her because she had breakfast. The mother started to love the girl.
>
> Soon after that, the boy cooked lunch for the girl. As a result, the girl had lunch. The girl felt joy that she had lunch.

Figure 6.14: Example story for Reward Type 1

ASP Time	ASP Line	Description of ASP Constraint	Story Time	Matching Part of Story
T1	9	Beneficiary feels Gratitude.	2	The mother felt gratitude towards the girl about cooking breakfast for her because she had breakfast.
T1	11	Protagonist does not feel any Distress after this point.	2	The girl does not feel any Distress after $T = 2$.
T2	10	Protagonist feels Joy.	3	The girl felt joy that she had lunch.
T2	12	Protagonist does not feel Relief.	3	The girl does not feel Relief at $T = 3$.
T2	13	Protagonist feels Admiration towards some other agent.	3	The girl feels Admiration towards the boy, but this is removed by the Reward Type 1 emotion filter.
T2	14	Beneficiary does not feel any Joy.	3	The mother does not feel Joy at $T = 3$.

Table 6.17: Reward Type 1 emotions matched to story elements

This example exhibits two features which occurred by chance, but actually improve the story. The first is the fact that breakfast is referenced before lunch. There are no constraints in the Family domain on when particular meals should occur relative to one another, so it is quite conceivable for a character to prepare dinner, only to have someone make them breakfast later that day. In this case, the appropriate ordering makes the story coherent. The second feature is the superfluous event of the girl becoming hungry. The story still matches the moral rule if this event is removed, but the fact that the girl cooked breakfast for her mother even though she herself was hungry actually serves to emphasise the selflessness of her good deed.

Reward Type 2

Reward Type 2 is more specific than Type 1: the ASP rule provided in Listing 5.32 requires reciprocal Gratitude. The story in Figure 6.15 exemplifies this moral using the Animals domain. The squirrel is the original beneficiary: she feels Gratitude towards the wolf, the story's protagonist, for releasing her. To reward him, the squirrel then kills the toad, whom the wolf hates. We illustrate how this story conforms to the ASP rule for Reward Type 2 in Table 6.18.

In a lush valley there lived a toad, a squirrel and a wolf. The wolf hated the toad.

One chilly autumn day the wolf caught the squirrel. As a result, he had the squirrel and the squirrel was not free. The squirrel started to hate the wolf.

A short time later the wolf let go of the squirrel. As a result, the squirrel was free. The squirrel felt joy that she was free. The squirrel felt gratitude towards the wolf about letting her go because she was free. The squirrel started to love the wolf.

More time passed, and the squirrel attacked and killed the toad. As a result, the toad was dead. The wolf felt joy that the toad was dead. The wolf felt joy that the squirrel was alive. The wolf felt gratitude towards the squirrel about attacking the toad because the toad was dead.

Figure 6.15: Example story for Reward Type 2

ASP Time	ASP Line	Description of ASP Constraint	Story Time	Matching Part of Story
T1	9	Beneficiary feels Gratitude.	2	The squirrel felt gratitude towards the wolf about letting her go because she was free.
T2	10	Protagonist feels Gratitude.	3	The wolf felt gratitude towards the squirrel about attacking the toad because the toad was dead.
T2	11	Beneficiary does not feel any Joy.	3	The squirrel does not feel Joy at $T = 3$.

Table 6.18: Reward Type 2 emotions matched to story elements

We will make two remarks about this story, both with reference to object-based emotions. The story shows how sophisticated plot lines can emerge from very simple rules, as a result of the relationships between characters. The squirrel's action of killing the toad, which serves as the wolf's reward, has no direct effect on the wolf. However, it nevertheless causes him Joy, because he hated the toad. Without any reference to emotions, the events in this story would not make sense. The story also highlights how our simplistic definitions of Love and Hate (refer to Section 4.4.4) result in emotionally fickle characters: the squirrel hates the wolf at T = 1, then loves him at T = 2. A more sophisticated model of these two emotions would lead to significantly more realistic characters. We discuss this further in Chapter 9.

6.7 Limitations

MOSS has a number of limitations, which we outline in this section. Many of them stem from the limitations of our OCC emotion model, which were described in Section 4.6.5. We will mention those in passing here for completeness, but do not go into depth; they were covered in sufficient detail in Chapter 4. The system's limitations tie in to the areas for future work which will be presented in Chapter 9.

We begin with a brief review of the limitations of our OCC emotion model. The first of these, which ties back to the underlying action framework, is that we do not permit simultaneous events, which restricts the complexity of the stories that can be generated. The simplistic omniscient belief model is another significant limitation, as many interesting storylines can emerge from characters having incorrect beliefs. There

are also limitations in the emotion definitions themselves. Our definitions of Love and Hate are very simplistic, and the fact that we do not model emotion intensity restricts the representational finesse of the model. The reduced scope of the prospect-based emotions in our implementation as compared, to the original OCC descriptions, inhibits modelling retrospective Hope and Fear. Finally, emotions can only be experienced immediately following the event that caused them, which prevents characters from being able to experience retrospective emotions about earlier events.

As mentioned in Section 6.2.6, story generation in MOSS is controlled exclusively at the plot level. There are no rules that require characters to behave rationally. We do not model goals at all, and desires and ideals are implemented only to facilitate the emotion model; they do not place any restrictions on characters' behaviour. For example, consider the MOSS-generated story presented in Figure 6.16. The knight's action does not make any sense if he loves the fairy, but there are no constraints defined in the Story Planner to prevent it. Sometimes, irrational behaviour may be desirable in a story, but we still identify this as a limitation, because there is no simple way to enforce coherent character behaviour. Achieving this effectively would require a considerably more sophisticated character model than that which we have implemented.

> Long ago in a far away land there lived a knight and a fairy. The knight loved the fairy.
>
> The knight feared that the fairy would not be alive. One summer's morning the knight killed the fairy. As a result, the fairy was dead. The knight felt his fears were confirmed that the fairy was dead.

Figure 6.16: Example of irrational character behaviour

Another limitation is that *clasp* will produce many solutions which are nearly identical. For example, consider two stories which contain exactly the same sequence of actions, only performed by different characters, who nevertheless feel exactly the same emotions. These stories are different from the solver's perspective, but for a reader, they convey the same plot, and are essentially the same story. Consequently, obtaining truly different stories (i.e. with a different plot) is not as simple as requiring *clasp* to produce multiple solutions. To obtain a variety of storylines for evaluation, we run the system multiple times, but progressively introduce additional constraints that prevent the sequence of actions generated by the previous run from being generated again. We

describe this approach in more detail in Chapter 7.

MOSS stories do not explicitly express causality. For simple stories which do not have superfluous events this is not a serious limitation, as causality can be inferred. For example, if Bob steals from Alice, and then Alice punches him immediately afterwards, most readers will infer that Alice punched Bob because he stole from her (i.e. in retaliation). However, in more complex stories, which may contain many other events between Bob's theft and Alice punching him, this kind of inference by the reader cannot always be assumed. The stories we currently deal with are relatively simple, and thus tracing causality is not critical for them to be understood, but it is nevertheless an important consideration if MOSS were to be used for generating more sophisticated stories.

Finally, as mentioned in Section 6.2.6, due to the nature of ASP (i.e. that the problem definition must be completely grounded to find a solution), the exact length of the story must be specified in advance, through the time declaration. Thus, a single run of the system cannot generate stories of varying length. This is not a limitation that can be readily resolved, however, as long as ASP is used for the underlying planner. We simply need to run MOSS multiple times, for each desired story length. This can be automated using an external script if the time declaration is provided as a separate file; we do this when generating stories of varying length for testing.

6.8 Summary

In this chapter, we described the overall architecture of the Moral Storytelling System (MOSS), including the ASP Story Planner (which combines the Action, Belief, Emotion, and Moral Layers described in Chapters 4 and 5) and the Text Generator Perl script. We used a running example of a simple story domain to illustrate how each component contributes to story generation. After providing an overview of the three story worlds we implement, a fairytale world, an animal world, and a family world, we presented examples of stories generated using each of MOSS's moral rules. For each story, we explained how it conforms to the corresponding rule, and identified story features that highlight key aspects of the story generation process. Finally, we identified and discussed some key system limitations. In the following chapter, we describe our evaluation process for MOSS-generated stories.

Chapter 7

Evaluation

Everything that can be counted doesn't
necessarily count; everything that counts
cannot necessarily be counted.

Albert Einstein

One of the greatest challenges in the field of computational storytelling is evaluation [180]. There is no right or wrong in storytelling. Instead, there is an inherent subjectivity: a story loved by one person might be hated by another. This makes objective evaluation impossible. Nevertheless, some form of evaluation is necessary to determine whether we have achieved our aims. The only way to judge whether a story conveys a moral is for people to read it. If they understand the moral, then the story was successful.

However, it is not quite so simple. Say a reader does not perceive the correct moral in an automatically generated story. How do we know whether this is due to a flaw in our system, or simply a consequence of the selected story world, which may be too restrictive to represent that particular moral effectively? How do we know a human-authored story would perform any better? The answer is, we don't. Even human-authored stories would not be expected to convey each moral perfectly to every reader; there is too much ambiguity in storytelling and story interpretation. We need a way to assess our system fairly, without implicitly expecting it to outperform a human storyteller.

In this chapter, we present our approach to evaluating the Moral Storytelling System (MOSS): a survey-based evaluation comparing MOSS-generated stories to human-

authored stories and deliberately moral-free event sequences. Section 7.1 provides an overview of the experiment design, along with our hypothesis. We begin by generating stories using each of MOSS's moral rules, as outlined in Section 7.2. This is Stage 1 of the evaluation process. In Section 7.3, we describe Stage 2, which involves obtaining human-authored stories for each of the MOSS domains. Section 7.4 moves on to Stage 3, and explains the structure and format of our survey. We present the results in Section 7.5, and discuss qualitative participant feedback in Section 7.6. We acknowledge the limitations of our approach in Section 7.7, and discuss the implications of our results in Section 7.8. We conclude with a summary of this chapter in Section 7.9.

7.1 Experiment Design

The crux of our evaluation process is to have human participants read a selection of MOSS-generated stories, and identify the moral each story conveys.[1] The proportion of stories classified correctly provides a measure of system performance. However, the difficulty is that story interpretation is extremely subjective. Even human-authored stories would not be expected to score perfectly in such a test, making it hard to define good performance. To address this, we require baselines for comparison. Given that people have an innate storytelling ability [148], human-authored stories can be considered an upper threshold on storytelling performance. Deliberately moral-free sequences of events can serve as the lower threshold. Comparing our system to both allows us to assess its performance.

We divide our evaluation process into three stages. Stage 1 involves producing a range of system-generated stories using MOSS. This includes both stories with morals, and deliberately moral-free sequences of events. The moral-free event sequences are generated using constraints which prevent any emotion patterns corresponding to our moral rules from occurring. In Stage 2, human authors are asked to compose stories which convey the selected morals, within the same story worlds used by MOSS. Stage 3 is the online survey in which participants read a random selection of stories (including MOSS-generated stories, human-authored stories, and moral-free event sequences), and attempt to classify them according to their moral. The evaluation process is shown as a flowchart in Figure 7.1.

[1]This study was approved by UNSW Human Research Advisory Panel 'H,' with approval number 08/2013/05.

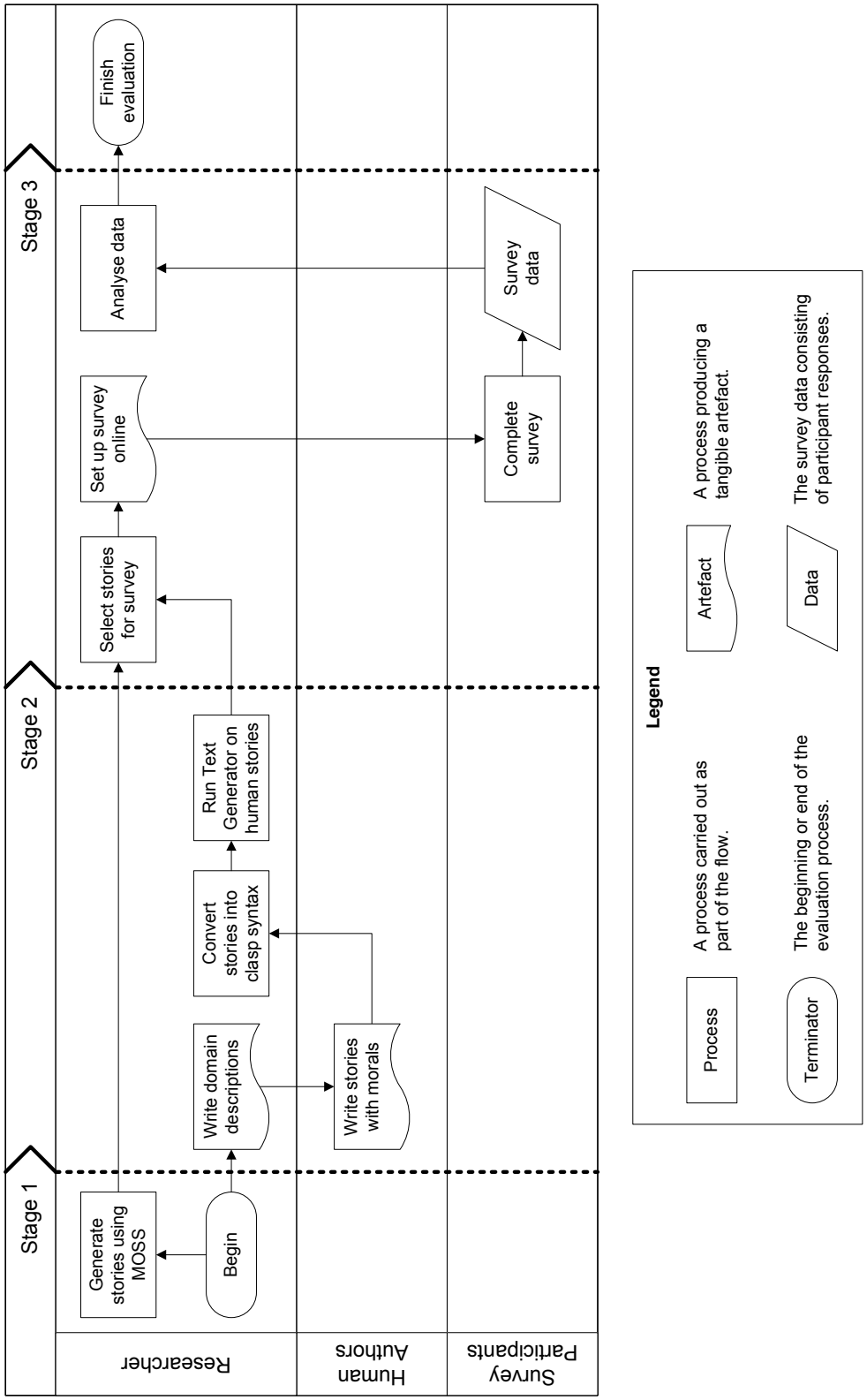

Figure 7.1: Flowchart of the evaluation process

7.1.1 Hypothesis

Due to people's innate storytelling ability, we expect human-authored stories to most effectively convey the intended morals. Conversely, moral-free sequences of events should perform the worst, with no moral distinguished by readers most of the time. MOSS-generated stories are expected to lie between these two extremes, but much closer to the upper threshold of human story performance. We expect similar patterns for story coherence and interest: human-authored stories should come out in front, but MOSS-generated stories should be more coherent and more interesting than the moral-free event sequences, which enforce no plot-level structure.

7.2 Stage 1 - Automatically Generated Stories

As part of Stage 1, we produced stories with morals, which we will refer to as MOSS-generated stories for the remainder of this work, and also deliberately moral-free sequences of events, which we will call moral-free stories. Both sets of stories were automatically generated using MOSS, but with different constraints on the solutions. We explain the story generation process in depth for each set in Sections 7.2.1 and 7.2.2.

7.2.1 MOSS-Generated Stories

To produce MOSS-generated stories, we follow the process described in Chapter 6. The additional consideration in producing stories for evaluation, however, is that the examples selected for a given moral should be distinctly different from one another. Evaluating four Retribution stories, for instance, which all consist of exactly the same sequence of events performed by different combinations of characters, is not very informative. As identified in Section 6.7, one of the limitations of the system is that a very large number of near-identical stories are produced. In some cases, the first few hundred solutions may differ only in which characters or objects are involved (due to the nature of the solving process, similar solutions are found first, before significantly different solutions are produced).

To overcome this hurdle, we run the solver multiple times, producing only one solution each time, but progressively adding constraints that prevent the previous event sequence from being generated again (regardless of the characters/objects involved). To illustrate, consider the following example: a knight steals treasure from a dragon, and

then the dragon kills the knight. This story involves two events. We show the `happens` predicates corresponding to this story in Listing 7.1 (we do not show the other predicates that would be part of the *clasp* output, as they are not necessary for our explanation). From this output, we produce the constraint shown in Listing 7.2. This constraint says that stories in which an agent, `Agent1`, steals an object from some other agent, `Agent2`, after which `Agent2` kills `Agent1`, are not acceptable solutions. By adding this constraint to the ASP code the next time the solver is run, we guarantee any further solutions generated will be distinctly different from the first one. This process is repeated until either no solutions remain, or 200 distinct solutions have been produced.

```
1 happens(knight,steals_from(dragon,treasure),0).
2 happens(dragon,kills(knight),1).
```

Listing 7.1: Example happens predicates

```
1 :- happens(Agent1,steals_from(Agent2,Object1),0),
2    happens(Agent2,kills(Agent1),1),
3    agent(Agent1),
4    agent(Agent2),
5    object(Object1).
```

Listing 7.2: Constraint produced from Listing 7.1

We also include stories of varying length in our survey. Although for any given moral rule there is a minimum story length (equal to the number of unique time-points required by the rule), longer stories can also satisfy the requisite emotion patterns. In stories of minimum length, every event is necessary to convey the moral, but in longer stories, this is not necessarily the case; some events may be superfluous. The distraction of superfluous events may affect readers' understanding of morals. As such, we ran the solver with several different time declarations for each moral, generating stories ranging in length from 1 to 5.[2] This was repeated for each of the three storytelling domains described in Sections 6.4.1–6.4.3. At the conclusion of this process, we had a large pool of unique stories for each moral (more precisely, for each moral type, in the case of morals with two different ASP rules) of varying length.

[2]Stories of length 1 can only exist for Realistic Expectations and Recklessness, because all the other moral rules require emotions to occur during at least 2 unique time-points.

7.2.2 Moral-Free Stories

Moral-free sequences of events, presented as stories, serve as the lower baseline for our evaluation. Our goal is to determine whether MOSS-generated stories, by enforcing moral-specific emotion sequences, convey morals better than stories without those emotion sequences. Thus, what we require from our moral-free stories is that they do not contain emotion patterns which match any of our moral rules. To achieve this, we use a set of constraints which preclude any of our moral definitions from occurring in the solutions; these constraints are shown in Listing 7.3.[3] Apart from these constraints, no other restrictions are placed on moral-free stories at the plot level. As with the MOSS-generated stories, we run the system multiple times, progressively adding constraints to prevent near-identical stories from being repeatedly generated, and also produce stories of varying length.

```
1  :- retribution_t1.
2  :- retribution_t2.
3
4  :- greed_t1.
5  :- greed_t2.
6
7  :- pride.
8
9  :- realistic_expectations.
10
11 :- recklessness_t1.
12 :- recklessness_t2
13
14 :- reward_t1.
15 :- reward_t2.
```

Listing 7.3: Constraint set used to generate moral-free stories

[3]Note that the `maxtime` condition was removed from the moral rules for use in generating moral-free stories, because otherwise the emotion patterns would be permitted, as long as the last emotion did not occur in the last time-point. The goal was to prevent them from occurring anywhere in the story.

As was explained in Chapter 6 (refer to Section 6.3.3), the MOSS Text Generator uses moral-specific emotion filters to determine which emotions to display in the story discourse, so readers are not overwhelmed with excessive emotion descriptions. This introduces another variable: whether the choice of emotions to display plays a significant role in a reader's understanding of the moral of a story. To take this into account, we produce moral-free stories using each of the moral-specific filters described in Section 6.3.3, and also some using no filter (i.e. all emotions included in the story text).

Figure 7.2 shows an example of a moral-free story. The events are completely unrelated, resulting in an extremely disjoint narrative that does not make any sense. The text in this case has been generated using the Greed Type 1 emotion filter, which permits Hope, Satisfaction, and Distress (refer to Table 6.5). Unlike for MOSS-generated stories, where moral-specific emotions are displayed only in the time-points matching the relevant ASP rule, for moral-free stories they are displayed whenever they occur. There are no moral predicates (e.g. `greed_t1`) in the clasp output for moral-free stories, since these stories do not satisfy any of the moral rules, so the Text Generator cannot match emotions to specific time-points.

Long ago in a far away land there lived a unicorn and a dwarf. The unicorn hated the dwarf.

One day the unicorn became hungry. The unicorn felt distress that he was hungry.

A short time later the dwarf picked a poisonous mushroom.

The dwarf hoped that he would have the rose. The day after that, the dwarf picked a rose. As a result, he had the rose. The dwarf felt satisfaction that he had the rose.

Figure 7.2: Moral-free story produced using the Greed Type 1 emotion filter

7.3 Stage 2 - Human-Authored Stories

Stage 2 of our evaluation process involves obtaining human-authored stories to compare to those that were MOSS-generated. Simply asking people to write stories conveying a given moral with no other restrictions would result in a variety of stories on a wide range of topics. MOSS, on the other hand, has limited domains to work with. To ensure a fair comparison, participants were provided with detailed descriptions of the MOSS story worlds, including the available characters, objects, fluents, and possible actions (including their preconditions and consequences), as well as a list of emotions characters could feel. The domain descriptions corresponding to each of the three story worlds described in Chapter 6 are included in Appendix D. Participants were instructed to write stories conveying the given morals, but only within the supplied domain restrictions.

However, even within the same domains, the language of a story written by a person is immediately distinguishable from auto-generated text. We are not aiming to evaluate the discourse, so to eliminate this bias, we asked participants to provide stories as a list of events, their consequences, and the resulting character emotions. We then manually expressed these stories in *clasp* output format, so they could be passed as input to the MOSS Text Generator, thus automatically generating the text. To demonstrate this process, Figure 7.3 presents a transcript of one Reward story provided by a human author for the Fairytale domain. The same story once it has been translated into *clasp* output syntax is shown in Figure 7.4.[4] The resulting text, produced when this is passed to the Text Generator, is provided in Figure 7.5. No emotion filter was applied to the human-authored stories, to give people complete flexibility in which emotions they considered relevant. This creates a level playing field, as both the human participants and MOSS are working within the same domains and with the same style of discourse.

[4]Note that the predicate list was provided to the Text Generator as a single, space-separated line, as per *clasp*'s output format. We add line breaks and spacing in Figure 7.4 for readability.

The Fairy hopes for a rose.
The Knight picks a rose and pricks his finger.
The Knight gives the Fairy the rose.
The Fairy feels joyous.
The Fairy heals the Knights finger.
The Knight feels grateful.

Figure 7.3: Human-authored story for Reward

```
Answer: 1
hope(fairy,becametrue(has(fairy,rose)),0)
happens(knight,picks_rose,0)
succeeds(knight,picks_rose,0)

happens(knight,gives_to(fairy,rose),1)
succeeds(knight,gives_to(fairy,rose),1)

joy(fairy,becametrue(has(fairy,rose)),2)
happens(fairy,heals(knight),2)
succeeds(fairy,heals(knight),2)

gratitude(knight,fairy,heals(knight),becametrue(neg(has_pricked_finger(knight))),3)
```

Figure 7.4: Hand-written clasp output corresponding to Figure 7.3

Long ago in a far away land there lived a knight and a fairy.

One day the knight picked a rose, but pricked his finger.

The next day the knight gave the rose to the fairy. As a result, the fairy had the rose. The fairy felt joy that she had the rose.

The day after that, the fairy healed the knight. As a result, the knight did not have a pricked finger. The knight felt gratitude towards the fairy about healing him because he did not have a pricked finger.

Figure 7.5: Text generated from the clasp output in Figure 7.4

Five independent human authors participated in Stage 2, producing a total pool of 143 stories: 48 for the Fairytale domain, 51 for the Animals domain, and 44 for the Family domain.[5] Authors were free to write as many stories as they wished, for whichever morals they chose. In Table 7.1, we summarise the number of stories provided for each moral by domain. This comprised the total human-authored pool, from which we selected stories to include in the survey. For those domain/moral combinations where we had more stories available than required, 4 stories were selected at random, as will be described in Section 7.4.1.

Moral	Fairytale	Animals	Family	ALL DOMAINS
Retribution	5	5	4	14
Greed	8	9	8	25
Pride	9	9	7	25
Realistic Expectations	10	11	9	30
Recklessness	10	11	8	29
Reward	6	6	8	20
TOTAL	48	51	44	143

Table 7.1: Number of human-authored stories by domain and moral

7.3.1 Author Feedback

In addition to their stories, we also gathered informal feedback from the human authors about the task. In particular, authors were encouraged to provide comments about any aspects of the task they found difficult, but they were free to provide feedback of any nature. The following notable points were raised:

- Some authors had difficulty writing stories for certain morals, due to the limited number of actions available, and associated conditions. As a result, they felt that the stories for a given moral tended to be quite similar to each other. This is not surprising. People are naturally used to writing stories without having to

[5]These numbers include duplicates; i.e. when different authors supplied identical or near-identical stories (by near-identical we mean stories in which the event sequences are the same, but different characters are involved). Also note that one story was provided for the Fairytale domain which the author labelled as conveying both Reward and Retribution (for different characters). This story was counted as Reward in Table 7.1, because that was the first moral listed by the author.

follow such strict rules, which is the reason we provided domain restrictions in the first place: to ensure a consistent scope between MOSS-generated stories and human-authored stories. The reduced number of possible stories is an expected side-effect of limited story domains. Building richer, more complex story worlds would expand the story possibilities available for each moral.

- Authors had differing views about how morals should be conveyed. For instance, in the case of Retribution, one author believed the emotions a character felt due to their reprehensible action inherently served as retribution for that action. The example they gave (based on the Family domain) was if the son threw a cricket ball, which is a potentially destructive action, and then felt a negative emotion after breaking the window, he was getting what he deserved through experiencing that emotion. On the other hand, a different author had trouble writing Retribution stories for the Family domain, because they felt that the majority of available negative outcomes resulted from accidents, which meant they were not applicable for Retribution. This is an interesting viewpoint, because it implies that for Retribution to be conveyed, the punishment must result from an intentional action. This is not the case in many of Aesop's fables, where Retribution can be served through natural events. This clearly demonstrates the subjectivity inherent in storytelling, and how much it can affect interpretation. The second author would undoubtedly not have understood Retribution from any of the first author's stories.

- One author commented that in order to convey Greed, a character would have to desire more of an item they already possess, and attempt to obtain it. This cannot be conveyed within any of our domains, due to the restriction that a character can only possess one of any particular object at a time (this was discussed in Section 4.5.4). This is an example of a domain limitation which directly impacts the system's capability of conveying certain morals, and will be discussed as a possible extension in Chapter 9.

- Some authors found themselves writing the same (or extremely similar) stories for different morals, in particular Realistic Expectations and Recklessness. This supports the notion of an inherent similarity between these two morals, as indicated by the ILP results presented in Chapter 5, at least in the context of our limited domains.

- One author interpreted Pride as hubris,[6] in the sense that any expectation of success where it was not assured was certain to lead to a downfall. Their assumption was that if a character attempted an action, that meant they were confident of its success, so any action failure, and the resultant shame, was a demonstration of Pride. This is another example of subjectivity in interpretation; not every reader would necessarily make the same assumptions, without which the moral in these stories cannot be understood.

7.4 Stage 3 - Survey

The final stage of our evaluation process is to present the stories produced in Stages 1 and 2 to readers, and ask them to identify the moral of each. We do this using an online survey. Our approach is based on a similar philosophy to a simple Turing test [169], in that participants are not told which stories are automatically generated, and which are human-authored. The uniformity of language across the board, due to the use of a consistent text generation process for all stories (as described in Section 7.3), removes any bias at the discourse level, ensuring that our evaluation focusses on the fabula. In Section 7.4.1, we explain the story selection process, followed by a description of the survey format in Section 7.4.2. We outline the recruitment process and summarise the demographic information of participants in Section 7.4.3.

7.4.1 Story Selection

Our total pool of stories, as described in Sections 7.2 and 7.3, consists of stories spread across three different domains and six morals. To ensure we do not introduce a bias towards particular domains, we need to ensure our survey contains an even spread of stories for each moral from every domain. We also need to ensure the stories are uniformly distributed between MOSS-generated stories, human-authored stories, and moral-free stories.

To allow us to include stories of varying length, and examples of both Type 1 and Type 2 moral definitions where applicable, we select 4 MOSS-generated stories per

[6]Hubris is defined as "excessive pride or self-confidence" [1], and was a common theme in ancient Greek drama. Given the Greek origin of Aesop's fables, this interpretation is likely to be a good match for the moral of Pride as expressed in the collection. However, the assumption that every attempted action implies a character's overconfidence in its success is extremely strong, and unlikely to hold true for the majority of modern readers.

domain for each moral, yielding 24 MOSS-generated stories for each domain, and 72 in total. We use an identical spread of human-authored stories: 4 stories per moral, per domain, so there are also 72 included in the survey. For the moral-free stories, our pool consists of examples generated using each of the six moral-specific emotion filters, plus stories generated with no filter. To obtain a roughly equal number of moral-free stories in total, we include 3 for each emotion filter and 4 with no filter for each domain, giving us 22 moral-free stories per domain, and 66 overall. This produces a survey pool of 210 stories, 70 from each domain. Figure 7.6 illustrates the selection process using a flowchart. Specific stories within a particular domain for a given moral were selected at random, and discarded only if a near-identical story was already included in the set.[7] A representative selection of the stories used for the evaluation survey is provided in Appendix E, and the full set is available online.[8]

Although using such a large number of stories is important to achieve a comprehensive evaluation, the difficulty is that no single survey participant can be expected to read and respond to 210 stories. We managed this by providing each participant with 9 stories selected at random from the total story pool, based on the following spread:

- 3 MOSS-generated stories (1 per domain).
- 3 human-authored stories (1 per domain).
- 3 moral-free event sequences (1 per domain).

After completing these 9 stories, participants were given the option of responding to more. If they chose to do so, the system selected another 9 stories at random with the same breakdown, after excluding those stories the participant had already responded to from the selection pool, to ensure they never saw the same story twice. Responses were saved after every individual story the participant completed, in case they decided to discontinue the survey part way through a block of 9 stories.

Although story selection is randomised, we need to ensure an approximately equal number of responses for each story; having hundreds of responses for one story and none for another is not very useful for evaluation. For this reason, the selection algorithm

[7] As explained earlier, by near-identical we mean stories with the same sequence of events, but different characters or objects. This was not an issue for the MOSS-generated or moral-free stories, due to the progressive addition of constraints during generation (as described in Section 7.2). However, near-identical stories did appear in the human-authored set.

[8] The stories used for the evaluation survey are available at: `www.cse.unsw.edu.au/~msarlej/moss/stories`.

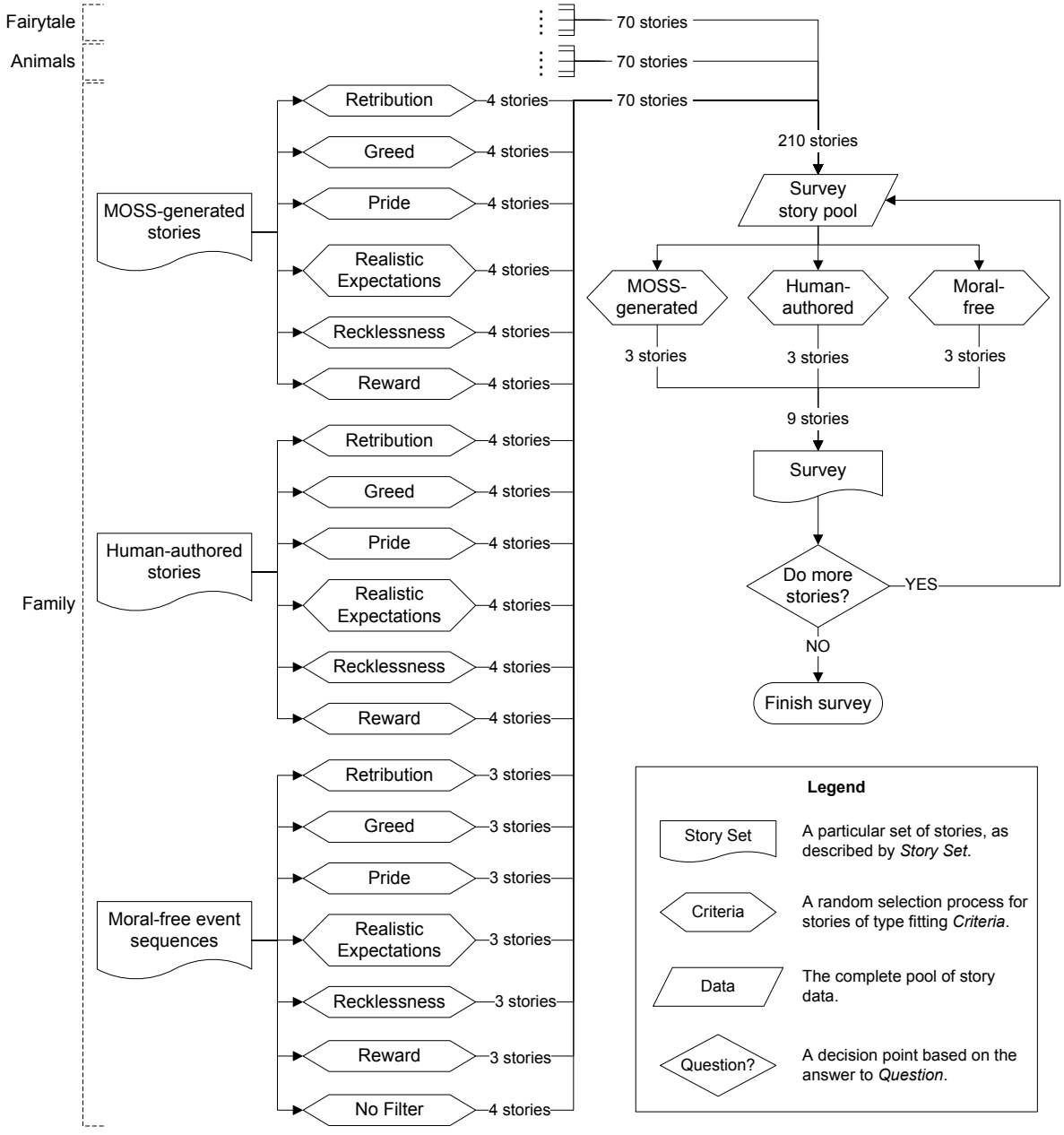

Figure 7.6: Story selection process for evaluation survey

was designed to favour stories with less responses. Stories were selected at random from a subset of the total pool consisting of those stories with the smallest number of responses to date.

7.4.2 Survey Format

The evaluation survey was conducted online. Prior to commencing the survey, participants were provided with definitions of the six morals we deal with, in order to ensure a consistent interpretation of terminology between participants. These definitions were simpler than those listed in Table 3.3, to make them easier for participants to remember. We provide the definitions used for the survey in Table 7.2. We also collected basic demographic information, including gender, age, English proficiency, education, and country of residence. We will discuss the demographic spread of participants in Section 7.4.3.

Moral	Definition
Retribution	You get what you deserve for doing something bad.
Reward	You get what you deserve for doing something good.
Greed	You should not be greedy.
Realistic Expectations	You should have realistic expectations for yourself.
Recklessness	You should not do things recklessly, but consider your actions.
Pride	You should not be excessively proud.

Table 7.2: Moral definitions provided to survey participants

Following the demographic questions, participants were presented with stories one-by-one, on separate pages, along with a series of questions. Figure 7.7 shows a single page of the survey, corresponding to one story. The first few questions focus on the moral of the story, asking participants to decide whether the story has a moral, and if so what that moral is. These questions are presented in multiple-choice format, providing participants with a fixed list of morals to choose from. We recognise this as a limitation, in that participants are funnelled into selecting one of our six morals. However, we make this trade-off to ensure we are able to categorise and interpret the results. Without a fixed-choice list, participants may use inconsistent terminology in describing morals, which would make it extremely difficult to aggregate the data.

Story 1

In a distant kingdom there lived a unicorn and a dwarf. The dwarf hated the unicorn.

The dwarf hoped that the unicorn would not be alive. One day the dwarf tried to kill the unicorn, but failed. The dwarf felt disappointment that the unicorn was alive.

Please answer the following questions based on the story above:

1. Does this story have a **moral**?
   ◯ Yes
   ◯ No

2. If you selected 'Yes' above, **which morals** does the story convey (you may choose more than one)?
   ☐ Reward
   ☐ Retribution
   ☐ Greed
   ☐ Realistic Expectations
   ☐ Recklessness
   ☐ Pride

3. Which moral is **most clear**?
   ◯ Reward
   ◯ Retribution
   ◯ Greed
   ◯ Realistic Expectations
   ◯ Recklessness
   ◯ Pride

4. **How clearly** does this moral come across?
   (Not clearly at all) ◯ 1 ◯ 2 ◯ 3 ◯ 4 ◯ 5 ◯ 6 ◯ 7 (Extremely clearly)

5. Does the story **make sense**?
   (Not coherent at all) ◯ 1 ◯ 2 ◯ 3 ◯ 4 ◯ 5 ◯ 6 ◯ 7 (Extremely coherent)

6. How **interesting** is this story?
   (Not interesting at all) ◯ 1 ◯ 2 ◯ 3 ◯ 4 ◯ 5 ◯ 6 ◯ 7 (Extremely interesting)

7. (Optional) Would you like to make any comments about this story?

Continue

Figure 7.7: Example survey page

In addition to asking participants to decide on each story's moral, the survey also asks them to rate, on a numeric scale, how clearly the moral is conveyed, how coherent the story is, and how interesting the story is. This allows us to compare MOSS-generated stories to human-authored and moral-free stories in terms of these criteria, to determine whether there are significant differences between the three groups. Participants were also given the opportunity to comment on each story if they wished. This qualitative feedback is useful in helping us understand readers' thought processes when interpreting stories, as well as identifying specific issues with their structure or content.

7.4.3 Participant Demographics

Participants were recruited for the survey via a range of electronic mailing lists and Facebook. There were no specific requirements for participation, other than a sufficient grasp of the English language to read and understand the stories. A total of 80 participants responded to our survey: 43% male and 56% female. The majority of participants were between 22 and 29 years of age (68%), although all age groups (18–21, 22–25, 26–29, 30–34, 35–39, 40–44, 45–49, and 50+) were represented. Figure 7.8 shows the spread of respondents by age group.

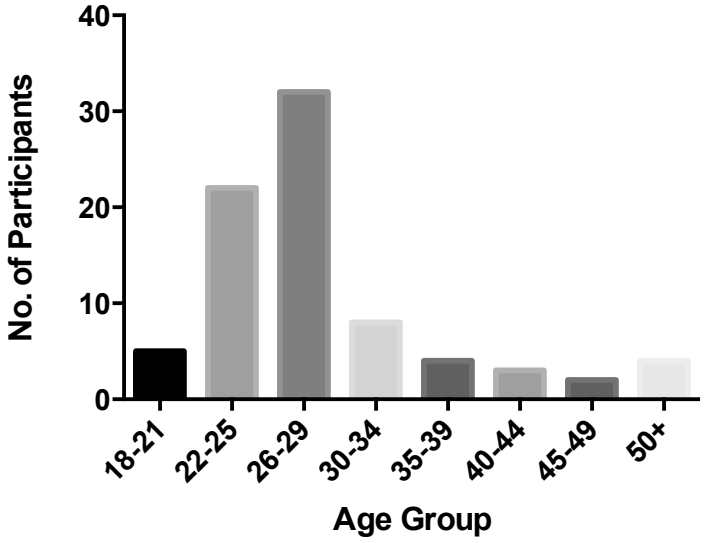

Figure 7.8: Age distribution of survey participants

99% of participants were native or fluent English speakers, and the majority had completed a university degree: 45% undergraduate, 31% postgraduate. We summarise the overall distribution by education level in Figure 7.9, and Figure 7.10 breaks this down into areas of tertiary study, where applicable. Most participants currently reside in Australia (84%), with 10% in the USA, 4% in Germany, and 1% in the UK.[9] Cultural differences may affect story interpretation, so we also asked participants to specify the country where they have spent the largest portion of their lives. This data is presented in Figure 7.11. 9 countries are represented, including Australia, the USA, Germany, New Zealand, Algeria, France, Indonesia, Malaysia, and the UK. However, given that 76% of participants had spent the largest portion of their lives in Australia, there was insufficient data for the remaining countries to make an analysis based on this criterion meaningful.

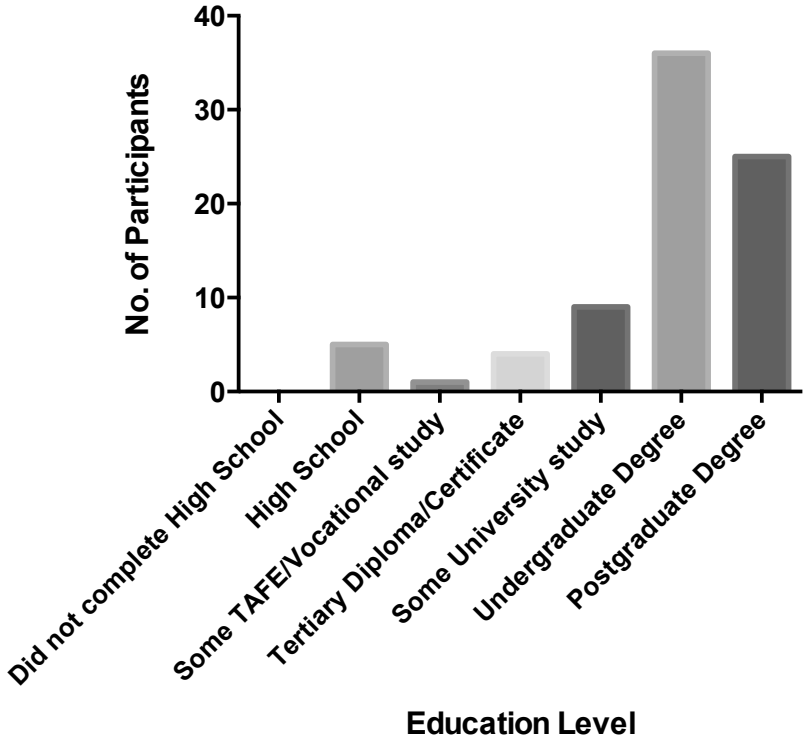

Figure 7.9: Distribution of survey participants based on education level

[9] 1 participant chose not to supply their country of residence, and therefore these percentages only sum to 99%.

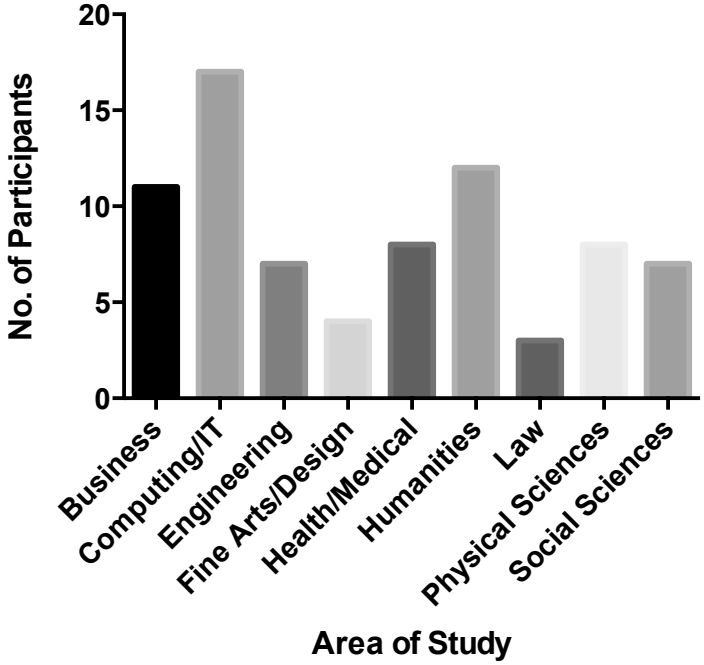

Figure 7.10: Distribution of survey participants based on area of study

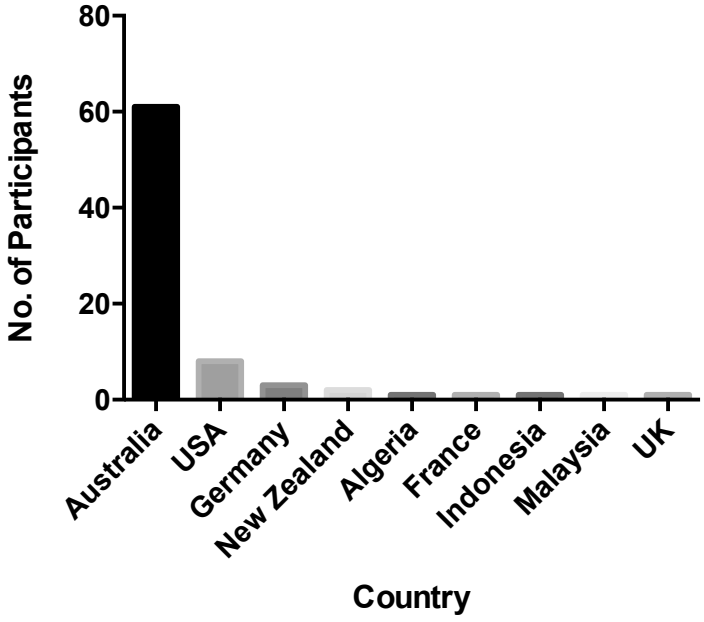

Figure 7.11: Distribution of survey participants based on country

7.5 Results

We collected a total of 841 story responses across all 80 participants, an average of 10.5 stories per person, and covering each story an average of 4.0 times. In aggregate, this yields an average of 140.2 responses per moral. The crux of our evaluation is to compare how often morals were identified correctly in MOSS-generated stories relative to human-authored and moral-free stories. Due to the large number of stories used in the evaluation, there are insufficient responses per story to be meaningful. Instead, we analyse the data in aggregate, by moral.

In Section, 7.5.1 we compare the three groups of stories based on the performance of individual morals. Section 7.5.2 compares the Type 1 and Type 2 rules for those morals which had multiple rules defined. In Section 7.5.3, we perform a similar analysis based on the moral groups that were identified in Chapter 5, followed by a comparison of Type 1 and Type 2 rule performance by moral group in Section 7.5.4. Finally, in Sections 7.5.5 and 7.5.6 we analyse the coherence and interest respectively of the three groups of stories.

7.5.1 Individual Morals

We present the aggregate survey data in three separate confusion matrices: one for MOSS-generated stories (Table 7.3), one for human-authored stories (Table 7.4), and one for moral-free stories (Table 7.5).[10] This allows us to compare the recall and precision per moral between the three groups. Recall is the percentage of stories with a given moral which readers identify correctly. It is calculated as shown in Equation 7.1, where TP stands for the number of true positives (i.e. correctly classified stories), and FN is the number of false negatives (i.e. stories with a particular moral classified as having a different moral).

$$Recall = \frac{TP}{TP + FN} \tag{7.1}$$

Precision is the percentage of stories readers classify as having a particular moral which do have that moral. The relevant formula is shown in Equation 7.2. As above, TP represents true positives. FP stands for false positives (i.e. stories classified by readers

[10] In the case of moral-free stories, the first column of the confusion matrix, representing the actual class of a given example, has a different interpretation. Moral-free stories have no actual class, because they are moral-free. Instead, this column identifies which emotion filter was used to generate the discourse.

as having that moral which actually have a different moral).

$$Precision = \frac{TP}{TP + FP} \tag{7.2}$$

	Retr	Greed	Pride	Real Exp	Reck	Reward	None	RECALL
Retribution	**27**	2	0	1	4	1	11	58.7%
Greed	16	**19**	1	1	2	0	6	42.2%
Pride	12	7	**11**	0	2	1	14	23.4%
Real Exp	5	0	3	**6**	4	0	28	13.0%
Recklessness	7	2	0	12	**9**	0	16	19.6%
Reward	4	0	3	1	0	**27**	10	60.0%
PRECISION	38.0%	63.3%	61.1%	28.6%	42.9%	93.1%		

Table 7.3: Confusion matrix for human-authored stories

	Retr	Greed	Pride	Real Exp	Reck	Reward	None	RECALL
Retribution	**25**	1	1	0	2	2	16	53.2%
Greed	14	**12**	0	0	0	1	21	25.0%
Pride	11	5	**3**	3	0	1	21	6.8%
Real Exp	1	3	1	**10**	1	0	33	20.4%
Recklessness	5	1	0	9	**10**	3	20	20.8%
Reward	5	2	1	0	0	**28**	13	57.1%
PRECISION	41.0%	50.0%	50.0%	45.5%	76.9%	80.0%		

Table 7.4: Confusion matrix for MOSS-generated stories

	Retr	Greed	Pride	Real Exp	Reck	Reward	None	RECALL
Retribution	**1**	2	0	3	1	3	28	2.6%
Greed	0	**1**	1	0	2	3	34	2.4%
Pride	1	1	**3**	2	2	6	23	7.9%
Real Exp	1	4	0	**3**	1	0	26	8.6%
Recklessness	0	1	1	2	**1**	1	35	2.4%
Reward	1	1	0	1	1	**2**	32	5.3%
All Emotions	6	2	3	3	1	3	32	
PRECISION	10.0%	8.3%	37.5%	21.4%	11.1%	11.1%		

Table 7.5: Confusion matrix for moral-free stories

Both the recall and precision values are relatively low across the board, even for human-authored stories. This is likely a side-effect of the restricted story worlds. As discussed in Section 7.3.1, a number of the authors participating in Stage 2 commented on the difficulty of writing within the supplied constraints. Nevertheless, differences are evident when comparing the percentages between groups. Although performance varies between morals, in most cases human-authored stories performed the best in terms of both recall and precision, and moral-free stories the worst. As expected, MOSS-generated stories usually lie in between. The only exceptions are Pride (where they perform worse than moral-free)[11] and Realistic Expectations (where they perform better than human).

To judge whether these differences are significant given the sample size, we use the Wilson score method [178] to calculate 95% confidence intervals for the recall and precision values. Figure 7.12 shows the confidence intervals for recall. From the graph it is clear that both human-authored and MOSS-generated stories performed significantly better than moral-free stories for three morals: Retribution, Greed, and Reward. For the remaining morals (Pride, Realistic Expectations, and Recklessness), the error bars overlap with moral-free stories, not only for MOSS-generated stories but human-authored stories as well. This may indicate that those morals were more difficult to express within the given domains. The differences in precision, as shown in Figure 7.13, were less conclusive; only Reward shows a significant difference between human-authored and MOSS-generated stories as compared to moral-free stories. The difference between MOSS and moral-free stories is also significant for Recklessness, but this is not the case for human-authored stories.

[11] In the case of Pride, it may be worth noting that it is the only moral which shares its name with one of the OCC emotions. This may contribute to the stronger performance of the moral-free stories, because the Pride emotion filter displays the Pride emotion; seeing this in the story text may prompt readers to select Pride as the story's moral.

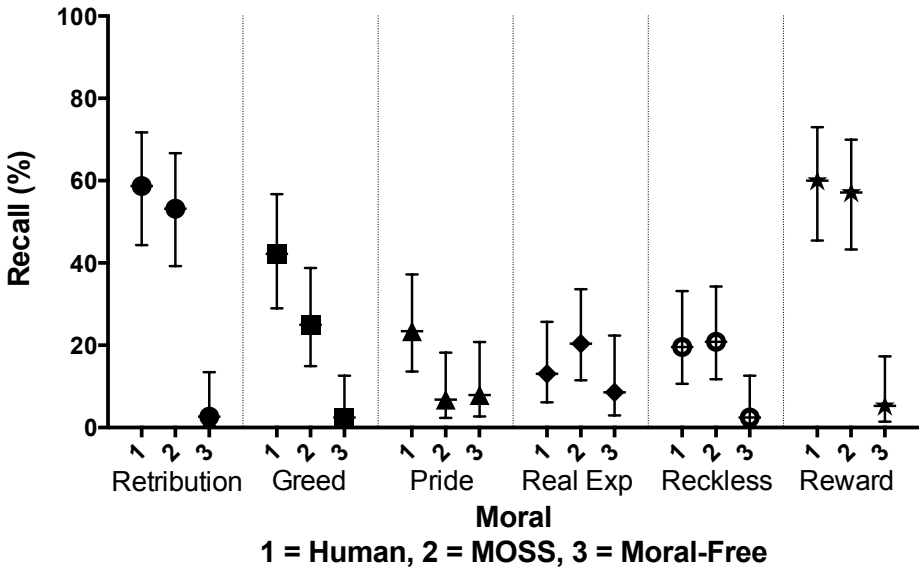

Figure 7.12: Recall by moral showing 95% confidence intervals

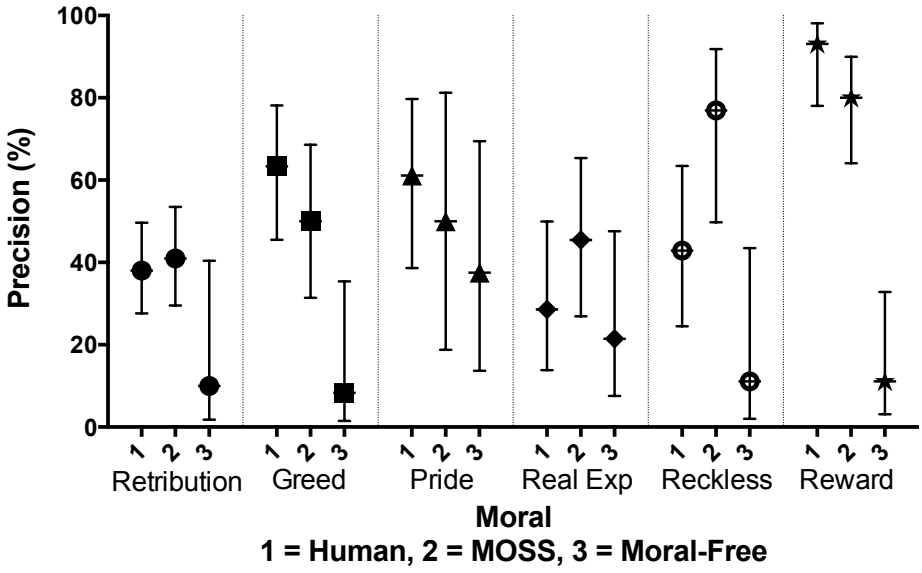

Figure 7.13: Precision by moral showing 95% confidence intervals

Survey participants were given the opportunity (in Question 2) to select multiple morals if they felt a story conveyed more than one. In Table 7.6, we summarise the percentage of responses where the correct moral was one of those selected in Question 2, regardless of whether it was chosen as the clearest moral in Question 3. We show the results as a percentage of those responses where the participant indicated the story did have a moral (in the YES columns), and also as a percentage of total responses for that moral, regardless of whether the participant indicated the story had a moral or not (in the ALL columns). In most cases, the percentages are considerably higher than the recall values shown in Tables 7.3–7.5. Figure 7.14 graphs the data from Table 7.6 as a percentage of total responses, showing 95% confidence intervals. It is evident from the graph that the difference in performance between both human-authored and MOSS-generated stories as compared to moral-free stories is more significant than it was based on the single-moral classification summarised in the confusion matrices.

Moral	Percentage of Responses Where Correct Moral Is One of Those Selected (YES = as % of responses with 'Moral = Yes', ALL = as % of all responses)					
	Human		MOSS		Moral-Free	
	YES	ALL	YES	ALL	YES	ALL
Retribution	88.6%	67.4%	93.6%	61.7%	10.0%	2.6%
Greed	79.5%	68.9%	70.4%	39.6%	14.3%	2.4%
Pride	66.7%	46.8%	26.1%	13.6%	40.0%	15.8%
Real Exp	33.3%	13.0%	62.5%	20.4%	44.4%	11.4%
Recklessness	56.7%	37.0%	46.4%	27.1%	16.7%	2.4%
Reward	94.3%	73.3%	91.7%	67.4%	50.0%	7.9%
Overall	73.7%	50.9%	68.3%	38.6%	22.5%	5.7%

Table 7.6: Percentage of responses where correct moral is one of those selected

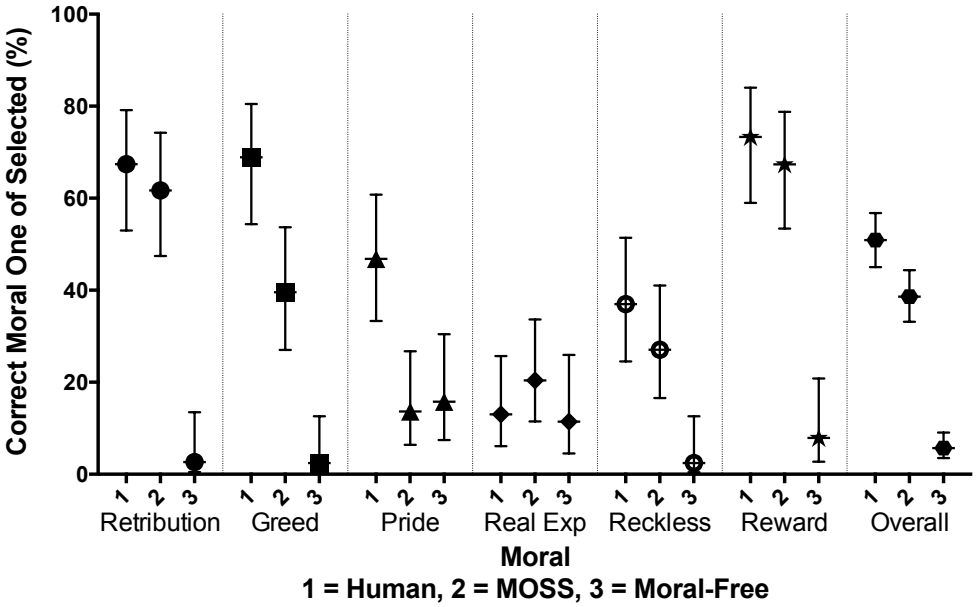

Figure 7.14: Responses where correct moral is one of those selected as a percentage of total responses showing 95% confidence intervals

These results show that participants often perceived the correct moral, but observed other morals at the same time. To a degree, this is another indicator of the subjectivity in story interpretation. It also reveals that although morals are conveyed by the stories, they are not sufficiently emphasised. Given that this is the case for human-authored stories as well (in fact, there is a greater improvement in the performance of human-authored stories than MOSS-generated stories), it may be a consequence of the simplistic discourse generation, rather than the events or emotions taking place within the stories.

Participants were also asked to rate how strongly the moral of each story was conveyed. We summarise the average ratings, showing standard deviation, in Table 7.7. Overall, participants rated the human-authored stories as conveying morals most strongly, followed by MOSS-generated stories, and finally moral-free event sequences. The overall variation between both human-authored and MOSS-generated stories compared to moral-free stories is statistically significant based on a two-tailed Welch t-test [175], with $p < 0.0001$. The difference between human-authored and MOSS-generated stories is also significant, but slightly less so, with $p = 0.0038$.

Moral	Average Moral Strength Rating ± Standard Deviation		
	Human	MOSS	Moral-Free
Retribution	4.0 ± 2.0	4.2 ± 1.8	2.6 ± 2.0
Greed	4.5 ± 1.8	4.2 ± 2.0	2.1 ± 1.7
Pride	4.0 ± 2.1	3.3 ± 1.7	3.1 ± 1.8
Real Exp	3.0 ± 1.7	2.7 ± 1.4	2.7 ± 2.0
Recklessness	3.9 ± 1.9	3.1 ± 1.9	1.8 ± 1.2
Reward	4.9 ± 1.9	3.8 ± 1.9	2.2 ± 1.7
No Filter	–	–	2.8 ± 1.9
OVERALL	4.1 ± 2.0	3.6 ± 1.8	2.5 ± 1.8

Table 7.7: Average moral strength ratings by story type and moral

We show the distribution of moral strength ratings as histograms in Figures 7.15 (human-authored stories), 7.16 (MOSS-generated stories), and 7.17 (moral-free stories). The skew of the moral strength ratings for the moral-free stories towards the low end of the scale is particularly evident. The difference between human-authored and MOSS-generated stories is less dramatic, but nevertheless it can be discerned that human-authored stories were generally rated as conveying morals more strongly.

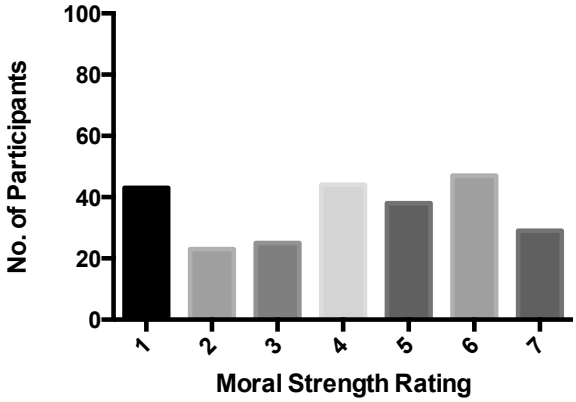

Figure 7.15: Moral strength ratings for human-authored stories

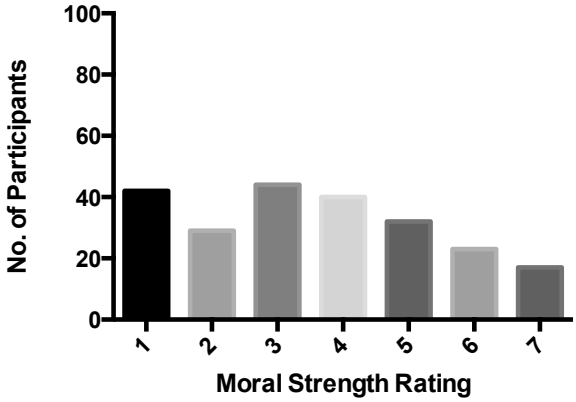

Figure 7.16: Moral strength ratings for MOSS-generated stories

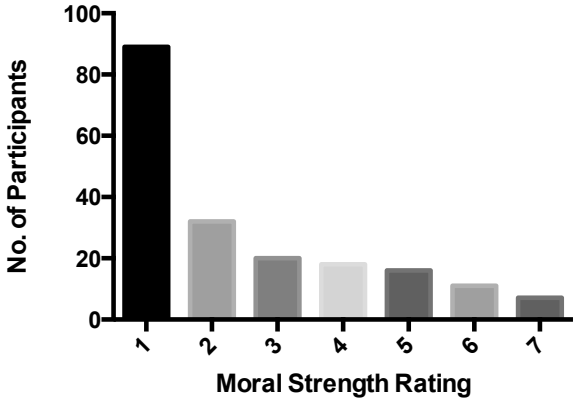

Figure 7.17: Moral strength ratings for moral-free stories

7.5.2 Individual Morals by Type

For some morals (Retribution, Greed, Recklessness, and Reward), we developed two different ASP rules, as described in Chapter 5. In these cases, stories of both types (Type 1 and Type 2) were included in the evaluation, allowing us to compare the performance of the alternative rules. Table 7.8 shows the confusion matrix for stories separated by type: T1 stands for Type 1, and T2 for Type 2. The precision values are the same as those provided in Table 7.4, because survey participants were classifying stories into morals, not moral types. As such, we cannot compare precision between the Type 1 and Type 2 rules. We can, however, compare recall; the matrix shows separate recall values for each moral type (where applicable).

	Retr	Greed	Pride	Real Exp	Reck	Reward	None	RECALL
Retrib T1	**9**	1	1	0	1	1	10	39.1%
Retrib T2	**16**	0	0	0	1	1	6	66.7%
Greed T1	8	**4**	0	0	0	1	11	16.7%
Greed T2	6	**8**	0	0	0	0	10	33.3%
Pride	11	5	**3**	3	0	1	21	6.8%
Real Exp	1	3	1	**10**	1	0	33	20.4%
Reckless T1	3	1	0	5	**4**	3	8	16.7%
Reckless T2	2	0	0	4	**6**	0	12	25.0%
Reward T1	3	2	1	0	0	**10**	4	50.0%
Reward T2	2	0	0	0	0	**18**	9	62.1%
PRECISION	41.0%	50.0%	50.0%	45.5%	76.9%	80.0%		

Table 7.8: Confusion matrix for MOSS-generated stories by moral rule type

Based on the recall values in Table 7.8, the Type 2 version of each moral tends to perform better. In most cases, the Type 2 rule was more specific than Type 1, so this outcome is not surprising (the exception is Recklessness, where neither rule is more specific than the other, but they provide different takes on the moral). To get a sense of whether these differences are significant, we graph recall for the Type 1 and Type 2 versions of these four morals, showing 95% confidence intervals, in Figure 7.18. There is a considerable overlap in error bars in all cases, meaning the variation in performance is not conclusive. A substantially larger pool of data would be required to properly assess the significance of these differences.

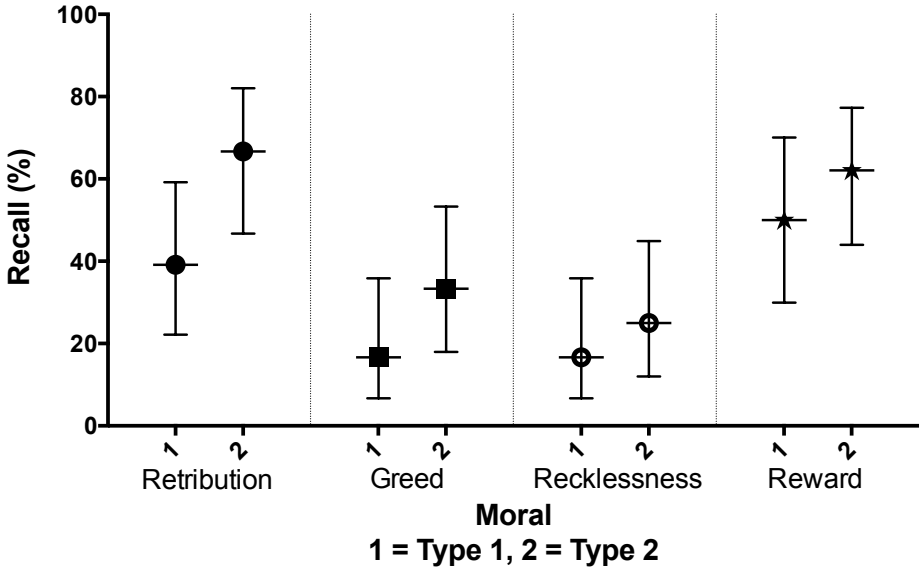

Figure 7.18: Recall by moral type showing 95% confidence intervals

7.5.3 Moral Groups

In the human-authored and MOSS-generated confusion matrices presented in Section 7.5.1, we observe similar patterns to those that were identified via the ILP experiments described in Chapter 5: Greed and Pride are commonly misclassified as Retribution, and Realistic Expectations and Recklessness are often confused. As discussed in Section 5.3.4, the similarities between Retribution, Greed, and Pride, and between Realistic Expectations and Recklessness, make sense. Greed and Pride can be considered as specific subclasses of Retribution. Realistic Expectations and Recklessness involve very similar emotions; this is supported by the author feedback described in Section 7.3.1, where authors found themselves writing almost identical stories for both morals. To further investigate these relationships, we perform a similar analysis to that described in Section 7.5.1, but based on moral groups. As in Section 5.3.5, we group Retribution, Greed, and Pride, and Realistic Expectations with Recklessness. The resulting confusion matrices are provided in Tables 7.9 (human-authored stories), 7.10 (MOSS-generated stories), and 7.11 (moral-free stories).

	Retr/Greed/Pride	Real Exp/Reck	Reward	None	RECALL
Retr/Greed/Pride	**95**	10	2	31	68.8%
Real Exp/Reckless	17	**31**	0	44	33.7%
Reward	7	1	**27**	10	60.0%
PRECISION	79.8%	73.8%	93.1%		

Table 7.9: Confusion matrix for human-authored stories for moral groups

	Retr/Greed/Pride	Real Exp/Reck	Reward	None	RECALL
Retr/Greed/Pride	**72**	5	4	58	51.8%
Real Exp/Reckless	11	**30**	3	53	30.9%
Reward	8	0	**28**	13	57.1%
PRECISION	79.1%	85.7%	80.0%		

Table 7.10: Confusion matrix for MOSS-generated stories for moral groups

	Retr/Greed/Pride	Real Exp/Reck	Reward	None	RECALL
Retr/Greed/Pride	**10**	10	12	85	8.5%
Real Exp/Reckless	7	**7**	1	61	9.2%
Reward	2	2	**2**	32	5.3%
All Emotions	11	4	3	32	
PRECISION	33.3%	30.4%	11.1%		

Table 7.11: Confusion matrix for moral-free stories for moral groups

Overall, both recall and precision are higher than they were for individual morals, particularly for the human-authored and MOSS-generated stories, which is a good indicator that these groupings are reasonable. We calculated 95% confidence intervals for both sets of values, again using the Wilson score method, and provide the corresponding graphs in Figures 7.19 and 7.20. This time, there is a clear difference between the human-authored and MOSS-generated story performance as compared to moral-free stories; there is no overlap in error bars in any case, for recall or precision.

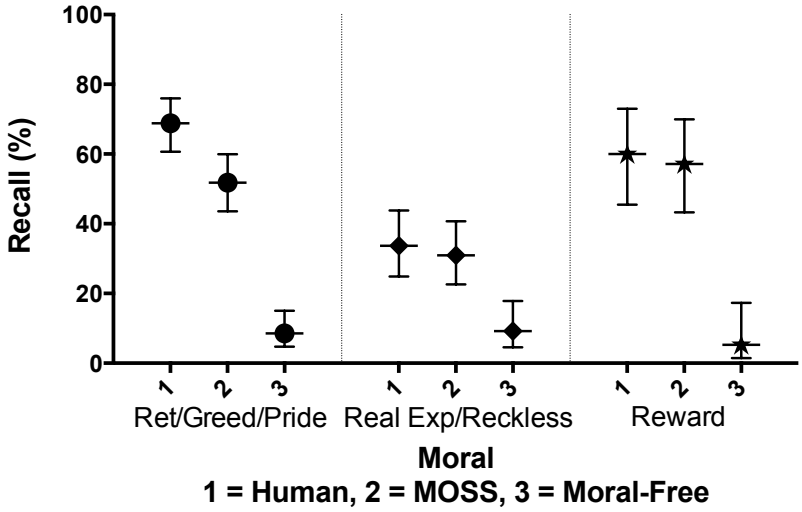

Figure 7.19: Recall by moral group showing 95% confidence intervals

Figure 7.20: Precision by moral group showing 95% confidence intervals

7.5.4 Moral Groups by Type

As was done for individual morals in Section 7.5.2, we also compare the performance of the Type 1 and Type 2 rules (where applicable) with respect to the moral groups described in Section 7.5.3. Table 7.12 presents the relevant confusion matrix. As explained in Section 7.5.2, readers were not asked to distinguish between the different types of each moral. Consequently, we can only compare the recall values between Type 1 and Type 2 rules, because precision can only be calculated as a combined value. The 95% confidence intervals for the recall values in Table 7.12 are shown in Figure 7.21. As was the case in Section 7.5.2, all error bars overlap, and thus we cannot conclude that any of the differences in performance between Type 1 and Type 2 rules are significant.

	Retr/Greed/Pride	Real Exp/Reck	Reward	None	RECALL
Retribution T1	**11**	1	1	10	47.8%
Retribution T2	**16**	1	1	6	66.7%
Greed T1	**12**	0	1	11	50.0%
Greed T2	**14**	0	0	10	58.3%
Pride	**19**	3	1	21	43.2%
Realistic Exp	5	**11**	0	33	22.4%
Recklessness T1	4	**9**	3	8	37.5%
Recklessness T2	2	**10**	0	12	41.7%
Reward T1	2	0	**18**	9	62.1%
Reward T1	6	0	**10**	4	50.0%
PRECISION	79.1%	85.7%	80.0%		

Table 7.12: Confusion matrix for MOSS-generated stories for moral groups by type

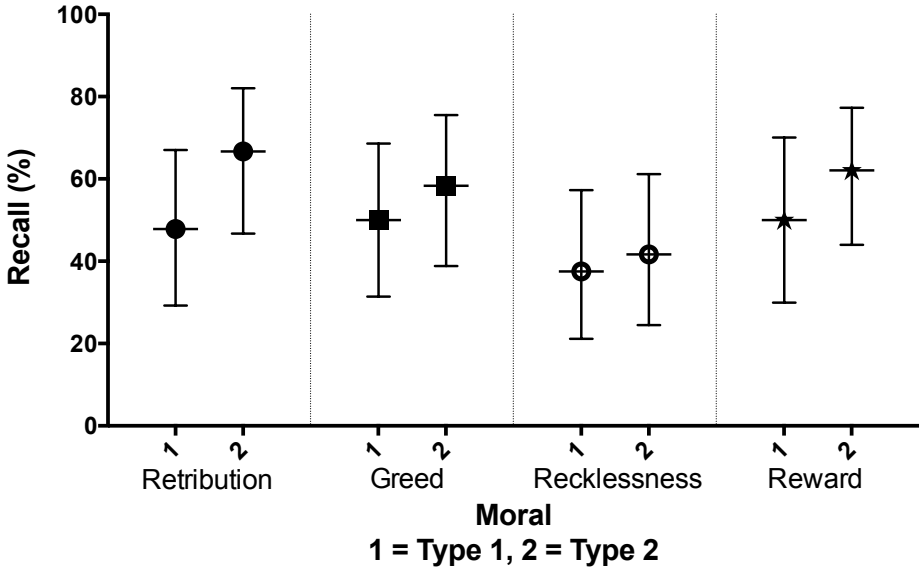

Figure 7.21: Recall for moral groups by type

7.5.5 Coherence

Survey participants rated the coherence of stories on a numeric scale, ranging from 1 to 7. We similarly aggregate the data by moral, and use the Welch t-test to gauge the significance of any differences between story types. It is important to note here that, apart from the physical restrictions on actions enforced in the Action Layer (for example, an agent cannot give someone an object unless that object is in their possession), we made no attempt to enforce coherence in the MOSS-generated stories. The only restrictions on plot were those imposed by the emotion requirements comprising the moral rules defined in Chapter 5.

Table 7.13 presents the average coherence ratings by story type and moral, along with standard deviation. Across all six morals, the human-authored stories have the highest coherence rating, followed by MOSS-generated stories. Moral-free stories are rated least coherent. The overall variation is statistically significant based on a Welch t-test with $p < 0.0001$ in all cases (i.e. between human-authored and MOSS-generated, human-authored and moral-free, and MOSS-generated and moral-free). Despite no attempt to enforce coherence, MOSS's moral rules produced more sensible stories than those based on moral-free sequences of events.

Moral	Average Coherence Rating ± Standard Deviation		
	Human	MOSS	Moral-Free
Retribution	5.0 ± 1.7	3.7 ± 1.8	3.1 ± 1.7
Greed	5.1 ± 1.8	4.2 ± 2.1	3.3 ± 2.1
Pride	4.6 ± 1.4	4.2 ± 1.6	4.1 ± 1.7
Real Exp	4.4 ± 1.8	3.9 ± 1.8	2.8 ± 1.9
Recklessness	5.1 ± 1.6	4.4 ± 2.0	3.1 ± 2.0
Reward	5.5 ± 1.3	3.9 ± 1.6	3.2 ± 1.7
No Filter	–	–	3.3 ± 1.7
OVERALL	4.9 ± 1.6	4.1 ± 1.8	3.3 ± 1.8

Table 7.13: Average coherence ratings by story type and moral

Figures 7.22, 7.23, and 7.24 show the distribution of coherence ratings visually for human-authored, MOSS-generated, and moral-free stories respectively. Figure 7.22 is clearly skewed towards the high end of the scale, while Figure 7.24 is skewed towards the low end. Figure 7.23, which represents the ratings for the MOSS-generated stories, peaks in the middle.

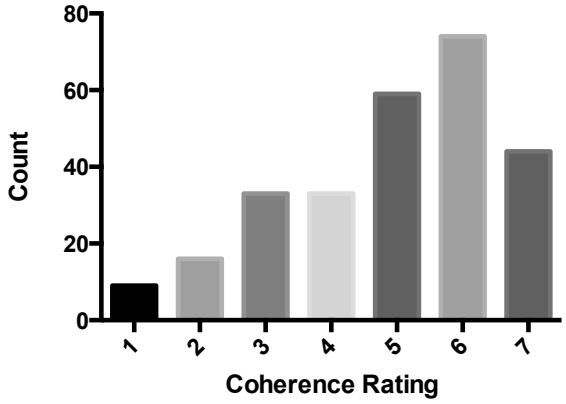

Figure 7.22: Distribution of coherence ratings for human-authored stories

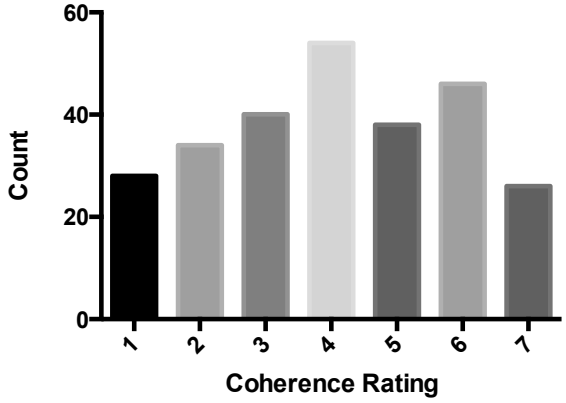

Figure 7.23: Distribution of coherence ratings for MOSS-generated stories

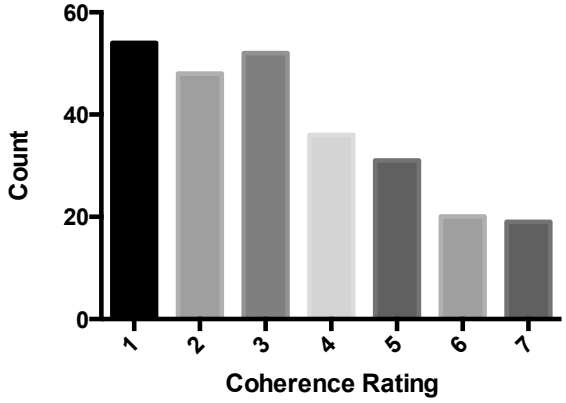

Figure 7.24: Distribution of coherence ratings for moral-free stories

7.5.6 Interest

Participants were also asked to rate the interest of each story, also using a numeric scale ranging from 1 to 7. We present the average interest ratings, along with standard deviation, in Table 7.14. Again, the results are shown per moral and overall. Interest was low across the board, which is likely a consequence of the limited story domains. As with the coherence ratings, interest is highest for human-authored stories and lowest for moral-free stories across all morals, with MOSS-generated stories lying in between. A Welch t-test shows the difference in interest overall between human-authored and MOSS-generated stories is significant with $p = 0.0039$. The difference between both human-authored and MOSS-stories compared to moral-free stories is more significant: $p < 0.0001$ in both cases.

Moral	Average Interest Rating ± Standard Deviation		
	Human	MOSS	Moral-Free
Retribution	3.4 ± 1.7	3.2 ± 1.7	2.2 ± 1.3
Greed	3.8 ± 1.8	3.1 ± 1.7	2.5 ± 1.6
Pride	3.6 ± 1.6	3.3 ± 1.3	3.2 ± 1.5
Real Exp	3.3 ± 1.5	3.0 ± 1.5	2.5 ± 1.7
Recklessness	3.4 ± 1.8	3.0 ± 1.6	2.3 ± 1.3
Reward	3.8 ± 1.4	3.2 ± 1.7	2.5 ± 1.1
No Filter	–	–	2.6 ± 1.5
OVERALL	3.5 ± 1.7	3.1 ± 1.6	2.5 ± 1.5

Table 7.14: Average interest ratings by story type and moral

We show the distribution of ratings for each group of stories in Figures 7.25 (human-authored stories), 7.26 (MOSS-generated stories), and 7.27 (moral-free stories). The distributions for human-authored and MOSS-generated stories are similar, whereas the distribution for moral-free stories is more noticeably skewed towards the low end of the scale.

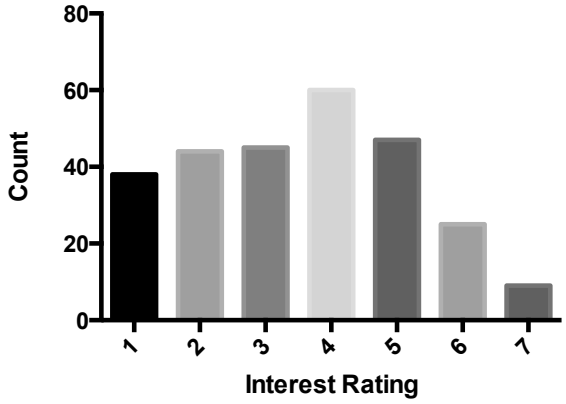

Figure 7.25: Distribution of interest ratings for human-authored stories

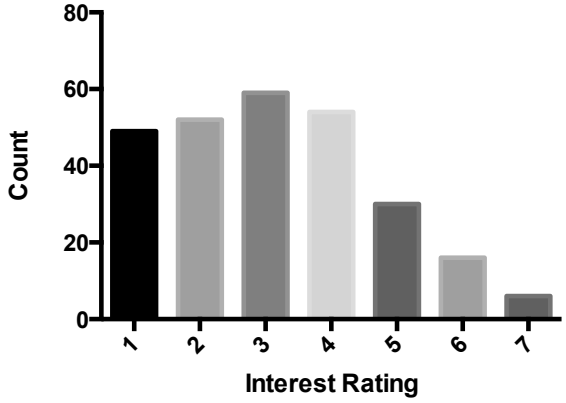

Figure 7.26: Distribution of interest ratings for MOSS-generated stories

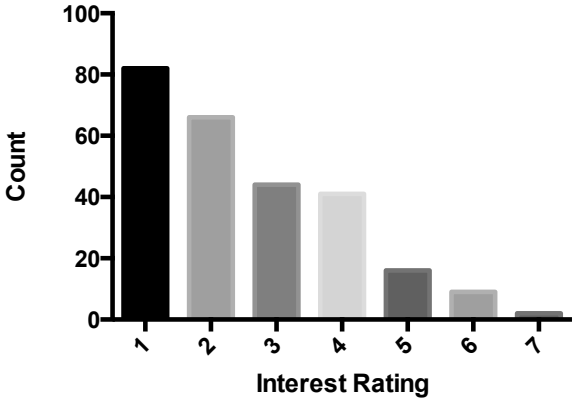

Figure 7.27: Distribution of interest ratings for moral-free stories

7.5.7 Key Outcomes

In this section, we summarise the key outcomes of the analysis presented in Sections 7.5.1–7.5.6. We began with an analysis of individual moral performance, and found:

- Both recall and precision for individual morals were relatively low across the board. In terms of recall, the performance of human and MOSS stories was significantly better than moral-free stories for Retribution, Greed, and Reward. No significant differences were observed for precision.
- There was a marked improvement in performance for both human-authored and MOSS-generated stories when we analysed the data based on whether the correct moral was one of those selected by the participant, but not necessarily chosen as the clearest. This shows readers often identified the correct moral in a story, but simultaneously perceived other morals, which they believed were conveyed more strongly.
- Morals were judged to be conveyed most strongly by human-authored stories, followed by MOSS-generated stories, then moral-free stories.
- For those morals where alternative rules were defined, the variation between Type 1 and Type 2 performance was inconclusive. No significant difference was observed based on the data available.

When morals were grouped, in the same manner as in Chapter 5 (i.e. Retribution, Greed, and Pride together, and Realistic Expectations with Recklessness), we note the following:

- Recall and precision are much higher for moral groups than individual morals, and human-authored and MOSS-generated stories perform significantly better than moral-free stories in all cases.
- As was the case when analysed based on individual morals, no statistically significant variation was observed between the performance of Type 1 and Type 2 moral rules.

Analysing participants' ratings of coherence and interest showed:

- Human-authored stories were generally rated the most coherent, but MOSS-generated stories were significantly more coherent than moral-free stories, despite no specific attempt to enforce coherence.

- Interest was relatively low across the board, which likely results from the limited story worlds. Overall, human stories were rated the most interesting, but MOSS stories were deemed significantly more interesting than moral-free stories, despite no specific attempt to make the stories interesting.

7.6 Qualitative Feedback

Survey participants were given the option of providing comments about each story. In this section, we summarise the most notable feedback received, and propose system modifications that could resolve the issues raised. In many cases, straightforward modifications to the text generation would be sufficient to address the deficiencies readers identify.

7.6.1 Emotions After Death

By far the most common criticism of the stories was that characters continued to experience emotions after their death. As explained in Chapter 4, we permit characters to feel emotions in the time-step immediately following the event that causes their death, so the solver is able to match these emotions to the relevant moral rule. Figure 7.28 shows an example of a MOSS-generated story in which two characters (the bear and the lion) feel emotions after they die. Such posthumous emotions make a story sound absurd, or even inadvertently comical.

> Deep in the jungle there lived a bear, a lion and a bee. The lion hated the bee. The bee loved the bear.
>
> Early one morning the lion attacked and killed the bear. As a result, the bear was dead. The bear felt distress that he was dead. The bear felt anger towards the lion about attacking him because he was dead. The bee started to hate the lion.
>
> Not long afterwards the bee attacked and killed the lion. As a result, the lion was dead. The lion felt distress that he was dead. The lion felt distress that the bee was alive.

Figure 7.28: Characters feeling emotions after death

This issue could be addressed at the discourse level. Without making any changes to the planning component, the Text Generator could be modified so that any emotions occurring after a character has died are not displayed. This approach implicitly assumes readers will infer that characters consider death to be undesirable in order to understand certain morals. However, that notion is generally an accepted one (except, of course, in stories that explicitly state a character wishes to die).

7.6.2 Confusing Relationship Dynamics

Readers were not always able to understand the relationship dynamics in the stories, in particular as defined by the Love/Hate emotions between characters. Questions such as "why did the dwarf start to hate the troll?" show that readers did not ascribe the changes in object-based emotions to the preceding admirable or reproachable action. This question arose with regards to the story presented in Figure 7.29. While the reader was able to infer a reason for the wizard's hatred of the troll (i.e. that the troll kidnapped him), the dwarf's hatred was perplexing.

> Once upon a time there lived a wizard, a troll and a dwarf.
>
> One summer's morning the troll kidnapped the wizard. As a result, he had the wizard and the wizard was not free. The troll felt joy that he had the wizard. The troll felt pride about kidnapping the wizard. The wizard and the dwarf started to hate the troll.
>
> The next day the dwarf stole the wizard from the troll. As a result, the troll didn't have the wizard anymore. The troll felt distress that he didn't have the wizard anymore.

Figure 7.29: Unexplained object-based emotions

The simplest way to resolve this is to modify the Text Generator to include the action which caused the Love or Hate when describing these emotions. It might help readers make this leap of inference if the last sentence in the first paragraph of the story shown in Figure 7.29 were replaced with: "The wizard and the dwarf started to hate the troll because they disapproved of the troll kidnapping the wizard." The trade-off of such an approach would be the increased verbosity of the tale, which may also irk readers.

7.6.3 Characters' Motivations

Some readers wanted clearer reasons for why characters performed certain actions: "nobody's behaviour is ever explained, so there is no logical plot/characterization for the reader to identify with and learn from." This particular comment was in reference to a moral-free story, so it may have been less coherent than most, but similar comments were also made with reference to the other groups of stories. For example:

- "[...] a lack of reason behind their actions."
- "There are actions without causes [...]"
- "[...] lacking in sufficient details/explanation/requiring too much inference."

In addition to frustrating readers when they are unclear, characters' motivations sometimes play a role in how readers interpret a story. For example, one participant asks: "Why did the princess want the gold? Maybe for greed? To help someone else?" We provide the relevant story, which was human-authored, in Figure 7.30. The implication is that understanding the princess's motivation would have helped the reader deduce the moral.

> Long ago in a far away land there lived a dwarf and a princess.
>
> One summer's morning the princess stole gold from the dwarf. As a result, the dwarf didn't have gold anymore. The dwarf felt distress that he didn't have gold anymore. The princess felt remorse about stealing gold because the dwarf didn't have gold anymore.

Figure 7.30: Unclear character motivations

As mentioned in Chapter 6, we do not explicitly model characters' goals or motivations. Although we do model desires and ideals, which are derived from goals [118], this is only to facilitate our emotion model; our interest is at the plot level, not the character level. The best way to address this issue would be to extend MOSS with a more sophisticated character model. The additional information modelled as a result could be included in the text of the stories to provide explanations for characters' actions. We discuss this as a key avenue for future work in Chapter 9.

7.6.4 Understanding Why

Characters' motivations were not the only aspect of the stories where readers sometimes struggled to make the necessary inferences. Similar comments were made about the outcomes of certain events, or why they failed. For example, the following comment was made with regards to the human-authored story shown in Figure 7.31:

> I think the story is missing a "why," not just a "what" and "to whom." It makes a few leaps of logic like "fail at making dinner = dirty dishes."

This can be resolved by modifying the text associated with particular events, to better explain their consequences, or provide reasons for why actions failed. In its current form, the Text Generator outputs nothing more than a simplistic description of what happens, without offering any explanations.

> In a modern inner-city apartment there lived a boy.
>
> One afternoon the boy tried to cook dinner for himself, but failed and burnt himself. As a result, he was hurt and the dishes were not clean. The boy felt distress that the dishes were not clean. The boy felt distress that he was hurt. The boy felt remorse about trying to cook dinner for himself because he was hurt.

Figure 7.31: Understanding why

7.6.5 Unrealistic Stories

Some readers commented that many of the stories were not realistic. For example:

- "Stories aren't realistic – bee trying to catch a squirrel/toad."
- "This story was possibly inadvertently funny because the notion of a squirrel catching a bear is ridiculous."
- "How can a bird catch a wolf?"
- "A bird killing a bear is also kind of unrealistic."

What makes these comments interesting is that animals in stories, particularly fairytales or fables, often do things that are not at all realistic. However, the reader's awareness

that the story is fictional, and the animals have been anthropomorphised, allows them to suspend disbelief and become immersed in the tale regardless. However, multiple readers were unable to do this for the animals domain (each of the comments above was made by a different participant). The expectations set up for a reader at the beginning of a story are critical in managing suspension of disbelief [105], so these comments may indicate that it was not clear from this domain whether or not the animals should be thought of as real. No such comments appeared for the other domains; the fairytale domain was very obviously a fantasy world, whereas the family domain did not contain any actions that could be considered unrealistic.

One participant commented on some stories being unrealistic because characters were able to capture (and therefore possess) others. They made two comments to this effect, regarding two different stories:

- "You can't 'possess' someone else."
- "How does one 'have' a bee?"

This, however, is likely just a result of that specific reader's interpretation of the idea of possession. For example, it is easy to imagine having a bee: one could catch it and put it in a jar. In the same way it can be argued that you can possess someone else by keeping them captive, which is exactly what these stories attempt to portray. Again, these comments arose only with respect to the animals domain, so it may be another example of a reader expecting the animals to behave in a realistic fashion.

7.6.6 Introducing Objects

The Text Generator produces an introductory paragraph at the beginning of each story which describes the setting, and lists all the characters that will feature. However, no mention is made of any objects those characters possess. This can result in confusion for readers when such objects are suddenly referenced:

> The story also needs to first note the Dwarf has an Axe and that the Dragon has a Treasure before it explains how the Knight coveted those objects and stole them.

Figure 7.32 shows the MOSS-generated story about which this comment was made. The knight hopes for "the axe" and "the treasure," but neither object is introduced beforehand.

Long ago in a far away land there lived a knight, a dragon and a dwarf.

The knight hoped that he would have the axe. One summer's morning the knight stole the axe from the dwarf. As a result, he had the axe and the dwarf didn't have the axe anymore. The knight felt satisfaction that he had the axe. The knight felt shame about stealing the axe. The dwarf started to hate the knight. The dragon started to love the knight.

The knight hoped that he would have the treasure. Later that day the knight stole the treasure from the dragon. As a result, he had the treasure and the dragon didn't have the treasure anymore. The knight felt satisfaction that he had the treasure. The knight felt shame about stealing the treasure. The dragon started to hate the knight.

The day after that, the dragon killed the knight. As a result, the knight was dead. The knight felt distress that he was dead.

Figure 7.32: Objects are not introduced

This could be addressed in two ways. The simplest to implement would be to mention all relevant objects in the introduction. In this example, the first paragraph could be replaced with:

> Long ago in a far away land there lived a knight, a dragon and a dwarf. The dragon had treasure. The dwarf had an axe.

Alternatively, each object could be introduced immediately prior to the first time it is referenced. Taking this approach, the second paragraph would begin with:

> The dwarf had an axe. The knight hoped that he would have the axe.

For simple stories, either approach would serve the required purpose. As stories become more complicated, it would be more effective to take the second approach, as readers may not readily remember everything that was listed in the introductory paragraph as they move through the story.

7.6.7 Writing Style

A number of comments were made about the writing style of the stories. Interestingly, both positive and negative comments appeared. Given the simplistic nature of the text generation, comments such as "better writing style needed" and "the writing style is not very artistic which made the story quite fragmented and non-cohesive" were expected. However, some participants found merit in the discourse:

- "The unusual English makes the story somewhat humorous, and therefore more interesting."
- "[...] the setting and terseness of the story was at least a bit artful."

There were also comments which affirmed our success in masking the differences between human-authored and MOSS-generated stories, by automatically generating the text of both. Each of the following comments was made with respect to a different human-authored story:

- "The story telling needs more heart. I would guess it's not written by a human author."
- "It sounds very clinical, like a computer wrote it."

Text generation was not the focus of our work, but it is clear that improvements to this aspect of the system could significantly impact not only readers' enjoyment of the stories, but also their effectiveness at conveying morals.

7.6.8 Preconceived Significance

Readers bring with them a history, past experiences which affect how they interpret any story they read [150]. Consequently, including events or objects in the story domain that have some special significance in popular stories (such as fairytales) can trigger unintended reactions. In our case, having picking a rose and possibly pricking one's finger as an available action could potentially be attributed greater significance than warranted, as evidenced by the following comment:

> I'm not sure that the Princess pricking her finger on the thorn really had any place in this story, it usually signifies something important, but nothing happened here.

This relates to the story presented in Figure 7.33, which is human-authored. The princess picks the rose in order to give it to the knight, as a reward for killing the troll who had her captive. The fact that she pricks her finger while doing so has no bearing on the story, but because it is mentioned, readers expect some kind of significance.

Once upon a time there lived a troll, a knight and a princess.

One day the troll kidnapped the princess. As a result, the princess was not free. The princess felt distress that she was not free.

The princess feared that she would not be free. A short time later the knight rescued the princess from the troll. As a result, the princess was free. The princess felt joy and relief that she was free.

Some time later the knight killed the troll. As a result, the troll was dead. The knight felt pride about killing the troll. The princess felt gratitude towards the knight about killing the troll because the troll was dead. The princess felt admiration towards the knight about killing the troll.

Later the princess picked a rose, but pricked her finger.

Eventually the princess gave the rose to the knight. As a result, the knight had the rose. The knight felt satisfaction that he had the rose.

Figure 7.33: Preconceived significance

The setting itself can also influence a reader's interpretation of a story, if they are primed to think of particular settings as more suited to conveying morals. For example, one participant made the following comment:

> I also think I was more primed to find a moral in a story taking place in a more rural setting that [*sic*] in a city setting.

The rural setting results from nothing more than the randomly selected sentence beginning: "In a small country town." It is for this reason that generating stories with morals in multiple domains is of particular importance: it reduces the dependence of the results on the characteristics of a specific story world.

7.6.9 Differences in Interpretation of Morals

Even though definitions were provided for all morals at the beginning of the survey, some readers still used their own interpretation. For example, in explaining how they distinguish between Greed and Retribution, one participant said the following:

> But I do think it's interesting that I put greed higher than retribution; I'm not sure I can think that retribution can be a moral very often. The story teaches that greed can lead to retribution; therefore DON'T BE GREEDY. Not, DON'T BE A VICTIM OF RETRIBUTION.

"Don't be a victim of retribution" is not how Retribution as a moral should be interpreted based on the definitions we supplied to participants. Rather, it should be more like, "don't do bad things to others, or bad things will happen to you." This different understanding of the moral likely affected the reader's perception of Retribution in stories.

Another reader's interpretation of Pride provides a similar example:

> I think these days we start to use "greed" at least partly for what we used to call "pride." People who are too cocky and think that they are so great that they can do whatever they want.

Not surprisingly, this reader misclassified the corresponding story (which was MOSS-generated with the moral Pride) as Greed, despite obviously identifying features of Pride in the story in order to make the comment. We show this story in Figure 7.34. Undoubtedly there were other participants who also interpreted the morals differently from the supplied definitions.

Once upon a time there lived a wizard, a unicorn, a dragon and a dwarf. The unicorn hated the dwarf.

One day the unicorn kidnapped the wizard. As a result, the wizard was not free. The wizard and the dwarf started to hate the unicorn. The dragon started to love the unicorn.

The next day the dragon stole the wizard from the unicorn. As a result, he had the wizard. The dragon felt joy that he had the wizard. The dragon felt pride about stealing the wizard. The wizard, the unicorn and the dwarf started to hate the dragon.

More time passed, and the dwarf stole the wizard from the dragon. As a result, the dragon didn't have the wizard anymore. The dragon felt distress that he didn't have the wizard anymore.

Figure 7.34: Different interpretations of morals

7.6.10 Choice of Connectives

As described in Chapter 6, in stories featuring multiple events, the Text Generator randomly chooses connectives to begin the description of each new event. These connectives vary in the temporal distance they signify, which may affect readers' interpretations of how closely consecutive events are related to one another. The following comment, in which the participant juxtaposed their own responses to two different stories, is a perfect illustration:

> It's also interesting that I did not infer that the mother bought ice cream for the daughter directly because the daughter made her a picture, like I did for the husband-wife combo cooking/cleaning for each other. This was because of both power dynamics/family roles and the amount of time passing between gifts.

The reference to "the amount of time passing" is particularly notable here. One of these stories, shown in Figure 7.35, uses the connective "that evening" between the cooking and the cleaning. The other (Figure 7.36), which is also longer, separates the related events with "more time passed," and then "a little later." It may be more effective, especially for those morals that involve reciprocal actions (i.e. Retribution and Reward) to select connectives which are more immediate.

In a small country town there lived a father and a mother.

One weekend the father cooked lunch for the mother. As a result, the dishes were not clean and the mother had lunch. The mother felt joy that she had lunch. The mother felt gratitude towards the father about cooking lunch for her because she had lunch. The mother started to love the father.

That evening the mother washed the dishes. As a result, the dishes were clean. The father felt joy that the dishes were clean. The father felt joy that the dishes were not broken. The father felt gratitude towards the mother about washing the dishes because the dishes were clean.

Figure 7.35: Connectives indicating a shorter temporal distance

In a nice house on a quiet street there lived a mother and a girl.

One day the girl drew a picture.

Later that day the girl gave the picture to the mother. As a result, the mother had the picture. The mother felt joy that she had the picture. The mother felt gratitude towards the girl about giving the picture to her because she had the picture.

More time passed, and the mother bought ice-cream.

A little later the mother gave ice-cream to the girl. As a result, the girl had ice-cream. The girl felt joy that she had ice-cream. The girl felt gratitude towards the mother about giving ice-cream to her because she had ice-cream.

Figure 7.36: Connectives indicating a greater temporal distance

7.6.11 Looking for Meaning

Generally, readers expect to find some kind of meaning or point in a story [65]. This is evidenced by some of the comments that were made about moral-free stories as readers searched for meaning. For example, a reader made the following comment about the moral-free story shown in Figure 7.37:

> This story makes sense AND is more interesting if the vase is actually an urn and the mother is actually passed.

Some participants even tried to interpret stories in a figurative way:

> I'm beginning to think the wolf is symbolic of our inner demons which cause us to attack to achieve our goals or as a reward for doing so.

The fact that readers can be quite forgiving is advantageous for storytelling systems, because it allows a system to make small (or sometimes even large) errors in plot structure, and readers will come up with their own explanations for why it makes sense.

In a modern inner-city apartment there lived a boy and a mother.

One chilly autumn day the boy got the vase off a high shelf.

The mother hoped that she would have breakfast. The next day the boy tried to cook breakfast for the mother, but failed and burnt himself. As a result, he was hurt and the dishes were not clean.

More time passed, and the boy put the vase on a high shelf.

Figure 7.37: Readers looking for meaning

Even when readers cannot find meaning in a story, sometimes this in itself is enough to prompt interest. The moral-free story shown in Figure 7.38 presents a sequence of completely disconnected events. Nevertheless, it prompted the following comment:

> Is this a riddle? Am I suppose [*sic*] to find the sense somewhere between the lines? Therefore kind of interesting!

Although this will not necessarily help a story convey the intended moral, it alleviates the burden of producing flawless stories. Provided that a system is able to generate reasonably coherent storylines, the reader's imagination will do the rest.

In a small country town there lived a father, a boy, a mother and a girl. The boy loved the girl.

The father hoped that he would have breakfast. One afternoon the mother cooked breakfast for the father. As a result, the dishes were not clean. The girl started to love the mother.

A short time later the girl messed up the boy's room. As a result, the boy's room was not clean. The mother felt distress that the boy's room was not clean.

Figure 7.38: Looking for meaning between the lines

7.6.12 Suggesting Modifications

Some readers, when they were unable to discern a moral in a particular story, speculated about how the story could be modified to convey a moral. For instance, the following comment relates to the story in Figure 7.39:

> I suppose if the wizard suffered as a result of his actions there would be a moral. But he gets away with everything.

This is actually a human-authored story intended to convey Realistic Expectations (i.e. the fairy's attempt to free the princess from the spell was unrealistic), but the reader was evidently looking for a moral more akin to Retribution.

Once upon a time there lived a wizard, a fairy and a princess.

One summer's morning the wizard cast a spell on the princess. As a result, the princess was under a spell. The princess felt distress that she was under a spell. The wizard felt gratification about casting a spell on the princess because the princess was under a spell.

A short time later the fairy tried to free the princess from the spell, but failed. The wizard felt satisfaction that the princess was under a spell. The wizard felt gloating towards fairy because the princess was under a spell. The fairy felt shame about trying to free the princess from the spell.

Figure 7.39: Suggesting modifications: punish the wizard

Another example of this kind of feedback, for the Family domain this time, concerns the story shown in Figure 7.40:

> It's senseless that the father takes food from his son unless he is a bad father. The story would have a moral if the father took food from the son and shared it between the three of them. The boy could still be angry that he got less food and yet realizes that it was the right thing to do and feels bad for feeling gloating for the little girl.

This is also a human-authored story, intended to convey Pride, but again the reader is looking for a different moral, and provides suggestions for how it can be achieved. This kind of feedback can be useful in making system improvements. In our case, there were a relatively small number of such comments, but a broader user study, which specifically asks participants to consider how stories could be improved, would be a valuable part of the development process.

In a nice house on a quiet street there lived a father, a boy and a girl.

One day the boy became hungry.

Later that day the girl became hungry.

Soon after that, the boy cooked lunch for himself.

Then the girl tried to cook lunch for herself, but failed and burnt herself. The boy felt joy that he had lunch. The boy felt gloating towards the girl because he had lunch.

A little later the father took lunch from the boy. As a result, he had lunch and the boy didn't have lunch anymore. The boy felt resentment towards the father because the father had lunch. The boy felt anger towards the father about taking his lunch because he didn't have lunch anymore.

Figure 7.40: Suggesting modifications: share the lunch

7.6.13 Realistic Expectations and Recklessness

In Chapter 5 (refer to Section 5.3.4), we identified that Realistic Expectations and Recklessness were often confused by the ILP learner. The evaluation survey showed the same thing: participants often confused these morals, not only for MOSS-generated stories, but human-authored stories as well. In fact, as mentioned in Section 7.3.1, the human authors themselves had difficulty separating them. Both morals are conveyed by stories in which characters suffer as an unintended result of their own actions. In the case of Realistic Expectations, this takes the form of a failed action; the character intends a particular outcome, but is unable to achieve it, hence their expectations are unrealistic. In the case of Recklessness, the moral can be conveyed through a failed action recklessly undertaken (sometimes difficult to distinguish from an unrealistic action), or through an action that was successful, but either did not have the intended outcome, or had additional negative outcomes. In both cases, the character experiences Distress at the end of the story, and this Distress results from their own action.

Some survey participants provided feedback that points to techniques for differentiating Realistic Expectations from Recklessness in stories. In distinguishing between these two morals, it appears that characters' beliefs and expectations, and the reader's awareness of them, play a pivotal role in which moral is understood. Figure 7.41 shows one of the MOSS-generated Recklessness stories that featured in the survey. One participant made the following comment regarding this story:

> How am I supposed to know if the troll picked the poisonous mushroom by accident or not?

Another participant made a similar comment about the same story:

> Was I to know that the troll knew the mushroom was poisonous? Because I don't.

If the troll knew the mushroom was poisonous, but ate it anyway, this would clearly be a reckless action. If not, however, then it is simply an honest mistake, and not necessarily reckless at all (unless one considers it reckless to eat a mushroom without knowing for certain that it is not poisonous, which these readers did not). Thus, to understand these morals readers must be informed about what characters know in advance, and what they expect from their actions.

Once upon a time there lived a troll.

One day the troll picked a poisonous mushroom.

Later that day the troll became hungry.

The troll hoped that he would not be hungry. Some time later the troll ate the mushroom. The troll felt satisfaction that he was not hungry. The troll felt distress that he was dead.

Figure 7.41: MOSS-generated story for Recklessness

There are basic changes which could be made at the discourse level to significantly improve readers' recognition of these morals. For Realistic Expectations, it would simply be a matter of explicitly stating a character's conviction that they will perform the action successfully. Their subsequent failure, juxtaposed with their prior confidence, would signal Realistic Expectations to a reader. For Recklessness, it is not quite as simple because, as explained in Section 5.5.6, we identify two reasons for which an action can be classified as reckless: either the agent has not considered the potential negative side-effects of the action, or the agent does anticipate a potential negative outcome but performs the action anyway. In the first case, the discourse should emphasise the character's confidence that nothing could possibly go wrong; in the second case, the character's awareness of the possible negative outcome should be highlighted, followed by their blatant disregard of this fact. Thus, even two stories with identical sequences of events could be tailored to one moral or the other, by modifying what the reader is told.

7.7 Limitations

In this section, we discuss the limitations not only of our approach to evaluation, but also of the study as a whole. We do not discuss limitations specific to the emotion model or the system implementation here; these were covered in detail in Chapters 4 and 6 respectively (refer to Sections 4.6.5 and 6.7). We begin with the limitations of the evaluation process in Section 7.7.1, before stepping back to assess the more general limitations of the study in Section 7.7.2.

7.7.1 Limitations of the Evaluation

There are several limitations of our approach to evaluation. Perhaps the most significant of these is that the survey is based on a forced-choice categorisation: participants are funnelled into selecting one of the six morals listed (or none), without the option of specifying an arbitrary moral. As such, participants may pick what they consider to be the closest moral, even though it is possible a better fitting moral exists. However, allowing participants to freely specify any moral (e.g. using a free text field) comes with its own set of problems. There is considerable ambiguity in most natural languages; it is therefore likely that different participants may use very different terminology to describe the same moral, or conversely the same term to describe different morals. The resulting data would be impossible to aggregate accurately. Even with a fixed set of morals and supplied definitions, readers' interpretations sometimes varied (refer to Section 7.6.9). By providing a limited set of responses, we ensure that the survey data can be analysed appropriately.

The use of numeric rating scales for certain criteria (i.e. moral strength, coherence, and interest) should also be interpreted with caution. Although we can calculate averages and compare the values between groups of stories, the ratings themselves are subjective. There is no way to judge the degree of consistency between participants; what an interest rating of 2 means to one participant may be completely different to what it means to another participant. Even for a single rater, the difference between a rating of 1 and 2 may not be the same as the difference between a rating of 4 and 5. Thus, while these ratings appear to be interval variables, and can be analysed as such, the subjectivity of the responses mean they are more ordinal in nature [54].

Another important limitation is the amount of data; a larger pool of responses would have produced more reliable results, and reduced the margin of error. A large number of stories were included in the evaluation (210 in total, as described in Section 7.4.1), and yet this set only contains 4 stories per moral of each type.[12] A more robust study would evaluate a larger number of stories for each moral, and gather more responses per story. However, this is difficult to achieve because a single survey participant cannot be expected to respond to hundreds of stories. As such, a very large number of participants is required to obtain enough data per story to be meaningful. In our case, we were only able to analyse the survey data in aggregate, per moral, and even then a larger number

[12] 4 stories per moral are included for MOSS-generated and human-authored stories, but only 3 per emotion filter for the moral-free stories, plus 4 with no filer. Story selection was described in depth in Section 7.4.1.

of responses would have been beneficial.

Much of the qualitative feedback provided by survey participants was less concerned with the events taking place within a story (the fabula) than by the way the story was presented (the discourse). This emphasises the importance of discourse in story interpretation. However, in our study, the discourse was deliberately simplistic, which may have reduced the stories' effectiveness in conveying morals. To properly assess the impact of discourse quality, the evaluation could also have incorporated stories with human-authored text. Authors could have been provided with event sequences corresponding to MOSS-generated and moral-free stories, and asked to write the corresponding text. Comparing the performance of the two groups of stories (automatically generated text versus human-authored text) would allow us to gauge the degree to which discourse influences the recognition of morals. The downside of this approach, however, is that it would double the number of stories comprising the evaluation survey, and therefore necessitate a much larger group of participants.

7.7.2 General Limitations

The most significant limitation of our study is the restricted domains to which all stories (MOSS-generated, human-authored, and moral-free) are confined. For human-authored stories in particular, such limited and precisely defined domains are not realistic. When people tell stories naturally, they do so without such restrictions. It is therefore not surprising that some of the authors participating in Stage 2 commented on the difficulty of writing within the supplied domains (refer to Section 7.3.1). It is important to remember that the performance of the human-authored stories, as presented in Section 7.5, is not a true reflection of how effectively human-authored stories convey morals in general. However, that was not the goal of the study; our aim was to define a baseline relative to which MOSS-generated stories could be evaluated. Human-authored stories would undoubtedly perform better if the story world was unrestricted, as too would system-generated stories. Unfortunately, generating stories in an infinite domain is beyond the capabilities of any current storytelling system, so there is no way, at present, to fairly compare them to the full power of human storytelling. Story worlds can certainly be made more complex than the simplistic domains we used, but will still fall short of the nearly limitless scope of a human author's imagination.

7.8 Discussion

In the preceding sections, we presented the results of our evaluation of MOSS's story generation capabilities, and acknowledged the system's limitations. Here we discuss the key implications of our study for moral-based story generation. In Section 7.8.1, we examine the similarities between certain morals which were indicated by the survey results, and how they impact story generation. In Section 7.8.2, we propose the development of a moral taxonomy to guide the design of moral-based storytelling systems.

7.8.1 Similarities Between Morals

Performing cross-validation on the moral rules which were derived using ILP (described in Chapter 5) revealed a strong resemblance between certain morals. This is corroborated by the results of our final evaluation: the same morals were frequently confused by survey participants, not only for MOSS-generated stories, but human-authored stories as well. This is a strong indicator of inherent similarities between the morals concerned, at least in terms of the character emotions that arise in the relevant stories. While it is certainly possible that a broader corpus with more examples of each moral would yield superior rules that better characterise them, it is also conceivable that distinguishing between closely related morals requires an extended representation, beyond just character emotions. Two groups of related morals emerged from our study: Retribution, Greed, and Pride, and Realistic Expectations and Recklessness.

Retribution, Greed, and Pride

Retribution is a very broad moral. The definition we provided in Chapter 3 (refer to Table 3.3) was as follows:

> If a character performs a reprehensible action, they will be punished.

There are no restrictions on what the action can be, provided it is reprehensible. We model this using Anger (i.e. Distress plus Reproach), because that is how the moral is manifested in Aesop's fables. However, an action can be considered reprehensible as long as it is against the ideals of society or other characters, regardless of whether it directly causes Distress. Thus, replacing Anger with Reproach in our moral rules for Retribution (refer to Listings 5.23 and 5.24) could also yield a valid representation. This aligns closely with Greed and Pride, which similarly involve a character being

punished for reprehensible behaviour. The only difference is the more specific nature of the action: the character must exhibit greed or pride. As such, Greed and Pride can be considered specific subclasses of Retribution: a story conveying Greed or Pride is, at the same time, also conveying Retribution.

The results of our evaluation support this relationship, through the large number of Greed and Pride stories that readers misclassified as Retribution. The relevant emotion patterns share key common elements (Reproach towards the protagonist, followed by the protagonist feeling Distress), making it difficult to differentiate them using emotions alone. This leads us to conclude that while high-level morals, such as Retribution, can be adequately represented in terms of emotion sequences, more specific subclasses of morals require another level of representation to characterise them effectively.

Realistic Expectations and Recklessness

Both Realistic Expectations and Recklessness are conveyed through characters suffering as an unintended result of their own actions. In the case of Realistic Expectations, this takes the form of a failed action. A character intends a particular outcome, but is unable to achieve it, which demonstrates that their expectations were unrealistic. In the case of Recklessness, the moral can either be conveyed through a failed action recklessly undertaken (sometimes difficult to distinguish from an unrealistic action), or through an action that was successful, but had undesirable consequences. For both morals, the protagonist feels Hope for a desired outcome, followed by Distress about the actual outcome of performing the unrealistic/reckless action.[13]

The similar emotion patterns defining these morals resulted in MOSS generating similar stories for both, leading to frequent confusion by survey participants. However, confusion between Realistic Expectations and Recklessness occurred not only in the set of MOSS-generated stories, but the human-authored stories as well. In fact, some of the participants authoring stories for the human-authored set commented about writing the same, or extremely similar, stories for both (refer to the discussion of author feedback in Section 7.3.1). This indicates an inherent similarity between these morals, at least in terms of the character emotions arising in the relevant stories.

[13]Hope and Distress do not appear explicitly the relevant moral rules (as presented in Chapter 5), but must arise to produce the emotions which are specified. In the case of Realistic Expectations (refer to Listing 5.5.5), the protagonist must feel Disappointment, which implies both Distress and a preceding Hope. In the case of Recklessness Type 1 (refer to Listing 5.10), Distress is explicitly included in the rule, and Hope is implied by Satisfaction. Recklessness Type 2 does not require Hope, but Distress is required through Remorse.

The implication of this similarity is that emotions alone are insufficient to differentiate between Realistic Expectations and Recklessness: the same story could convey either. However, an extended representation may not be necessary if they can be distinguished at the discourse level, as discussed in Section 7.6.13. Describing the same sequence of events differently, by emphasising a character's confidence in the success of their action, or showing their awareness of possible negative side-effects, could direct readers to one interpretation over the other.

We conclude from this that emotions alone, while capable of characterising certain morals, are unable to adequately distinguish between those that are closely related. Multiple levels of representation are needed, with emotions positioned at the highest level of abstraction. To identify what these other levels of representation are, the relationships between morals need to be identified. We conjecture that morals can be organised into a hierarchy to achieve this end, and discuss this notion in the following section.

7.8.2 Developing a Taxonomy of Morals

Based on the results of our experiments, participant feedback, and the points raised in the preceding discussion, it appears that broad, high-level morals, such as Retribution and Reward, can be represented effectively using only character emotions. However, isolating more specific subclasses of these morals (for example, Greed and Pride) requires additional representational features. In some cases, this may be achievable at the discourse level, as suggested in Section 7.8.1 regarding Realistic Expectations and Recklessness. For other morals, rules may need to incorporate information about characters' beliefs (for example, representing Lies would surely involve the notion of incorrect beliefs), or even physical constraints (for instance, according to some Stage 2 authors, demonstrating Greed requires the character to accumulate multiple instances of a particular object).

In light of this, it would be valuable to develop a comprehensive taxonomy of story morals, which identifies the story features necessary at each level to represent those morals effectively. Figure 7.42 shows a fragment of such a classification; the hierarchical structure portrays which morals are subclasses of others. We do not mean to imply that this classification is optimal, or even necessarily accurate, but include it as an illustration of what needs to be achieved. There are a myriad of morals that can appear in stories. In Chapter 3, we identified 28 distinct morals in a corpus of only

85 fables. Constructing a full taxonomy would require a thorough analysis of all the morals appearing across a broad range of fable collections, to identify any relationships between them, as well as minimal representational requirements. Such a classification would serve as a cornerstone for moral-based storytelling systems.

7.9 Summary

In this chapter, we described our approach to evaluating the Moral Storytelling System (MOSS), by comparing MOSS-generated stories to human-authored stories and moral-free event sequences. We presented the results of our survey-based evaluation, finding that, overall, MOSS-generated stories conveyed morals significantly better than moral-free stories, though human-authored stories had the best performance. The coherence and interest of MOSS-generated stories also surpassed that of moral-free stories, despite no attempt to enforce coherence or interest at the plot level. Again, as expected, human-authored stories were deemed the most coherent and interesting. We also presented key elements of participant feedback, in the form of comments pertaining to specific stories, and suggested system modifications to address common concerns. After identifying the limitations of our approach to evaluation, we concluded with a discussion of the implications of the results, and proposed the development of a taxonomy of morals to guide further research in moral-based storytelling. In the following chapter, we draw final conclusions based on our work.

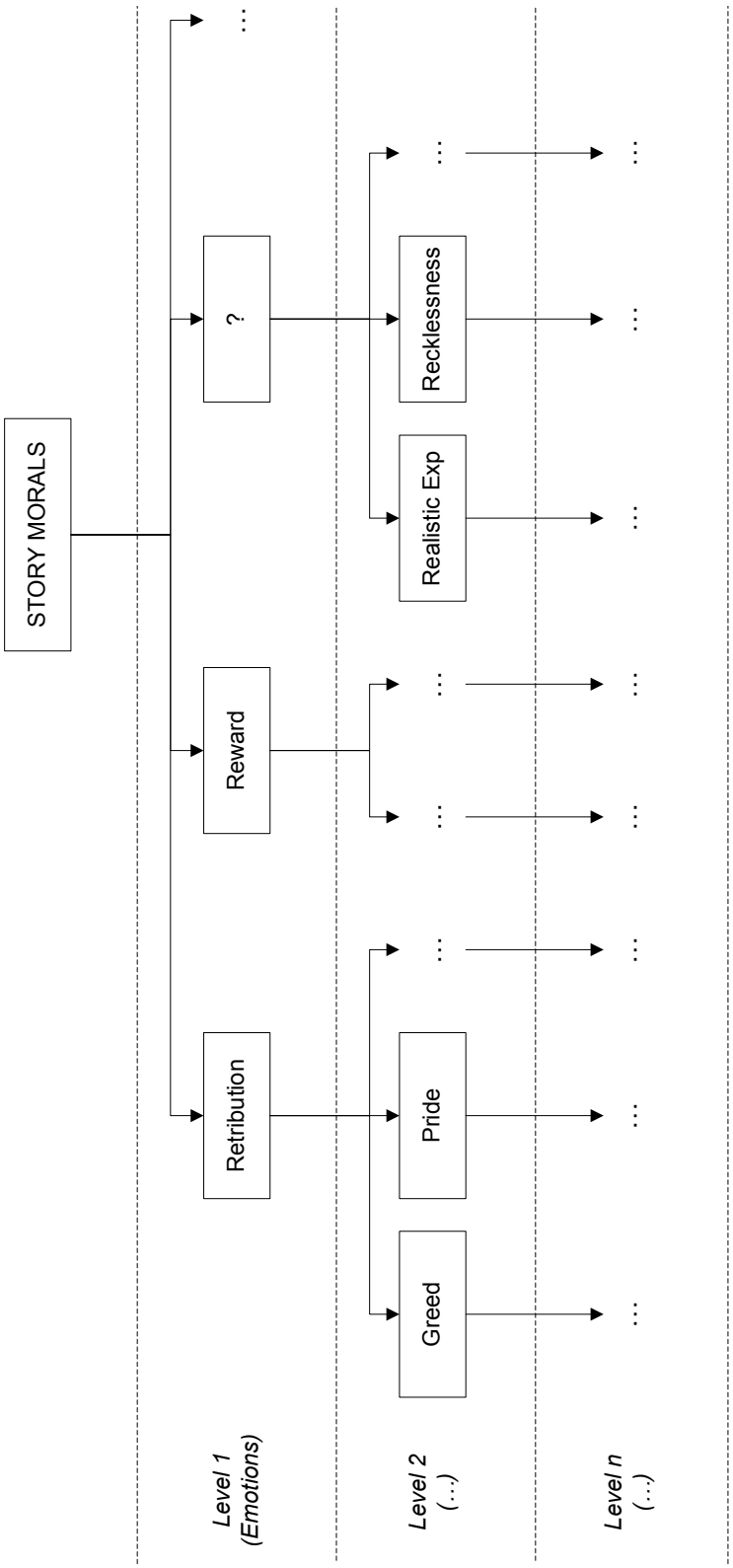

Figure 7.42: A fragment of a possible taxonomy of story morals

Chapter 8

Conclusion

In literature and in life we ultimately pursue,
not conclusions, but beginnings.

Sam Tanenhaus
Literature Unbound

A key aspect of storytelling that has been overlooked by the majority of storytelling systems to date is teaching a lesson: the moral of the story. In this thesis, we address two key research questions: is there a relationship between character emotions and the moral of a story, and can this relationship be leveraged to develop a representation for story morals in terms of emotions, thereby facilitating the automatic generation of stories with morals. We address the first research question in Section 8.1, by asserting a correlation between morals and character emotions. Section 8.2 is directed at the second research question: we assess the efficacy of emotions as a representation for morals, and suggest augmenting them with other story features to better distinguish those that are closely related. Section 8.3 provides a final reflection on our Moral Storytelling System (MOSS), and Section 8.4 summarises our conclusions.

8.1 Morals and Emotions: Are They Related?

Our first objective was to determine whether there is a relationship between the moral of a story and the emotions experienced by the characters during the course of that story. Our approach was to produce emotion data from a selection of Aesop's fables, and look for common emotion patterns among stories with the same moral. Our analysis

was conducted using a combination of inductive logic programming (ILP) and manual inspection.

Even with a limited corpus, we were able to identify emotions corresponding to particular story morals through the use of ILP. Despite the high margin of error in the results, due in part to the small number of examples available for each moral, they support our hypothesis: the moral of a story is related to the character emotions arising in that story. Although the rules produced using ILP could not be applied directly to story generation, they provided a useful starting point for developing the emotion-based ASP rules for morals which were ultimately used by MOSS to generate stories.

8.2 Emotions as a Representation for Morals

The second goal of our research was to gauge the viability of emotions as a representation for morals, in the context of story generation. To investigate this, we developed emotion-based ASP rules corresponding to six common morals: Retribution, Greed, Pride, Realistic Expectations, Recklessness, and Reward. These rules served as the foundation for our moral-based storytelling system: MOSS. Our survey-based evaluation of the stories MOSS produced showed that they conveyed morals significantly better than moral-free event sequences, though they were outperformed by human-authored stories. In Section 8.2.1, we use the results of this study to conclude that emotions can serve as a representation for morals. In Section 8.2.2, we address the related question of whether emotions alone provide a sufficiently comprehensive representation, or whether they should be augmented with other story features to produce better results.

8.2.1 Can Emotions Represent Morals?

The premise underlying our work was the assumption that morals and character emotions are closely related, and therefore emotions can be used as a foundation for representing morals. The results presented in Chapter 7 support this hypothesis, in that MOSS-generated stories convey morals almost as effectively as human-authored stories in the given domains, and certainly far better than moral-free event sequences. This confirms the feasibility of an emotion-based representation for morals. This approach has a number of advantages for developing storytelling systems.

A key benefit of using emotions as representational units is the ease of expressing sophisticated concepts. This results from the fact that emotions themselves are a complex phenomenon. A single emotion arises due to a particular combination of an event with a character's desires and ideals; a specific sequence of emotions therefore imposes significant constraints on a story. Emotions themselves are not simple to model, but once this is achieved, they yield significant representational power, particularly when compared to simpler representational units, such as Lehnert's plot units [87]. On the other hand, emotions are much simpler than highly structured representations like Dyer's Thematic Abstraction Units [48], Turner's Planning Advice Themes [170], and Wilensky's point prototypes [177]. This intermediate level of abstraction provides a balance between simplicity and power.

Emotions allow readers to empathise with the characters in a story. Empathy, in turn, plays a role in story interpretation: understanding how the events in a story affect characters emotionally helps readers perceive the morals in stories. In this way, emotions are closely tied to morals at a conceptual level. Although the precise definitions of emotion words can vary, people have an innate sense of what they mean, at least within a given culture. Consequently, rather than stemming from arbitrary, computationally derived correlations between vaguely related concepts, emotion-based moral rules have an obvious semantic interpretation. This greatly simplifies the process of developing, testing, and refining such rules, and can also make them relevant to fields outside the sphere of computing, such as literature or psychology. At the same time, any developments in these fields which shed light on the nature of the relationship between morals and emotions can be applied to computational storytelling.

8.2.2 Are Emotions Enough?

In the previous section, we asserted that emotions can be used to represent morals. The caveat, however, is that although we have shown emotions to be a valuable representational tool in this context, that is not to say emotion-based representations for morals cannot be improved by incorporating other story features. Rather, emotions should be thought of as one level of representation, which may be sufficient to fully capture some morals, but must be augmented with additional elements to effectively portray others.

The results of our study suggest that high-level morals, such as Retribution and Reward, can be adequately represented using emotions alone. More specific subclasses of these broad morals, such as Greed and Pride, require additional levels of representation.

Inherently similar morals, such as Realistic Expectations and Recklessness, also cannot be distinguished effectively based on emotions alone, though discourse-level techniques can help readers identify them correctly.

8.3 Moral Storytelling System

Our Moral Storytelling System (MOSS) uses specific patterns of OCC emotions to generate stories that convey morals. Its present scope encompasses six morals, which only scratches the surface of the morals that can be found in fables, let alone stories more generally. However, the system successfully pilots the use of emotions for representing morals, in that overall, MOSS-generated stories proved to be more coherent, more interesting, and convey morals better than the control group of moral-free event sequences.

We adopted a declarative approach to implementation, using answer set programming, which proved useful for controlling story trajectories due to the ease of specifying a variety of plot-level constraints. This also makes the system readily extendable, not only to include other morals, but also richer character models, and constraints based on other story features. Although MOSS has limitations, it provides a useful platform for investigating emotion-based plot control, not only in the context of morals, but also more generally.

8.4 Summary

To judge whether emotions can serve as a viable representation for morals, we developed a storytelling system, MOSS, which uses emotion-based rules to generate stories with morals. Our evaluation shows that the resulting stories convey morals almost as effectively as human-authored stories within the same restricted domains, and significantly better than the baseline group of moral-free event sequences. This validates the use of emotions for representing story morals. However, the strong resemblance between certain morals renders emotions alone insufficient as a complete representation. Rather, they lay a foundation which can be built upon using other aspects of story to produce a comprehensive model for moral-based story generation. Our research is only the first step down the long road of emotion-driven, moral-based storytelling. In the following chapter, we speculate about where this road can lead through a discussion of

key avenues for future work.

Chapter 9

Future Work

> *I don't think traditional fiction is dying, and I don't think the universal grammar will ever change. But I do think storytelling will evolve in new directions over the next fifty years.*
>
> Jonathan Gottschall
> *The Storytelling Animal*

It was a rainy Saturday afternoon. John reclined on the couch to watch a holo-story. Lucy sat on the floor nearby, playing with her interactive fairytale. Mike was in his room, headphones on, oblivious to the outside world as he played the protagonist in the latest interactive virtual-reality narrative. Any—or all—of these are viable directions for storytelling to take. Some are not far off, or already in existence. The nature of how we experience story has undeniably evolved. What has not changed is its critical role in our lives:

> Story, in other words, continues to fulfill its ancient function of binding society by reinforcing a set of common values and strengthening the ties of common culture. Story enculturates youth. It defines the people. It tells us what is laudable and what is contemptible. It subtly and constantly encourages us to be decent instead of decadent. Story is the grease and glue of society: by encouraging us to behave well, story reduces social friction while uniting people around common values. Story homogenizes us; it makes us one. This is part of what Marshall McLuhan had in mind with his idea of the global village. Technology has saturated widely dispersed people with

> the same media and made them into citizens of a village that spans the world. [65]

Morals are central to these functions of story. Our work on automatically generating stories with morals is only a first step towards incorporating morals into the new forms of storytelling made possible by technology.

In this chapter, we identify future directions for work in this area. We begin with specific extensions that can be made to MOSS's Action, Belief, Emotion, and Moral Layers in Sections 9.1, 9.2, 9.3, and 9.4 respectively, before moving on to broader areas of future work. In Section 9.5, we suggest enhancements in character modelling, before discussing the potential for interactivity in Section 9.6. Section 9.7 touches on the value inductive logic programming for developing moral representations. In Section 9.8, we take a step back and identify how MOSS could be used as a tool for investigating other aspects of storytelling, unrelated to morals. Although our focus has been on producing the fabula, not the discourse, in Section 9.9 we identify text generation as a key area for improvement. Finally, in Section 9.10 we reflect on the current state of moral-based story generation, and the road ahead.

9.1 Extending the Action Layer

The story generation capabilities of any system are strongly dependent on the underlying action model. To produce more sophisticated stories, MOSS requires a more sophisticated action model. The obvious way to extend the Action Layer would be to increase the domain complexity, by including a broader range of events/actions, characters, objects, and so on. However, changes to the action model can also be made at a more fundamental level, to address the limitations we identified in Chapter 6. In this section, we discuss three possible extensions which would significantly improve the system's expressivity: permitting characters to possess multiple instances of objects (Section 9.1.1), extending the notion of success and failure (Section 9.1.2), and modelling speech acts (Section 9.1.3).

9.1.1 Possessing Multiple Instances of Objects

As mentioned in Chapter 4 (refer to our discussion of the Action Layer, in Section 4.5.4), we do not track quantity in our action model. As a result, characters can only

possess one of any particular object at a time. One of the authors participating in Stage 2 of our evaluation commented on this limitation with respect to conveying the moral Greed (refer to Section 7.3.1). In this author's view, showing Greed would require a character to obtain multiple instances of the same object, which none of our domains permit. Implementing numeric quantities in our action model, and thus allowing us to track how many instances of a particular object each character has at any given time-point, would allow MOSS to generate stories in which characters accumulate items. This could directly facilitate an improved representation for Greed.

9.1.2 Success and Failure

At present, MOSS treats the notion of success as binary: an action can succeed, or it can fail (i.e. not succeed). Real life is rarely that simple. An action can certainly succeed or fail completely, but partial success is also possible, to varying degrees. For instance, consider a simple example in which a knight attempts to steal a chest of treasure from a dragon. Using our current approach to success and failure, there are two possible outcomes: either the knight successfully steals the treasure, or he does not. However, to model this action realistically, there should be several possible outcomes. We list some possibilities below:

1. The knight obtains the treasure (success).
2. The knight does not obtain the treasure (failure).
3. The knight is eaten by the dragon (also failure, but in a more extreme sense).
4. The knight obtains the treasure chest, but it is empty (partial success).

Most real actions can have more than two possible outcomes, depending on specific circumstances. The simplest way to achieve this would be to extend MOSS's action model to incorporate separate `succeeds` predicates corresponding to each possible (independent) consequence. The price would be a significantly more complex action model. While this would allow for a much wider range of possible stories, there would doubtless be an impact on the runtime of the ASP grounder and solver.

9.1.3 Speech Acts

At present, all the events and actions available in MOSS's storytelling domains, as well as their consequences, are physical. This does not allow for communicative actions

(i.e. an agent passing on information—true or false—to another agent). Modelling speech acts not only makes it possible for characters to pass on information, but also to lie. Lying is the third most common moral appearing in Aesop's fables, as identified in Chapter 3. It cannot be handled by MOSS in its present form, not only because characters are unable to communicate, but also because they cannot have incorrect beliefs. Thus, for characters to be able to lie, changes would also need to be made at the belief level (we discuss this in Section 9.2.2). At the action level, beliefs would need to be treated as possible consequences of actions (specifically, as consequences of speech acts).

9.2 Extending the Belief Layer

As explained in Chapter 4, MOSS's belief model is extremely simplistic. Given that emotions are derived directly from characters' beliefs, implementing a more sophisticated belief model would significantly impact the emotion model, and in turn the system's story generation capabilities. We discuss removing the omniscience assumption in Section 9.2.1, which would in turn facilitate characters having incorrect beliefs, discussed in Section 9.2.2.

9.2.1 Removing Omniscience

MOSS treats all characters as omniscient: their beliefs correctly reflect what is true in the story world.[1] This includes beliefs about other characters' desires, which directly impact the fortunes-of-others emotions. The omniscience assumption limits the stories MOSS can generate, because many interesting storylines can emerge as a result of characters acting on incomplete information. Modelling speech acts (Section 9.1.3) would provide a mechanism through which characters can update their beliefs (i.e. when information is communicated to them by another character).

As mentioned in Chapter 3, belief revision is a complex issue, which has been widely studied in its own right [11, 17, 33, 59, 114, 171]. We would need to specify the conditions under which characters adopt new beliefs, or change existing beliefs. This could depend on many factors, including their level of trust for other characters, which

[1] As explained in Chapter 4, all characters who are located onstage are aware of all actions taking place, and their outcomes. Characters located offstage do not have beliefs, and thus cannot experience emotions.

we currently do not model, as well as how their existing beliefs originated. For example, if a character knows something about the world first-hand (i.e. they saw the relevant event happen), they might not be prepared to believe another character telling them otherwise. On the other hand, if a belief is second-hand, it should be more open to revision.

9.2.2 Incorrect Beliefs

Removing omniscience from the belief model (discussed in Section 9.2.1), in addition to allowing for gaps in characters' knowledge of the world, opens the door for incorrect beliefs. The information characters communicate to others does not necessarily have to be true. Similarly, this would enable characters to misinterpret events, and to have incorrect beliefs about others' desires. The potentially inappropriate fortunes-of-others emotions that may arise as a result (for example, feeling HappyFor towards someone when in fact they are experiencing Distress rather than Joy) could also lead to interesting interactions between characters.

9.3 Extending the Emotion Layer

Emotions play an important role in stories: they allow readers to empathise with the characters. For characters to be realistic and believable, their emotions need to be consistent with their traits and personality. As such, improvements to MOSS's Emotion Layer will produce more realistic characters, which readers can relate to better. Given that we use emotions as our foundation for representing morals, enhancing the emotion model can also lead to stories that more effectively convey morals.

MOSS's emotion model makes some simplifications to the OCC theory, as described in Chapter 4. The most obvious extension is to model emotion intensity, which we discuss in Section 9.3.1. Incorporating mood, which we describe in Section 9.3.2, could lead to more sophisticated characters. Finally, we suggest implementing character-specific definitions for certain emotions (the fortunes-of-others emotions in particular) in Section 9.3.3.

9.3.1 Intensity Variables

Human emotion is not black and white. There are countless shades of grey: every emotion can be experienced at a variety of intensities. In Ortony et al.'s words, this results in an "infinitude of phenomenally possible emotions." [118] As part of the OCC theory, Ortony et al. describe a number of intensity variables, which determine the strength of an emotion. They break these variables down into three main categories:

1. Central intensity variables, which affect all emotions of a particular type (i.e. event-based, attribution, or attraction): desirability, praiseworthiness and appealingness.

2. Global variables, which affect all emotions: sense of reality, proximity, unexpectedness, and arousal.

3. Local variables, which only relate to particular groups of emotions: likelihood, effort, realisation, desirability-for-other, liking, deservingness, strength of cognitive unit, expectation-deviation, and familiarity.

To truly capture the subtleties that can arise in characters' reactions and interactions, an emotion model needs to address intensity in some way.

MOSS does not model emotion intensity; we identified this as a limitation in Section 4.6.5. While this is sufficient for simple stories, as plot lines become more complicated, so do the emotions characters experience. Even modelling a small subset of the intensity variables identified by Ortony et al. (e.g. just the central intensity variables) would markedly increase the depth of the emotion model, and give much finger-grained control over a story. This in turn would allow for more specific moral definitions. For example, in the context of Retribution, the punishment for a reproachable action could depend on how serious the transgression. Killing someone, for instance, should attract a more severe punishment than stealing a loaf of bread. This would ultimately make stories more believable.

Having a numeric value associated with desires and ideals would also facilitate more consistent character behaviour.[2] For instance, generally a character would not be expected to steal from someone they like. However, if they desire an object above a certain threshold (relative to how much they like the other character), they might break this

[2]Although desires and ideals are defined in the Belief Layer, not the Emotion Layer, we discuss them in this section because associating a numeric valence with these elements directly influences, and to a large extent defines, emotion intensity.

ethic and steal from them anyway. A concrete example might be a person who is starving or broke stealing from their friends out of desperation. As explained in Chapter 6, in its present form, MOSS does not attempt to make characters behave rationally, and so characters' feelings towards each other do not influence their actions. However, if a more sophisticated character model were incorporated (this will be discussed in Section 9.5.4), modelling the strength of desires and ideals would facilitate more believable behaviours.

Similarly, modelling emotion intensity would facilitate a more realistic model of the object-based emotions, Love and Hate. As discussed in Chapter 4 (refer to Section 4.4.4), we adopt a very simplistic implementation of these emotions, based directly on Admiration and Reproach. The result is that characters' Love or Hate for each other can fluctuate erratically during a story, often due to relatively minor events. Modelling intensity would allow us to define a threshold for when these emotions can change: e.g. only when a character performs an action inciting significant Admiration or Reproach, or when there is a sufficient build-up of these emotions resulting from several less significant actions. This would make the relationship dynamics between characters more sensible, and easier for readers to understand. Several survey participants commented on the confusing nature of the Love/Hate relationships in MOSS-generated stories (refer to Chapter 7, Section 7.6.2).

9.3.2 Modelling Mood

In their discussion of the OCC theory, Adam et al. [8] highlight the distinction between emotion and mood: emotions are momentary and transient, whereas mood is an affective state that has a longer duration. MOSS currently does not take mood into account.[3] Incorporating a model of mood would result in more sophisticated emotional patterns, because a character's mood could affect the way they react emotionally to certain events. For example, a character in a bad mood might react more strongly to a slightly negative outcome than they would if they were in a good mood (assuming a model of emotion intensity, as described in Section 9.3.1). Ortony et al. raise the idea of a threshold value for certain emotions, whereby a good mood would increase the threshold for experiencing negative emotions, and decrease it for positive emotions. A bad mood would have the opposite effect. Characters may also be more self-absorbed

[3]Although Love and Hate persist through time in our implementation of the OCC theory (refer to Section 4.5.3), they are not moods.

if they are in a bad mood, and less likely to feel fortunes-of-others emotions such as HappyFor or Pity. In this way, incorporating mood into the emotion model would help make characters more realistic. There has been previous work on implementing mood in the context of believable character agents (for example, Burkitt and Romano's work [30]), which could be applied to our system.

9.3.3 Character-Specific Emotion Models

As explained in Chapter 6, MOSS's Emotion and Moral Layers are decoupled from the Action and Belief Layers. Consequently, all characters in all domains follow a consistent emotion model. However, personality can conceivably be a factor in how emotions are experienced. For example, a selfish character might not feel emotions such as Pity as readily as a nice character (or at all). Furthermore, some characters may be more emotional than others; i.e. they experience emotions more readily, and more strongly, as a result of the same situations (this would correspond to a lower threshold for particular emotions, as per Section 9.3.1). In light of this, it may be of value to provide personality-specific rules for certain emotions. These could be defined for individual characters, or based on personality types.

9.4 Extending the Moral Layer

In MOSS, we implemented rules for six common morals. These rules are based on the emotion patterns arising in Aesop's fables. There are a number of ways in which the system could be extended at the moral level. We discuss three of these as notable avenues for future work: incorporating a broader range of morals (Section 9.4.1), generating stories with multiple morals (Section 9.4.2), and conveying morals using parallel story lines (Section 9.4.3).

9.4.1 A Broader Range of Morals

The holy grail of moral-based story generation would be a system that can generate a story which conveys any moral the user specifies. This goal lies well outside the scope of current systems. For that matter, we cannot be certain whether it is achievable at all. However, we can take steps towards it by gradually increasing the number of morals we

are able to represent. This not only results in a more useful system, but also increases our understanding of morals as a concept.

Our work focusses on a narrow range of morals. The next step is to broaden this to include others. This will require an analysis of additional fable collections (the most renowned of these were identified in Chapter 3), to obtain sufficient examples of other morals to identify meaningful emotion patterns. Developing the moral taxonomy proposed in Chapter 7 would also be of value, to identify relationships between morals, as well as other levels of representation that may assist in conveying them effectively. For example, the extensions described for MOSS's Action and Belief Layers (refer to Sections 9.1 and 9.2) facilitate the inclusion of morals that cannot be handled by the system in its present form, such as Lies and Superficiality.

9.4.2 Multiple Morals

Very simple stories, such as fables, are usually tailored to convey one particular moral. However, as stories become longer and more complex, they can contain many different morals, either intertwined or occurring in different parts of the story. In Dyer's words [48]:

> Actually, there is no such thing as 'the' moral of a story when the story is complex.

MOSS, in its present form, can already be used to generate stories with multiple morals; it is simply a matter of specifying multiple moral constraints at the same time. We did not explore this possibility in this study, leaving it as an avenue for future work.

9.4.3 Conveying Morals by Comparison

Morals can be conveyed by the juxtaposition of one situation with another. According to Booker [25], the same essential story can be told in two ways:

1. Good: a protagonist with positive characteristics successfully develops and gets a happy ending.

2. Bad: a protagonist with negative qualities gets what could be a happy ending, but it is tainted in some way.

This notion can be used in comparative tales, which follow two characters with parallel paths, but different attitudes or goals. The moral is demonstrated through one character getting a happy ending, but the other not. This is akin to the idea of clones, also called mirror characters or reflection characters [105], which are used to show what might happen to the main protagonist if they took a different path. The use of clones in this way is common in stories. McDonald [105] uses the *Three Little Pigs* as an example: the first two pigs are clones, whose failure is used to emphasise the third pig's success. This approach could be applied to any of the six morals MOSS currently handles, by defining the required parallel emotion sequences as part of the moral rules. Comparing the performance of the resulting stories to those that do not use clones would allow us to gauge the impact of this technique on readers' understanding of morals.

9.5 Characters

Characters are an important part of stories. Their actions drive the plot forward. They serve as the foundation for the character-centric approach to narrative generation, but are often neglected in author-centric systems. As identified in Chapter 2, the main shortcoming of the author-centric approach is that the characters in the stories do not behave in a consistent or believable fashion. This is evidenced by some of the examples of MOSS-generated stories which were presented in Chapter 6: for example, a knight loving a fairy, fearing she will die, and then killing her, only to feel distress about it. Such stories arise because MOSS is author-centric: constraints on stories are enforced only at the plot level, without taking into account how characters' actions relate to their desires or emotions. We took this approach because our focus was on how emotions can be used to generate sequences of events conveying morals: only plot-level control is required to investigate this. To achieve more realistic characters, and thus improve the coherence and believability of MOSS-generated stories, we need to incorporate a more sophisticated character model.

Presently, MOSS characters are defined based on their desires, ideals, and any relevant physical properties (e.g. for the Animals domain, one such property is whether the character is a carnivore, herbivore, or omnivore). Although characters' physical properties serve as action preconditions, desires and ideals are used only to define emotions, and have no impact on character behaviour. In this section, we discuss enhancements that could be made to our basic character model, including automatically defining char-

acters (Section 9.5.1), allowing for characters' personalities to evolve during a story (Section 9.5.2), explicitly modelling goals (Section 9.5.3), and imposing constraints to ensure rational character behaviour (Section 9.5.4).

9.5.1 Automatically Defining Characters

Although all the characters used to generate stories for this work were pre-defined, there is no reason why characters' personalities cannot be automatically designated by the system, as part of the solving process. In the most basic sense, this would simply be a matter of leaving standards and desires to be allocated as appropriate to produce the required emotion sequences. Research could also be undertaken into how varying characters' ideals and desires impacts story generation. If all characters have the same standards, storylines that are interesting (at least from a narrative perspective) rarely arise. Differences in standards, and the resulting behaviours, cause conflict, which is a crucial aspect of interesting stories [127].

9.5.2 Evolving Personality

Currently in MOSS, characters are completely defined programmatically in each storytelling domain (i.e. their properties, desires, and ideals), and these definitions are fixed.[4] In real life, and therefore in all but the simplest of stories, characters develop and change as a story progresses. In fact, morals are often conveyed through character development: a character learns important life lessons during the course of the story, and through them so does the reader. Burkitt and Romano [30] describe a model for believable characters which allows personality to evolve based on the emotional experiences characters have. This would be an extremely useful extension for a storytelling system attempting to convey morals.

9.5.3 Goals

In MOSS, we do not model characters' goals directly. They are indirectly taken into account through characters' desires, but these desires are hard-coded in the Belief Layer. Agents' standards, or ideals, are similarly hard-coded. There has been some work by Battaglino, Damiano, and Lesmo [22] on relating agents' standards to their goals and

[4] Although desires have a time parameter and can change during the course of a story, this is based on fixed domain-specific circumstances.

values. They define the praiseworthiness of an action based on the goal that motivates its execution, and the blameworthiness of an action based on its effects in terms of putting an agent's values at stake. This provides a scheme for deriving ideals from characters' goals, which would reduce the aspects of character that need to be hard-coded in the system, as well as allowing characters' ideals to evolve accordingly if their goals change.

9.5.4 Rational Behaviour

In its current form, MOSS makes no attempt to make characters behave rationally. Constraints on story trajectories exist only at the plot level, in terms of the required emotion sequences, without taking into account whether a particular action is consistent with a character's desires, ideals, or personality. As explained in Chapter 6, this can lead to confusing and inconsistent behaviour. A more sophisticated character model, in which characters' desires, ideals, and goals play a role in determining their actions, would result in more believable characters and plot lines. Achieving this would be aided by modelling the intensity of desires and emotions, as proposed in Section 9.3.1. It is reasonable for a character to risk adverse consequences if the intended outcome of their action is desirable enough, but not if the adverse consequences far outweigh the desirability of that outcome.

9.6 Interactivity

One of the key benefits computing brings to storytelling is the potential for interactivity. A system with the capacity to generate stories with morals in arbitrary domains is already valuable in an educational context, by allowing stories to be tailored to the interests of individual readers. This is particularly relevant for children's education, because individually tailored stories would motivate children to learn by keeping them interested. Making these stories interactive takes the potential for educational applications one step further, also fostering engagement by virtue of interactivity.

An interactive moral-based storytelling system would allow readers to make decisions on behalf of the characters, and adapt the story to convey appropriate moral outcomes. To illustrate with a simple example, if the reader selects a reprehensible action, the character will suffer as a result. On the other hand, if they choose to act in a virtuous way, they will be rewarded. This would provide readers with an opportunity to

learn by experience, without having to perform such actions themselves. In its present form, MOSS is not interactive. The system generates complete stories for the specified moral, with no room for intermediate user input. Introducing an interactive component would be a step towards producing a useful tool for children's moral education.

9.7 Inductive Logic Programming

Inductive logic programming (ILP) is a useful technique for automatically deriving rules which characterise a data set. In the context of storytelling, it can be applied to data corresponding to existing stories in order to learn rules which can then be used for story generation. In our case, the story data was emotion-based, though the technique could also be applied to other story features.

Our use of ILP was limited by the small number of fables contributing emotion data. However, the technique shows promise. With a sufficiently broad corpus, there is no reason why rules produced using ILP could not be used directly for generating stories with morals, without the need for manual tweaking. This would require producing emotion data from other collections of fables, to increase the number of examples available for each moral. ILP could also be applied to the stories human authors composed as part of our evaluation process (refer to Chapter 7, Section 7.3). This would not only broaden the corpus, but also incorporate stories written from a more modern perspective; most popular fable collections date back at least decades, if not hundreds of years. It may yield useful insights about the way storytelling has evolved to compare the rules derived from Aesop's fables to those based on stories written by modern authors.

The other important factor in theory construction, aside from the size of the corpus, is the background knowledge provided to the ILP system (in our case, Aleph). Aleph can only construct rules using the predicates defined in the background knowledge file. Our goal was to find rules expressing morals as temporal sequences of emotions, so we only defined two basic temporal relationships for Aleph to use. Defining a broader range of predicates, either temporal or otherwise, would allow Aleph to leverage other relationships between character emotions, potentially leading to better rules. The trade-off would be a significant increase in the search space, which would also make the choice of search algorithm, another aspect of ILP we did not investigate, more important.

Thus far, ILP has not been broadly applied to the development of storytelling systems. However, the possibility of automating rule development for story generation,

moral-based or otherwise, certainly warrants a comprehensive exploration of the potential for ILP in the narrative space.

9.8 Alternative Applications

The primary goal of our system was moral-based story generation. However, it can also serve as a tool for investigating other aspects of stories. Although MOSS was designed to generate stories with particular morals, the Moral Layer can be removed, and replaced with rules corresponding to some other characteristic of interest. For example, it may be worth investigating whether particular sequences of character emotions result in stories that are more interesting, more suspenseful, more enjoyable, and so forth. This could even be approached as a machine learning problem, whereby the system generates stories at random, and readers rate them according to various properties. This would allow the system, over time, to learn which emotional trajectories are most suited to achieving particular storytelling goals.

9.9 Text Generation

Given our focus throughout this work on fabula generation, most of the suggestions for future work presented in this chapter relate to improvements in story planning. However, readers do not perceive the fabula directly; they experience it through the discourse. As such, the way a story is presented can significantly impact interpretation.

As discussed in Chapter 7, there are a number of relatively straightforward changes that could be made to MOSS's text generation component which would yield significant improvements in story quality. For example, omitting posthumous emotions and irrelevant consequences from the discourse can make stories appear more coherent and sensible. Even just describing the same sequence of events in a different way can change which moral readers perceive in a story (refer to Section 7.6.13).

In addition to such basic changes, general improvements in the language, to make it less simplistic and repetitive, would significantly impact the quality of the resulting stories. However, effective text generation is a difficult problem in its own right. As Callaway and Lester [31] point out, story generation and natural language generation have progressed largely separately, due to the difficulty of the problems in both fields. A key area of future work for storytelling systems in general is better integration with state

of the art work in natural language generation. Without it, automatically generated stories will never truly rival those that are human-authored.

9.10 The Final Word

As evidenced by the areas of future work identified in this chapter, there is still a long way to go before we can realise the full potential of automatic, moral-based story generation. Although further refinement in modelling both emotions and morals is required for practical, real-world applications, our work lays the foundation for an emotion-driven approach, establishing character emotions as a promising representational technique for morals. Despite the remaining challenges, the value of moral-based story generation, particularly in areas such as education, makes it a goal worthy of pursuit.

Bibliography

[1] "hubris, *n*". In *Oxford Dictionaries*. Oxford University Press. Accessed January 9, 2014. http://www.oxforddictionaries.com/definition/english/hubris.

[2] "moral, *n*". In *Oxford Dictionaries*. Oxford University Press. Accessed November 29, 2013. http://www.oxforddictionaries.com/definition/english/moral.

[3] "narrative, *n*". In *Oxford Dictionaries*. Oxford University Press. Accessed February 25, 2014. http://www.oxforddictionaries.com/definition/english/narrative.

[4] "point, *n*". In *Oxford Dictionaries*. Oxford University Press. Accessed November 29, 2013. http://www.oxforddictionaries.com/definition/english/point.

[5] "reckless". In *Oxford Dictionaries*. Oxford University Press. Accessed March 27, 2013. http://www.oxforddictionaries.com/definition/english/reckless.

[6] "retribution, *n*". In *Collins English Thesaurus*. HarperCollins Publishers. Accessed November 29, 2013. http://www.collinsdictionary.com/dictionary/english-thesaurus/retribution.

[7] "theme, *n*". In *Oxford Dictionaries*. Oxford University Press. Accessed November 29, 2013. http://www.oxforddictionaries.com/definition/english/theme.

[8] Carole Adam, Andreas Herzig, and Dominique Longin. A logical formalization of the OCC theory of emotions. *Synthese*, 168(2):201–248, 2009.

[9] Aesop. *Aesop: The Complete Fables*. Penguin Classics. Penguin Books, London, UK, 1998. Translated by Olivia and Robert Temple; introduction by Robert Temple.

[10] Hans Christian Andersen. *The Complete Fairy Tales*. Wordsworth Editions Ltd, Hertfordshire, 2009.

[11] Douglas E. Appelt and Kurt Konolige. A Non-Monotonic Logic for Reasoning about Speech Acts and Belief Revision. In *Non-Monotonic Reasoning*, volume 346 of *Lecture Notes in Computer Science*, pages 164–175. Springer-Verlag, 1989.

[12] Aristotle. *Poetics*. `http://classics.mit.edu/Aristotle/poetics.html`, 350 BC. Translated by S.H. Butcher. Accessed 22nd July 2010.

[13] Wendy S. Ark, D. Christopher Dryer, and Davia J. Lu. The Emotion Mouse. In *HCI International 1999: the 8th International Conference on Human-Computer Interaction*, volume 1, pages 818–823, Munich, Germany, 1999. Lawrence Eribaum Associates, Inc.

[14] Magda B. Arnold. *Emotion and Personality*, volume 1. Columbia University Press, New York, 1960.

[15] Augusta Baker and Ellin Greene. *Storytelling: Art and Technique*. R. R. Bowker Company, New York, 1977.

[16] Mieke Bal. *Narratology: Introduction to the Theory of Narrative*. University of Toronto Press, Toronto, 2nd edition, 1997.

[17] K. Suzanne Barber and Jisun Park. Agent Belief Autonomy in Open Multi-agent Systems. In *Agents and Computational Autonomy*, volume 2969 of *Lecture Notes in Computer Science*, pages 7–16. Springer-Verlag, 2004.

[18] Roland Barthes. *S/Z*. Blackwell, Oxford, UK, 1990. Translated by Richard Miller; preface by Richard Howard.

[19] Roland Barthes and Lionel Duisit. An Introduction to the Structural Analysis of Narrative. *New Literary History*, 6(2):237–272, 1975.

[20] Joseph Bates. The Role of Emotion in Believable Agents. *Communications of the ACM*, 37(7):122–125, 1994.

[21] Cristina Battaglino and Rossana Damiano. Emotional Appraisal of Moral Dilemma in Characters. In David Oyarzun, Federico Peinado, R. Michael Young, Ane Elizalde, and Gonzalo Mndez, editors, *Interactive Storytelling*, volume 7648

of *Lecture Notes in Computer Science*, pages 150–161. Springer Berlin Heidelberg, 2012.

[22] Cristina Battaglino, Rossana Damiano, and Leonardo Lesmo. Moral Appraisal and Emotions. In *Workshop on Emotional and Empathetic Agents, 11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2012)*, Valencia, Spain, 2012.

[23] Bruno Bettelheim. *The Uses of Enchantment: The Meaning and Importance of Fairy Tales*. Alfred A. Knopf, New York, 1976.

[24] Timothy Bickmore, Laila Bukhari, Laura Pfeifer Vardoulakis, Michael Paasche-Orlow, and Christopher Shanahan. Hospital Buddy: A Persistent Emotional Support Companion Agent for Hospital Patients. In *Proceedings of the 12th International Conference on Intelligent Virtual Agents (IVA 2012)*, pages 492–495, Santa Cruz, California, 2012. Springer Berlin Heidelberg.

[25] Christopher Booker. *The Seven Basic Plots*. Bloomsbury Academic, London, 2006.

[26] Rafael H. Bordini, Lars Braubach, Mehdi Dastani, Amal El Fallah Seghrouchni, Jorge J Gomez-Sanz, João Leite, Gregory O'Hare, Alexander Pokahr, and Alessandro Ricci. A Survey of Programming Languages and Platforms for Multi-Agent Systems. *Informatica*, 30(1):33–44, 2006.

[27] Claude Bremond. *Logique du récit*. Collection Poétique. Éditions du Seuil, Paris, 1973.

[28] William F. Brewer. To Assert That Essentially All Human Knowledge and Memory is Represented in Terms of Stories Is Certainly Wrong. In Robert S. Jr. Wyer, editor, *Knowledge and Memory: The Real Story*, volume 8 of *Advances in Social Cognition*. Psychology Press, New York, 2014.

[29] Donald E. Brown. *Human Universals*. McGraw-Hill, New York, 1991.

[30] Mark Burkitt and Daniela M. Romano. The Mood and Memory of Believable Adaptable Socially Intelligent Characters. In *Proceedings of the 8th International Conference on Intelligent Virtual Agents*, pages 372–379, Tokyo, Japan, 2008. Springer-Verlag.

[31] Charles B. Callaway and James C. Lester. Narrative Prose Generation. *Artificial Intelligence*, 139(2):213–252, 2002.

[32] Walter B. Cannon. The James-Lange Theory of Emotions: A Critical Examination and an Alternative Theory. *The American Journal of Psychology*, 39(1/4):106–124, 1927.

[33] John Cantwell. A Formal Model of Multi-Agent Belief Interaction. *Journal of Logic, Language and Information*, 15(4):303–329, 2006.

[34] M. Cavazza, F. Charles, and S.J. Mead. Characters in Search of an Author: AI-Based Virtual Storytelling. In *Virtual Storytelling*, volume 2197 of *Lecture Notes in Computer Science*, pages 145–154. Springer-Verlag, 2001.

[35] M. Cavazza and D. Pizzi. Narratology for Interactive Storytelling: A Critical Introduction. In *Technologies for Interactive Digital Storytelling and Entertainment*, volume 4326 of *Lecture Notes in Computer Science*, pages 72–83. Springer, 2006.

[36] Émile Chambry. *Ésope Fables, Texte Établi et Traduit par Émile Chambry*. Collection des Universités de France, Paris, 1927.

[37] David Joseph Chaplin and Abdennour El Rhalibi. IPD for Emotional NPC Societies in Games. In *2004 ACM SIGCHI International Conference on Advances in Computer Entertainment Technology (ACE 2004)*, pages 51–60, Singapore, 2004. ACM Press.

[38] Fred Charles, Miguel Lozano, Steven J. Mead, Alicia Fornes Bisquerra, and Marc Cavazza. Planning Formalisms and Authoring in Interactive Storytelling. In *The 1st International Conference on Technologies for Interactive Digital Storytelling and Entertainment (TIDSE)*, Darmstadt, Germany, 2003.

[39] Seymour Chatman. *Story and Discourse: Narrative Structure in Fiction and Film*. Cornell University Press, Ithaca, 1978.

[40] Yun-Gyung Cheong and R. Young. Narrative Generation for Suspense: Modeling and Evaluation. *Interactive Storytelling*, pages 144–155, 2008.

[41] Philip R. Cohen and Hector J. Levesque. Intention Is Choice with Commitment. *Artificial Intelligence*, 42(2-3):213–261, 1990.

[42] Monte Cook, Jonathan Tweet, and Skip Williams. *Dungeon's and Dragons Player's Handbook: Core Rulebook I, v.3.5*. Wizards of the Coast, 2003.

[43] Rossana Damiano and Antonio Pizzo. Emotions in Drama Characters and Virtual Agents. In I. Horswill, E. Hudlicka, C. Lisetti, and J.D. Velasquez, editors, *AAAI Spring Symposium on Emotion, Personality, and Social Behaviour*, Spring Symposium Series. AAAI Press, 2008.

[44] João Dias and Ana Paiva. Feeling and Reasoning: A Computational Model for Emotional Characters. In *Progress in Artificial Intelligence*, volume 3808 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2005.

[45] Minh Binh Do and Subbarao Kambhampati. Planning as Constraint Satisfaction: Solving the planning-graph by compiling it into CSP. *Artificial Intelligence*, 132(2):151–182, 2001.

[46] Charles Dolan. *Memory Based Processing for Cross Contextual Reasoning: Reminding and Analogy Using Thematic Structures*. PhD thesis, University of California, Los Angeles, 1984.

[47] Marcy H. Dorfman and William F. Brewer. Understanding the points of fables. *Discourse Processes*, 17(1):105–129, 1994.

[48] Michael George Dyer. *In-Depth Understanding: A Computer Model of Integrated Processing for Narrative Comprehension*. The MIT Press, Cambridge, Massachusetts, 1983.

[49] Umberto Eco and Jean-Claude Carriere. *This is Not the End of the Book*. Vintage Books, London, 2012. Translated by Polly McLean.

[50] Paul Ekman. Basic Emotions. In T. Dalgleish and M. Power, editors, *Handbook of Cognition and Emotion*. John Wiley & Sons, Sussex, UK, 1999.

[51] Abdennour El Rhalibi, Nick Baker, and Madjid Merabti. Emotional Agent Model and Architecture for NPCs Group Control and Interaction to Facilitate Leadership Roles in Computer Entertainment. In Newton Lee, editor, *2005 ACM SIGCHI International Conference on Advances in Computer Entertainment Technology (ACE 2005)*, pages 156–163, Valencia, Spain, 2005. ACM Press.

[52] Clark Elliott. *The Affective Reasoner: A process model of emotions in a multi-agent system*. PhD thesis, Northwestern University, 1992.

[53] Paolo Ferraris and Vladimir Lifschitz. Mathematical Foundations of Answer Set Programming. In *We Will Show Them! Essays in Honour of Dov Gabbay*, pages 615–664. King's College Publications, 2005.

[54] Andy Field. *Discovering Statistics Using SPSS*. SAGE Publications Ltd, London, 3rd edition, 2009.

[55] Monika Fludernik. *An Introduction to Narratology*. Routledge, Abingdon, UK, 2009.

[56] Henry Thomas Francis and Edward Joseph Thomas. *Jātaka Tales*. The University Press, Cambridge, 1916.

[57] Adam Frank, Andrew Stern, and Ben Resner. Socially Intelligent Virtual Petz. In *AAAI Fall Symposium on Socially Intelligent Agents*, pages 43–45, Cambridge, Massachusetts, 1997.

[58] Nico H. Frijda. *The Emotions*. Cambridge University Press, New York, 1986.

[59] Peter Gärdenfors. *Belief Revision*, volume 29 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, Cambridge, 2003.

[60] Martin Gebser, Roland Kaminski, Benjamin Kaufmann, Max Ostrowski, Torsten Schaub, and Marius Schneider. Potassco: The Potsdam Answer Set Solving Collection. *AI Communications*, 24(2):105–124, 2011.

[61] Martin Gebser, Roland Kaminski, Benjamin Kaufmann, Max Ostrowski, Torsten Schaub, and Sven Thiele. A User's Guide to gringo, clasp, clingo, and iclingo. Technical report, University of Potsdam, November 2008.

[62] Michael Gelfond and Vladimir Lifschitz. The Stable Model Semantics for Logic Programming. In *Fifth International Conference and Symposium of Logic Programming (ICLP 88)*, pages 1070–1080, 1988.

[63] Alfonso Gerevini and Derek Long. Preferences and Soft Constraints in PDDL3. In *Proceedings of the ICAPS-2006 Workshop on Preferences and Soft Constraints in Planning*, pages 46–54, 2006.

[64] Enrico Giunchiglia, Yuliya Lierier, and Marco Maratea. Answer Set Programming based on Propositional Satisfiability. *Journal of Automated Reasoning*, 36(4):345–377, 2006.

[65] Jonathan Gottschall. *The Storytelling Animal: How Stories Make Us Human*. Houghton Mifflin Harcourt, New York, 2012.

[66] Benjamin Goudey. A comparison of Situation Calculus and Event Calculus. Technical report, University of Melbourne, 2007.

[67] Jonathan Gratch. Émile: Marshalling Passions in Training and Education. In *4th International Conference on Autonomous Agents*, pages 325–332, Barcelona, Spain, 2000.

[68] A.J. Greimas. *Structural Semantics: An Attempt at a Method*. University of Nebraska Press, Lincoln, Nebraska, 1983. Translated by Daniele McDowell, Ronald Schleifer, and Alan Velie.

[69] Charlie Hargood, David E. Millard, and Mark J. Weal. Investigating a thematic approach to narrative generation. In *Proceedings of the International Workshop on Dynamic and Adaptive Hypertext*, Torino, Italy, 2009.

[70] Joel Chandler Harris. *The Complete Tales of Uncle Remus*. Houghton Mifflin, Boston, 1955.

[71] Knut Hartmann, Sandra Hartmann, and Matthias Feustel. Motif Definition and Classification to Structure Non-linear Plots and to Control the Narrative Flow in Interactive Dramas. In *Virtual Storytelling*, volume 3805 of *Lecture Notes in Computer Science*, pages 158–167. Springer-Verlag, 2005.

[72] Louis Hébert. *Tools for Text and Image Analysis: An Introduction to Applied Semiotics*. Texto!, 2006. Translated by Julie Tabler. Available at: `http://www.revue-texto.net/Parutions/Livres-E/Hebert_AS/Hebert_Tools.html`.

[73] Eva Hudlicka. To feel or not to feel: The role of affect in human-computer interaction. *International Journal of Human-Computer Studies*, 59:1–32, 2003.

[74] Marco Iacoboni. *Mirroring People*. Farrar, Straus and Giroux, New York, 2008.

[75] Judy R. Jablon and Michael Wilkinson. Using Engagement Strategies to Facilitate Children's Learning and Success. *YC Young Children*, 61(2):12–16, 2006.

[76] William James. What is an Emotion? *Mind*, 9(34):188–205, 1884.

[77] Kevser Koc and Cary A. Buzzelli. The Moral of the Story Is ... Using Children's Literature in Moral Education. *YC Young Children*, 59(1):92–97, 2004.

[78] Lawrence Kohlberg. *The Development of Modes of Thinking and Choices in years 10 to 16*. PhD thesis, University of Chicago, 1958.

[79] Ignacy Krasicki. *Bajki i Przypowieści*. Biblioteka Arcydzieł. Ksiegarnia Polska, Warszawa, 1950.

[80] Simone Kriglstein and Günter Wallner. HOMIE: An Artificial Companion for Elderly People. In *ACM Conference on Human Factors in Computing Systems (CHI 2005)*, pages 2094–2098, Portland, Oregon, 2005.

[81] Ivan Andreevich Krylov. *Krylov's Fables*. J. Cape, 1936. Translated by B. Pares.

[82] Vickie E. Lake. Linking Literacy and Moral Education in the Primary Classroom. *The Reading Teacher*, 55(2):125–129, 2001.

[83] Carl Georg Lange. The Emotions. In Knight Dunlap, editor, *The Emotions: Volume I*. Williams & Wilkins Company, Baltimore, 1922.

[84] Richard S. Lazarus. *Emotion and Adaptation*. Oxford University Press, New York, 1991.

[85] Michael Lebowitz. Creating Characters in a Story-Telling Universe. *Poetics*, 13(3):171–194, 1984.

[86] Michael Lebowitz. Story-telling as Planning and Learning. *Poetics*, 14:483–502, 1985.

[87] Wendy G Lehnert. Plot Units and Narrative Summarization. *Cognitive Science*, 5(4), 1981.

[88] Nicola Leone, Gerald Pfeifer, Wolfgang Faber, Thomas Eiter, Georg Gottlob, Simona Perri, and Francesco Scarcello. The DLV System for Knowledge Representation and Reasoning. *ACM Transactions on Computational Logic*, 7(3):499–562, 2006.

[89] Hector J. Levesque, Raymond Reiter, Yves Lespérance, Fangzhen Lin, and Richard B. Scherl. GOLOG: A logic programming language for dynamic domains. *The Journal of Logic Programming*, 31(1-3):59–83, 1997.

[90] Jayne Elizabeth Lewis. *The English Fable: Aesop and Literary Culture, 1651–1740*. Cambridge University Press, Cambridge, 1996.

[91] Vladimir Lifschitz. Answer set programming and plan generation. *Artificial Intelligence*, 138(1-2):39–54, 2002.

[92] Fangzhen Lin and Yuting Zhao. ASSAT: Computing Answer Sets of A Logic Program By SAT Solvers. *Artificial Intelligence*, 157(1-2):115–137, 2004.

[93] Christine Lisetti and Fatma Nasoz. Using Noninvasive Wearable Computers to Recognize Human Emotions from Physiological Signals. *EURASIP Journal on Applied Signal Processing*, 11:1672–1687, 2004.

[94] James N. Loehlin. *The Cambridge Introduction to Chekhov*. Cambridge University Press, Cambridge, 2010.

[95] Jean Mandler and Nancy Johnson. Remembrance of Things Parsed: Story Structure and Recall. *Cognitive Psychology*, 9(1):111–151, 1977.

[96] Raymond A. Mar and Keith Oatley. The Function of Fiction is the Abstraction and Simulation of Social Experience. *Perspectives on Psychological Science*, 3(3):173–192, 2008.

[97] Raymond A. Mar, Keith Oatley, Jacob Hirsh, Jennifer dela Paz, and Jordan B. Peterson. Bookworms versus nerds: Exposure to fiction verses non-fiction, divergent associations with social ability, and the simulation of fictional story worlds. *Journal of Research in Personality*, 40(5):694–712, 2006.

[98] William March. *99 Fables*, volume 94 of *Library of Alabama Classics*. University of Alabama Press, 2011. Edited by W.T. Going. Illustrated by R. Brough.

[99] Stacy Marsella and Jonathan Gratch. EMA: A process model of appraisal dynamics. *Cognitive Systems Research*, 10:70–90, 2009.

[100] Diane T. Marsh, Felicisima C. Serafica, and Carl Barenboim. Interrelationships among Perspective Taking, Interpersonal Problem Solving, and Interpersonal

Functioning. *The Journal of Genetic Psychology: Research and Theory on Human Development*, 138(1):37–48, 1981.

[101] Elizabeth J. Marsh and Lisa K. Fazio. Learning errors from fiction: Difficulties in reducing reliance on fictional stories. *Memory and Cognition*, 34(5):1140–1149, 2006.

[102] Elizabeth J. Marsh, Michelle L. Meade, and Henry L. Roediger III. Learning facts from fiction. *Journal of Memory and Language*, 49(4):519–536, 2003.

[103] Michael Mateas and Andrew Stern. A Behavior Language for Story-Based Believable Agents. *IEEE Intelligent Systems*, 17(4):39–47, 2002.

[104] Michael Mateas and Andrew Stern. Structuring Content in the Façade Interactive Drama Architecture. In *Proceedings of the First Artificial Intelligence and Interactive Digital Entertainment Conference*, pages 93–98, 2005.

[105] Brian McDonald. *Invisible Ink*. Libertary Editions, Seattle, 2010.

[106] Marshall McLuhan. *Understanding Media: The Extension of Man*. The MIT Press, Cambridge, Massachusetts, 1994. Introduction by Lewis H. Lapham.

[107] James R. Meehan. TALE-SPIN, An Interactive Program that Writes Stories. In *Proceedings of the 5th international joint conference on Artificial intelligence*, volume 1, pages 91–98, Cambridge, USA, 1977. Morgan Kaufmann Publishers Inc.

[108] Philipp Michel and Rana El Kaliouby. Real Time Facial Expression Recognition in Video using Support Vector Machines. In *5th International Conference on Multimodal Interfaces (ICMI 2003)*, pages 258–264, Vancouver, Canada, 2003. ACM Press.

[109] B. W. Mott, C. B. Callaway, L. S. Zettlemoyer, S. Y. Lee, and J. C. Lester. Towards Narrative-Centered Learning Environments. In *Proceedings of the 1999 AAAI Fall Symposium on Narrative Intelligence*, pages 78–82, 1999.

[110] Erik T. Mueller. *Commonsense Reasoning*. Morgan Kaufmann Publishers, San Francisco, CA, 2006.

[111] Erik T Mueller. Discrete Event Calculus Reasoner Documentation. Technical report, IBM Thomas J. Watson Research Center, 2008.

[112] Stephen Muggleton. *Inductive Logic Programming*, volume 38 of *The APIC Series*. Academic Press Limited, London, 1992.

[113] George Peter Murdock. The Common Denominator of Cultures. In Ralph Linton, editor, *The Science of Man in the World Crisis*. Columbia University Press, New York, 1945.

[114] Hai H. Nguyen. Belief Revision in a Fact-Rule Agent's Belief Base. In *Agent and Multi-Agent Systems: Technologies and Applications*, volume 5559 of *Lecture Notes in Computer Science*, pages 120–130. Springer-Verlag, 2009.

[115] Paolo Nichelli, Jordan Grafman, Pietro Pietrini, Kimberley Clark, Kyu Young Lee, and Robert Miletich. Where the brain appreciates the moral of a story. *NeuroReport*, 6(17):2309–2313, 1995.

[116] Peter Norvig. Frame Activated Inferences in a Story Understanding Program. *8th International Joint Conference on Artificial Intelligence*, 2:624–626, 1983.

[117] OECD. Special Focus: Measuring Leisure in OECD Countries. In *Society at a Glance 2009: OECD Social Indicators*. OECD Publishing, 2009.

[118] Andrew Ortony, Gerald L. Clore, and Allan Collins. *The Cognitive Structure of Emotions*. Cambridge University Press, Cambridge, 1988.

[119] Anne Pellowski. *The World of Storytelling*. R. R. Bowker Company, New York, 1977.

[120] James W. Pennebaker and Janel D. Seagal. Forming a Story: The Health Benefits of Narrative. *Journal of Clinical Psychology*, 55(10):1243–1254, 1999.

[121] Rafael Pérez y Pérez. Employing emotions to drive plot generation in a computer-based storyteller. *Cognitive Systems Research*, 8(2):89–109, 2007.

[122] Rafael Pérez y Pérez and Mike Sharples. MEXICA: A computer model of a cognitive account of creative writing. *Journal of Experimental and Theoretical Artificial Intelligence*, 13(2):119–139, 2001.

[123] Robert Plutchik. *Emotion: A Psychoevolutionary Synthesis*. Harper & Row, New York, 1980.

[124] Mannes Poel, Rieks op den Akker, Dirk Heylen, and Anton Nijholt. Emotion based Agent Architectures for Tutoring Systems: The INES Architecture. In R Tappi, editor, *Cybernetics and Systems 2004: Workshop on Affective Computational Entities (ACE 2004)*, pages 663–667, Vienna, Austria, 2004.

[125] J. Porteous and M. Cavazza. Controlling Narrative Generation with Planning Trajectories: The Role of Constraints. In *Interactive Storytelling*, pages 234–245, 2009.

[126] Vladimir Propp. *Morphology of the Folktale*. University of Texas Press, Austin, Texas, 2nd edition, 1968.

[127] Ayn Rand. *The Art of Fiction: A Guide for Writers and Readers*. Plume, New York, 2000. Edited by Tore Boeckmann; introduction by Leonard Peikoff.

[128] Anand S. Rao. AgentSpeak(L): BDI agents speak out in a logical computable language. In *7th European Workshop on Modelling Autonomous Agents in a Multi-Agent World*, Eindhoven, The Netherlands, 1996.

[129] W. Scott Neal Reilly. *Believable Social and Emotional Agents*. PhD thesis, Carnegie Mellon University, 1996.

[130] Raymond Reiter. *Knowledge In Action: Logical Foundations for Specifying and Implementing Dynamical Systems*. Massachusetts Institute of Technology, Cambridge, Massachusetts, 2001.

[131] M. Riedl and R.M. Young. Character-Focused Narrative Generation for Execution in Virtual Worlds. In *Proceedings of the 2nd International Conference on Virtual Storytelling*, Toulouse, France, 2003.

[132] Mark Owen Riedl. *Narrative Generation: Balancing Plot and Character*. PhD thesis, North Carolina State University, 2004.

[133] Paola Rizzo. Why Should Agents Be Emotional for Entertaining Users? A Critical Analysis. In *Affective Interactions: Towards a New Generation of Computer Interfaces*, volume 1814 of *Lecture Notes in Computer Science*, pages 166–181. Springer Berlin Heidelberg, 2000.

[134] Giacomo Rizzolatti and Laila Craighero. The Mirror-Neuron System. *Annual Review of Neuroscience*, 27:169–192, 2004.

[135] Sam Roberts. An Introduction to Progol. Technical report, Oxford University, 1997.

[136] Yvonne Rogers, Helen Sharp, and Jenny Preece. *Interaction Design: Beyond Human-Computer Interaction*. John Wiley & Sons, West Sussex, United Kingdom, 3rd edition, 2011.

[137] Ira J. Roseman. Cognitive Determinants of Emotions: A Structural Theory. In Phil Shaver, editor, *Review of Personality and Social Psychology*, volume 5, pages 11–36. Sage, Beverly Hills, 1984.

[138] Christian Roth, Peter Vorderer, and Christoph Klimmt. The Motivational Appeal of Interactive Storytelling: Towards a Dimensional Model of the User Experience. In *Interactive Storytelling*, pages 38–43, 2009.

[139] Stuart Russel and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, New Jersey, 2nd edition, 2003.

[140] James A. Russell. A Circumplex Model of Affect. *Journal of Personality and Social Psychology*, 39(6):1161–1178, 1980.

[141] Kevin Ryan. The Narrative and the Moral. *The Clearing House*, 64(5):316–319, 1991.

[142] Malcolm Ryan, Nicholas Hannah, and Joshua Lobb. The tale of Peter Rabbit: a case-study in story-sense reasoning. In *Proceedings of the 4th Australasian Conference on Interactive Entertainment*, Melbourne, Australia, 2007. RMIT University.

[143] Vítor Santos Costa, Luís Damas, Rogério Reis, and Rúben Azevedo. YAP User's Manual (Version 6.2.0). Technical report, Universidade do Porto, 2000. Available at: `http://www.dcc.fc.up.pt/~vsc/Yap/documentation.html`.

[144] Sebastian Sardina and Lin Padgham. A BDI Agent Programming Language with Failure Handling, Declarative Goals, and Planning. *Autonomous Agents and Multi-Agent Systems*, 23(1):18–70, 2011.

[145] Margaret Sarlej and Malcolm Ryan. A Discrete Event Calculus Implementation of the OCC Theory of Emotion. In *The 4th Workshop on Intelligent Narrative Technologies*, pages 57–64, Stanford University, Palo Alto, California, 2011. AAAI Press.

[146] Margaret Sarlej and Malcolm Ryan. Representing Morals in Terms of Emotion. In *The 8th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE2012)*, pages 69–74, Stanford University, Palo Alto, California, 2012. AAAI Press.

[147] Margaret Sarlej and Malcolm Ryan. Generating Stories with Morals. In Hartmut Koenitz, Tonguc Ibrahim Sezen, Gabriele Ferri, Mads Haahr, Digdem Sezen, and Güven Çatak, editors, *The 6th International Conference on Interactive Digital Storytelling*, volume 8230 of *Lecture Notes in Computer Science*, pages 217–222, Istanbul, Turkey, 2013. Springer.

[148] Michelle Scalise Sugiyama. Food, foragers, and folklore: the role of narrative in human subsistence. *Evolution and Human Behavior*, 22(4):221–240, 2001.

[149] Roger C. Schank. *Dynamic memory: A theory of reminding and learning in computers and people*. Cambridge University Press, Cambridge, 1982.

[150] Roger C. Schank. *Tell Me a Story: Narrative and Intelligence*. Northwestern University Press, 1995.

[151] Roger C. Schank and Robert P. Abelson. Knowledge and Memory: The Real Story. In Robert S. Jr. Wyer, editor, *Knowledge and Memory: The Real Story*, volume 8 of *Advances in Social Cognition*. Psychology Press, New York, 2014.

[152] Roger C. Schank, Gregg C. Collins, Ernest Davis, Peter N. Johnson, Steve Lytinen, and Brian J. Reiser. What's the Point? *Cognitive Science*, 6:255–275, 1982.

[153] Klaus R. Scherer. Emotion as a process: Function, origin and regulation. *Social Science Information*, 21:555–570, 1982.

[154] Matthias Scheutz. Useful Roles of Emotions in Artificial Agents: A Case Study from Artificial Life. In *The 19th National Conference on Artificial Intelligence (AAAI 2004)*, pages 42–47, San Jose, California, 2004. The AAAI Press.

[155] Harold Schlosberg. Three Dimensions of Emotion. *The Psychological Review*, 61(2):81–88, 1954.

[156] Wolf Schmid. *Narratology: An Introduction*. De Gruyter, Berlin, 2010.

[157] Jaspreet Shaheed and Jim Cunningham. Agents making moral decisions. In *Proceedings of the ECAI08 Workshop on Artificial Intelligence in Games*, 2008.

[158] George Shannon. Storytelling and the Schools. *The English Journal*, 68(5):50–51, 1979.

[159] Vishnu Sharma. *Panchatantra*. Jaico Publishing House, Mumbai, 1949. Translated by A.W. Ryder.

[160] Khalil Shihab and Nida Chalabi. Emotional Agents in Computer Games. *International Journal of Computers and Communications*, 2(4):102–107, 2008.

[161] Patrik Simons, Ilkka Niemelä, and Timo Soininen. Extending and Implementing the Stable Model Semantics. *Artificial Intelligence*, 138(1-2):181–234, 2002.

[162] Ashwin Srinivasan. P-Progol User Manual. Technical report, Oxford University, 1998. Available at: `http://www.cs.ox.ac.uk/activities/machlearn/PProgol/ppman.html`.

[163] Ashwin Srinivasan. The Aleph Manual. Technical report, Computing Laboratory, Oxford University, 2003. Available at: `http://www.cs.ox.ac.uk/activities/machlearn/Aleph/aleph.html`.

[164] Nancy L. Stein and Margaret Policastro. Chapter 4: The Concept of a Story: A Comparison Between Children's and Teachers' Viewpoints. In Heinz Mandl, Nancy L. Stein, and Tom Trabasso, editors, *Learning and Comprehension of Text*, pages 113–155. Lawrence Erlbaum Associates, Hillsdale, New Jersey, 1984.

[165] Mariët Theune, Sander Faas, Anton Nijholt, and Dirk Heylen. Emotional Characters for Automatic Plot Creation. In *Technologies for Interactive Digital Storytelling and Entertainment*, 2003.

[166] Perry Thorndyke. Cognitive Structures in Comprehension and Memory of Narrative Discourse. *Cognitive Psychology*, 9(1):77–110, 1977.

[167] James Thurber. *Fables for Our Time and Famous Poems Illustrated*. Harper-Collins, New York, 1983.

[168] Boris Tomashevsky. Thematics. In *Russian Formalist Criticism: Four Essays*. University of Nebraska Press, Lincoln, Nebraska, 1965. Translated by Lee T. Lemon and Marion J. Reis; introduction by Lee T. Lemon and Marion J. Reis.

[169] Alan M. Turing. Computing Machinery and Intelligence. *Mind*, 59(239):433–460, 1950.

[170] Scott R. Turner. *MINSTREL: A Computer Model of Creativity and Storytelling*. PhD thesis, University of California, 1993.

[171] Johan van Benthem. Dynamic logic for belief revision. *Journal of Applied Non-Classical Logics*, 17(2):129–155, 2007.

[172] Paul C. Vitz. The Use of Stories in Moral Development. *American Psychologist*, 45(6):709–720, 1990.

[173] Noah Wardrip-Fruin. *Expressive Processing: Digital Fictions, Computer Games, and Software Studies*. The MIT Press, Cambridge, Massachusetts, 2009.

[174] David Watson and Auke Tellegan. Toward a Consensual Structure of Mood. *Psychological Bulletin*, 98(2):219–235, 1985.

[175] B. L. Welch. The Significance of the Difference Between Two Means when the Population Variances are Unequal. *Biometrika*, 29(3/4):350–362, 1938.

[176] Daniel Weld. An Introduction to Least Commitment Planning. *AI Magazine*, 15(4):27–61, 1994.

[177] Robert Wilensky. Story grammars versus story points. *The Behavioral and Brain Sciences*, 6:579–623, 1983.

[178] Edwin B. Wilson. Probable Inference, the Law of Succession, and Statistical Inference. *Journal of the American Statistical Association*, 22(158):209–212, 1927.

[179] Sun Yu, Li Zhiping, and Xia Youming. A Model of Intelligent Tutoring Systems with Emotional Pedagogical Agents. In *The 7th International Conference on Computer Science and Education (ICCSE 2012)*, pages 1328–1332, Melbourne, Australia, 2012.

[180] Jichen Zhu. Designing an Interdisciplinary User Evaluation for the Riu Computational Narrative System. In *Proceedings of the 5th International Conference on Interactive Digital Storytelling (ICIDS2012)*, pages 126–131, San Sebastián, Spain, 2012.

Appendix A

OCC Emotion Survey Sample

In this appendix, we include illustrative excerpts from the survey conducted to evaluate our ASP model of the OCC theory, as described in Chapter 4 (refer to Section 4.6). Section A.1 shows the instructions provided to participants, including emotion definitions. In Section A.2, we provide the survey page corresponding to one of Aesop's fables. The remainder of the survey followed the same format.

A.1 Introduction and Task Description

You (the research
participant) are invited to participate in a study about character emotions in fables (short stories). We (the investigators) hope to learn what emotions readers expect particular characters to experience in response to specific events in the selected fables.

Description of Study

If you decide to participate, we will ask you to complete an online survey about character emotions in Aesop's fables [1]. Completing the survey should take approximately 30 minutes. You will be provided with outlines of 6 fables and asked to select (using checkboxes) which emotions you believe each character should be feeling at each step. You may select "None of the above" if you believe a character was not emotionally affected by the previous event, or the emotion they experienced is not available in the list.

If you decide not to complete the entire survey, it is important that you skip through to the last page (leaving any remaining questions unanswered) and click 'Submit'. If you do not do this, your results will not be recorded.

The emotions used in this survey are based on the OCC Theory of Emotion [2], and are defined briefly below:

Emotions directed towards an action:

- **Pride:** A character approves of one of their own actions.
- **Shame:** A character disapproves of one of their own actions.
- **Admiration:** A character approves of another character's action.
- **Reproach:** A character disapproves of another character's action.

Emotions directed towards the outcomes of events:

- **Joy:** A character is pleased about the outcome of an event.
- **Distress:** A character is displeased about the outcome of an event.

Emotions directed towards an action and one of its outcomes:

- **Gratification:** A character approves of one of their own actions and is pleased about the outcome.
- **Remorse:** A character disapproves of one of their own actions and is displeased about the outcome.
- **Gratitude:** A character approves of another character's action and is pleased about the outcome.
- **Anger:** A character disapproves of another character's action and is displeased about the outcome.

Emotions directed towards another character with regards to an outcome:

- **Happy-For:** A character is pleased about an outcome that is desirable for another character.
- **Gloating:** A character is pleased about an outcome that is undesirable for another character.

Confidentiality and Disclosure of Information

The survey is completely anonymous, and you will not be required to provide any personal or identifying information. By completing the survey you indicate your consent for us to use the acquired data for this and related studies. Aggregate data may also be published in academic journals, conference proceedings and online.

Complaints may be directed to:

Ethics Secretariat
The University of New South Wales
Sydney 2052 AUSTRALIA
Phone: +61 2 9385 4234
Fax: +61 2 9385 6648
Email: ethics.sec@unsw.edu.au

Any complaint you make will be investigated and you will be informed of the outcome.

Feedback to Participants

If you are interested in receiving feedback about the results of this survey please feel free to contact Margaret Sarlej on msarlej@cse.unsw.edu.au.

References

[1] Aesop. 'Aesop: The Complete Fables.' Penguin Books, London, UK, 1998.
[2] Ortony, A; Clore, G.L and Collins, A. 'The Cognitive Structure of Emotions.' Cambridge University Press, Cambridge, 1988.

Continue »

Powered by Google Docs

A.2 Survey Excerpt

1. Aesop's Fable No. 3 - The Fox and the Eagle

Characters:
Fox, Eagle

An eagle and a fox, having become friends, decided to live near one another and be neighbours. They believed that this proximity would strengthen their friendship. So the eagle flew up and established herself on a very high branch of a tree, where she made her nest. And the fox, creeping about among the bushes which were at the foot of the same tree, made her den there, depositing her babies right beneath the eagle.

But, one day when the fox was out looking for food, the eagle, who was very short of food too, swooped down to the bushes and took the fox cubs up to her nest and feasted on them with her own young.

When the fox returned, she was less distressed at the death of her little ones than she was driven mad by frustration at the impossibility of ever effectively avenging herself. For she, a land animal, could never hope to pursue a winged bird. She had no option but to content herself, in her powerlessness and feebleness, with cursing her enemy from afar.

Now it was not long afterwards that the eagle did actually receive her punishment for her crimes against her friend.

Some men were sacrificing a goat in the countryside and the eagle swooped down on the altar, carrying off some burning entrails, which she took up to her nest. A strong wind arose which blew the fire from the burning entrails into some straw that was in the nest. The eaglets were singed and, as they were not yet able to fly, when they leaped from the nest they fell to the ground. The fox rushed up and devoured them all in front of the eagle's eyes.

Event 1: The Fox and the Eagle are friends. While the Fox is away hunting, the Eagle eats the Fox's cubs.

The action is carried out by the Eagle.

There are two consequences of this event:

a) The Eagle and her eaglets are no longer hungry; and
b) The Fox loses her cubs.

1.1.1. Which of the following (if any) would you expect the EAGLE to feel about her own action of eating the Fox's cubs?

- [ ] Approving of her action (Pride)
- [ ] Disapproving of her action (Shame)
- [ ] None of the above

1.1.2. Which of the following (if any) would you expect the FOX to feel about the Eagle's action of eating the Fox's cubs?

- [ ] Approving of the Eagle's action (Admiration)
- [ ] Disapproving of the Eagle's action (Reproach)
- [ ] None of the above

1.1.3. Which of the following (if any) would you expect the EAGLE to feel about no longer being hungry?

- [ ] Pleased about this outcome (Joy)
- [ ] Displeased about this outcome (Distress)
- [ ] None of the above

1.1.4. Which of the following (if any) would you expect the FOX to feel about the Eagle no longer being hungry?

- [ ] Pleased about this outcome (Joy)
- [ ] Displeased about this outcome (Distress)
- [ ] None of the above

1.1.5. Which of the following (if any) would you expect the EAGLE to feel about her own action with regards to no longer being hungry?

- [ ] Approving of her action and pleased with the outcome (Gratification)
- [ ] Disapproving of her action and displeased with the outcome (Remorse)
- [ ] None of the above

1.1.6. Which of the following (if any) would you expect the FOX to feel about the Eagle's action with regards to the Eagle no longer being hungry?

- [ ] Approving of the Eagle's action and pleased with the outcome (Gratitude)
- [ ] Disapproving of the Eagle's action and displeased with the outcome (Anger)
- [ ] None of the above

1.1.7. Which of the following (if any) would you expect the EAGLE to feel towards the FOX with regards to the Eagle no longer being hungry?

- [ ] Pleased that the outcome is desirable for the Fox (Happy-For)
- [ ] Pleased that the outcome is undesirable for the Fox (Gloating)
- [ ] Displeased that the outcome is desirable for the Fox (Resentment)
- [ ] Displeased that the outcome is undesirable for the Fox (Pity)
- [ ] None of the above

1.1.8. Which of the following (if any) would you expect the FOX to feel towards the EAGLE with regards to the Eagle no longer being hungry?

- [ ] Pleased that the outcome is desirable for the Eagle (Happy-For)
- [ ] Pleased that the outcome is undesirable for the Eagle (Gloating)
- [ ] Displeased that the outcome is desirable for the Eagle (Resentment)
- [ ] Displeased that the outcome is undesirable for the Eagle (Pity)
- [ ] None of the above

1.1.9. Which of the following (if any) would you expect the EAGLE to feel about the Fox losing her cubs?

- [ ] Pleased about this outcome (Joy)
- [ ] Displeased about this outcome (Distress)
- [ ] None of the above

1.1.10. Which of the following (if any) would you expect the FOX to feel about losing her cubs?

- [ ] Pleased about this outcome (Joy)
- [ ] Displeased about this outcome (Distress)
- [ ] None of the above

1.1.11. Which of the following (if any) would you expect the EAGLE to feel about her own action with regards to the Fox losing her cubs?

- [ ] Approving of her action and pleased with the outcome (Gratification)
- [ ] Disapproving of her action and displeased with the outcome (Remorse)
- [ ] None of the above

1.1.12. Which of the following (if any) would you expect the FOX to feel about the Eagle's action with regards to the Fox losing her cubs?

- [ ] Approving of the Eagle's action and pleased with the outcome (Gratitude)
- [ ] Disapproving of the Eagle's action and displeased with the outcome (Anger)
- [ ] None of the above

1.1.13. Which of the following (if any) would you expect the EAGLE to feel towards the FOX with regards to the Fox losing her cubs?

- [ ] Pleased that the outcome is desirable for the Fox (Happy-For)
- [ ] Pleased that the outcome is undesirable for the Fox (Gloating)
- [ ] Displeased that the outcome is desirable for the Fox (Resentment)
- [ ] Displeased that the outcome is undesirable for the Fox (Pity)
- [ ] None of the above

1.1.14. Which of the following (if any) would you expect the FOX to feel towards the EAGLE with regards to the Fox losing her cubs?

- [ ] Pleased that the outcome is desirable for the Eagle (Happy-For)
- [ ] Pleased that the outcome is undesirable for the Eagle (Gloating)
- [ ] Displeased that the outcome is desirable for the Eagle (Resentment)

- [ ] Displeased that the outcome is undesirable for the Eagle (Pity)
- [ ] None of the above

1.1.15. Which of the following (if any) would you expect the EAGLE to feel towards the FOX after this event?

- [ ] Like
- [ ] Dislike
- [ ] None of the above

1.1.16. Which of the following (if any) would you expect the FOX to feel towards the EAGLE after this event?

- [ ] Like
- [ ] Dislike
- [ ] None of the above

Event 2: The Eagle brings a flaming carcass back to its nest. It catches fire, causing the eaglets to fall from the nest.

The action is carried out by the Eagle.

There is one consequence of this event:

a) The eaglets fall from the nest.

1.2.1. Which of the following (if any) would you expect the EAGLE to feel about her own action of bringing a flaming carcass back to her nest?

- [ ] Approving of her action (Pride)
- [ ] Disapproving of her action (Shame)
- [ ] None of the above

1.2.2. Which of the following (if any) would you expect the FOX to feel about the Eagle's action of bringing a flaming carcass back to her nest?

- [ ] Approving of the Eagle's action (Admiration)
- [ ] Disapproving of the Eagle's action (Reproach)
- [ ] None of the above

1.2.3. Which of the following (if any) would you expect the EAGLE to feel about the eaglets falling from the nest?

- [ ] Pleased about this outcome (Joy)
- [ ] Displeased about this outcome (Distress)
- [ ] None of the above

1.2.4. Which of the following (if any) would you expect the FOX to feel about the eaglets falling from the nest?

- [ ] Pleased about this outcome (Joy)

- [ ] Displeased about this outcome (Distress)
- [ ] None of the above

1.2.5. Which of the following (if any) would you expect the EAGLE to feel about her own action with regards to the eaglets falling from the nest?

- [ ] Approving of her action and pleased with the outcome (Gratification)
- [ ] Disapproving of her action and displeased with the outcome (Remorse)
- [ ] None of the above

1.2.6. Which of the following (if any) would you expect the FOX to feel about the Eagle's action with regards to the eaglets falling from the nest?

- [ ] Approving of the Eagle's action and pleased with the outcome (Gratitude)
- [ ] Disapproving of the Eagle's action and displeased with the outcome (Anger)
- [ ] None of the above

1.2.7. Which of the following (if any) would you expect the EAGLE to feel towards the FOX with regards to the eaglets falling from the nest?

- [ ] Pleased that the outcome is desirable for the Fox (Happy-For)
- [ ] Pleased that the outcome is undesirable for the Fox (Gloating)
- [ ] Displeased that the outcome is desirable for the Fox (Resentment)
- [ ] Displeased that the outcome is undesirable for the Fox (Pity)
- [ ] None of the above

1.2.8. Which of the following (if any) would you expect the FOX to feel towards the EAGLE with regards to the eaglets falling from the nest?

- [ ] Pleased that the outcome is desirable for the Eagle (Happy-For)
- [ ] Pleased that the outcome is undesirable for the Eagle (Gloating)
- [ ] Displeased that the outcome is desirable for the Eagle (Resentment)
- [ ] Displeased that the outcome is undesirable for the Eagle (Pity)
- [ ] None of the above

1.2.9. Which of the following (if any) would you expect the EAGLE to feel towards the FOX after this event?

- [ ] Like
- [ ] Dislike
- [ ] None of the above

1.2.10. Which of the following (if any) would you expect the FOX to feel towards the EAGLE after this event?

- [ ] Like
- [ ] Dislike
- [ ] None of the above

Event 3: The Fox eats the eaglets in front of the Eagle.

The action is carried out by the Fox.

There are two consequences of this event:

a) The Fox is no longer hungry; and

b) The Eagle loses her eaglets.

1.3.1. Which of the following (if any) would you expect the FOX to feel about her own action of eating the eaglets?

- [ ] Approving of her action (Pride)
- [ ] Disapproving of her action (Shame)
- [ ] None of the above

1.3.2. Which of the following (if any) would you expect the EAGLE to feel about the Fox's action of eating the eaglets?

- [ ] Approving of the Fox's action (Admiration)
- [ ] Disapproving of the Fox's action (Reproach)
- [ ] None of the above

1.3.3. Which of the following (if any) would you expect the FOX to feel about no longer being hungry?

- [ ] Pleased about this outcome (Joy)
- [ ] Displeased about this outcome (Distress)
- [ ] None of the above

1.3.4. Which of the following (if any) would you expect the EAGLE to feel about the Fox no longer being hungry?

- [ ] Pleased about this outcome (Joy)
- [ ] Displeased about this outcome (Distress)
- [ ] None of the above

1.3.5. Which of the following (if any) would you expect the FOX to feel about her own action with regards to no longer being hungry?

- [ ] Approving of her action and pleased with the outcome (Gratification)
- [ ] Disapproving of her action and displeased with the outcome (Remorse)
- [ ] None of the above

1.3.6. Which of the following (if any) would you expect the EAGLE to feel about the Fox's action with regards to the Fox no longer being hungry?

- [ ] Approving of the Fox's action and pleased with the outcome (Gratitude)
- [ ] Disapproving of the Fox's action and displeased with the outcome (Anger)
- [ ] None of the above

1.3.7. Which of the following (if any) would you expect the FOX to feel towards the EAGLE with

regards to the Fox no longer being hungry?

- [ ] Pleased that the outcome is desirable for the Eagle (Happy-For)
- [ ] Pleased that the outcome is undesirable for the Eagle (Gloating)
- [ ] Displeased that the outcome is desirable for the Eagle (Resentment)
- [ ] Displeased that the outcome is undesirable for the Eagle (Pity)
- [ ] None of the above

1.3.8. Which of the following (if any) would you expect the EAGLE to feel towards the FOX with regards to the Fox no longer being hungry?

- [ ] Pleased that the outcome is desirable for the Fox (Happy-For)
- [ ] Pleased that the outcome is undesirable for the Fox (Gloating)
- [ ] Displeased that the outcome is desirable for the Fox (Resentment)
- [ ] Displeased that the outcome is undesirable for the Fox (Pity)
- [ ] None of the above

1.3.9. Which of the following (if any) would you expect the FOX to feel about the Eagle losing her eaglets?

- [ ] Pleased about this outcome (Joy)
- [ ] Displeased about this outcome (Distress)
- [ ] None of the above

1.3.10. Which of the following (if any) would you expect the EAGLE to feel about losing her eaglets?

- [ ] Pleased about this outcome (Joy)
- [ ] Displeased about this outcome (Distress)
- [ ] None of the above

1.3.11. Which of the following (if any) would you expect the FOX to feel about her own action with regards to the Eagle losing her eaglets?

- [ ] Approving of her action and pleased with the outcome (Gratification)
- [ ] Disapproving of her action and displeased with the outcome (Remorse)
- [ ] None of the above

1.3.12. Which of the following (if any) would you expect the EAGLE to feel about the Fox's action with regards to the Eagle losing her eaglets?

- [ ] Approving of the Fox's action and pleased with the outcome (Gratitude)
- [ ] Disapproving of the Fox's action and displeased with the outcome (Anger)
- [ ] None of the above

1.3.13. Which of the following (if any) would you expect the FOX to feel towards the EAGLE with regards to the Eagle losing her eaglets?

- [ ] Pleased that the outcome is desirable for the Eagle (Happy-For)
- [ ] Pleased that the outcome is undesirable for the Eagle (Gloating)
- [ ] Displeased that the outcome is desirable for the Eagle (Resentment)

- [ ] Displeased that the outcome is undesirable for the Eagle (Pity)
- [ ] None of the above

1.2.14. Which of the following (if any) would you expect the EAGLE to feel towards the FOX with regards to the Eagle losing her eaglets?

- [ ] Pleased that the outcome is desirable for the Fox (Happy-For)
- [ ] Pleased that the outcome is undesirable for the Fox (Gloating)
- [ ] Displeased that the outcome is desirable for the Fox (Resentment)
- [ ] Displeased that the outcome is undesirable for the Fox (Pity)
- [ ] None of the above

1.3.15. Which of the following (if any) would you expect the FOX to feel towards the EAGLE after this event?

- [ ] Like
- [ ] Dislike
- [ ] None of the above

1.3.16. Which of the following (if any) would you expect the EAGLE to feel towards the FOX after this event?

- [ ] Like
- [ ] Dislike
- [ ] None of the above

« Back Continue »

Appendix B

Aleph Mode Declarations

To aid the interpretation of the moral rules produced by Aleph, which were presented in Chapter 5 (refer to Section 5.4.3), in this appendix we provide mode declarations for the emotions, to clarify their parameter structure. Listing B.1 includes mode declarations corresponding to each of the OCC emotions except Love and Hate, which were not used in generating rules due to their persistent nature. As explained in Section 5.2.2: *Mode Declarations*, `story`, `agent`, `consequence`, and `t` indicate parameter type, and `+` or `-` signify whether they are input or output variables. For those emotions with multiple agent variables, the first represents the agent experiencing the emotion, and the second is the agent it is directed towards.

```
% Mode declarations for OCC emotions (except Love and Hate)

% Event-based emotions
:- modeb(*,joy(+story,-agent,-consequence,-t)).
:- modeb(*,distress(+story,-agent,-consequence,-t)).

% Fortunes-of-others emotions
:- modeb(*,happyfor(+story,-agent,-agent,-consequence,-t)).
:- modeb(*,pity(+story,-agent,-agent,-consequence,-t)).
:- modeb(*,gloating(+story,-agent,-agent,-consequence,-t)).
:- modeb(*,resentment(+story,-agent,-agent,-consequence,-t)).

% Agent-based emotions
% Self-directed
:- modeb(*,pride(+story,-agent,-event,-t)).
:- modeb(*,shame(+story,-agent,-event,-t)).
% Other-directed
:- modeb(*,admiration(+story,-agent,-agent,-event,-t)).
:- modeb(*,reproach(+story,-agent,-agent,-event,-t)).

% Compound emotions
% Self-directed
:- modeb(*,gratification(+story,-agent,-event,-consequence,-t)).
:- modeb(*,remorse(+story,-agent,-event,-consequence,-t)).
% Other-directed
:- modeb(*,anger(+story,-agent,-agent,-event,-consequence,-t)).
:- modeb(*,gratitude(+story,-agent,-agent,-event,-consequence,-t)).

% Prospect-based emotions
% Unconfirmed
:- modeb(*,hope(+story,-agent,-consequence,-t)).
:- modeb(*,fear(+story,-agent,-consequence,-t)).
% Confirmed
:- modeb(*,satisfaction(+story,-agent,-consequence,-t)).
:- modeb(*,fearsconfirmed(+story,-agent,-consequence,-t)).
% Disconfirmed
:- modeb(*,disappointment(+story,-agent,-consequence,-t)).
:- modeb(*,relief(+story,-agent,-consequence,-t)).
```

Listing B.1: Aleph mode declarations for OCC emotions

Appendix C

Aleph Classification Tree Rules

In this appendix, we provide the rules produced by Aleph during the tree-learning experiments described in Chapter 5. Section C.1 presents the rules produced when learning individual moral classes, and Section C.2 presents the rules produced when learning to classify moral groups.

C.1 Learning Individual Morals

The rules shown in Listing C.1 were produced by Aleph when learning a classification tree for individual morals, as described in Section 5.3.4. The classification tree diagram shown in Figure 5.1 was constructed from these rules.

```
1  moral(A,B) :-
2     not fear(A,C,D,E), not gratitude(A,F,C,G,D,E), shame(A,C,G,E),
3     joy(A,C,D,H), anger(A,F,C,I,J,K), B=greed.
4
5  moral(A,B) :-
6     not fear(A,C,D,E), not gratitude(A,F,C,G,D,E), shame(A,C,G,E),
7     joy(A,C,D,H), not anger(A,F,C,I,J,K), B=greed.
8
9  moral(A,B) :-
10    not fear(A,C,D,E), not gratitude(A,F,C,G,D,E), shame(A,C,G,E),
11    not joy(A,C,D,H), remorse(A,C,G,D,E), B=realistic.
12
13 moral(A,B) :-
14    not fear(A,C,D,E), not gratitude(A,F,C,G,D,E), shame(A,C,G,E),
15    not joy(A,C,D,H), not remorse(A,C,G,D,E), B=realistic.
16
17 moral(A,B) :-
18    not fear(A,C,D,E), not gratitude(A,F,C,G,D,E), not shame(A,C,G,E),
19    satisfaction(A,C,D,E), distress(A,C,H,E), B=greed.
20
21 moral(A,B) :-
22    not fear(A,C,D,E), not gratitude(A,F,C,G,D,E), not shame(A,C,G,E),
23    satisfaction(A,C,D,E), not distress(A,C,H,E), B=recklessness.
24
25 moral(A,B) :-
26    not fear(A,C,D,E), not gratitude(A,F,C,G,D,E), not shame(A,C,G,E),
27    not satisfaction(A,C,D,E), pity(A,F,C,D,E), B=greed.
28
29 moral(A,B) :-
30    not fear(A,C,D,E), not gratitude(A,F,C,G,D,E), not shame(A,C,G,E),
31    not satisfaction(A,C,D,E), not pity(A,F,C,D,E), B=retribution.
32
33 moral(A,B) :-
34    fear(A,C,D,E), anger(A,F,G,H,I,J), gratitude(A,K,C,L,M,N),
35    B=reward.
36
37 moral(A,B) :-
38    fear(A,C,D,E), anger(A,F,G,H,I,J), not gratitude(A,K,C,L,M,N),
39    B=retribution.
40
41 moral(A,B) :-
42    fear(A,C,D,E), not anger(A,F,G,H,I,J), B=pride.
43
44 moral(A,B) :-
45    not fear(A,C,D,E), gratitude(A,F,C,G,D,E), B=retribution.
```

Listing C.1: Aleph's rules for individual moral classification tree

C.2 Learning Moral Groups

The rules shown in Listing C.2 were produced by Aleph when learning a classification tree for moral groups, as described in Section 5.3.5. (i.e. Retribution, Greed, and Pride were grouped, as were Realistic Expectations and Recklessness). The classification tree diagram shown in Figure 5.4 was constructed from these rules.

```
1  moral(A,B) :-
2     not gratitude(A,C,D,E,F,G), not fear(A,C,F,G), not pride(A,C,E,G),
3     joy(A,C,F,G), distress(A,C,H,I), B=ret_greed_pride.
4
5  moral(A,B) :-
6     not gratitude(A,C,D,E,F,G), not fear(A,C,F,G), not pride(A,C,E,G),
7     joy(A,C,F,G), not distress(A,C,H,I), B=ret_greed_pride.
8
9  moral(A,B) :-
10    not gratitude(A,C,D,E,F,G), not fear(A,C,F,G), not pride(A,C,E,G),
11    not joy(A,C,F,G), shame(A,C,E,G), B=real_reck.
12
13 moral(A,B) :-
14    not gratitude(A,C,D,E,F,G), not fear(A,C,F,G), not pride(A,C,E,G),
15    not joy(A,C,F,G), not shame(A,C,E,G), B=ret_greed_pride.
16
17 moral(A,B) :-
18    not gratitude(A,C,D,E,F,G), not fear(A,C,F,G), pride(A,C,E,G),
19    B=real_reck.
20
21 moral(A,B) :-
22    gratitude(A,C,D,E,F,G), gratitude(A,D,C,H,I,J), B=reward.
23
24 moral(A,B) :-
25    gratitude(A,C,D,E,F,G), not gratitude(A,D,C,H,I,J),
26    B=ret_greed_pride.
27
28 moral(A,B) :-
29    not gratitude(A,C,D,E,F,G), fear(A,C,F,G),
30    B=ret_greed_pride.
```

Listing C.2: Aleph's rules for grouped moral classification tree

Appendix D

Domain Descriptions Provided to Stage 2 Participants

In this appendix, we include the domain descriptions that were provided to the human authors participating in Stage 2 of our evaluation process (refer to Chapter 7, Section 7.3). Section D.1 presents the task description, which was common to all three story-telling domains. Sections D.2, D.3, and D.4 present the domain-specific instructions corresponding to the Fairytale, Family, and Animals story worlds respectively.

D.1 Task Description

Generating Stories with Morals - Stage 2

Task Description

Your task is as follows:

1. Choose one of the morals from the table below.
2. Using the characters, actions and emotions provided in the Domain Description on the following pages, construct a sequence of events that you believe would convey the selected moral.

Note that you do not need to provide the full text of the story, only list the events that would take place, and any emotions you think the characters should feel. There is no minimum length for stories; a story can consist of a single event if you believe it conveys the intended moral.

If you believe one or more of the morals listed below *cannot* be conveyed using only the events provided for this domain (for any reason), please note this in your response.

Moral Definitions

The following table lists the available morals, along with a brief definition of each:

Moral	**Definition**
Retribution	You get what you deserve for doing something bad.
Reward	You get what you deserve for doing something good.
Greed	You should not be greedy.
Realistic Expectations	You should have realistic expectations for yourself.
Recklessness	You should not do something recklessly, but consider your actions.
Pride	You should not be excessively proud.

D.2 Domain Description - Fairytale

Overview

This story domain is a simple fairytale world, with several typical fairytale characters and a list of actions they are able to perform, as well as a list of emotions they can feel. Some actions always succeed, while others may fail.

Characters

The following table lists the characters that exist in this domain, and what objects those characters have at the start of the story:

Character	What the Character Has Initially
Princess	crown
Knight	sword
Wizard	wand, spellbook
Dragon	treasure
Fairy	fairydust
Troll	
Unicorn	
Dwarf	gold, axe

Properties of Characters

Characters can have certain properties, which can change when actions happen. Initially, all characters are alive, free, uninjured and not under any spells.

The available properties are:

Property
is hungry
is alive
has object
is free
is enchanted
has pricked finger
is injured

Objects

The following objects exist in this story domain:

Object	Properties
apple	edible
crown	valuable
fairydust	magic
gold	valuable
mushroom	edible, poisonous
rose	edible
spellbook	magic
sword	weapon
treasure	valuable
wand	magic
axe	weapon

Actions

The following table lists the actions characters can perform in this domain, along with their prerequisites (what must be true to perform the action) and their outcomes (what happens as a result of the action). The table also shows which actions can fail when attempted (if an action fails, none of its outcomes happen).

Action	Can Fail?	Prerequisites	Outcomes (if successful)
eat	NO	1. Character must have the object they are eating.	1. The character is not hungry. 2. If the object was poisonous, character dies.
steal	YES	1. The character they are stealing from must have the object.	1. The character has the object. 2. The character they stole from does not have the object.
kill	YES	1. The character they are killing must be alive.	1. The character they killed dies.
give	NO	1. The character must have the object they are giving.	1. The character does not have the object. 2. The character they gave it to has the object.
rescue	YES	1. The character they are rescuing must be captive.	1. The character they rescued is now free.
kidnap	YES	1. The character they are kidnapping must be free.	1. The character they kidnapped is no longer free. 2. They now have the character they kidnapped.
cast spell	YES	1. The character they are casting a spell on cannot already be enchanted.	1. The character they cast a spell on is now enchanted.
free from spell	YES	1. The character they are freeing from the spell must be enchanted.	1. The character they freed from the spell is no longer enchanted.
pick rose	NO	1. The character must not already have a rose.	1. The character has a rose. 2. The character pricks their finger.
pick mushroom	NO	1. The character must not already have a mushroom.	1. The character has a mushroom.
climb tree to pick apple	NO	1. The character must not already have an apple. 2. The character must not be injured.	1. The character has an apple. 2. The character falls from the tree and is injured.
release	YES	1. The character must have the character they want to release.	1. The character no longer has the character they released. 2. The character they released is now free.
heal	YES	1. The character they want to heal must be injured or have a pricked finger.	1. The character they healed is no longer injured.
gets hungry	NO	1. The character must not already be hungry.	1. The character is hungry.

Emotions

The following table lists the emotions that characters are able to feel:

Joy	Relief	Pride	Gratitude
Distress	Disappointment	Shame	Anger
Hope	Happy-For	Admiration	Love
Fear	Pity	Reproach	Hate
Satisfaction	Resentment	Gratification	
Fears-Confirmed	Gloating	Remorse	

Examples

Below are examples of how you could write out an event and the associated character emotions.

Example 1: Action succeeding

Knight hopes for treasure.
Knight steals treasure from dragon.
Knight feels joy, dragon feels distress and anger.

Example 2: Action failing

Princess tries to kill the dragon, but fails.
Princess feels distress and disappointment, dragon feels reproach and gloating.

D.3 Domain Description - Family

Overview

This story domain is a simple household consisting of a father, a mother, a son and a daughter. There is a set of actions the characters are able to perform, as well as a list of emotions they can feel. Some actions always succeed, while others may fail. Sometimes there are conditions that must be met so that an action can be attempted (prerequisites).

Characters

The following table lists the characters that exist in this domain, and what objects those characters have at the start of the story:

Character	What the Character Has Initially
Father (adult)	
Mother (adult)	
Son (child)	a room, clothes
Daughter (child)	a room, clothes

All characters are able to access items classified as 'furniture', and there is one of each such item in the house. Characters can only move (give, take, etc) items that are 'movable'.

Properties of Characters

Characters and objects have certain properties, which can change when actions happen. Initially, all characters are not hungry and unhurt.

The available properties are:

Property
is hungry
is hurt
has object

Objects

The following objects exist in this story domain:

Object	Properties
room	cleanable
clothes	washable
dishes	furniture, washable, breakable
dinner	meal, edible, movable
lunch	meal, edible, movable
breakfast	meal, edible, movable
ice-cream	edible, buyable, movable
toy	buyable, movable
window	furniture, breakable
stove	furniture, breakable
vase	breakable, furniture, movable
picture	movable
flower	movable

Initially, the vase is on a high shelf.

Actions

The following table lists the actions characters can perform in this domain, along with their prerequisites (what must be true to perform the action) and their outcomes (what happens as a result of the action). The table also shows which actions can fail when attempted. If an action fails, none of its normal outcomes happen, but in some cases other things will happen instead.

You may only use actions from this table in your story.

Action	Can Fail?	Prerequisites	Outcomes
buy	NO	1. Character must be an adult. 2. The object must be Buyable. 3. The character must not already have the object.	1. The character has the object.
gives to	NO	1. The character must have the object they are giving. 2. The character they are giving it to must not have the object.	1. The character doesn't have the object anymore. 2. The character they gave it to now has the object.
gets hungry	NO	1. The character must not be hungry.	1. The character is hungry.
cook for	YES	1. The stove must not be broken. 2. The dishes must not be broken. 3. The character they are cooking for must not already have that meal. 4. If the character is the mother, this always succeeds. 5. For other characters this can fail.	SUCCEEDS: 1. The character they cooked for has the meal. 2. The dishes are dirty. FAILS: 1. The character burns themselves and is hurt. 2. The dishes are dirty.
eats	NO	1. The character must have the item they are eating. 2. The item must be edible.	1. The character is no longer hungry. 2. The character no longer has that object.
messes up	NO	1. The object being messed up must be Cleanable, and cannot already be messy.	1. The object is not clean.
cleans	YES	1. The object being cleaned must be Cleanable, but cannot already be clean.	SUCCEEDS: 1. The object is clean.
washes	YES	1. The object being washed must be Washable, but cannot already be clean. 2. The object must not be broken.	SUCCEEDS: 1. The object is clean. FAILS: 1. If the object is breakable, it breaks.
plays in mud	NO	1. The character must be a child. 2. Their clothes must be clean.	1. The character's clothes are dirty.
pushes over in the mud	YES	1. The character must be a child. 2. If the character they are trying to push over is an adult, this always fails.	SUCCEEDS: 1. The character pushed over is hurt. 2. The character pushed over has dirty clothes.
throws cricket ball	NO	1. The window must not already be broken.	1. The window is broken.
picks flower	NO	1. The character must not already have a flower.	1. The character has a flower.
draws picture	YES	1. The character must be a child. 2. The character must not already have a picture.	SUCCEEDS: 1. The character has a picture.

breaks	NO	1. The object must be Breakable, but cannot already be broken.	1. The object is broken.
fixes	YES	1. The object must be Breakable, and must be broken.	SUCCEEDS: 1. The object is no longer broken.
rides bike	YES		FAILS: 1. The character falls off and is hurt.
puts on high shelf	YES	1. The character must have the object. 2. If the character is an adult, this action always succeeds. 3. If the character is a child, this action can fail.	SUCCEEDS: 1. The object is on the high shelf. 2. The agent does not have the object anymore.
gets off high shelf	YES	1. The object must be on the high shelf. 2. If the character is an adult, this action always succeeds. 3. If the character is a child, this action can fail.	SUCCEEDS: 1. The object is not on the high shelf. 2. The character has the object. FAILS: 1. The object is not on the high shelf. 2. The object is broken if it's Breakable. 3. The character is hurt.
takes from	YES	1. The character must not have the object. 2. The character they are taking from must have the object. 3. If the character is an adult trying to take something from a child, they always succeed.	SUCCEEDS: 1. The character has the object. 2. The character they took it form does not have the object.

Emotions

The following table lists the emotions that characters are able to feel:

Joy	Relief	Pride	Gratitude
Distress	Disappointment	Shame	Anger
Hope	Happy-For	Admiration	Love
Fear	Pity	Reproach	Hate
Satisfaction	Resentment	Gratification	
Fears-Confirmed	Gloating	Remorse	

Examples

Below are examples of how you could write out an event and the associated character emotions.

Example 1: Action succeeding

The boy throws a cricket ball and smashes a window.
He feels shame and distress. His father feels anger.

Example 2: Action failing

The girl tries to ride a bike, but falls off and hurts herself.
She feels disappointment and distress.

D.4 Domain Description - Animals

Overview

This story domain consists of various animals interacting with one another. There is a set of actions the characters are able to perform, as well as a list of emotions they can feel. Some actions always succeed, while others may fail. Sometimes there are conditions that must be met so that an action can be attempted (prerequisites).

Characters

The following table lists the characters that exist in this domain, and some key facts about those characters:

Character	Facts About the Characters
Lion	carnivore
Bird	herbivore
Wolf	carnivore
Bee	herbivore, small
Bear	omnivore
Squirrel	herbivore
Toad	omnivore, poisonous, small

All characters are edible.

Properties of Characters

Characters and objects have certain properties, which can change when actions happen. Initially, all characters are alive, free, not hungry and don't have any objects.

The available properties for characters are:

Property
is hungry
is alive
has object
is free
in snare

Objects

The following objects exist in this story domain:

Object	Properties
honey	edible, vegetarian
meat	edible
berries	edible, vegetarian, collectable
nuts	edible, vegetarian, collectable
vine	edible, vegetarian, poisonous
pollen	
fish	edible

Actions

The following table lists the actions characters can perform in this domain, along with their prerequisites (what must be true to perform the action) and their outcomes (what happens as a result of the action). The table also shows which actions can fail when attempted. If an action fails, none of its normal outcomes happen, but in some cases other things will happen instead.

You may only use actions from this table in your story.

Action	Can Fail?	Prerequisites	Outcomes
collects pollen	YES	1. The character must not have any pollen. 2. If the character is a bee, this action always succeeds. 3. Otherwise, this action fails.	SUCCEEDS: 1. The character has pollen.
makes honey	YES	1. The character must have pollen. 2. If the character is a bee, this action always succeeds. 3. Otherwise, this action fails.	SUCCEEDS: 1. The character has honey. 2. The character no longer has pollen.
collects object	YES	1. The object must be 'collectable'. 2. The character must not already have the object. 3. If the character is 'small' this always fails.	SUCCEEDS: 1. The character has the object.
eats object	YES	1. The character must have the object. 2. The object must be edible. 3. This succeeds only if the object is suitable for the character's diet.	SUCCEEDS: 1. The character is not hungry. 2. If the object was poisonous, the character is no longer alive. FAILS: 1. The character chokes and dies.
catches	YES	1. The character they are trying to catch must be free. 2. Fails if the character is 'small' but their target is not.	SUCCEEDS: 1. The character has the other character. 2. The other character is not free.
lets go	NO	1. The character must have the character they want to let go.	1. The character they let go is free. 2. They don't have the character.
takes meat from snare	NO	1. The character must not already have meat. 2. The character must be free.	1. The character is in the snare. 2. The character is not free. 3. The character has the meat.
frees from snare	YES	1. The character must be free. 2. The character they want to free must be in a snare.	SUCCEEDS: 1. The character they freed is no longer in the snare.
gives to	NO	1. The character must have the object they are giving. 2. The character they are giving it to must not have that object.	1. The character no longer has the object. 2. The character they gave it to has the object.
takes from	YES	1. The character must not have the object. 2. The character they want to take it from must have the object.	SUCCEEDS: 1. The character has the object. 2. The character they took it from does not have the object.

gets hungry	NO	1. The character must not already be hungry.	1. The character is hungry.
attacks	YES	1. The character must be free. 2. The character they want to attack must be alive. 3. This always succeeds if the attacker is not 'small' but the target is 'small.'	SUCCEEDS: 1. The character they attacked is no longer alive. FAILS: 1. The attacker is no longer alive.
picks vine	NO	1. The character must not already have a vine.	1. The character has a vine.
catches fish	YES	1. The character must not already have a fish. 2. If the character is 'small', this always fails.	SUCCEEDS: 1. The character has a fish. FAILS: 2. The character drowns (is no longer alive).

Emotions

The following table lists the emotions that characters are able to feel:

Joy	Relief	Pride	Gratitude
Distress	Disappointment	Shame	Anger
Hope	Happy-For	Admiration	Love
Fear	Pity	Reproach	Hate
Satisfaction	Resentment	Gratification	
Fears-Confirmed	Gloating	Remorse	

Examples

Below are examples of how you could write out an event and the associated character emotions.

Example 1: Action succeeding

The bee gives honey to the bear.
The bee feels pride. The bear feels joy and gratitude.

Example 2: Action failing

The lion tries to catch the squirrel, but fails.
He feels disappointment and distress.

Appendix E

Sample Stories

In this appendix, we provide a broader sample of MOSS-generated, human-authored, and moral-free stories. For each type, we provide three example stories per moral, one for each story world. All stories included in this appendix were used in the final evaluation survey. The complete set of stories comprising the survey is available online.[1]

E.1 MOSS-Generated Stories

The stories presented in this section were produced by MOSS, enforcing the relevant moral rules (as defined in Chapter 5, Section 5.5).

[1]The stories used in the evaluation survey are available at: `www.cse.unsw.edu.au/~msarlej/moss/stories`.

E.1.1 Retribution

Long ago in a far away land there lived a wizard, a dwarf and a princess. The wizard loved the princess.

One summer's morning the dwarf killed the princess. As a result, the princess was dead. The wizard felt distress that the princess was dead. The wizard felt anger towards the dwarf about killing the princess because the princess was dead. The wizard started to hate the dwarf.

The next day the wizard killed the dwarf. As a result, the dwarf was dead. The dwarf felt distress that he was dead. The dwarf felt anger towards the wizard about killing him because he was dead.

Figure E.1: MOSS-generated Retribution story: Fairytale domain

In a modern inner-city apartment there lived a boy and a girl. The boy loved the girl.

One chilly autumn day the boy messed up the girl's room. As a result, the girl's room was not clean. The girl felt distress that her room was not clean. The girl felt anger towards the boy about messing up her room because her room was not clean. The girl started to hate the boy.

Not long afterwards the girl pushed the boy over in the mud. As a result, the boy's clothes were not clean and the boy was hurt. The boy felt distress that he was hurt.

Figure E.2: MOSS-generated Retribution story: Family domain

Deep in the jungle there lived a lion, a toad, a squirrel and a bird. The squirrel loved the bird.

One warm afternoon the lion attacked and killed the bird. As a result, the bird was dead. The squirrel felt distress that the bird was dead. The squirrel felt distress that the lion was alive. The squirrel felt anger towards the lion about attacking the bird because the bird was dead. The squirrel started to hate the lion.

The next day the lion attacked and killed the toad. As a result, the toad was dead.

Some time later the squirrel caught the lion. As a result, she had the lion and the lion was not free. The lion felt distress that he was not free. The lion felt anger towards the squirrel about catching him because he was not free.

Figure E.3: MOSS-generated Retribution story: Animals domain

E.1.2 Greed

In a distant kingdom there lived a unicorn, a dragon, a fairy and a princess.

The princess hoped that she would have the treasure. One day the princess stole the treasure from the dragon. As a result, she had the treasure and the dragon didn't have the treasure anymore. The princess felt satisfaction that she had the treasure. The dragon started to hate the princess. The unicorn and the fairy started to love the princess.

The next day the fairy stole the crown from the princess. As a result, she had the crown and the princess didn't have the crown anymore. The unicorn and the princess started to hate the fairy. The dragon started to love the fairy.

More time passed, and the unicorn stole the treasure from the princess. As a result, he had the treasure and the princess didn't have the treasure anymore. The princess felt distress that she didn't have the treasure anymore.

Figure E.4: MOSS-generated Greed story: Fairytale domain

In a modern inner-city apartment there lived a father, a boy and a mother.

One weekend the boy drew a picture.

Not long afterwards the boy cooked lunch for the mother. As a result, the dishes were not clean and the mother had lunch. The father and the mother started to love the boy.

The father hoped that he would have the picture. More time passed, and the father took the picture from the boy. As a result, he had the picture. The father felt satisfaction that he had the picture. The father felt shame about taking the picture. The boy and the mother started to hate the father.

The father hoped that he would have lunch. Afterwards the father took lunch from the mother. As a result, he had lunch and the mother didn't have lunch anymore. The father felt satisfaction that he had lunch. The father felt shame about taking lunch.

Eventually the boy took lunch from the father. As a result, the father didn't have lunch anymore. The father felt distress that he didn't have lunch anymore.

Figure E.5: MOSS-generated Greed story: Family domain

Deep in the jungle there lived a lion, a squirrel, a wolf and a bee. The squirrel hated the wolf.

One chilly autumn day the wolf picked a vine. As a result, he had the vine.

The squirrel hoped that she would have the vine. The next day the squirrel took the vine from the wolf. As a result, she had the vine and the wolf didn't have the vine anymore. The squirrel felt satisfaction that she had the vine. The squirrel felt shame about taking the vine. The wolf and the bee started to hate the squirrel.

The squirrel hoped that she would have the bee. Some time later the squirrel caught the bee. As a result, she had the bee and the bee was not free. The squirrel felt satisfaction that she had the bee. The squirrel felt shame about catching the bee. The lion and the wolf started to love the squirrel.

A few days later the lion caught the squirrel. As a result, he had the bee, he had the squirrel, the squirrel didn't have the bee anymore and the squirrel was not free. The squirrel felt distress that she didn't have the bee anymore. The squirrel felt distress that she was not free.

Figure E.6: MOSS-generated Greed story: Animals domain

E.1.3 Pride

Once upon a time there lived a wizard, a troll and a dwarf.

One summer's morning the troll kidnapped the wizard. As a result, he had the wizard and the wizard was not free. The troll felt joy that he had the wizard. The troll felt pride about kidnapping the wizard. The wizard and the dwarf started to hate the troll.

The next day the dwarf stole the wizard from the troll. As a result, the troll didn't have the wizard anymore. The troll felt distress that he didn't have the wizard anymore.

Figure E.7: MOSS-generated Pride story: Fairytale domain

In a nice house on a quiet street there lived a father, a boy and a mother.

One evening the mother bought a toy.

Not long afterwards the mother put the toy on a high shelf.

Soon after that, the boy got the toy off the high shelf. As a result, he had the toy. The boy felt joy that he was not hurt. The boy felt joy that he had the toy. The boy felt pride about getting the toy off the high shelf.

A little later the father took the toy from the boy. As a result, the boy didn't have the toy anymore. The boy felt distress that he didn't have the toy anymore.

Figure E.8: MOSS-generated Pride story: Family domain

In a lush valley there lived a bear, a lion, a squirrel and a wolf.

One warm afternoon the squirrel picked a vine. As a result, she had the vine.

The next day the wolf caught the bear. As a result, he had the bear and the bear was not free. The wolf felt joy that he had the bear. The wolf felt pride about catching the bear. The bear and the squirrel started to hate the wolf. The lion started to love the wolf.

Some time later the lion caught the wolf. As a result, he had the wolf, he had the bear, the wolf didn't have the bear anymore and the wolf was not free. The wolf felt distress that he didn't have the bear anymore. The wolf felt distress that he was not free.

Figure E.9: MOSS-generated Pride story: Animals domain

E.1.4 Realistic Expectations

In a distant kingdom there lived a unicorn, a dragon and a dwarf. The unicorn hated the dwarf.

One summer's morning the dwarf tried to kidnap the unicorn, but failed.

The dwarf hoped that he would have the treasure. Later that day the dwarf tried to steal the treasure from the dragon, but failed. The dwarf felt disappointment that he didn't have the treasure.

Figure E.10: MOSS-generated Realistic Expectations story: Fairytale domain

In a modern inner-city apartment there lived a father and a girl.

One weekend the girl rode a bike.

The father hoped that he would have dinner. That evening the girl tried to cook dinner for the father, but failed and burnt herself. As a result, she was hurt and the dishes were not clean. The father felt disappointment that he didn't have dinner.

Figure E.11: MOSS-generated Realistic Expectations story: Family domain

In a lush valley there lived a bear, a toad, a squirrel and a bee. The squirrel hated the toad.

One warm afternoon the toad caught the bee. As a result, he had the bee and the bee was not free. The bee started to hate the toad.

The squirrel hoped that she would have the bear. Later that day the squirrel tried to catch the bear, but failed. The squirrel felt disappointment that she didn't have the bear.

Figure E.12: MOSS-generated Realistic Expectations story: Animals domain

E.1.5 Recklessness

In a distant kingdom there lived a troll.

One summer's morning the troll became hungry.

The troll hoped that he would have an apple. The next day the troll climbed a tree to pick an apple. He picked the apple, but then fell out of the tree and got injured. As a result, he had the apple and he was injured. The troll felt distress that he was injured. The troll felt satisfaction that he had the apple.

Figure E.13: MOSS-generated Recklessness story: Fairytale domain

In a modern-city apartment there lived a boy.

The boy feared that he would be hurt. Early one morning the boy tried to ride a bike, but fell off. As a result, he was hurt. The boy felt his fears were confirmed that he was hurt.

Figure E.14: MOSS-generated Recklessness story: Family domain

In a lush valley there lived a toad and a wolf. The wolf hated the toad.

One day the wolf caught the toad. As a result, he had the toad and the toad was not free. The toad started to hate the wolf.

The wolf hoped that the toad would die. Later that day the wolf ate the toad, which was poisonous. As a result, the wolf was dead, he didn't have the toad anymore and the toad was dead. The wolf felt distress that he was dead. The wolf felt satisfaction that the toad was dead.

Figure E.15: MOSS-generated Recklessness story: Animals domain

E.1.6 Reward

Once upon a time there lived a dwarf and a princess.

One day the dwarf gave gold to the princess. As a result, the princess had gold. The princess felt joy that she had gold. The princess felt gratitude towards the dwarf about giving gold to her because she had gold. The princess started to love the dwarf.

A short time later the princess gave the crown to the dwarf. As a result, the dwarf had the crown. The dwarf felt joy that he had the crown. The dwarf felt gratitude towards the princess about giving the crown to him because he had the crown.

Figure E.16: MOSS-generated Reward story: Fairytale domain

In a small country town there lived a father, a boy, a girl and a mother. The boy loved the girl.

One weekend the boy pushed the girl over in the mud. As a result, the girl was hurt and the girl's clothes were not clean. The father, the girl and the mother started to hate the boy.

The next day the father washed the girl's clothes. As a result, the girl's clothes were clean. The mother felt joy that the girl's clothes were clean. The mother felt gratitude towards the father about washing the girl's clothes because the girl's clothes were clean. The boy, the girl and the mother started to love the father.

More time passed, and the mother cooked breakfast for the father. As a result, the dishes were not clean and the father had breakfast. The father felt joy that he had breakfast. The father felt gratitude towards the mother about cooking breakfast for him because he had breakfast.

Figure E.17: MOSS-generated Reward story: Family domain

In a large forest there lived a lion and a bee.

One day the lion took meat from a snare. As a result, he was caught in a snare, he had the meat and he was not free. The bee started to hate the lion.

The next day the bee freed the lion from the snare. As a result, the lion was free and the lion was not caught in a snare. The lion felt joy that he was not free. The lion felt joy that he was not caught in a snare. The lion felt gratitude towards the bee about freeing him from the snare because he was not caught in a snare and he was free. The lion started to love the bee.

More time passed, and the lion gave meat to the bee. As a result, the bee had meat. The bee felt joy that she had meat.

Figure E.18: MOSS-generated Reward story: Animals domain

E.2 Human-Authored Stories

The stories in this section were composed by human authors, as part of Stage 2 of the evaluation process described in Chapter 7. Authors provided event sequences and emotions for each moral, within the constraints of the MOSS storytelling domains (described in Appendix D). The story text was automatically generated using the MOSS Text Generator.

E.2.1 Retribution

Long ago in a far away land there lived a unicorn, a knight and a dragon.

One summer's morning the dragon killed the unicorn. As a result, the unicorn was dead. The dragon felt gloating towards unicorn because the unicorn was dead.

A short time later the knight killed the dragon. As a result, the dragon was dead. The knight felt gratification about killing the dragon because the dragon was dead.

Figure E.19: Human-authored Retribution story: Fairytale domain

In a nice house on a quiet street there lived a girl and a mother.

One weekend the mother bought a toy.

Later that day the mother gave the toy to the girl. As a result, the girl had the toy. The girl felt joy that she had the toy. The girl felt gratitude towards the mother about giving the toy to her because she had the toy.

Some time later the girl threw a cricket ball. As a result, the window was broken. The girl felt distress that the window was broken. The mother felt anger towards the girl about throwing a cricket ball because the window was broken. The mother felt reproach towards the girl about throwing a cricket ball.

The girl feared that she wouldn't have the toy anymore. Then the mother took the toy from the girl. As a result, the girl didn't have the toy anymore. The girl felt distress that she didn't have the toy anymore.

Figure E.20: Human-authored Retribution story: Family domain

Deep in the jungle there lived a squirrel and a bird.

One day the squirrel collected nuts.

Not long afterwards the bird became hungry.

The day after that, the bird took nuts from the squirrel. As a result, the squirrel didn't have nuts anymore. The squirrel felt anger towards the bird about taking nuts because she didn't have nuts anymore.

Afterwards the squirrel attacked and killed the bird. As a result, the bird was dead. The squirrel felt satisfaction that the bird was dead.

Figure E.21: Human-authored Retribution story: Animals domain

E.2.2 Greed

Once upon a time there lived a troll, a knight, a dragon, a dwarf and a princess.

One day the troll stole the crown from the princess. As a result, he had the crown and the princess didn't have the crown anymore. The princess felt resentment towards troll because the troll had the crown. The princess felt distress that she didn't have the crown anymore. The princess felt anger towards the troll about stealing the crown because she didn't have the crown anymore.

A short time later the troll stole the treasure from the dragon. As a result, he had the treasure and the dragon didn't have the treasure anymore. The dragon felt resentment towards troll because the troll had the treasure. The dragon felt distress that he didn't have the treasure anymore. The dragon felt anger towards the troll about stealing the treasure because he didn't have the treasure anymore.

More time passed, and the troll stole gold from the dwarf. As a result, the dwarf had gold and the dwarf didn't have gold anymore. The dwarf felt resentment towards troll because he had gold. The dwarf felt distress that he didn't have gold anymore. The dwarf felt anger towards the troll about stealing gold because he didn't have gold anymore.

A short while after that the knight killed the troll. As a result, the troll was dead. The dragon felt joy and satisfaction that the troll was dead. The dwarf felt joy and satisfaction that the troll was dead. The princess felt joy and satisfaction that the troll was dead.

Figure E.22: Human-authored Greed story: Fairytale domain

In a nice house on a quiet street there lived a boy, a girl and a mother.

Early one morning the mother bought ice-cream.

A short time later the mother bought a toy.

Some time later the mother gave ice-cream to the boy.

Then the mother gave the toy to the girl.

A short while after that the boy took the toy from the girl. The mother felt reproach towards the boy about taking the toy.

Afterwards the mother took ice-cream from the boy.

Figure E.23: Human-authored Greed story: Family domain

In a large forest there lived a bear and a squirrel.

Early one morning the squirrel collected berries.

A short time later the squirrel collected nuts.

More time passed, and the squirrel gave nuts to the bear. As a result, the bear had nuts. The bear felt joy that he had nuts. The squirrel felt happy for the bear because the bear had nuts.

Then the bear tried to take berries from the squirrel, but failed. The bear felt disappointment and distress that he didn't have berries. The squirrel felt reproach towards the bear about trying to take the berries.

A short while after that the squirrel took nuts from the bear. As a result, the bear didn't have nuts anymore. The bear felt distress that he didn't have nuts anymore.

Figure E.24: Human-authored Greed story: Animals domain

E.2.3 Pride

Long ago in a far away land there lived a knight, a dragon and a princess.

One summer's morning the dragon kidnapped the princess.

The next day the knight rescued the princess from the dragon. As a result, the dragon didn't have the princess anymore and the princess was free. The knight felt gloating towards dragon because the dragon didn't have the princess anymore. The princess felt relief that the dragon didn't have her anymore. The princess felt joy that she was free. The knight felt pride about rescuing the princess from the dragon. The dragon felt anger towards the knight about rescuing the princess from him because he didn't have the princess anymore.

Some time later the dragon killed the knight. As a result, the knight was dead. The dragon felt satisfaction that the knight was dead.

Figure E.25: Human-authored Pride story: Fairytale domain

In a modern inner-city apartment there lived a father, a boy and a girl.

One day the boy messed up his room.

The next day the boy cleaned his room. As a result, his room was clean. The father felt satisfaction that the boy's room was clean.

Soon after that, the father bought a toy.

Afterwards the father gave the toy to the boy. As a result, the boy had the toy. The boy felt gloating towards the girl because he had the toy.

A little later the girl messed up the boy's room.

Figure E.26: Human-authored Pride story: Family domain

In a large forest there lived a bee and a bird.

One warm afternoon the bee collected some pollen.

The next day the bird took pollen from the bee. The bird felt pride about taking pollen.

The day after that, the bird tried to make honey, but failed. The bird felt shame about trying to make honey.

Figure E.27: Human-authored Pride story: Animals domain

E.2.4 Realistic Expectations

Once upon a time there lived a wizard, a troll and a fairy. The wizard hated the troll.

The troll hoped that he would have the spellbook. One day the wizard gave the spellbook to the fairy. The troll felt disappointment that he didn't have the spellbook.

Figure E.28: Human-authored Realistic Expectations story: Fairytale domain

In a nice house on a quiet street there lived a girl.

One day the girl tried to ride a bike, but fell off. As a result, the girl was hurt. The girl felt distress that she was hurt. The girl felt shame about trying to ride a bike.

Figure E.29: Human-authored Realistic Expectations story: Family domain

Deep in the jungle there lived a bear and a bee.

One chilly autumn day the bee collected some pollen.

Not long afterwards the bee made honey. As a result, she had honey. The bee felt satisfaction that she had honey. The bee felt pride about making honey.

The day after that, the bear became hungry.

Then the bear took honey from the bee. As a result, the bee didn't have honey anymore. The bee felt anger towards the bear about taking honey because she didn't have honey anymore.

Eventually the bee tried to take honey from the bear, but failed.

Later the bear ate honey. The bee started to hate the bear.

A few days later the bee attacked the bear, but failed. As a result, she was dead. The bee felt distress that she was dead.

Figure E.30: Human-authored Realistic Expectations story: Animals domain

E.2.5 Recklessness

Long ago in a far away land there lived a knight and a princess. The knight loved the princess.

One day the knight picked a poisonous mushroom.

The next day the knight gave the mushroom to the princess.

Some time later the princess ate the mushroom. As a result, she was dead. The knight felt distress that the princess was dead.

Figure E.31: Human-authored Recklessness story: Fairytale domain

In a nice house on a quiet street there lived a girl.

One weekend the girl threw a cricket ball. As a result, the window was broken. The girl felt remorse about throwing a cricket ball because the window was broken.

Figure E.32: Human-authored Recklessness story: Family domain

Deep in the jungle there lived a lion and a squirrel.

One warm afternoon the lion took meat from a snare. As a result, he was caught in a snare. The lion felt distress that he was caught in a snare.

A short time later the squirrel freed the lion from the snare.

The day after that, the lion attacked and killed the squirrel. As a result, the squirrel was dead. The squirrel felt her fears were confirmed that she was dead.

Figure E.33: Human-authored Recklessness story: Animals domain

E.2.6 Reward

Long ago in a far away land there lived a knight and a fairy.

One day the knight picked a rose, but pricked his finger.

The next day the knight gave the rose to the fairy. As a result, the fairy had the rose. The fairy felt joy that she had the rose.

The day after that, the fairy healed the knight. As a result, the knight did not have a pricked finger. The knight felt gratitude towards the fairy about healing him because he did not have a pricked finger.

Figure E.34: Human-authored Reward story: Fairytale domain

In a modern inner-city apartment there lived a father and a mother.

One weekend the father became hungry.

Later that day the mother cooked dinner for the father. As a result, the father had dinner. The father felt gratitude towards the mother about cooking dinner for him because he had dinner.

Some time later the father ate dinner. The father felt joy that he was not hungry.

Later the father picked a flower.

Then the father gave the flower to the mother. As a result, the mother had the flower. The mother felt joy that she had the flower.

Figure E.35: Human-authored Reward story: Family domain

Deep in the jungle there lived a bear and a bird.

One warm afternoon the bear became hungry.

The next day the bear caught the bird. As a result, the bird was not free. The bear felt pity towards the bird because the bird was not free. The bird felt distress that she was not free.

The bird feared that she would die. Some time later the bear let go of the bird. As a result, the bird was free. The bird felt relief that she was alive. The bird felt joy that she was free. The bird felt gratitude towards the bear about letting her go because she was free.

Afterwards the bird collected nuts.

A short while after that the bird gave nuts to the bear. As a result, the bear had nuts. The bear felt gratitude towards the bird about giving nuts to him because he had nuts.

Eventually the bear ate nuts. The bear felt joy and satisfaction that he was not hungry.

Figure E.36: Human-authored Reward story: Animals domain

E.3 Moral-Free Stories

The stories in this section were automatically produced using the MOSS Story Planner, deliberately excluding the emotion patterns comprising the moral rules described in Chapter 5. Apart from this, no additional constraints were placed on the stories. The discourse was generated using each of the MOSS emotion filters, and also with no filter.

E.3.1 Retribution Filter

In a distant kingdom there lived a unicorn and a dwarf. The unicorn hated the dwarf.

One day the dwarf gave the axe to the unicorn. The unicorn started to love the dwarf.

The next day the dwarf picked a poisonous mushroom.

Some time later the dwarf picked a rose. As a result, he had the rose.

Figure E.37: Moral-free story using Retribution filter: Fairytale domain

In a small country town there lived a boy and a mother.

One weekend the mother bought ice-cream.

Later that day the boy messed up his room. As a result, his room was not clean. The mother felt distress that the boy's room was not clean. The mother felt anger towards the boy about messing up his room because the boy's room was not clean.

Figure E.38: Moral-free story using Retribution filter: Family domain

In a large forest there lived a toad and a squirrel.

Early one morning the toad attacked and killed the squirrel. As a result, the squirrel was dead. The squirrel felt distress that she was dead. The squirrel felt anger towards the toad about attacking her because she was dead.

A short time later the toad picked a vine. As a result, he had the vine.

Figure E.39: Moral-free story using Retribution filter: Animals domain

E.3.2 Greed Filter

In a distant kingdom there lived a unicorn.

One day the unicorn became hungry. The unicorn felt distress that he was hungry.

Figure E.40: Moral-free story using Greed filter: Fairytale domain

In a modern inner-city apartment there lived a boy and a girl. The boy loved the girl.

One day the boy played in the mud. As a result, his clothes were not clean.

The next day the boy tried to push the girl over in the mud, but failed. The boy felt shame about trying to push the girl over in the mud.

Soon after that, the boy picked a flower.

Figure E.41: Moral-free story using Greed filter: Family domain

Deep in the jungle there lived a toad and a squirrel. The squirrel hated the toad.

The squirrel hoped that she would have a vine. One chilly autumn day the squirrel picked a vine. As a result, she had the vine. The squirrel felt satisfaction that she had the vine.

The toad hoped that he would have the vine. Later that day the toad tried to take the vine from the squirrel, but failed. The toad felt distress that he didn't have the vine.

The toad hoped that he would have the vine. Some time later the toad picked a vine. As a result, he had the vine. The toad felt satisfaction that he had the vine.

Figure E.42: Moral-free story using Greed filter: Animals domain

E.3.3 Pride Filter

Once upon a time there lived a unicorn and a fairy.

One summer's morning the unicorn picked a rose. As a result, he had the rose. The unicorn felt joy that he had the rose.

A short time later the unicorn gave the rose to the fairy. As a result, the fairy had the rose. The fairy felt joy that she had the rose. The unicorn felt pride about giving the rose to the fairy.

Figure E.43: Moral-free story using Pride filter: Fairytale domain

In a small country town there lived a boy, a mother and a girl. The boy loved the girl.

Early one morning the mother picked a flower. As a result, she had the flower. The mother felt joy that she had the flower.

The next day the boy tried to push the girl over in the mud, but failed. The mother felt joy that the girl's clothes were clean. The girl felt joy that her clothes were clean. The girl felt joy that she was not hurt. The mother and the girl started to hate the boy.

More time passed, and the girl picked a flower. As a result, she had the flower. The girl felt joy that she had the flower.

Figure E.44: Moral-free story using Pride filter: Family domain

Deep in the jungle there lived a toad, a squirrel, a bee, a wolf and a bird. The toad hated the squirrel.

One day the toad tried to catch the bee, but failed. The toad felt distress that he didn't have the bee. The bee felt joy that she was free. The squirrel, the bee and the bird started to hate the toad.

Not long afterwards the wolf collected nuts. As a result, he had nuts. The wolf felt joy that he had nuts. The squirrel, the bee and the bird started to love the wolf.

More time passed, and the toad attacked and killed the squirrel. As a result, the squirrel was dead. The toad felt joy that the squirrel was dead. The squirrel felt distress that she was dead. The toad felt joy that he was alive. The squirrel felt distress that the toad was alive. The bee felt distress that the toad was alive. The wolf felt joy that the toad was alive. The bird felt distress that the toad was alive. The wolf started to love the toad.

A few days later the wolf tried to catch the bird, but failed. The wolf felt distress that he didn't have the bird. The bird felt joy that she was free.

Figure E.45: Moral-free story using Pride filter: Animals domain

E.3.4 Realistic Expectations Filter

Once upon a time there lived a troll, a unicorn, a knight and a princess.

One summer's morning the troll became hungry.

The troll hoped that he would have the sword. Later that day the troll stole the sword from the knight. As a result, he had the sword and the knight didn't have the sword anymore. The unicorn, the knight and the princess started to hate the troll.

Some time later the unicorn tried to kidnap the princess, but failed.

Figure E.46: Moral-free story using Realistic Expectations filter: Fairytale domain

In a modern inner-city apartment there lived a boy and a girl. The boy loved the girl.

One evening the girl messed up the boy's room. As a result, the boy's room was not clean.

Figure E.47: Moral-free story using Realistic Expectations filter: Family domain

In a large forest there lived a toad, a squirrel, a bee and a wolf.

The toad hoped that he would have the bee. One warm afternoon the toad tried to catch the bee, but failed. The toad felt disappointment that he didn't have the bee. The squirrel and the bee started to hate the toad.

The squirrel hoped that she would have a vine. Later that day the squirrel picked a vine. As a result, she had the vine.

The toad hoped that he would have a vine. More time passed, and the toad picked a vine. As a result, he had the vine.

Later the toad attacked and killed the wolf. As a result, the wolf was dead.

Figure E.48: Moral-free story using Realistic Expectations filter: Animals domain

E.3.5 Recklessness Filter

In a distant kingdom there lived a troll, a dragon and a fairy.

One day the dragon gave the treasure to the troll. As a result, he didn't have the treasure anymore and the troll had the treasure. The fairy started to hate the dragon.

Later that day the fairy gave fairydust to the troll.

Figure E.49: Moral-free story using Recklessness filter: Fairytale domain

In a small country town there lived a boy.

One afternoon the boy picked a flower.

Figure E.50: Moral-free story using Recklessness filter: Family domain

In a large forest there lived a toad.

The toad hoped that he would have a vine. Early one morning the toad picked a vine. As a result, he had the vine. The toad felt satisfaction that he had the vine.

Not long afterwards the toad tried to collect pollen, but failed.

More time passed, and the toad tried to collect pollen, but failed.

Figure E.51: Moral-free story using Recklessness filter: Animals domain

E.3.6 Reward Filter

Once upon a time there lived a troll and a dwarf.

One summer's morning the dwarf became hungry.

The next day the troll picked a poisonous mushroom.

Figure E.52: Moral-free story using Reward filter: Fairytale domain

In a nice house on a quiet street there lived a boy, a mother and a girl. The boy loved the girl.

One evening the boy tried to cook breakfast for the girl, but failed and burnt himself. As a result, he was hurt and the dishes were not clean.

Not long afterwards the mother tried to ride a bike, but fell off. As a result, she was hurt.

More time passed, and the girl rode a bike. The girl felt joy that she was not hurt.

Eventually the girl picked a flower. As a result, she had the flower. The girl felt joy that she had the flower.

Figure E.53: Moral-free story using Reward filter: Family domain

In a lush valley there lived a toad and a squirrel. The squirrel hated the toad.

One day the toad tried to catch the squirrel, but failed. The squirrel felt joy that she was free.

Later that day the toad tried to collect pollen, but failed.

Some time later the toad picked a vine. As a result, he had the vine. The toad felt joy that he had the vine.

Figure E.54: Moral-free story using Reward filter: Animals domain

E.3.7 No Filter

In a distant kingdom there lived a unicorn and a dwarf. The unicorn hated the dwarf.

One summer's morning the unicorn became hungry. The unicorn felt distress that he was hungry. The dwarf felt pity towards the unicorn because the unicorn was hungry.

The dwarf hoped that he would have a rose. Later that day the dwarf picked a rose. As a result, he had the rose. The unicorn felt resentment towards dwarf because the dwarf had the rose. The dwarf felt joy and satisfaction that he had the rose.

Figure E.55: Moral-free story with no filter: Fairytale domain

In a small country town there lived a father, a boy and a girl. The boy loved the girl.

One weekend the father bought a toy.

The girl feared that her clothes would not be clean. A short time later the boy tried to push the girl over in the mud, but failed. The father felt joy that the girl's clothes were clean. The boy felt happy for the girl because her clothes were clean. The boy felt happy for the girl because she was not hurt. The father felt reproach towards the boy about trying to push the girl over in the mud. The boy felt shame about trying to push the girl over in the mud. The father started to hate the boy.

Soon after that, the boy picked a flower.

Figure E.56: Moral-free story with no filter: Family domain

In a lush valley there lived a toad and a bee. The toad loved the bee.

One warm afternoon the bee collected some pollen. The bee felt pride about collecting pollen.

The toad hoped that he would have a vine. The next day the toad picked a vine. As a result, he had the vine. The toad felt joy and satisfaction that he had the vine.

The toad feared that the bee would die. The bee feared that she would die. More time passed, and the toad attacked the bee, but failed. As a result, the toad was dead. The toad felt distress that he was dead. The toad felt happy for the bee because he was dead. The bee felt joy that the toad was dead. The bee felt gloating towards the toad because the toad was dead. The toad felt joy and relief that the bee was alive. The toad felt happy for the bee because the bee was alive. The bee felt joy and relief that she was alive. The bee felt resentment towards the toad because she was alive. The bee felt reproach towards the toad about attacking her. The bee started to hate the toad.

Figure E.57: Moral-free story with no filter: Animals domain