Abstract

WARE, STEPHEN G. A Plan-Based Model of Conflict for Narrative Reasoning and Generation. (Under the direction of R. Michael Young.)

Narrative is one of the fundamental cognitive tools that we use to understand the world around us. When interacting with other humans we rely on a shared knowledge of narrative structure, but in order to enable this kind of communication with digital artifacts we must first formalize these narrative conventions. Narratology, computer science, and cognitive science have formed a symbiotic relationship around this endeavor to create computational models of narrative. These models provide us a deeper understanding of story structure and will enable us to create a fundamentally new kind of interactive narrative experience in which the author, the audience, and the machine all participate in the story composition process.

This document presents a computational model of narrative conflict, its empirical evaluation, and its deployment in an interactive narrative experience. Narratologists have described conflict in terms of the difficulties that an intelligent agent encounters while executing a plan to achieve a goal [33, 34]. This definition is inherently plan-based, and has been integrated into an existing model of narrative based on the data structures and algorithms of artificial intelligence planning—the process of constructing a sequence of actions to achieve a goal. The Conflict Partial Order Causal Link (or CPOCL) model of narrative represents the events of a story along with their causal structure and temporal constraints. It extends previous models by representing non-executed actions which describe how an agent intended to complete its plans even if those plans failed, thus enabling an explicit representation of thwarted plans and conflict. The model also includes seven dimensions which can distinguish one conflict from another and provide authors with greater control over story generation: *participants*, *topic*, *duration*, *balance*, *directness*, *intensity*, and *resolution*.

One valuable aspect of plan-based models is that they can be generated and modified automatically. Two story creation methods are discussed: the plan-space CPOCL algorithm that works directly with the rich CPOCL knowledge representation and the state-space Glaive algorithm which is significantly faster. Glaive achieves its speedup by incorporating research from fast forward-chaining state-space heuristic search planning and by using the constraints that a valid narrative plan must obey to calculate a more accurate heuristic. Glaive is fast enough to solve certain non-trivial narrative planning problems in real time.

This computational model of narrative conflict has been evaluated in a series of empirical experiments. The first validates the three discrete dimensions of conflict: *participants*, *topic*, and *duration*. It demonstrates that a human audience recognizes thwarted plans in static text stories in the same places that the CPOCL model defines them to exist. The second experiment

validates the four continuous dimensions—*balance*, *directness*, *intensity*, and *resolution*—by showing that a human audience ranks static text stories in the same order defined by the formulas for those dimensions.

The final experiment is an evaluation of an interactive narrative video game called *The Best Laid Plans*, which uses Glaive to generate a story at run time from atomic actions and without recourse to pre-scripted behaviors or story fragments. In this game, the player first acts out a plan to achieve a goal and then Glaive coordinates all the non-player characters in the game to thwart the player's plan. The game is evaluated relative to two other versions: a control in which the other characters do nothing and a scripted version in which the other characters are controlled by programs written by a human author. Players recognize intentionality and conflict in the stories Glaive produces more so than in the control and comparably to the human scripted version.

In summary, this document describes how a narratological definition of conflict as thwarted plans has been operationalized in plan data structures and incorporated into a narrative planning algorithm. The knowledge representation is rich enough that a human audience recognizes thwarted plans where the model defines them to exist. The algorithm is fast enough to be used in a real time interactive context for certain non-trivial story domains. This work represents one small advancement toward understanding human storytelling and leveraging that understanding in interactive systems.

A Plan-Based Model of Conflict for Narrative Reasoning and Generation

by
Stephen G. Ware

A dissertation submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Computer Science

Raleigh, North Carolina

2014

APPROVED BY:

David L. Roberts	Mark O. Riedl
James Lester	Robert St. Amant

R. Michael Young
Chair of Advisory Committee

Dedication

To my beloved parents, who were the first to tell me stories.

Biography

Stephen George Ware's interest in computer storytelling arose from his love of playing adventure games like *The Secret of Monkey Island* with his father on their IBM personal computer. Stephen taught himself various programming skills during high school in his spare time while learning to create video games of his own. His broader interests in the human mind, computer science, narratology, and interactive storytelling began at Loyola University, New Orleans. In addition to a Bachelor's Degree in Computer Science, he earned a second major in Philosophy with a focus on the philosophy of mind. He played *Dungeons and Dragons* with his friends in his spare time and wrote an undergraduate thesis on the wizard archetype under Dr. William T. Cotton of the English Department. These interests finally converged at North Carolina State University, where Stephen joined Dr. R. Michael Young's Liquid Narrative Group. Stephen's research at NCSU has focused on plan-based models of stories and how the planning algorithms which create those stories can be improved.

Acknowledgements

I wish to thank the many people without whom this work could never have happened. Brent Harrison and David L. Roberts, thank you for your assistance in designing the experiments in Chapter 4 and for your help analyzing the data. Christian Stith and Phillip Wright, thank you for your help in the development of *The Best Laid Plans* game used in Chapter 6. To the United States National Science Foundation, thank you for your financial support through the Human-Centered Computing Program, Grant IIS-0915598: *Plan-Based Models of Narrative Structure for Virtual Environments.*

To my parents, Steve and Helen Ware, you sacrificed so much to give me the financial and educational opportunities to get to graduate school. More importantly, you taught me to work hard, to think hard, to never give up, and to believe in my own self-worth. You taught me the really important things that get me out of bed in the morning and so much more than you give yourselves credit for.

The person who made the greatest impact on my scientific career is Dr. R. Michael Young, my advisor and mentor. Michael, you taught me to be a scientist and how to thrive in academia through your advice and your example. Most importantly, you taught me to pursue with great passion the problems that interest me and to pursue them *because* they are interesting and because of what they can tell us about the human mind. In the day to day grind, it is easy to forget the long term goals or to settle for those results which are easy to obtain. You inspired me never to lose sight of the noble purpose of scientific research.

I also wish to thank my committee for their support and guidance. Dr. David L. Roberts, you taught me experimental design and honored me by making me feel like a colleague. Dr. Mark O. Riedl, you paved the way for this work through your own research and your encouragement when I was just beginning. Dr. James Lester, you lent me your encyclopedic knowledge of our field and showed me by example a truly exemplary scientific career. Dr. Robert St. Amant, you taught me to love artificial intelligence, and you showed me that a planner can be both sophisticated and elegant.

All my teachers have played a part in my success, but some stand out in my memory. Ann Stevens and Lyla Marinelli, you were the first to make me love science. Katie King, you taught me to keep working on something until I'm happy with the result. Dr. David Brooks, you taught me most of what I know about pedagogy. Dr. William T. Cotton, you whet my appetite for graduate-level research. Dr. Carla Savage and Dr. Nagiza Samatova, you initiated me into the central mysteries of computer science.

I have learned as much from my peers as from my teachers. Gustavo Heudebert, you taught me that intelligence is laudable, not embarrassing. Andy Morgan and Elliot Sanchez, you made

me love philosophy and the human mind in the long debates that made our college nights so memorable. Brent Harrison, you single-handedly taught me more about computer science and experimental design than anyone else. I would not have survived my first year of graduate school without you. Rogelio E. Cardona Rivera, you continuously inspire me with the scope of your knowledge and your passion for research. Justus Robertson, you were the other half of many of the most interesting conversations I had in graduate school. You have all impacted me with your knowledge, but more importantly you have been the best friends I have ever had the privilege to know. I am also deeply indebted and forever grateful to all the member of the Liquid Narrative Group, past and present.

Finally, this work literally would not have been possible without the love and support of my wife, Leslie Forrest Ware. You engaged me in thoughtful discussions about psychology and philosophy. You applauded me when I was successful. You encouraged me when I felt hopeless. You picked up my slack at home while I was working 80 hour weeks and never complained. I would have quit several times if you had not been there. At a time when most relationships are strained, ours grew stronger thanks to your understanding and patience. Thank you for being so supportive and so encouraging, and know that I love you deeply. Thank you all, from the bottom of my heart.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Narrative is one of the fundamental cognitive tools that we use to understand the world around us. We choose the content and structure of the narratives we tell to others based on our communicative goals, and in turn we have expectations about the content and structure of the narratives we are told. When interacting with other humans we rely on a shared knowledge of these protocols, but in order to enable this kind of communication with digital artifacts we must first formalize these narrative conventions.

Narratology, computer science, and cognitive science have formed a symbiotic relationship around this endeavor to create computational models of narrative. Narratology observes normative features of human storytelling which can be formalized and implemented by computer science and used by cognitive science to investigate the role those norms play in human thought. This process is not a pipeline but a cycle—each part making contributions to and prompting further investigation by the others—by which we slowly illuminate the complex mental machinery of human storytelling. The process is inherently valuable because of what we learn about our minds, our stories, and our cultures. It is also valuable because it allows us to construct complex artificial machinery which can communicate with a human audience using the language of narrative.

Machines have long played a part in storytelling, primarily as the medium through which authors deliver stories to their audiences. Computers provide a means of making these experiences interactive, but as far as narrative is concerned the computer is being treated as little more than the printing press. A human author decides on a narrative at design time, encodes that single experience in great detail, and then the audience consumes it. First authoring, then experiencing. This has been generalized, by the video game industry for example, to a team of authors who provide a homogenous set of narratives that the audience can choose from, but a fundamentally different kind of interactivity is possible. By encoding human understanding into computational models of narrative we allow authorship to occur simultaneously with ex-

periencing. This provides the audience a tailored, context-sensitive experience and expands the expressive capabilities of the author. It is also a more intimate kind of communication. Rather than sharing a single narrative, the author shares a part of his mind, a part that embodies thousands of years of collected knowledge on human storytelling. So the end goal of research in computational models of narrative is not to replace the author with a machine, as some have claimed, but to magnify the participation of the the author, the audience, and the machine in a collaborative and fundamentally human storytelling process.

1.1 Plan-Based Models of Narrative

Narratives are often described as a sequence of causally and temporally related events constructed by an author for some communicative purpose. It is no surprise, then, that the artificial intelligence paradigm of planning is a popular source of data structures and algorithms for reasoning about stories. Planning is the computational process of constructing a sequence of actions which can be taken by one or more agents to achieve a goal. Plans and the associated planners which produce them provide a computational foundation on which models of narrative can be built and evaluated.

Plans have proven useful for representing both the *fabula* and *syuzhet* of a narrative. The *fabula* is the complete chronology of events in the story world seen from a god's eye view, where plans represent the actions each agent takes to achieve its goals. The *syuzhet* is the telling of the story, a selection of information from the fabula, where plans represent communicative acts taken by the author for some rhetorical purpose. It is often difficult to separate these two layers because they mutually constrain one another, but many computational models focus primarily on one or the other.

Another benefit of plan-based models is that they are generative—that is, they can be used not only to represent and analyze a sequence of events but also to create such a sequence. When evaluating a generative model, it is important to consider both the richness of the knowledge representation and the speed of the algorithms used to construct it. Both are important if the model is to be used in an interactive context, but many plan-based models of narrative have focused on one at the expense of the other.

1.2 Thesis

This document tells the story of a computational model of narrative conflict, the algorithms used to generate and adapt it, and its empirical evaluation. Conflict is an essential element of interesting stories. It structures the discourse, motivates the story's action, and engages the audience. Conflict can be operationalized in terms of the difficulties that an intentional agent

experiences while carrying out a plan to achieve a goal. This model is inherently plan-based. It reasons about possible worlds where an agent's plan succeeds or fails and how that plan can be thwarted by other agents or the environment. Specifically, I say that conflict occurs when a character forms a plan that is thwarted by another event in the story, or would have been thwarted if the event had succeeded. The thwarting event can be part of another character's plan (external conflict), part of the same character's plan for a different goal (internal conflict), or an accident or force of nature (environmental conflict).

The thesis I advance in this document is as follows: An operationalization of narrative conflict as thwarted plans can be incorporated into a computational model of story fabula which is representationally rich enough that a human audience will perceive conflict just when the model defines it to exist; this model can be generated and adapted by an algorithm which is efficient enough to create stories with recognizable conflicts at run time in an interactive virtual environment.

1.3 Roadmap

This document presents the development of this computational model of narrative conflict. Chapter 2 begins with a narratological discussion of why conflict is an essential element of stories. It also surveys how other computational systems have dealt with conflict and presents background on the planning systems used to create this model.

Chapter 3 describes the knowledge representation in formal detail. It presents the Partial Order Causal Link planning framework on which the model is built, and describes how that framework was extended first by Riedl and Young [62] to include intentionality and then by me to include conflict by representing actions which an agent intended to take but failed to execute. The chapter concludes with a discussion of how the Conflict Partial Order Causal Link (or CPOCL) representation constrains the expressiveness of plans to be more consistent with the narrative expectations of a human audience. Chapter 4 presents an evaluation of the CPOCL representation through two human subjects experiments. These experiments demonstrate that the CPOCL model can be used to predict how humans reason about conflict in static textual narratives.

Chapter 5 then moves to the problem of generating stories with conflict. It presents two different types of story planning algorithms. The CPOCL algorithm operates directly on the CPOCL knowledge representation, but its applicability is limited by its speed. The Glaive algorithm incorporates advances in state-space heuristic search planning to achieve a significant speedup. It incorporates the specific constraints of narrative planning into its heuristic and enables real time narrative planning for some non-trivial problems. Chapter 6 gives a computational evaluation of Glaive on a suite of narrative planning problems. It also describes how

Glaive has been used to control the plot of an interactive narrative adventure game called *The Best Laid Plans* while the game is being played. A final human subjects experiment demonstrates that players recognize thwarted plans in the plots generated by Glaive more so than in a control and comparably to plots specified by a human author at design time. Chapter 7 concludes the document and discusses the ample opportunities for further development.

Considered as a whole, this body of work tells the story of how the important narrative phenomena of conflict was recognized, operationalized, implemented, and empirically evaluated as a computational model which is representationally rich and efficiently generated. It represents one small advancement in the larger research program of formalizing the collective knowledge of human storytelling into a framework which will give rise to a fundamentally new kind of narrative, one in which the creative power is distributed evenly between the author, the audience, and the machine.

Chapter 2

Related Work

This work is inherently interdisciplinary, so relevant material from several fields is surveyed below. This chapter begins with a narratological and psychological discussion of why conflict is important to narrative and what rhetorical purpose it serves. I then discuss how previous narrative generation systems have dealt with conflict and describe other systems which have used quantifiable metrics to evaluate the quality of their stories. I also survey relevant technology from the classical AI planning community and how it is related to my work. I conclude by situating this work in the field of computational models of narrative.

2.1 Conflict in Fiction

Conflict is a key component of interesting stories. The *Routledge Encyclopedia of Narrative Theory* [34], *A Dictionary of Narratology* [59], and the *The Cambridge Introduction to Narrative* [1] provide subtly different definitions, but they all agree that it is essential. Abbott notes that it "is so often the life of the narrative" [1]. Herman goes so far as to declare it a "minimal condition for narrative" [34], while Brooks and Warren even tell us that "story means conflict" [13].

Other scholars analyzing computational storytelling have come to similar conclusions about the centrality of conflict [51, 65, 70, 67, 22, 19, 6, 50]. Szilas, creator of the IDtension narrative system, declares that "the notion of conflict is the core of the drama" [71]. Crawford argues that a narrative game is impossible without a thorough analysis of conflict [22]. Screenwriting handbooks also highlight the importance of a story's central struggle [24, 78, 36, 17, 74].

This universal agreement on the importance of conflict throws into sharp relief the lack of literature written on the subject. Anthropologists and sociologists have studied its historical importance, but their analysis provides little insight for generating fictional stories. Narratologists often refer to it, but always in an informal manner. This problem seems to stretch back

even to antiquity:

> To me, the amazing thing about [Aristotle's] *Poetics* is that for all the aspects of good screenwriting the *Poetics* addresses, it does not address everything directly. For example, take conflict. Everybody knows conflict is important, and its probably the dominant mode of action. Greek tragedy was loaded with conflict, so it's safe to say Aristotle assumed (as we can about movie storytelling) that conflict is a given. [74]

Narrative conflict seems to be so ingrained in our social and historical consciousness that critics do not bother to explain it. Unfortunately, machines have no such consciousness to fall back on, so a formal model needs to be distilled. Developing one such formal model is the first step for my work, and the next is to integrate that model into the story generation process.

Conflict serves at least three important purposes in a story. Firstly, it structures the discourse into a rising action that culminates in a climax (the resolution of a story's central conflict) and ends with falling action [34]. Chapters and scenes are often organized around a conflict so that its resolution (or lack of resolution) creates a meaningful segmentation of the discourse [1]. Secondly, conflict motivates the action. Stories are commonly described as beginning in a state of equilibrium which is destroyed by some conflict [75]. The characters in the story are motivated by this distrubance to form a plan—that is, to plot—a way of restoring that equilibrium. Thirdly, psychologists studying narrative claim that conflict engages the audience. It causes the audience to ask questions and form expectations about the outcome, which propels them forward through the plot [28, 1]. Readers also experience *anomalous suspense*, which is a form of engagement that occurs despite knowing the outcome of the story [28, 17, 1].

Many narratological definitions of conflict focus on the central role of actions, plans, and the thwarting of plans [1, 24, 23, 34]. One such plan-based definition of conflict is given by Herman:

> Yet a minimal condition for narrative can be defined as the thwarting of intended actions by unplanned, sometimes unplannable, events, which may or may not be the effect of other participants' intended actions. This is another way of expressing the intuition that stories prototypically involve *conflict*. [33]

Egri [24] and Dibell [23] distinguish conflict from the more general notion of *tension* by specifying that conflict is a property of thwarted intentional actions (i.e. plans). Tension is the general sense of opposition between forces, so while conflict delivers tension it is not the only source. In terms of the Belief Desire Intention (BDI) framework [11], conflict arises from intentions, not desires. Here my work differs from others like Szilas [71], who defines conflict as opposition between a character's actions and moral principles—something I would label as tension. The

definition I have chosen to operationalize is this: Conflict occurs when a character forms a plan that is thwarted by another event in the story, or would have been thwarted if the event had succeeded. The thwarting event can be part of another character's plan (external conflict), part of the same character's plan for a different goal (internal conflict), or an accident or force of nature (environmental conflict).

2.2 Conflict in Interactive Narrative and Story Generation

Non-interactive narratives, such as oral tradition, books, and films, rely on a human author (or authors) to design story conflicts appropriately for the genre and audience. Interactive narratives must somehow accommodate the intervention of the audience while still maintaining certain narrative properties that are important to the author. Riedl and Bulitko [63] present a framework for classifying interactive narrative systems based on design philosophy, and I will use this framework for a brief survey of how different kinds of systems have handled the problem of conflict.

Most interactive stories designed for commercial systems, such as Choose Your Own Adventure books and most narrative-focused video games, are classified as *Manually Authored Stories*. All possible branches of the story are created by human authors at design time to ensure that the author's vision is maintained no matter how the narrative is experienced. These systems rely on the author's understanding of conflict and story structure. They have the largest authorial burden and are least able to adapt a story to an individual audience.

Hybrid Systems use a combination of human authorship and an experience manager to create an interactive story. The human author generally provides story content in the form of chapters or scenes along with a story graph that formalizes how the audience may move from one plot element to the next. These systems reduce authorial burden and allow a wider variety of stories to arise, but the experience manager is responsible for maintaining the author's vision, and thus it needs some computational model of the narrative phenomena that the author wishes to create. Roberts and Isbell provide a survey of experience management techniques [64]. Notable systems that explicitly address the problem of conflict include Universe [41] and Mexica [57]. These systems combine pre-scripted plot fragments according to plot grammars to produce whole stories. Both of these systems rely on a human author to encode short-term conflicts into the individual scenes that make up the plot graph [41] and long-term conflicts (e.g. a plot resembling Freytag's Triangle) into the story graph [57]. These systems have many interesting design features, but they still rely on a human author to create conflict, so the general problem of building plot fragments that contain conflict from atomic actions remains unsolved.

On the opposite end of this spectrum lie *Automatically Generated Stories*. The systems that produce them use finer grained plot elements, usually corresponding to individual character

or story world actions. The information provided by the author is more general—usually a domain theory that describes when actions may occur and how they change the world. These systems can produce the widest range of stories, but their experience managers must assume more responsibility for the story's quality. More general computational models of narrative are needed to ensure that the story generated meets the author's requirements. The computational model of conflict I describe in this document is intended for use in these kinds of systems. Gervás [29] provides a survey of automatic story generators as creative systems. Below I will describe some of the notable systems which have explicitly addressed conflict.

Carbonell [15] described a rule-based system in which agents with conflicting goals reason about how they can interfere with and overcome the plans of their competitors. This kind of reasoning and counter-planning is of interest to narrative generation, but Carbonell's model is described at a high level of abstraction and he does not present a formal algorithm for generating conflicting plans.

Smith [68] generated conflict in stories using an adversarial game-playing algorithm. This approach oversimplifies the antagonist; which is not simply a malevolent force to make trouble for the protagonist, but a character with its own goals that should thwart the protagonist only when it is properly motivated.

Barber and Kudenko [6] create dramatic tension in their GADIN system with *dilemmas*—decisions the user must make which will negatively affect at least one character. When the player feels invested in those characters, these conflicts will engage the audience. GADIN detects when these dilemmas are applicable to the story, applies them immediately, and then elicits a resolution from the player. GADIN does not attempt to reason about the role of conflict in the future of the story, making it difficult to model the thematic and extended conflicts that provide important macro-structural narrative features.

Teaching conflict resolution strategies is a central element of the *Fearnot!* [2] and SIREN [20] narrative systems. Their model of conflict is based on organizational psychology research, as opposed to my narratology-inspired model, so they tend to emphasize different aspects. For example, conflict resolution games designed to teach real-world skills tend to focus on cooperation and compromise, whereas fictional conflicts are more often resolved by competition or trickery.

Gratch and Marsella [30, 44] analyze how the plans of multiple agents conflict with one another using the same causal threats on which my model is based. Their work is focused on producing appropriate emotional reactions for agents who find themselves in conflict. They also use domain knowledge such as the likelihood of the plan's success and the utility of the agents to calculate the intensity of the emotions. This is cloesly related to the kind of metrics I use to describe conflict, so this work is highly relevant and compatible with mine. My work differs from theirs in a few key aspects. Firstly, my work focuses not only on a model of conflict

but also in the planning process to produce those models. Secondly, my work has a specific narrative generation focus and is validated in a narrative context.

2.3 Story Metrics

Many researchers have explored the use of quantifiable metrics to describe properties of stories. Yannakakis [83] provides a survey that measures human perceptions like *fun* and *flow* in the context of video games. Peinado and Gervs [55] collected four metrics from human readers evaluating the quality of stories produced by their ProtoPropp system: *linguistic quality*, *coherence*, *interest*, and *originality*. My approach to the seven dimensions of narrative conflict differs from these because I attempt to measure story properties apart from their effects on the reader. The dimensions of conflict answer *who? what? when?* and *how?* questions and are designed so that readers can agree on their values even when they disagree on more subjective aspects such as how fun or interesting a conflict is.

At least four story systems have reasoned about conflict quantitatively. IDtension [71] assigns each action a "conflict value" for the degree to which a character is forced to act against its moral principles. MEXICA [57] measures the tension a reader perceives at each world state, allowing the system to craft a pattern of rising and falling action. Zambetta, Nash, and Smith [87] specify the ideal amount of conflict in a story as a system of differential equations that simulate an arms race scenario. These approaches are helpful as high-level control for the pace of a story, but do not reason about the individual motivations of the participants.

The AI Director of the game series *Left 4 Dead* [10] moderates the intensity of conflicts by controlling the number and frequency of enemies, distribution of power-ups, and geography of levels. The director monitors metrics like the player's health and accuracy to measure stress level and create a series of peaks and valleys in intensity that are similar to popular narratives in its domain. We hope to provide a model which can generalize to many domains, but we plan to use our metrics similarly to these systems, which produce rising and falling story arcs.

2.4 AI Planning

The artificial intelligence subfield of *planning* is devoted to reasoning about action and change. A plan is a sequence of actions that describes how the agents in a formally-defined world interact with places and things to transition from some initial world state to a goal state. Each action in a plan has preconditions which must be satisfied before it can be taken and a set of effects which become true after the action has been taken. A planning domain describes a set of action templates which generally correspond to verbs, e.g. "Someone stole something from someone else." A planning problem describes the nouns of the world. It also describes

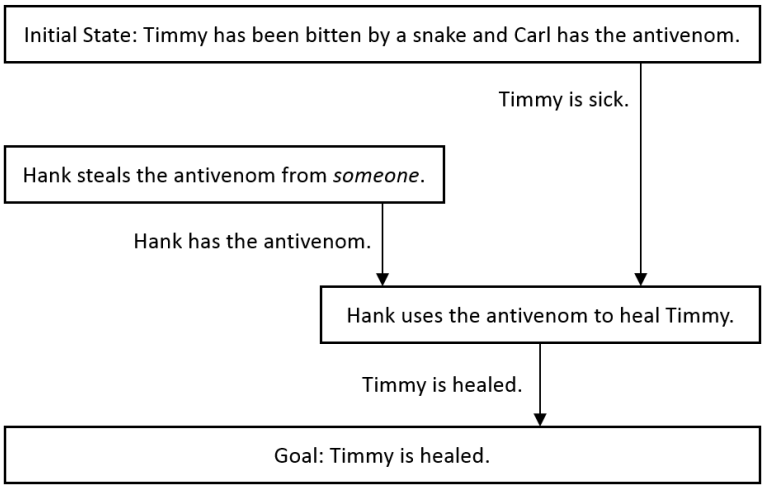


Figure 2.1: An example of an incomplete Partial Order Causal Link plan. Edges represent causal links that describe how the effect of an earlier step satisfies the precondition of a later step. During the plan construction process, this plan would be annotated with a set of flaws that indicate how it is incomplete. These flaws would be fixed one by one until a flawless plan is produced.

what state that world is in initially and specifies a goal that the world should be by the end. The purpose of a planning algorithm, or planner for short, is to solve a planning problem by finding a sequence of actions which achieves the goal. These actions are drawn from the domain but must be fully ground, e.g. "Hank stole the medicine from Carl." Story generation is the process of findings a sequence of actions involving characters, props, and locations, so planning-based methods have proven a popular foundation for many narrative systems (selected examples include [3, 18, 43, 58, 61, 62, 85, 86]).

Early research in Partial Order Causal Link (POCL) planning [81] has proven especially useful for story modeling because a POCL plan explicitly represents the temporal and causal constraints that exist between story actions. Story actions are partially ordered by a set of constraints that specify "step x occurs before step y." This means that a single plan represents many possible valid sequences that the steps might be executed in. They also contain a first class representation of causality in the form of causal links. If the steps in a POCL plan form a graph, then a causal link is a directed edge which indicates how the effect of an earlier step satisfies the precondition of a later step. Young [84] points out that these representation of time and causality are essential elements of a story's *fabula* as described by narratologists [5]. Human audiences reason about causal chains of events when experiencing a narrative, so the structures defined by causal links are especially important [76, 77]. POCL planning has been used to model narrative phenomena like character believability [62], suspense [4], student learning in a

narrative teaching environment [72], and the salience of past actions in a story discourse [16]. Many early planning systems, including POCL planners, contained some model of conflict and could reason about these conflicts specifically to remove them from plans. In contrast, my work leverages this representation to *retain* certain conflicts without sacrificing the plan's soundness.

POCL planning is a kind of refinement search through the space of plans [37]. A partial plan is constructed and annotated with flaws describing how it is incomplete. Search proceeds by iteratively choosing a flaw and repairing it until a flawless (and thus complete) plan is found. The least-commitment nature of POCL search allows the planner to express notions like "*someone* steals the medicine," where the variable *someone* is not bound to an actual character until it is needed. Partially-ordered steps and least-commitment variable bindings allow the planner many degrees of freedom during composition, but story evaluation is often made more difficult as a result. The rhetorical effect of "*someone* steals the medicine" can vary widely depending on when that step occurs and who *someone* is.

Though POCL planning is widely used in narrative modeling because of its helpful knowledge representation, such planners are often too slow to generate stories of significant length, and thus their practicality is limited, especially for real time systems. The last two decades of classical planning research have been dominated by forward-chaining state-space heuristic search. The Heuristic Search Planner, or HSP [9], begins at the initial state and moves through the space of states toward a goal state by applying fully ground steps whose preconditions are satisfied by the current state. The HSP heuristic is a domain-independent way to estimate any state's distance to a goal state. Variants of HSP with improved heuristics and other search enhancements have won the classical deterministic track at the biannual International Planning Competition every year since its inception in 1998, with one exception in 2002 [21]. Forward-chaining state-space planners construct totally ordered plans with no unbound variables. They lack the flexibility of POCL planners, but analyzing the quality of the unfinished story may be easier because all information is known about the story up to the current time (though the ending is still unknown, which can have a significant impact on the meaning of events at the beginning).

The state-space planning community has devoted much effort to the development of accurate heuristics which can be efficiently calculated—indeed, the names of many state-space planning systems are synonymous with the heuristics they use. Notable milestones include Bonet's HSP [9], Hoffmann and Nebel's Fast-Forward [35], and Helmert's Fast-Downward [32] systems, each of which defined a more accurate heuristic that bears the same name as the planning system. In Chapter 5 I will discuss how the constraints of narrative planning problems—specifically the constraint that actions must appear to be goal directed for the characters who take them—can be leveraged to improve these heuristics and thus the search process.

I will present two planning algorithms in Chapter 5. The first is the Conflict Partial Order

Causal Link (or CPOCL) algorithm, which was developed as a means to generate stories with the rich knowledge representation of POCL plans. The second is a forward-chaining state-space planner named Glaive, which was developed to capitalize on the speed of state-space planners so that it could be used to control the narrative of a real-time interactive system. Both algorithms have different strengths and weaknesses, so both are presented to provide a more complete understanding of how a complex narrative phenomena such as conflict can be integrated into a plan-based model.

2.5 Context of this Work

I will now attempt to contextualize my work in the landscape of other computational models of narrative and narrative generation systems.

- This work is founded primarily on narratology, meaning that it attempts to model narrative pheomena which have more or less universal applicability. This is opposed to systems (including many drama management systems [64]) which focus on modeling the preferences of an individual user and tailoring a narrative to that user. Obviously this is not a black and white distinction, because one advantage of any generative model is its ability to adapt to an individual player and most drama management systems make use of some general storytelling guidelines. However, my model does not construct a detailed player model to detirmine what kind of story to build; rather it constructs stories according to general principles described by narratology.

- I use planning techniques to construct stories from atomic actions. This sets my work apart from systems which rely on preauthored stories, story fragments (e.g. [41]), story grammars (e.g. [12]), and scripts (e.g. [66]). Building stories from atomic pieces allows a high degree of control over the details of the story but is very computationally expensive.

- I focus on extensive reasoning about the future to create a whole story, which differentiates this work from reactive planning architectures (e.g. [47]), which wait for certain situations to arise and then respond with pre-programmed scripts.

- My planners coordinate the resources in the story toward the author's goals. This places my work primarily in what Mateas [46] and Riedl and Bulitko [63] call the *strong story* camp. This is opposed to the *strong autonomy* camp (somtimes called the *emergent narrative* camp), in which the primary focus is the autonomy of the individual characters. However, like most systems, I attempt to strike a balance between these two positions. I leverage the same technique described by Rield and Young [62], where the algorithm

constructs a plan for the author's goal, but only out of steps which can be explained in terms of the goals of the individual agents who take them.

Chapter 3

The CPOCL Model of Conflict

Most research in planning to date has been for the purposes of efficiently coordinating real world resources for a single agent or a set of cooperating agents that all pursue the same goal. As we adapt models of planning to represent stories, we must reframe the problem in terms that distinguish between the author's goals and those of the individual agents, who sometimes cooperate and sometimes compete. Agents must be seen to pursue their own goals in order to seem believable, but the planner as a whole should still direct the story toward the author's desired outcome [62]. Since conflict is an essential element of interesting stories, we also need a means of modeling failed and thwarted plans. We say that conflict occurs when a character forms a plan that is thwarted by another event in the story, or would have been thwarted if the event had succeeded. This requires counterfactual reasoning—some way of representing other possible worlds. It also requires an explicit representation of how and why a plan fails.

This chapter describes a representation for stories that embodies this thwarted plans definition of conflict. It begins with the Partial Order Causal Link (or POCL) representation of plans. The POCL model is a rich data structure for representing a sequence of actions taken to achieve a goal. The events are partially ordered with respect to one another, which means that there are potentially many valid orders that the events could occur in. Causal links explain how the facts established by earlier events satisfy the preconditions of later events. The structures of a POCL plan provide a way to ensure that a plan is valid—that is, guaranteed to achieve a goal.

A causal link corresponds to a protected interval of time in the story during which a fact must remain true because it is needed. For example, it might represent the idea that, "Earlier, Hank robbed the shopkeeper in order to be in possession of the medicine. Later, Hank used that medicine to heal his son Timmy." When one of these protected intervals is violated, it means that a plan might not be able to achieve its goal. If some other character takes the medicine away from Hank before he is able to use it, the healing event can never occur. The

POCL planning paradigm provides a formal definition for this, called a threatened causal link, specifically so that these violations can be avoided.

My Conflict Partial Order Causal Link (or CPOCL) model leverages this idea of threatened causal links to represent conflict. Key to the thwarted plans definition is the idea of a character forming a plan which fails because some obstacle has made it impossible. But before this definition can be fully realized, the POCL model needs to be extended to include a representation of intentional actions.

Riedl and Young [62] describe a framework for intentional planning which organizes a POCL plan's events into groups based on character goals. These groups are called intention frames and have four important parts: a character who forms a plan, the goal that character is trying to achieve, something which motivated the character to achieve that goal, and a set of actions taken by the character in service of the goal. An example in text may be helpful here: "When young Timmy was bitten by a snake, his father Hank was motivated to heal him. In order to heal Timmy, Hank robbed the general store to get some medicine and then used that medicine to heal Timmy." Intentional POCL (or IPOCL) plans are one large plan that is organized into many smaller subplans for each characters that wants to achieve some goal.

This paradigm of intentional planning places additional constraints on which plans can be considered valid. Not only must the goal of the plan be achieved at the end, but all the actions taken by characters must be explained in terms of their personal goals. These constraints ensure that IPOCL plans more closely meet the expectations of the audience when told as a narrative. However, this representation has a significant limitation: every subplan must succeed. In other words, when a character adopts a goal, that character must either succeed in achieving that goal or never form a subplan at all.

The CPOCL model picks up where IPOCL left off. It allows some events in the story to be marked as non-executed events (and assumes that all others are executed). These non-executed events are actions which some character intended to take but failed to take because they were impossible for some reason. The uses of non-executed steps relaxes some of the constraints that IPOCL places on valid plans. In a CPOCL plan, it is acceptable for some causal links to be threatened as long as the later step is non-executed. Because a non-executed step never actually takes place, it does not matter if the causal links that explain the event are violated. Indeed, these threatened causal links allow the model to represent conflict as a data structure. Non-executed steps allow CPOCL to represent failed and partially executed subplans. It also allows a single plan to represent many possible worlds. A failed plan represents another way that the story might have gone—an alternate world. Reasoning about these alternate worlds gives the model additional reasoning power.

The definition of conflict as thwarted plans is a very broad one and is meant to cover a very broadly-defined narrative phenomenon. In order to give authors more control over the

kinds of stories which are generated, I also operationalize seven dimensions of narrative conflict from narratology which can be used to distinguish one conflict from another. Three of these dimensions can be directly observed in a CPOCL plan: *participants* (who is in conflict), *topic* (what fact they in conflict over), and *duration* (how long the conflict lasts). The other four are continuous values that require some additional domain information that is commonly available in narrative virtual environments. Given some way to measure a character's utility and the likelihood of a plan's success, I define *balance* (how closely matched the participants are), *directness* (how close the participants are to one another), *intensity* (how much is at stake), and *resolution* (the result).

This section presents only the CPOCL model itself—that is, only the data structures used to represent stories. A discussion of the planning process is reserved for Chapter 5.

3.1 Example Story World

For the purposes of illustrating various narrative planning concepts, I have provided a small example domain (Figure 3.1) and problem (Figure 3.2). The syntax is an extension of the Planning Domain Definition Language, or PDDL, a standard in the classical planning community. This sample setting is intentionally very simple; it is designed to be just large enough to demonstrate each of the features discussed in this section. The story world is set in the old American West. It defines a space of short stories about how a young boy named Timmy is saved (or not saved) from a deadly snakebite. His father, Hank, can save him by stealing antivenom from Carl, the town shopkeeper, but this theft cause sheriff William to hunt down Hank and dispense frontier justice. A detailed explanation of how to read the domain and problem is given below.

Figure 3.1 is the planning domain, which describes all the possible actions that can take place. It begins with a list of `:requirements` that allows a planner to recognize whether or not it implements enough features to reason about this domain. The `:types` define an ontology of things, with `object` being the default supertype of all things. This domain defines three high-level types: `status`, `person`, and `item`. A `sheriff` is a specific type of `person`. A `medicine` is a specific type of `item`. This domain defines three `:constants` which must be part of every story world. These three constants are the three kinds of status that a person can have: `Healthy`, `Sick`, and `Dead`.

The list of `:predicates` defines which kind of relationships can exist in this story world. The `status` predicate indicates that a person has one of the three possible statuses. The `has` predicate indicates that a person is currently in possession of an item. The `owns` predicate indicates that a person is the rightful owner of an item (independent of whether or not they have that item). The `armed` predicate indicates that a person has a weapon. The `parent`

```
(define (domain western)
  (:requirements :strips :equality :negative-preconditions :typing :intentionality)
  (:types status - object
          sheriff - person
          medicine - item)
  (:constants Healthy Sick Dead - status)
  (:predicates (status ?person - person ?status - status)
               (has ?person - person ?item - item)
               (owns ?person - person ?item - item)
               (armed ?person - person)
               (parent ?parent - person ?child - person))
  ;;; Action: A person gets bitten by a snake.
  (:action snakebite
    :parameters   (?victim - person)
    :precondition (status ?victim Healthy)
    :effect       (and (not (status ?victim Healthy))
                       (status ?victim Sick)
                       (intends ?victim (status ?victim Healthy))))
  ;;; Action: A sick person dies.
  (:action die
    :parameters   (?victim - person)
    :precondition (status ?victim Sick)
    :effect       (and (not (status ?victim Sick))
                       (status ?victim Dead)))
  ;;; Action: One person steals an item from another, angering the sheriff.
  (:action steal
    :parameters   (?thief - person ?item - item ?owner - person ?sheriff - sheriff)
    :precondition (and (not (= ?thief ?owner))
                       (status ?thief Healthy)
                       (owns ?owner ?item)
                       (has ?owner ?item))
    :effect       (and (not (has ?owner ?item))
                       (has ?thief ?item)
                       (intends ?sheriff (status ?thief Dead)))
    :agents       (?thief))
  ;;; Action: One person uses medicine to heal a sick person.
  (:action heal
    :parameters   (?healer - person ?medicine - medicine ?patient - person)
    :precondition (and (not (status ?healer Dead))
                       (has ?healer ?medicine)
                       (status ?patient Sick))
    :effect       (and (not (status ?patient Sick))
                       (status ?patient Healthy)
                       (not (has ?healer ?medicine)))
    :agents       (?healer ?patient))
  ;;; Action: One person shoots and kills another.
  (:action shoot
    :parameters   (?killer - person ?victim - person)
    :precondition (and (status ?killer Healthy)
                       (armed ?killer))
    :effect       (and (not (status ?victim Healthy))
                       (not (status ?victim Sick))
                       (status ?victim Dead))
    :agents       (?killer)))
```

Figure 3.1: A simple narrative planning domain for stories in the old American West in PDDL (Planning Domain Definition Language) syntax.

```
(define (problem save-timmy)
  (:domain western)
  (:objects Timmy Hank Carl - person
            William - sheriff
            Antivenom - medicine)
  (:init ;;; Timmy is dying and wants to get better.
         (status Timmy Sick) (intends Timmy (status Timmy Healthy))
         ;;; Hank is Timmy's father. He is armed.
         (parent Hank Timmy) (armed Hank) (status Hank Healthy) (intends Hank (status Hank Healthy))
         ;;; Hank also wants his son to get better.
         (intends Hank (status Timmy Healthy))
         ;;; Carl is the shopkeeper. He has antivenom which can save Timmy, but wants to keep it.
         (has Carl Antivenom) (owns Carl Antivenom) (intends Carl (has Carl Antivenom))
         (status Carl Healthy) (intends Carl (status Carl Healthy))
         ;;; William is the sheriff. He is also armed.
         (armed William) (status William Healthy) (intends William (status William Healthy)))
  (:goal ;;; The author's goal is for Timmy to be sick no longer.
         (not (status Timmy Sick))))
```

Figure 3.2: A simple narrative planning problem for the Western domain in Figure 3.1 in PDDL (Planning Domain Definition Language) syntax.

Plan A	
`(die Timmy)`	Timmy died.
Plan B	
`(heal Carl Timmy)`	Carl the shopkeeper healed Timmy using his medicine.
Plan C	
`(shoot Hank Timmy)`	Hank shot his son Timmy.
Plan D	
`(steal Hank Antivenom Carl William)`	Hank stole antivenom from the shop, which angered sheriff William.
`(heal Hank Antivenom Timmy)`	Hank healed his son Timmy using the stolen antivenom.
Plan E	
`(steal Hank Antivenom Carl William)`	Hank stole antivenom from the shop, which angered sheriff William.
`(heal Hank Antivenom Timmy)`	Hank healed his son Timmy using the stolen antivenom.
`(shoot William Hank)`	Sheriff William shot Hank for his crime.
Plan F	
`(steal Hank Antivenom Carl William)`	Hank stole antivenom from the shop, which angered sheriff William.
`(shoot William Hank)`	Sheriff William shot Hank for his crime.
`<heal Hank Antivenom Timmy>`	Hank intended to heal his son Timmy using the stolen antivenom.
`(die Timmy)`	Timmy died.
Plan G	
`(steal Hank Antivenom Carl William)`	Hank stole antivenom from the shop, which angered sheriff William.
`<shoot William Hank>`	Sheriff William intended to shoot Hank for his crime.
`(snakebite Hank)`	Hank got bitten by a snake.
`<heal Hank Antivenom Hank>`	Hank intended to heal himself using the stolen antivenom.
`(heal Hank Antivenom Timmy)`	Hank healed his son Timmy using the stolen antivenom.

Figure 3.3: Seven example solution plans (translated into natural language) for the problem in Figure 3.2. Steps in angle brackets are steps that were intended but not executed.

predicate indicates that one person is the mother or father of another.

Having established a vocabulary, the domain then defines five kinds of actions. The first action describes a `?victim` (which must be of type `person`) getting bitten by a snake. The `:precondition` indicates that before this action can occur, the `?victim` must be `Healthy`. After this action occurs, the `:effect` changes the world in three ways: the `?victim`'s status is no longer `Healthy`, the `?victim`'s status is `Sick` and the `?victim` intends to become `Healthy` again. The `die` action has a very similar structure. Before a `?victim` can die, he or she must have a status of `Sick`. After a victim dies, he or she is no longer `Sick` and instead is `Dead`.

The `steal` action describes how a `?thief` steals an `?item` from its `?owner` and angers the town `?sheriff`. The precondition of this action indicates that the `?thief` and the `?owner` may not be the same character. It also specifies that the `?thief` must be `Healthy`, the `?owner` must be the rightful owner of the `?item`, and that the `?owner` must be in possession of the `?item`. After this action, the `?owner` no longer has possession of the `?item`, the `?thief` has possession of the `?item`, and the `?sheriff` intends that the `?thief` be dead. The `:agents` keyword indicates that the `?thief` must be an intelligent agent who takes this action for some purpose. In other words, any story which contains a steal action must provide an explanation for why the `?thief` would want to steal something.

The `heal` action describes how a `?healer` uses some `?medicine` to heal a sick `?patient`. The `?healer` must not be `Dead`, the `?healer` must be in possession of the `?medicine`, and the patient must be `Sick`. After the action, the `?patient` is no longer `Sick`, the `?patient` is `Healthy`, and the `?healer` no longer has the `?medicine`. Both the `?healer` and the `?patient` must have a reason to take this action.

The final `shoot` action describes how a `?killer` shoots and kills some `?victim`. The `?killer` must be `Healthy` and must be armed. After the action, the `?victim` is no longer `Healthy` or `Sick`; the `?victim` is `Dead`. The `?killer` must have a reason to take this action.

The planning problem in Figure 3.2 describes all the people and items in the story world, as well as the initial state that the world is in. There are four people: `Timmy`, `Hank`, `Carl`, and `William`. `William` is also the `sheriff` (which is a subtype of `person`). There is one item, the `Antivenom` which is a kind of `medicine`. `Timmy` is a young boy. Initially, he is `Sick` and he intends to be `Healthy` again. `Hank` is `Timmy`'s father. He is `Healthy` and armed, and he intends that his son be `Healthy`. `Carl` is the town shopkeeper. He is the rightful owner of the `Antivenom` and he current has the `Antivenom`. `William` is the town sheriff and is also armed. The author's goal for the story is that `Timmy` is no longer `Sick`. This can be accomplished two ways: either `Timmy` is healed or `Timmy` dies.

Figure 3.3 gives seven plans which are solutions to the example problem. The increasing complexity of the solutions demonstrates the development of narrative plan representations from a classical plan, to the IPOCL plan model of Riedl and Young [62], to my CPOCL plan

model described here.

3.2 Representing Conflict in Partial Order Causal Link Plans

A plan is a sequence of steps that describes, in some formal language, how a world transitions from its beginning, or initial state, to its end, or goal state [53]. Classical planning generally uses the language of function-free ground predicate literals. Sometimes it also allows the use of other syntactic niceties such as boolean expressions, first order quantifiers, and conditional effects [81] (for simplicity, the example domain in Figure 3.1 does not use any of these). The various objects in the story world, such as characters, items, and places, are represented as logical constants. Relationships between these objects are described with predicates.

3.2.1 The Classical POCL Model

This section presents the definitions for classical partial order causal link planning on which intentional and conflict planning are built.

Definition 1 (state)**.** A *state* is a single function-free ground predicate literal or a conjunction of such literals describing what is true and false in a hypothetical story world.

Definition 2 (goal)**.** A *goal* is a literal or conjunction of literals which must be true in some state.

Definition 3 (planning problem)**.** A *planning problem* is composed of an initial state, which describes the configuration of the story world before planning begins, and a goal, which must be true in the final state of the story world after the plan has been executed.

Actions that change the world are described using templates. This formalism was originally described by Fikes and Nilsson as STRIPS [26].

Definition 4 (operator)**.** An *operator* is a template for an action which can occur in the world. It is defined as a two-tuple $\langle P, E \rangle$ where P is a set of *preconditions*—literals which must be true before the action can be carried out—and E is a set of *effects*—literals which are made true after the action is carried out. In the original STRIPS formalism the effects of an operator were divided into an add list and a delete list, but this can be generalized as one set of effects which contains both positive literals (e.g. p) and negative literals (e.g. $\neg p$).

The example domain in Figure 3.1 defines five operators: `snakebite`, `die`, `steal`, `heal`, and `kill`. The literals in an operator's preconditions and effects can use variables terms for generality.

Definition 5 (parameters)**.** The set of all variables used in an operators preconditions and effects are its *parameters*. To restrict the search space, a parameter can declare a *type* that limits which constants can legally be bound to that variable.In Figure 3.1, parameters are represented as tokens beginning with a `?`, and each parameter's type appears after the dash.

Definition 6 (planning domain)**.** The set of all operators which can be used to solve a problem is that problem's *planning domain*.

Definition 7 (step)**.** A *step* is a ground instance of an operator, i.e. one in which all the operator's parameters have been bound to constants. A step represents an actual event in the story. The example plans in Figure 3.3 are composed of steps.

Definition 8 (start step)**.** A *start step* is a special kind of step with no preconditions and effects equal to the initial state of the planning problem. A problem's start step can be thought of as the first step of every plan for that problem.

Definition 9 (end step)**.** An *end step* is a special kind of step with preconditions equal to the goals of the planning problem and no effects. A problem's end step can be through of as the last step of every plan for that problem.

Definition 10 (plan)**.** A *plan* is an ordered sequence of steps meant to solve a planning problem.

POCL planning operates in a least commitment fashion. When a new step is added to a plan, its parameters are not yet bound to specific constants. This allows a partial POCL plan to express notions such as "Someone healed Timmy using the antivenom" by including an instance of the `heal` operator but leaving the `?healer` parameter unbound. During the process of constructing a complete POCL plan, all unbound parameters must eventually be given a binding.

Definition 11 (binding constraint)**.** A *binding constraint* is a two-tuple $\langle v, t \rangle$ such that v is a variable corresponding to one parameter in a step, and t is either a constant or another parameter. When t is a constant, the binding constraint indicates that the value of parameter v is that constant. When t is another parameter, the binding constraint indicates that the parameters v and t must be assigned the same value. Binding constraints are transitive.

In addition to a set of steps, S, POCL plans also have a set of binding constraints, B, which track the values of each step's parameters. POCL planning also adopts a least commitment approach with regards to the order in which steps are taken.

Definition 12 (ordering constraint)**.** An *ordering constraint* $s < t$ indicates that a step s must occur before a step t.

POCL plans are said to be partially ordered because they contain a set of ordering constraints, O, which strives to express the minimum number of restrictions on how steps are ordered. In story terms, this means imposing as few constraints as possible on the story's discourse. A partially ordered plan represents many possible totally ordered plans—one for each valid topological sort of it steps.

POCL plans also explicitly track how the preconditions of each step are made true by previous steps using *causal links*. If we imagine a POCL plan as a graph in which the nodes are steps, causal links are labeled edges.

Definition 13 (causal link)**.** A *causal link* is an edge $s \xrightarrow{p} u$ whose tail is a step s with effect p and whose head is a step u with precondition p. A causal link $s \xrightarrow{p} u$ implies the ordering constraint $s < u$.

Adding a causal link to a plan often imposes binding constraints on the parameters of the tail and head steps. For example, say the plan contains an instance of `steal` such as `(steal ?thief Antivenom Carl)`. This step would have the effect `(has ?thief Antivenom)`. Say the plan also contains the step `(heal Hank Antivenom Timmy)`, which had the precondition `(has Hank Antivenom)`. A causal link can be drawn from `steal` to `heal` with the label `(has Hank Antivenom)`, but only if we impose the binding constraint $\langle$`?thief`, `Hank`$\rangle$. This logical unification process is that of finding a most general unifier, or *MGU*, which is described in more detail by Weld [81].

POCL plans maintain a set of causal links, L, to explain how the preconditions of each step become satisfied. The ordering constraints implied by causal links, as well as any other ordering constraints imposed on the plan, make the set of steps into a directed acyclic graph.

Definition 14 (causal parents and children)**.** If there exists a causal link $s \xrightarrow{p} u$, then the step s is said to be the *causal parent* of step u. Step u is said to be the *causal child* of s.

Definition 15 (causal ancestors and descendents)**.** A step's *causal ancestors* are all steps in the the transitive closure of the causal parent relationship. A step's *causal descendents* are all steps in the transitive closure of the causal child relationship.

Definition 16 (POCL plan)**.** A *partially ordered causal link plan*, or *POCL plan*, is a four-tuple $\langle S, B, O, L\rangle$, such that S is a set of steps, B is a set of binding constraints on the parameters in S, O is a set of ordering constraints that defines a partial ordering of the steps in S, and L is a set of causal links between the steps in S.

Due to the partially ordered nature of POCL plans, it may be possible to order a third step between the tail and head steps of a causal link. If this third step undoes the condition established by the causal link, the plan may fail to achieve the goal.

Definition 17 (threatened causal link)**.** A causal link $s \xrightarrow{p} u$ is a *threatened causal link* when there exists a step t with effect $\neg p$ such that the ordering $s < t$ and $t < u$ are valid in some total ordering of the steps s, t, and u. In other words, step t could possibly undo the effect p of step s before it is needed by step u.

Definition 18 (valid POCL plan)**.** A POCL plan is said to be *valid* if (1) every parameter of every step is bound to a constant, (2) for every total ordering of its steps, each step's preconditions are met immediately before it is taken, and (3) the goal is true after all steps have been taken.

Causal links and threatened causal links were originally designed to reason about what constraints has to be placed on a plan to ensure that it would succeed. Weld [81] provides sufficient conditions for the validity of a POCL plan in terms of causal links:

Theorem 19. *A POCL plan is valid if (1) for every precondition p of every step $u \in S$, there exists a causal link $s \xrightarrow{p} u$ from some step s, and (2) none of the causal links in L are threatened [81].*

Note that a threatened causal link is something to be avoided or eliminated in a POCL plan. One of the key extensions I make to this representation is a way to preserve certain threatened causal links while still guaranteeing that the plan will achieve its goal.

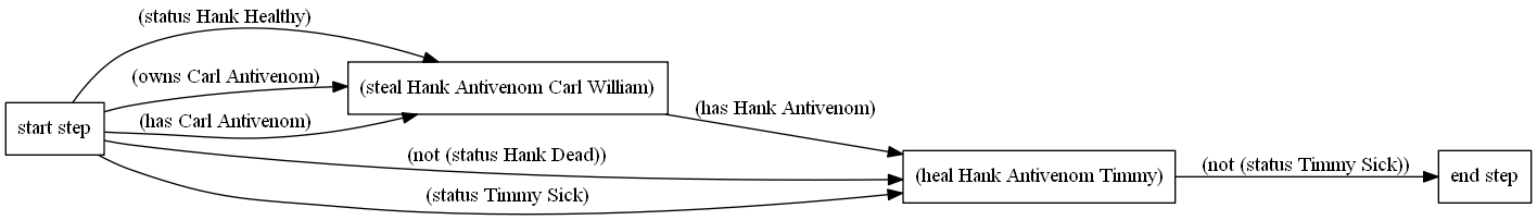
An example of a valid POCL plan is given in Figure 3.4. First, Hank steals the Antivenom from Carl, then Hank uses that Antivenom to heal Timmy. Note the causal link that runs from `steal` to `heal`. It explains how the earlier `steal` step satisfies a precondition of the later `heal` step, namely that Hank must have the Antivenom. Because a causal link implies an ordering constraint between its steps, the `steal` step must occur before the `heal` step.

3.2.2 The Intentional Plan Model

Intentional planning augments the knowledge representation of classical planning to incorporate the goals of individual characters in order to increase the appearance of character believability [62].

Definition 20 (character)**.** Some set of story world constants are designated as *characters*, intentional agents which should be seen to pursue their own goals.

Definition 21 (intentional operator)**.** An *intentional operator* is a three-tuple $\langle P, E, C \rangle$, such that P is a set of preconditions, E is a set of effects, and C is a set of characters who must consent to the execution of that action. The example domain given in Figure 3.1 lists the consenting characters for each operator after the `:characters` token. When no such token appears, the list of consenting characters is assumed to be empty.



Steps
`0 start step`
`1 end step`
`2 (heal ?healer ?medicine ?patient)`
`3 (steal ?thief ?item ?owner ?sheriff)`

Bindings
⟨`?healer,Hank`⟩ ⟨`?medicine,Antivenom`⟩
⟨`?patient,Timmy`⟩ ⟨`?thief,Hank`⟩
⟨`?item,Antivenom`⟩ ⟨`?owner,Carl`⟩
⟨`?sheriff,William`⟩

Orderings
$0 < 1$, $0 < 2$, $2 < 1$, $0 < 3$, $3 < 1$, $3 < 2$

Causal Links
0 — `(status Hank Healthy)`→ 3
0 — `(owns Carl Antivenom)` → 3
0 — `(has Carl Antivenom)`→ 3
0 — `(not (status Hank Dead))` → 2
3 — `(has Hank Antivenom)` → 2
0 — `(status Timmy Sick)` → 2
2 — `(not (status Timmy Sick))` → 1

Figure 3.4: A POCL plan for Plan D in Figure 3.3.

Definition 22 (intentional step)**.** An *intentional step* is a ground instance of an intentional operator.

Definition 23 (happening)**.** An intentional step for which $C = \emptyset$, i.e. a step with no consenting characters, is called a *happening*. Happenings represent accidents and the forces of nature. Start steps and end steps are happenings.

Definition 24 (motivation)**.** A *motivation* is a modal predicate of the form `(intends ?character ?goal)` where `?character` is a character and `?goal` is a ground predicate literal that the character wishes to make true. Motivations may appear in the effects of intentional operators (and thus also in the effects of intentional steps) and in a problem's initial state (and thus also in the effects of start steps).

Definition 25 (intentional parents and children)**.** An earlier step s is said to be the *intentional parent* of a later step u for character c if step s is the causal parent of u, $c \in C$ for s, and $c \in C$ for u. In other words, s is the causal parent of u and the steps share a consenting character. Step u is said to be the *intentional child* of step s for character c.

Definition 26 (intentional ancestors and descendents)**.** A step's *intentional ancestors* are all steps in the the transitive closure of the intentional parent relationship. A step's *intentional descendents* are all steps in the transitive closure of the intentional child relationship.

Intentional planning defines structures called *intention frames* that explain how a character adopts a goal, pursues that goal, and then eventually achieves it.

Definition 27 (motivating step)**.** A *motivating step* for character c and goal g is an intentional step whose effects contain the motivation `(intends` c g`)`. A motivating step describes how a character came to adopt a goal.

Definition 28 (satisfying step)**.** A *satisfying step* for a character c and a goal g is an intentional step which requires the consent of c and whose effects contain g. A satisfying step describes how a character achieved a goal.

Definition 29 (subplan)**.** A character c's *subplan* to achieve some goal g is a set of steps T such that T contains a satisfying step σ for character c and goal g, and every other step in T is an intentional ancestor of σ. Another equivalent way to express this definition is that T must contain a satisfying step σ, every other step in T must be the causal parent of σ, and every step in T requires the consent of character c.

Definition 30 (intention frame)**.** An *intention frame* is a five-tuple $\langle c, g, m, \sigma, T \rangle$ where c is a character, g is some ground predicate literal that c wants to make true, m is a motivating

step for c and g, σ is a satisfying step for c and g, and T is a subplan for c and g such that $\sigma \in T$. All steps in T must be ordered after the motivating step m. The steps in T are said to be members of T's intention frame.

Definition 31 (IPOCL plan)**.** An *intentional partial order causal link plan*, or *IPOCL plan*, is a five-tuple $\langle S, B, O, L, I \rangle$ such that S is a set of intentional steps, B is a set of binding constraints, O is a set of ordering constraints, L is a set of causal links, and I is a set of intention frames.

The process of intentional planning is that of organizing a plan's steps into intention frames to ensure that every step can be explained in terms of the motivations of the characters who take them. When a step that requires the consent of some character is not a member of any intention frame for that character, the plan is incomplete.

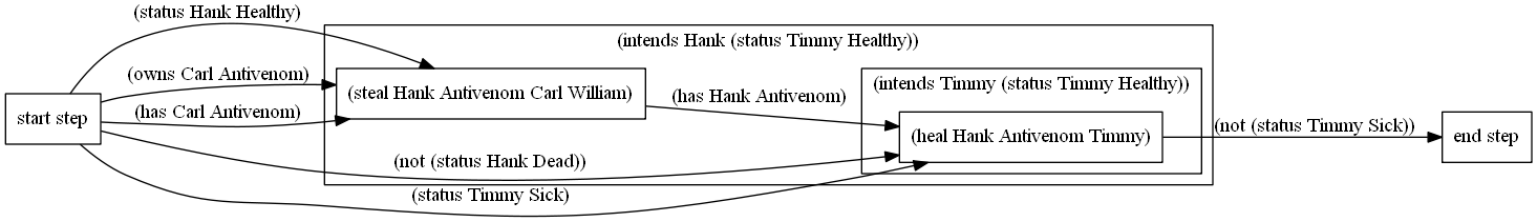
Definition 32 (orphan)**.** An intentional step is an *orphan* if it has c as a consenting character but is not a member of any intention frame for character c.

Definition 33 (valid IPOCL plan)**.** An IPOCL plan is *valid* if it is valid according to Definition 18 and it contains no orphans [62]. In other words, for every total ordering of intentional steps (1) every parameter of every step is bound to some constant (2) the preconditions of every step are satisfied before the step is taken, (3) the resulting state of the story world after all steps have been taken is a goal state, and (4) for every character c which must consent to a step s, s is the intentional ancestor of some satisfying step for one of c's goals.

An example valid IPOCL plan is given in Figure 3.5. In terms of the steps it contains, this plan is identical to the POCL plan in Figure 3.4: First Hank steals the Antivenom and then he uses it to heal Timmy. Note the addition of the intention frames which make this an IPOCL plan. The first frame describes Hank's subplan to heal his son. It is motivated by the start step (in other words, before the story starts). It is eventually satisfied by the `heal` step, which achieves Hank's goal. The `steal` step is also part of this frame because it is the intentional ancestor of the satisfying step—in other words, there is a causal link running from `steal` to `heal` that helps to explain how Hank was able to achieve his goal. Recall that the `heal` step requires the consent of two characters: the healer and the person being healed. That means that there must also exist a second intention frame to explain why Timmy allowed Hank to heal him. Timmy allowed that to happen because it was in service of his goal to be healthy again.

3.2.3 The Conflict Plan Model

My work on a computational model of conflict builds on this model of intentional planning. It leverages the existing notion of threatened causal links that arise between different subplans to



Steps
`0 start step`
`1 end step`
`2 (heal ?healer ?medicine ?patient)`
`3 (steal ?thief ?item ?owner ?sheriff)`

Bindings
⟨`?healer,Hank`⟩ ⟨`?medicine,Antivenom`⟩
⟨`?patient,Timmy`⟩ ⟨`?thief,Hank`⟩
⟨`?item,Antivenom`⟩ ⟨`?owner,Carl`⟩
⟨`?sheriff,William`⟩

Orderings
$0 < 1$, $0 < 2$, $2 < 1$, $0 < 3$, $3 < 1$, $3 < 2$

Causal Links
0 — `(status Hank Healthy)` → 3
0 — `(owns Carl Antivenom)` → 3
0 — `(has Carl Antivenom)` → 3
0 — `(not (status Hank Dead))` → 2
3 — `(has Hank Antivenom)` → 2
0 — `(status Timmy Sick)` → 2
2 — `(not (status Timmy Sick))` → 1

Intention Frames
⟨$c =$`Hank`, $g =$`(status Timmy Healthy)`, $m = 0, \sigma = 2, T = \langle 2, 3 \rangle$ ⟩
⟨$c =$`Timmy`, $g =$`(status Timmy Healthy)`, $m = 0, \sigma = 2, T = \langle 2 \rangle$ ⟩

Figure 3.5: An IPOCL plan for Plan D in Figure 3.3

capture how the subplans of characters interfere with one another. One of the key extensions to the intentional representation is that steps in a plan can be marked as either executed or non-executed.

Definition 34 (executable step)**.** An *executable step* is a ground instance of an intentional operator. It is a four-tuple $\langle P, E, C, x \rangle$ where P, E, and C are defined as for the step's operator, and x is a boolean flag that is true if the step is an *executed step* and false if the step is a *non-executed step*.

Note that this definition overrides the previous definition of step (Definition 7) by adding the boolean flag that indicates whether or not the step is executed. This notion of executed and non-executed steps should not be confused with the notion of "steps already executed" and "steps not yet executed" when carrying out a plan. A non-executed step is one which *will never be executed*. Non-executed steps exist to track what a character intended to do but did not succeed in doing. Happenings (steps not intended by any character) require some special reasoning.

Definition 35 (executable happenings)**.** For each happening h (including the start and end steps), h must be an executed step, and there must exist an intention frame whose character is $fate$, whose goal is $\emptyset$, whose motivating step is the start step of the plan, whose satisfying step is h, and for whom $T = \{h\}$. In other words, each happening is assigned to its own intention frame intended by $fate$.

This convention of assigning happenings to their own special intention frames for $fate$ exists to simplify the definition of conflict, which occurs between two intention frames. Otherwise, special cases would need to be defined any time a happening is involved in a conflict.

By allowing the subplans of some characters to contain non-executed steps, we can ensure that each character's subplans are consistent while still allowing some subplans to fail. In addition to subplans that change the world, we also need a way to represent the desire that things remain the same. For example, all characters in the example problem begin alive and wish to remain alive. We can accomplish this using non-executed steps.

Definition 36 (persistence step)**.** A *persistence step* is a non-executed step with a single precondition g, a single effect g, and a single consenting character c. All persistence steps are ordered to occur simultaneously with the plan's end step.

Persistence steps are not instances of any operator, but placeholder satisfying steps for the intention frames of character c. They exist to satisfy the goals of characters which are already satisfied and thus do not require any other steps to make true. The presence of a persistence step allows a causal link to be created with the label of the fact that the character wishes to

remain true. These causal links (more specifically, threats to these causal links) can be used to represent conflict.

The introduction of non-executed steps into the model has implications for causal and intentional relationships. A non-executed step cannot be the causal parent of an executed step. This is because a non-executed step will never actually occur, therefore its effects cannot be used to satisfy the preconditions of events that will occur. This implies that a non-executed step cannot be the intentional parent of an executed step. Similar implications apply to the causal child, intentional child, causal ancestor, intentional ancestor, causal descendent, and intentional descendent relationships.

Definition 37 (conflict plan)**.** A *conflict plan* is an intentional plan composed of executable steps.

Having laid out the role of executable steps, we can now define conflict as one intention frame interfering with another.

Definition 38 (conflict)**.** A *conflict* is a four-tuple $\left\langle c_1, c_2, s \xrightarrow{p} u, t \right\rangle$ such that:

- c_1 is a character such that $c_1 \neq fate$
- c_2 is any character, possibly $fate$ and possibly the same as c_1
- $s \xrightarrow{p} u$ is a causal link between step s and step u with label p, henceforth a *conflict link*
- t is a step with effect $\neg p$ that threatens the causal link $s \xrightarrow{p} u$
- there exists an intention frame $f_1 = \langle c_1, g_1, m_1, \sigma_1, T_1 \rangle$ such that $u \in T_1$
- there exists an intention frame $f_2 = \langle c_2, g_2, m_2, \sigma_2, T_2 \rangle$ such that $t \in T_2$ and $f_1 \neq f_2$
- either t or u (or both) are non-executed steps

While this definition is long and cumbersome, it corresponds to a very simple idea: one character intends to take a step u, but some step potentially prevents u from being taken. Either the character will succeed in taking step u, in which case the thwarting event must fail, or the character will fail to take step u, in which case the thwarting event may succeed. It is also possible that both the character and the thwarting event will fail. The thwarting event might be in the subplan of another character (narratologists call this *external conflict*), a subplan of the same character for a different goal (called *internal conflict*), or it might be a step intended by fate (called *conflict with the environment*).

Definition 39 (CPOCL plan)**.** A *conflict partial order causal link plan*, or *CPOCL plan*, is a five-tuple $\langle S, B, O, L, I \rangle$ such that S is a set of executable steps, B is a set of binding constraints, O is a set of ordering constraints, L is a set of causal links, and I is a set of intention frames.

Definition 40 (valid CPOCL plan)**.** A CPOCL plan is said to be *valid* if it adheres to Definition 33 and all executed steps have only executed steps as causal ancestors. In other words, a CPOCL plan is valid if (1) it is a valid IPOCL plan, (2) no causal link runs from a non-executed step to an executed step.

A POCL and IPOCL plan can only be valid if they are free of threatened causal links [81, 62]. The CPOCL definition of conflict allows certain threatened causal links to remain in a plan (those which are part of a conflict as described in Definition 38) without any risk that they will prevent the goal from being achieved.

An example valid CPOCL plan is given in Figure 3.7. The important new element in this plan are the non-executed steps (drawn with dashed borders). Consider Hank's subplan to heal Timmy. It is identical to the subplan from the previous IPOCL plan in Figure 3.5, except that the `heal` step is non-executed. This means that Hank never succeeds in healing Timmy. This is because one of the causal links that explains how `heal` was possible is threatened, specifically William shoots Hank before `heal` can happen, so Hank is no longer alive. This threatened causal link is a conflict link; it runs from an executed step to a non-executed step. Therefore it does not prevent the plan for achieving the author's goal, but it does provide helpful information about how William's plan thwarts Hank's plan.

Unlike in the previous examples, Timmy is never healed in this story. There cannot exist a causal link from `heal` to the end step because `heal` is non-executed and the end step must be executed. Rather, the `die` step is uses to satisfy the author's goal in this plan. Because `die` does not have any consenting characters, it is placed in an intention frame for Fate.

Also note the persistence steps that appear in the bottom right of the diagram. These persistence steps and their intention frames explain the character goals that were already achieved at the start of the story. For example, Carl was already healthy at the start of the story, and he wanted to stay that way. A persistence step and a single causal link from the start step demonstrate that indeed he stayed healthy throughout the story. However, Carl also intended to keep the Antivenom. That goal was eventually violated when Hank stole the Antivenom. This is represented by the fact that the causal link which satisfies the precondition of `(persist (has Carl Antivenom))` is a conflict link.

3.3 The Expressivity of Conflict Plans

The definition of a valid CPOCL plan defines the set of solutions that a narrative planner strives to produce. In order to demonstrate how the CPOCL model contributes to story expression, we should consider how this solution space compares to those defined by the models on which CPOCL is built. The example domain and problem, in Figure 3.1 and Figure 3.2 respectively,

Steps

0 start step
1 end step
2 (persist (has Carl Antivenom))
3 (persist (status Hank Healthy))
4 (persist (status Carl Healthy))
5 (persist (status William Healthy))
6 (die ?victim-1)
7 (heal ?healer ?medicine ?patient)
8 (steal ?thief ?item ?owner ?sheriff)
9 (shoot ?killer ?victim-2)

Bindings

⟨?victim-1,Timmy⟩⟨?healer,Hank⟩
⟨?medicine,Antivenom⟩ ⟨?patient,Timmy⟩
⟨?thief,Hank⟩ ⟨?item,Antivenom⟩
⟨?owner,Carl⟩ ⟨?sheriff,William⟩
⟨?killer,William⟩ ⟨?victim-2,Hank⟩

Orderings

$0 < 1$, $0 < 2$, $0 < 3$, $0 < 4$, $0 < 5$, $0 < 6$,
$6 < 1$, $6 < 2$, $6 < 3$, $6 < 4$, $6 < 5$, $0 < 7$,
$7 < 1$, $7 < 2$, $7 < 3$, $7 < 4$, $7 < 5$, $0 < 8$,
$8 < 1$, $8 < 2$, $8 < 3$, $8 < 4$, $8 < 5$, $8 < 7$,
$8 < 9$, $0 < 9$, $9 < 1$, $9 < 2$, $9 < 3$, $9 < 4$,
$9 < 5$

Causal Links

0 — (status William Healthy)→ 9
0 — (armed William)→ 9
0 — (status Hank Healthy)→ 8
0 — (owns Carl Antivenom) → 8
0 — (has Carl Antivenom)→ 8
0 — (not (status Hank Dead)) → 7
8 — (has Hank Antivenom) → 7
0 — (status Timmy Sick) → 7
0 — (status Timmy Sick) → 6
0 — (status William Healthy) → 5
0 — (status Carl Healthy) → 4
0 — (status Hank Healthy) → 3
0 — (has Carl Antivenom) → 2
6 — (not (status Timmy Sick)) → 1

Intention Frames

⟨c =Hank, g =(status Timmy Healthy),
$m = 0, \sigma = 7, T = \langle 8, 7 \rangle$ ⟩

⟨c =Timmy, g =(status Timmy Healthy),
$m = 0, \sigma = 7, T = \langle 7 \rangle$ ⟩

⟨c =Carl, g =(has Carl Antivenom),
$m = 0, \sigma = 2, T = \langle 2 \rangle$ ⟩

⟨c =Hank, g =(status Hank Healthy),
$m = 0, \sigma = 3, T = \langle 3 \rangle$ ⟩

⟨c =Carl, g =(status Carl Healthy),
$m = 0, \sigma = 4, T = \langle 4 \rangle$ ⟩

⟨c =William, g =(status William Healthy),
$m = 0, \sigma = 5, T = \langle 5 \rangle$ ⟩

⟨c =William, g =(status Hank Dead),
$m = 8, \sigma = 9, T = \langle 9 \rangle$ ⟩

⟨c =Fate, $g = \emptyset$,
$m = 0, \sigma = 6, T = \langle 6 \rangle$ ⟩

Figure 3.6: The CPOCL data structure for Plan F in Figure 3.3

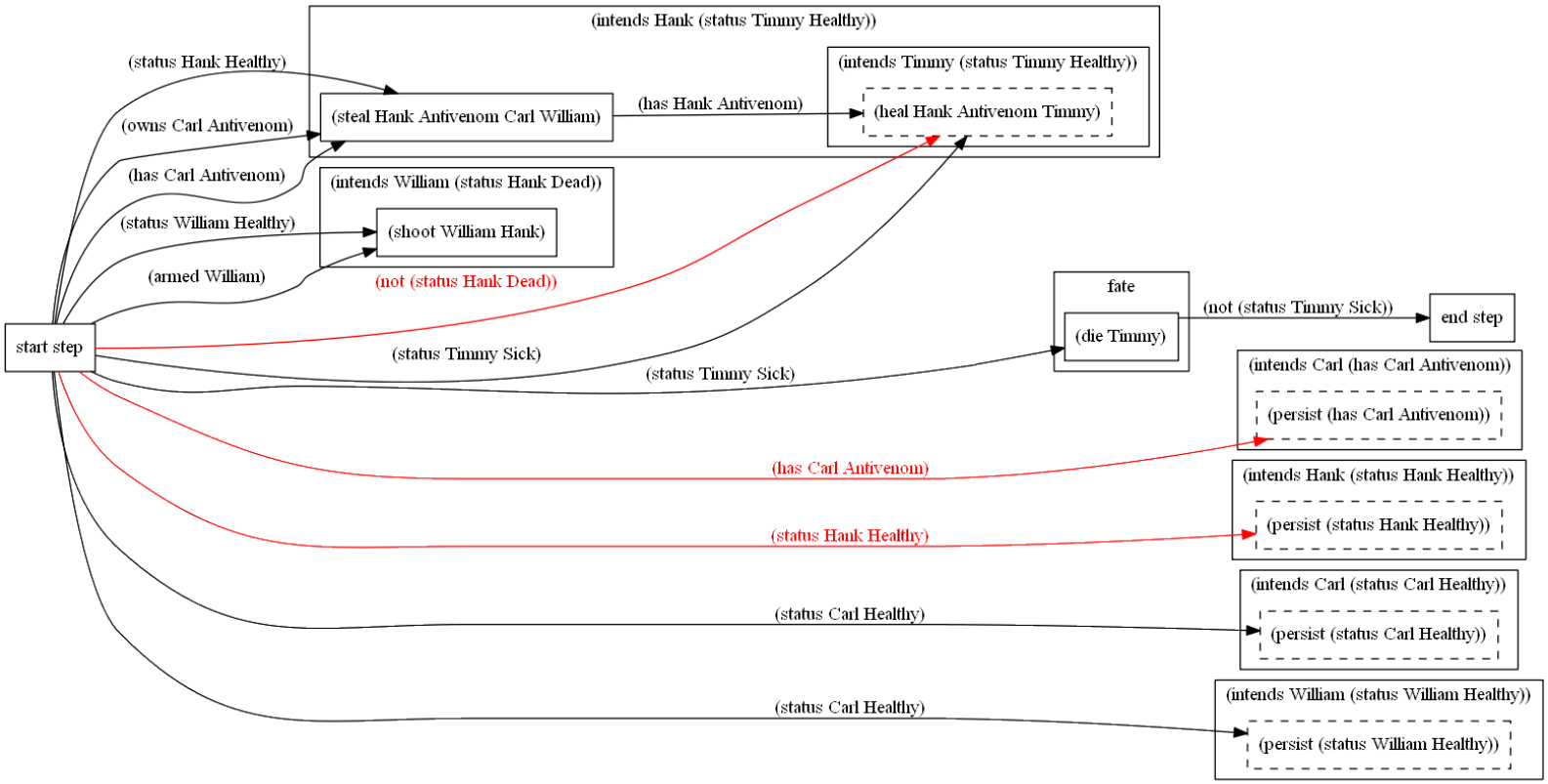


Figure 3.7: A CPOCL plan for Plan F in Figure 3.3. Steps with dashed borders are non-executed. Conflict links are in red.

will serve as a good case study for understanding the progression from POCL to IPOCL to CPOCL in terms of what can be represented.

Any solution which is guaranteed to reach the goal from the initial state can be represented as a POCL plan. Consider the example solutions in Figure 3.3. If we ignore the non-executed steps in Plans F and G, all of these plans can be generated by a POCL planner that performs complete search. In some cases this may be sufficient for story generation. In Plan A, Timmy dies. This satisfies the goal condition and would make for a believable (all be it short) story. However, POCL plans are also liable to represent stories with undesirable properties. In Plan B the shopkeeper heals Timmy. This does not strain credulity too much, except that the shopkeeper violates his goal to keep the antivenom for no apparent reason. However, Plan C, in which Hank shoots his son, is a clear violation of the expectations of the audience. These plans are said to be coherent because all the steps could happen that way, but they are not believable, because the agents do not behave according to expectations [62].

The intentional planning framework restricts the solution space of the POCL model to those plans in which every step is taken in service of some character goal (except happenings, which need not be intended by anyone). This excludes solutions like Plan B and Plan C in which characters act without motivation. After Plan A, Plan D is now the shortest valid IPOCL plan: Hank steals the antivenom and uses it to heal Timmy. Both of these actions clearly contribute to Hank's goal, so they form a story which is both coherent and believable.

IPOCL significantly narrows the POCL solution space in order to exclude plans that are not believable. But this new solution space is perhaps too narrow. In an IPOCL plan, either an agent does not pursue its goal at all, or it must succeed in pursuing its goal. IPOCL cannot represent failed subplans, and this is an essential element of conflict, which is an essential element of stories. The purpose of the CPOCL model is to widen the IPOCL solution space to include stories in which all characters act believably but some may fail to achieve their goals. This representational benefit is achieved through the use of non-executed steps.

Plan F and Plan G are examples of stories which can be modeled as CPOCL plans using non-executed steps to represent subplans that fail due to conflict. Even if we consider only the executed steps, Plan F cannot be produced by an intentional planner because it requires the use of non-executed steps during the reasoning process. In Plan F, sheriff William shoots Hank before Hank can use the antivenom to heal his son. William would have never been motivated to shoot Hank if Hank had not stolen the antivenom, and Hank would never have stolen the antivenom if he had not been motivated to heal Timmy. This is an example of a story in which *some* of a subplan is needed to motivate the rest of the story, but *all* of that subplan does not occur because it it interrupted by conflict. Stories like this are excluded from the IPOCL solution space.

One might argue that IPOCL plans can represent conflict, such as in Plan E where sheriff

William shoots Hank after Hank successfully heals his son. In some sense this is true; clearly William's subplan to shoot Hank violates Hank's desire to stay healthy. However, this only works because Hank does not form a subplan to stay healthy. Recall that in the IPOCL model, either a subplan does not exist or it succeeds. So the only way to express this conflict in an IPOCL plan is to ensure that Hank never forms a subplan to stay alive. CPOCL handles this in a way which is both representationally richer and semantically more helpful: Hank forms a subplan to stay alive, but it fails.

The CPOCL model represents an important expansion to the IPOCL model even if we disregard non-executed steps once planning is finished. But non-executed steps are more than just place-holders during search that enable a wider variety of stories to be produced and an explicit representation of conflict. They are helpful in and of themselves because they represent the inner worlds of the characters along with alternate paths the story could have taken. Having a record of what a character was planning to do, even if it failed to do it, can be useful to an interactive narrative system that may need to replan in response to unexpected user actions or generate dialog that is consistent with a character's plans.

Note that the CPOCL model subsumes the IPOCL model, which means it includes plans with no conflict such as Plan A and Plan D. A CPOCL plan can contain conflict, but it is not required to contain conflict. This poses an interesting question for a narrative planner: If a problem can be solved without recourse to the use of conflict, should it be? If the narrative planner can achieve the author's goals without bringing the characters into conflict, forcing them into conflict may seem artificial to the audience. When a problem can be solved without conflict, it may indicate that the initial state and goal of the story need to be revised rather than the planner. This limitation is discussed in more detail in Chapter 7.

3.4 Seven Dimensions of Narrative Conflict

The CPOCL model of conflict was conceived to represent all forms of narrative conflict that can be operationalized as thwarted plans. This is a very broad category of phenomena, so in order to provide greater control over the content of stories, I describe seven dimensions that can be used to distinguish one conflict from another. These dimensions were synthesized from various narratological sources (especially [34, 22, 24, 73]). Each one is meant to represent an important aspect of conflict, but I make no claim that this list is exhaustive.

1. Participants - the opposing forces between which the conflict occurs
2. Topic - the world condition which makes the conflicting subplans incompatible
3. Duration - the length of time during which the participants are in conflict

4. Balance - how evenly matched the participants are

5. Directness - how close the participants are to one another

6. Intensity - what is at stake

7. Resolution - the outcome of the conflict and its effects on the participants

The first three dimensions can be directly observed in a CPOCL plan with no additional information. They answer "who?" "why?" and "when?" respectively. *Participants* and *topic* correspond to discrete symbols in the plan, and *duration* can be measured based on the ordering of steps. These first three dimensions were evaluated in an experiment which also lent credence to the overall effectiveness of the CPOCL model [79].

The last four dimensions can be approximated with minimal additional information that might already be available in many narrative systems. *Balance*, *directness*, *intensity*, and *resolution* are defined as continuous values. A second experiment [80] has demonstrated that my formulas for these dimensions can rank a set of stories similarly to human readers.

The Purpose of the Dimensions

Different stories utilize these dimensions in different ways. Consider the growing conflict between Luke and Darth Vader in *Star Wars* in terms of the dimension of *directness*. Initially Darth Vader is a physically and emotionally distant enemy, but as Episodes IV, V, and VI progress, Luke gets physically closer and emotionally closer to Vader, culminating in a final face-to-face duel between father and son. The same is true of Ahab and Moby Dick and numerous other epic protagonist/antagonist pairs. But this trend of increasing directness does not hold across all stories. Consider the *Tom and Jerry* cartoon series. Most episodes begin with Tom the cat chasing Jerry the mouse in a very direct conflict. As the plot thickens, Tom resorts to more complicated, less direct methods of tricking or trapping Jerry. The same is true of Roadrunner and Coyote cartoons and others of this genre. So the way in which authors use the dimensions of conflict depends on the genre and the purpose of the story. For this reason, I have attempted to define these phenomena independently of their effect on the audience. In other words, readers should be able to agree on how direct a conflict is even if they do not agree on the rhetorical purpose of the conflict or how much they like it.

Another reason for this motivation lies in the fact that story planning as I describe it is meant to produce the *fabula* of a story—that is, the complete sets of events that occur in chronological order. Before a story is presented to the audience, the author must also decide on a *syuzhet*—a telling of the story—which may leave out important information or tell events out of order. The way in which a conflict is presented changes its effect on the audience. From the

god's-eye view of the *Star Wars* fabula, Luke and Darth Vader are always father and son, so this element of their directness never changes. However, by strategically withholding this important information from Luke and from the audience, the storyteller is able to achieve a perceived increase in the directness of the protagonist and antagonist through the syuzhet. Because the CPOCL model of conflict deals only with the fabula level, it is important to separate the dimensions of conflict from their effects on the audience. In other words, I am attempting to point out important features of conflict without committing to the specific artistic purposes of those features. The dimensions of conflict are meant to provide a well-defined toolbox for artists to use as they see fit. Having stated this important caveat, I now present the definitions for each of the seven dimensions.

3.4.1 Participants

The *participants* of a conflict, labeled c_1 and c_2 in Definition 38, are the two characters associated with the conflicting intention frames. Example Plan G has examples of three different kinds of conflict. Hank's plan to steal the antivenom and heal his son is in conflict with sheriff William's plan to shoot Hank. This conflict between two agents is external conflict where $c_1 \neq c_2$. The step where Hank gets bitten by a snake is a happening, which represents an accident or the forces of nature because $c_2 = fate$. This step thwarts Hank's plan to stay alive, and is an example of conflict with the environment. This snakebite prompts Hank to plan to use the stolen antivenom on himself, but that new plan conflicts with his original plan of healing his son. This is an example of internal conflict, where $c_1 = c_2$.

3.4.2 Topic

The *topic* of a conflict is the condition which makes the two character plans incompatible—the label of the threatened causal link. Textually, it can be expressed as "c_1 intends step u, which requires p, but c_2 intends step t, which causes $\neg p$." For example, "Hank intends to heal Timmy, which requires Hank not to be dead, but William intends to shoot Hank, which will cause Hank to be dead."

One important direction of future work will be to reason about the topic of conflict at a higher level. For example, a reader might say that William's duty to uphold the law is in conflict with Hank's duty to care for his son, or they might say that it is a conflict between the letter of the law and the spirit of the law. CPOCL is not yet able to reason about the topic of a conflict at this level of abstraction.

3.4.3 Duration

The *duration* of a conflict is the interval of time during which both participants intend their incompatible plans. The steps of a CPOCL plan are partially ordered, so to calculate duration, some valid total ordering O must be chosen. Let $\text{index}(s, O)$ be the index of step $s \in O$ such that the placeholder start step has index 0, the first executed step has index 1, the second executed step index 2, and so on until the placeholder end step, which has index n. By definition, all persistence steps also have index n. A non-executed step is defined to have an index equal to the first executed step which occurs after it. In example Plan G, the first step `(steal Hank Antivenom Carl William)` has index 1. The second step `<shoot William Hank>` is non-executed, so it has the same index as the third step `(snakebite Hank)`, which is index 2.

A story can now be envisioned as a sequence of n states. t_0 is the initial state of the story, occurring before the first step (i.e. the step with index 1). t_1 is the state after step 1 has occurred, t_2 the state after step 2, etc. The duration of a conflict is the number of states during which c_1 intends f_1 and c_2 intends f_2. To determine this, we need to know when intention frames begin and end. The beginning is simply the state after the motivating step, but detecting the end is more complicated.

The end of an intention frame is the state by which a character has abandoned its plan. If all of the steps in an intention frame are executed, the frame ends once the last step is executed. If some of the steps in the frame are non-executed, the frame ends after the last executed step. One important exception to this rule exists: if the first non-executed step in a frame is step t of a conflict (the head step of a threatened casual link), then the intention frame ends after step u (the threatening step). The reason for this exception comes from the nature of conflict: if a character abandoned a plan because it was thwarted, he should intend the plan up until the time when the plan gets thwarted.

Let the function $\Omega(f)$ return the index of the state by which intention frame f has ended. Recall that m_1 and m_2 are the motivating steps of the two conflicting intention frames. Now, we can define the duration of a conflict as:

$$\begin{aligned} \text{start} &= \max(\text{index}(m_1), \text{index}(m_2)) \\ \text{end} &= \min(\text{index}(t), \text{index}(u), \Omega(f_1), \Omega(f_2)) \\ \text{duration} &= \text{end} - \text{start} \end{aligned}$$

An example will help to make this more clear. At the beginning of example Plan F, Hank decides to steal the antivenom and heal his son. This subplan exists in the first state, t_0. After Hank succeeds in stealing the antivenom, in other words state t_1, it causes sheriff William to form the plan of shooting Hank. At this moment, Hank and William come into conflict. It

is resolved when William wins out over Hank by shooting him in step 2. Even though Hank intends steps with an index higher than 2—specifically the non-executed step where he heals Timmy—he is forced to abandon his plan at time 2 when he is killed by William. Thus, the conflict begins at time 1 and ends at time 2, having a duration of 1.

Additional Vocabulary for Balance, Directness, Intensity, and Resolution

The next four dimensions of conflict take on continuous values in the range [0,1] or [-1,1]. Each is measured from some participant's point of view—that is, one of the characters c_1 or c_2 who is involved in the conflict. For example, intensity(c_1) can be read as "how intense the conflict is for character c_1." This use of point of view implies that the values can be asymmetrical.

These dimensions also require some additional domain information in order to be approximated:

- utility(c, T) measures how satisfied character c is with the state of the world after the sequence of steps T occurs. utility$(c, \emptyset)$ is the character's utility before the conflict begins. This function might correspond to a player's score or level. An example utility function is given in Algorithm 1.

- $\pi(T)$ measures how likely some sequence of steps T is to succeed. Many systems, especially role playing games, involve statistical models of how likely an action is to succeed based on, for example, a character's skill plus a dice roll. An example likelihood function is given in Algorithm 2.

One of the parameters to both of these functions is T, a set of steps after which the dimension is measured. Recall that a partially ordered story can have many possible total orderings. This means that multiple conflicts can be interleaved with one another. When analyzing a conflict, we wish to consider only the steps which are part of that conflict. Generally this means only the steps in T_1 (character c_1's subplan) and T_2 (character c_2's subplan), but other steps from the story may need to occur in order to enable those steps.

Let T_1' be all steps in T_1 plus their causal ancestors, such that the time index of each step in T_1' is higher than the start time of the conflict. T_1' is *only* those future steps which need to occur to carry out the rest of c_1's subplan. Let T_2' be the same for T_2. These sets of steps are used in several of the dimensions below.

3.4.4 Balance

Balance measures the relative likelihood of each side in the conflict to succeed (regardless of the actual outcome). When the head step u of a conflict link $s \xrightarrow{p} u$ is a persistence step, then balance(c_1) is simply $1 - \pi(T_2')$ and balance(c_2) $= \pi(T_2')$. In other words, when c_1 wants some

fact to remain true, the balance of the conflict for c_1 is the inverse of the probability that the opponent will succeed. A more general formula is needed when not dealing with persistence steps. Assuming that one side or the other will prevail:

$$\text{balance}(c_1) = \frac{\pi(T_1')}{\pi(T_1') + \pi(T_2')}$$

The range of *balance* is $[0, 1]$. If c_1 is likely to prevail, i.e. $\pi(T_1')$ is close to 1, then balance is high for c_1. If the opponent is is more likely to prevail, then balance is low for c_1. In example Plan F, T_1' is the remaining portion of Hank's subplan once he comes into conflict with sheriff William: to heal Timmy. Based on the example likelihood function provided in Algorithm 2, $\pi(T_1') = 1$. William's subplan, T_2', is to shoot Hank. $\pi(T_2') = 0.5$ because the shootout has only a 50% probability of succeeding (since Hank also has a gun). Thus, balance$(Hank) = 0.667$, which is skewed in Hank's favor. This makes sense when we consider that Hank's plan has no chance of failing as long as no one interferes, whereas William's plan might fail even if he starts the shootout. Note that a value of 0.5 corresponds to a most balanced conflict from the point of view of the author, because both participants are equally likely to succeed.

3.4.5 Directness

Directness measures how close the participants are to one another:

$$\text{directness}(c_1) = \frac{\sum_{i=1}^{n} \text{closeness}_i(c_1, c_2)}{n}$$

For simplicity, the only type of *closeness* measured in the example domain is family closeness, which is 0 if the participants are not related or 1 if the participants are related or the participants are the same person. Other types of closeness can be measured, such as physical position, friendship, and interpersonal closeness (which is high when characters carry out their own plans and low when characters accomplish their plans vicariously through other characters [80]). This formula might also be made a weighted average based on genre expectations. The range of *directness* and each form of *closeness* is $[0, 1]$.

Hank and William are not family. Hence, directness$(Hank) = 0$. Example Plan G demonstrates a conflict between Hank and his son Timmy in which closeness$(Hank) = 1$, the maximum value. Additional types of closeness were measured in the example domain used to validate this formula.

3.4.6 Intensity

Intensity measures how much is at stake in the conflict. It is the difference between how high a participant's utility will be if she prevails and how low it will be if she fails. Because there are

many possible ways that a plan might fail, I estimated the utility of failure as the character's utility if its opponent succeeds:

$$\text{intensity}(c_1) = |\text{utility}(c_1, T_1') - \text{utility}(c_1, T_2')|$$

The range of *intensity* is $[0, 1]$. Two factors influence this formula: how much can be gained and how much can be lost. Situations which are high risk (failure results in a low utility) or high reward (success results in a high utility) have medium intensity, while situations which are both high risk and high reward have high intensity. Like balance, intensity is measured regardless of the actual outcome.

Hank's conflict with sheriff William in Plan F has the maximum intensity; $\text{intensity}(Hank) = 1$. If Hank succeeds, he and his son will both live, which would yield $\text{utility}(Hank, T_1') = 1$. If he fails, he will die and his son will be left to die of the snakebite, which would yield $\text{utility}(Hank, T_2') = 0$. This is a high risk and high reward situation, so the stakes are high.

3.4.7 Resolution

Resolution measures the change in utility a participant experiences after a conflict ends. Let E be the set of executed steps from T_1' and T_2'. In other words, E is how the conflict actually plays out. It may contain some steps from both subplans, but it cannot contain all steps from both subplans (because the subplans conflict).

$$\text{resolution}(c_1) = \text{utility}(c_1, E) - \text{utility}(c_1, \emptyset)$$

The range of resolution is $[-1, 1]$. Timmy was already dying when the conflict between Hank and William began in Plan F, so $\text{utility}(Hank, \emptyset) = 0.4$. Hank dies in the shootout, so $\text{utility}(Hank, E) = 0$. Thus, $\text{resolution}(Hank) = 0 - 0.4 = -0.4$ because the outcome is poor for him.

Algorithm 1 A simple utility function utility(c, T) for a character c after the sequence of steps T in the example domain given in Figure 3.1.

Require: A character c and the state of the story world after the steps T are taken.
Ensure: The utility of c after T, in the range [0,1].

Let $utility = 0$.
if `(status` c `Healthy)` **then** ▷ Characters want to be healthy.
    $utility += 0.7$
end if
if there exists an item i such that `(owns` c i`)` **then**
    if `(has` c i`)` **then** ▷ Characters want to have the things that they own.
        $utility += 0.3$
    else
        $utility += 0$
    end if
else
    $utility += 0.3$ ▷ The character does not own anything.
end if
if there exists a character h such that `(parent` c h`)` **then**
    return $0.4 \cdot utility + 0.6 \cdot$ UTILITY(h) ▷ Parents value the utility of their children higher.
else
    return $utility$
end if

Algorithm 2 A simple likelihood function $\pi(T)$ for some sequence of steps T in the example domain given in Figure 3.1.

Require: A sequence of n steps $T = \{s_1, s_2, ..., s_n\}$ and the current state of the story world.
Ensure: The likelihood that T will succeed, in the range [0,1].

```
return ∏_{i=1}^{n} LIKELIHOOD(s_i)

procedure LIKELIHOOD(step s)
    if s is an instance of (snakebite ?victim) then
        return 0.05                                   ▷ Snakebites are quite rare.
    end if
    if s is an instance of (steal ?thief ?item ?owner ?sheriff) then
        if (armed ?thief) then                        ▷ Armed robbers are more likely to succeed.
            return 0.8
        else
            return 0.2
        end if
    end if
    if s is an instance of (shoot ?killer ?victim) then
        if (armed ?victim) then                       ▷ Armed victims are harder to kill.
            return 0.5
        else
            return 1
        end if
    end if
    return 1                                          ▷ All other steps are 100% likely to happen as planned.
end procedure
```

Chapter 4

Evaluation of the CPOCL Model

Two empirical experiments were carried out to validate that the CPOCL model and the seven dimensions represent aspects of the human understanding of narrative conflict. The results demonstrate that a human audience can recognize and reason about thwarted plans in stories, and thus the CPOCL operationalization of conflict can be used facilitate this kind of narrative reasoning. It is important to emphasize that these experiments were *not* designed to validate the thwarted plans definition of conflict or the definitions of the seven dimensions given in a the previous chapter. It is assumed that these narratological definitions, which have been provided by experts and critically analyzed in the narratology community, will be useful for reasoning about stories. These experiments were designed to test whether or not the CPOCL model and the seven dimensions have accurately operationalized those definitions.

The first experiment evaluates how a human audience answers "who?" "why?" and "when?" questions about conflict. This corresponds to the three discrete dimensions: *participants*, *topic*, and *duration*. 27 subjects were given a simple natural language definition of conflict as thwarted plans and then asked to report the conflicts they noticed in three short text stories. When reporting a conflict, they provided a pair of characters, the action taken by the first character that thwarted an action by the second character, and the start and end time of the conflict. Subjects showed moderate agreement on which conflicts exist (Fleiss's $\kappa = 0.51$). When CPOCL is treated as a subject and asked to report conflicts based on threatened causal links, it also agreed closely with the human audience (accuracy = 0.92).

The second experiment was designed to evaluate the four continuous dimensions of conflict: *balance*, *directness*, *intensity*, and *resolution*. 30 subjects were given simple natural language definitions of these concepts and asked to rank 4 short text stories from highest to lowest for each dimension. Each story involved the same set of characters, locations, and objects but presented a different sequence of events designed to create high and low values for the formulas of those four dimensions. Subjects agreed on a best order to rank the stories in for all four

dimensions. The formulas presented in the previous chapter can also be treated as a subject and asked to rank the stories. For *balance*, *directness*, and *resolution* the ordering defined by the formulas was the most popular ordering according to the human audience. For *intensity*, the ordering defined by the formula was the fifth most popular ordering (of 24) according to the human audience.

4.1 Validating Participants, Topic, and Duration

The first experiment evaluated the plan-based structure of CPOCL, along with the three dimensions that can be directly observed in a CPOCL plan: *participants*, *topic*, and *duration* [79]. Human subjects were given three short stories and asked to list all the conflicts they observed. They also answered "who?" "why?" and "when?" questions for each conflict. The data collected was used to evaluate two hypotheses:

1. Subjects will report conflicts similarly to one another.

2. Subjects will report conflicts that are similar to those defined by CPOCL when the stories are modeled as CPOCL plans.

The experiment was conducted as an online survey, and subjects were recruited via e-mail and social networking websites. No compensation or incentives were offered. Subjects completed a tutorial to familiarize themselves with the interface, and then each subject reported conflicts for all three stories, which were presented in a random order. The stories were modeled as CPOCL plans and then translated into natural language using simple templates. The stories took place in three different domains: the American west (Figure 4.1), a medieval fantasy kingdom (Figure 4.2), and futuristic outer space (Figure 4.3).

27 people responded to the survey by finishing one or more stories. Of those, 23 users finished all three stories. There were 16 male and 11 female subjects. The most common age range was 26-35. In total, 486 conflicts were reported across the three stories. If a subject reported no conflicts for a story, that subject's data was not included in the analysis for that story.

Figure 4.4 shows the interface which subjects used to perform the annotations. It allowed subjects to move backward and forward through time at will. At each moment, they were shown the story up to that point along with thought bubbles for each character (including Fate) that described the character's current plan. Because CPOCL is a model of story *fabula*, subjects were intentionally given this god's-eye-view of the story.

Subjects were asked to list all the conflicts they noticed via a point-and-click interface. A conflict was reported as a 6-tuple $\langle c_1, c_2, s_1, s_2, b, e \rangle$ which was composed of:

The Snakebite
Once upon a time in the Wild West, there lived a cattle rancher named Hank and his young son Timmy. Not far from their ranch was a small town that had a saloon and a general store. William was the sheriff of the town, and it was his job to arrest and imprison anyone who broke the law. Carl owned the general store, and he sold all sort of things, including a powerful antivenom to cure snakebites. Then, one day...

Hank's son Timmy got bitten by a snake and became sick.
Hank went to the general store.
Hank tied up Carl.
Hank stole the antivenom from Carl.
William went to the general store.
William tied up Hank.
William took the antivenom from Hank.
William untied Carl.
William gave the antivenom to Carl.
William took Hank to jail.
(Hank escaped from jail.)
(Hank went to the Ranch.)
(Hank healed Timmy using the antivenom.)
Timmy died of his snakebite.

The end.

Figure 4.1: "The Snakebite," one of three stories annotated by human subjects in the experiment described in Section 4.1. The domain used for this story is a more expressive version of the domain in Figure 3.1. Steps in parentheses are non-executed steps.

True Riches
Once upon a time in a small village there lived a beautiful maiden named Talia. She was in love with a handsome thief named Rory, but Rory was too poor to support her. One day, Talia caught the eye of the kingdom's prince, Vince, and he also fell in love with Talia. Talia did not love Prince Vince, but he was very rich. Then one day...

Rory proposed to Talia.
Prince Vince proposed to Talia.
Gargax got hungry.
Gargax went to the village.
Gargax devoured Prince Vince.
(Talia married Prince Vince.)
Rory went to Gargax's cave.
Rory stole Gargax's treasure.
Rory went to the village.
Talia married Rory.

The end.

Figure 4.2: "True Riches," one of three stories annotated by human subjects in the experiment described in Section 4.1. Steps in parentheses are non-executed steps.

- c_1, the first character
- c_2, the second character
- s_1, an action from c_1's thought bubble
- s_2, an action from c_2's thought bubble that thwarts c_1's plan[1]
- b, the time when the conflict begins
- e, the time when the conflict ends

This information describes the *participants*, *topic*, and *duration* of each conflict. The order of participants was ignored—in other words $\langle c_1, c_2, s_1, s_2, b, e\rangle = \langle c_2, c_1, s_2, s_1, b, e\rangle$.

4.1.1 Inter-Subject Agreement

Before evaluating CPOCL, it had to be established that subjects agreed amongst themselves. For this, I used Fleiss's κ coefficient, a standard metric for measuring inter-rater agreement.

[1]Subjects were able to report any two steps that met these criteria. This made it possible to report two steps which do not thwart one another—that is, no effect of the first step negates any precondition of the second step and vice versa. However, none of these so-called invalid conflicts were ever reported by enough users to be considered correct conflicts.

The Lizard Beast of Mydrox
Many years in the future, space explorers will travel from planet to planet attempting to make peaceful contact with alien races. This is the story of Zoe, a space explorer orbiting the planet Mydrox in her starship. Deep in a cave on the planet Mydrox lives a dangerous Lizard Beast. One day...

Zoe teleported to the surface of planet Mydrox.
The Lizard Beast walked to the surface of planet Mydrox.
The Lizard Beast started a fight with Zoe, which made Zoe angry.
(Zoe slew the Lizard Beast of Mydrox.)
(The Lizard Beast of Mydrox slew Zoe.)
Zoe calmed the Lizard Beast with a soothing song.
A massive volcano on the surface of planet Mydrox began to erupt.
Zoe teleported to her ship.
The Lizard Beast walked to its underground nest.
A massive volcano erupted, covering the surface of planet Mydrox with magma, but no one was killed.
(Zoe made peace with the Lizard Beast.)

The end.

Figure 4.3: The Lizard Beast of Mydrox," one of three stories annotated by human subjects in the experiment described in Section 4.1. Steps in parentheses are non-executed steps.

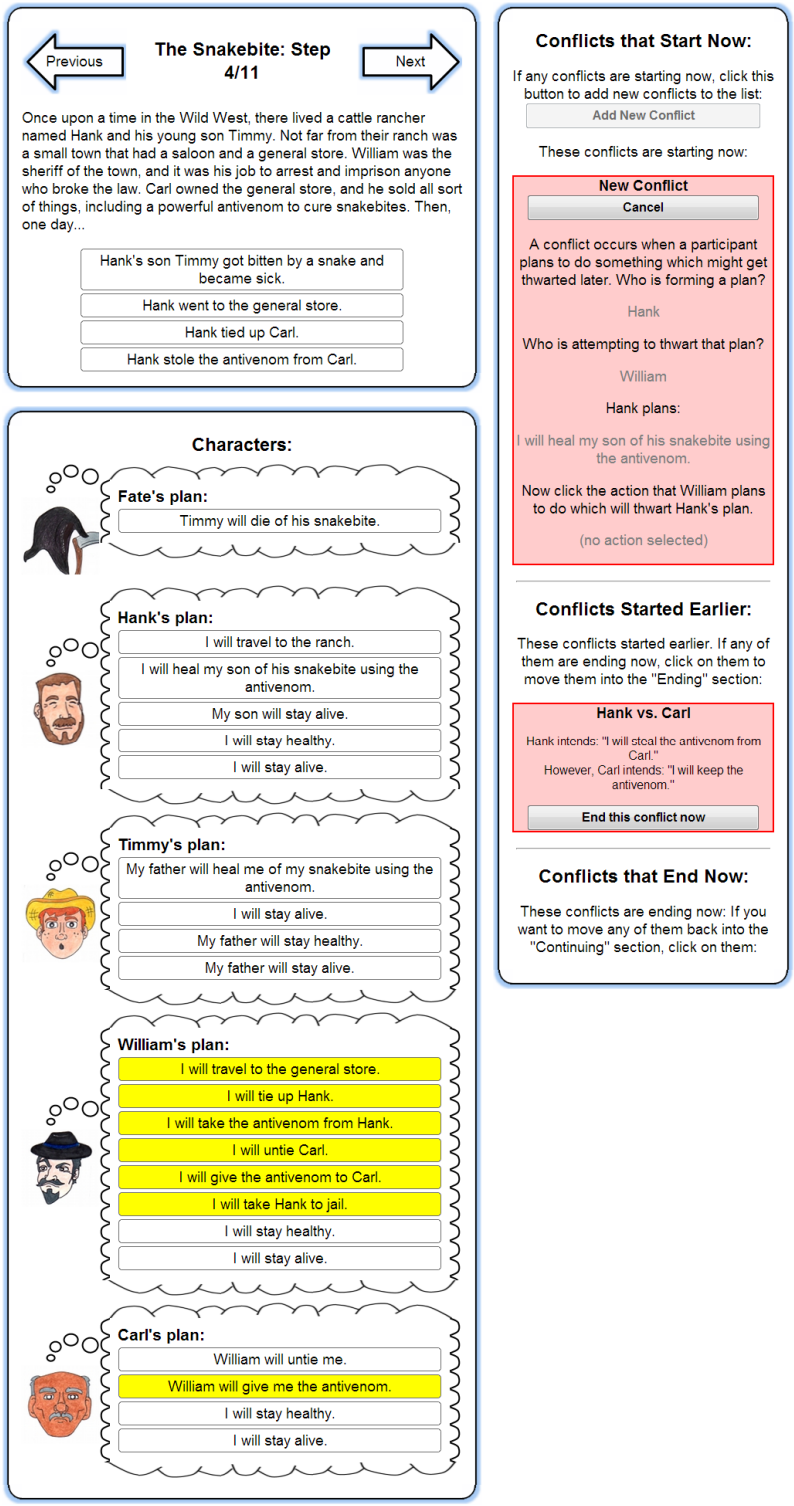


Figure 4.4: The interface by which human subjects annotated the stories in the experiment described in Section 4.1.

Table 4.1: Inter-subject agreement (Fleiss's κ) amongst subjects for the experiment described in Section 4.1.

		Exact Agreement	
Story	**Subjects**	**Questions Answered**	κ
The Snakebite	25	66	0.31
True Riches	24	35	0.33
The Lizard Beast of Mydrox	25	49	0.17
Average			**0.27**

		w/ Overlapping Durations	
Story	**Subjects**	**Questions Answered**	κ
The Snakebite	25	31	0.49
True Riches	24	18	0.59
The Lizard Beast of Mydrox	25	21	0.44
Average			**0.51**

Fleiss's κ is similar to Cohen's κ, except that it can be used for three or more raters. Fleiss's κ reaches 1 if users agreed completely and -1 if users disagreed completely. This calculation assumes that all participants answered a set multiple-choice questions, so in order to apply it I had to find a suitable interpretation of the data that was collected.

The most straightforward interpretation would be to consider every conflict that could possibly have been reported as a question that was implicitly answered as *true* if the user reported it or *false* if the user did not report it. However, this would artificially inflate the κ value with an abundance of true negatives (conflicts which were possible to report but were not reported). To account for this, I re-defined the range of "all possible conflicts" to be only those which were reported by at least 1 subject. This was less than 1% of all the conflicts that could possibly have been reported via the interface.

The first column of Table 4.1, labeled *Exact Agreement*, shows the κ values achieved for each story. *Subjects* is the number of subjects who finished that story. *Questions Answered* is the number of possible possible conflicts (i.e. number of questions) to which subjects implicitly answered "true" by reporting it or "false" by not reporting it. The average κ for all three stories was 0.27.

Many of the conflicts reported by subjects had the same participants, same topic, and overlapping (but not exactly the same) duration. To account for this, a second set of κ values was calculated such that these conflicts were considered the same. The results are shown in the second column of Table 4.1, labeled *w/ Overlapping Durations*. Allowing for overlapping duration reduced the range of reported conflicts by about half for each story and increased the

Table 4.2: Threshold values for each story

Story	**min θ**	**max θ**	**Average Accuracy**
Western	12 (48%)	19 (76%)	80%
Fantasy	8 (33%)	21 (88%)	81%
Space	4 (16%)	16 (64%)	80%

κ values. The average κ for all three stories when allowing for overlapping duration was 0.51.

Recall that any κ value above 0 represents agreement. Landis and Koch [40] published a set of labels associated with κ values. This table is not universally accepted, but it may be helpful in interpreting the results. Based on this table, subjects demonstrated "fair agreement" about which conflicts exist in the three stories, and "moderate agreement" when allowing for overlapping durations.

4.1.2 Subject Agreement with CPOCL

In order to evaluate CPOCL's performance, I had to establish which conflicts were considered correct out of all the ones reported. For each story, some threshold θ must be chosen such that if θ or more subjects reported a conflict, that conflict is defined as correct for that story. Choosing a θ value makes it possible evaluate the accuracy of an individual human subject. Keeping in mind that this data can be interpreted as a number of questions implicitly answered as *true* if the subject sees a conflict and *false* if the subject does not see it, I define accuracy to be the number of questions answered correctly divided by the total number of questions answered (conflicts with the same participants, the same topic, and overlapping durations are considered the same).

For each story, I chose the lowest value of θ that maximized the average accuracy of subjects. These θ values are given in Table 4.2 in the *min* θ column. Consider the Western story as an example. If I define a correct conflict as one reported by 12 or more subjects (that is, $\theta = 12$ or 48% of subjects), then the average subject's accuracy in reporting conflicts is 80%. $\theta = 12$ is the lowest value of θ that achieves the highest possible average accuracy of 80%. I could have chosen θ as high as 19 and observed the same average accuracy (given in Table 4.2 as *max* θ), but since many subjects reported exhaustion during the experiment, I chose the lowest θ in order to utilize as much data from subjects as possible. As Table 4.2 shows, there exists a θ value for each story such that the average subject achieves 80% or 81% accuracy. This is further evidence that subjects agree about which conflicts exist.

I then compared CPOCL's performance to both a naïve baseline and the performance of the average individual human subject for two tasks. For the first task, CPOCL was treated as a subject. The set of conflicts that it defines was compared to those reported by humans. The

Table 4.3: Confusion matrices for CPOCL's prediction of conflict in stories relative to human subjects.

The Snakebite	
True Pos: 10	False Pos: 4
False Neg: 0	True Neg: 17

True Riches	
True Pos: 4	False Pos: 4
False Neg: 0	True Neg: 11

The Lizard Beast of Mydrox	
True Pos: 6	False Pos: 3
False Neg: 0	True Neg: 12

resulting confusion matrices are shown in Table 4.3. A true positive is a conflict defined by CPOCL that at least θ subjects reported. A false positive is a conflict defined by CPOCL that fewer than θ subjects reported. A false negative is a conflict reported by at least θ subjects that CPOCL does not define. A true negative is a conflict which was not defined by CPOCL and reported by fewer than θ subjects. Recall that, in order to avoid a preponderance of true negatives, all the conflicts used in these calculations were reported by at least 1 subject. Summary statistics for this task are presented in Table 4.4.

CPOCL performs relatively well on this task considering the extremely low probability of guessing correctly. I define a random guess as follows: Choose two characters from the story at random; choose a start and end time at random such that the start time is less than or equal to the end time; choose two actions at random such that the first action is from one of the first character's intention frames, the second action is from one of the second character's intention frames, and both actions occur after the start time. For all three stories, the chances that a random guess was correct according to θ subjects was less than 0.02% (about 1 in 5000), even allowing for overlapping durations.

It is also possible to compare the model's performance to that of each individual user. Precision (true positive rate) is the most meaningful statistic for this comparison because it expresses the fraction of conflicts reported that were correct. These results are visualized in Figure 4.5. In short, CPOCL does much better than random guessing (a very naïve baseline) but not as well as the average human subject (the ideal).

Another way to evaluate the CPOCL model is to test how well it can recognize when a given pair of characters are in conflict. This second task asks this question both of subjects and of CPOCL: For every discrete time step, and for every pair of characters, are those characters

Table 4.4: CPOCL's accuracy, precision, and recall for both tasks.

	First Task		
Story	Accuracy	Precision	Recall
The Snakebite	0.87	0.71	1.00
True Riches	0.79	0.50	1.00
The Lizard Beast of Mydrox	0.86	0.67	1.00
Average	**0.84**	**0.63**	**1.00**

	Second Task		
Story	Accuracy	Precision	Recall
The Snakebite	0.98	0.90	1.00
True Riches	0.90	0.61	1.00
The Lizard Beast of Mydrox	0.87	0.82	0.86
Average	**0.92**	**0.77**	**0.95**

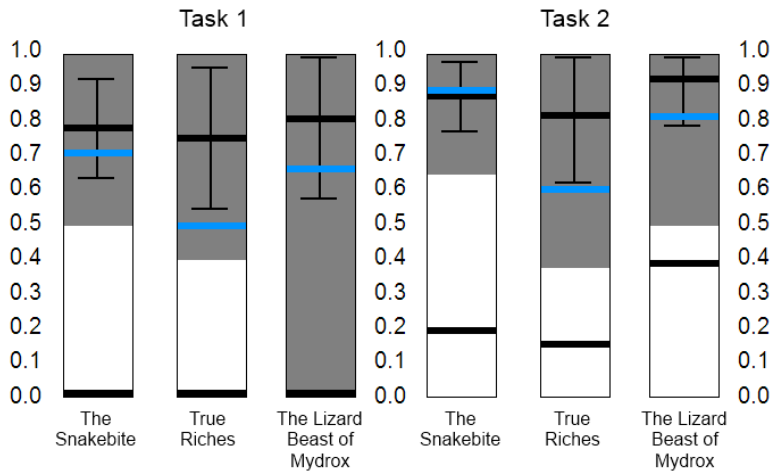


Figure 4.5: CPOCL's precision vs. a naïve baseline and the individual human subject for both tasks described in Section 4.1. The gray region represents the range of precision values for subjects. The higher solid black bar is the average user precision (+/- one standard deviation). The lower solid black bar is the precision of the random baseline. The solid blue bar is CPOCL's precision.

Table 4.5: CPOCL's performance on the second task relative to naïve baselines: always "no" and always "yes"

	Accuracy		
Story	**No**	**CPOCL**	**% Improvement**
The Snakebite	0.81	0.98	21%
True Riches	0.85	0.90	6%
The Lizard Beast of Mydrox	0.61	0.87	42%
Average			**23%**

	Precision		
Story	**Yes**	**CPOCL**	**% Improvement**
The Snakebite	0.19	0.90	462%
True Riches	0.15	0.61	395%
The Lizard Beast of Mydrox	0.39	0.82	110%
Average			**322%**

in conflict at that time? Figure 4.6 gives a visualization of the results. True positives indicate that users and CPOCL both answered "yes." True negatives indicate that users and CPOCL both answered "no." False positives indicate that CPOCL answers "yes," but users answers "no." False negatives indicate that users answers "yes," but CPOCL answered "no." Summary statistics for the recognition task are presented in Table 4.4.

A naïve baseline for this task is to always answer "yes" or "no" to every question. Answering "no" will yield the highest accuracy, and answering "yes" will yield the highest precision. I compared CPOCL's performance to these two models, and the results are presented in Table 4.5.

When compared to individual human users, CPOCL did better on the recognition task and even outperformed the average subject for the Western story (see Figure 4.5).

4.1.3 Discussion

Some conflicts defined by the model are understandably counter-intuitive to subjects due to a mismatch between how people think about actions and the knowledge representation of a STRIPS-style story domain. For example: William intends to take the antivenom from Hank, but Hank intends to travel back to his ranch. This conflict arises because the `take` action requires that both characters be at the same location (in this case, the general store). If Hank travels to his ranch, he will no longer be at the general store and the `take` action will fail. William can still take the antivenom from Hank, but he can no longer take the antivenom from Hank *at the general store*. This suggests that subjects do not think about steps in terms of

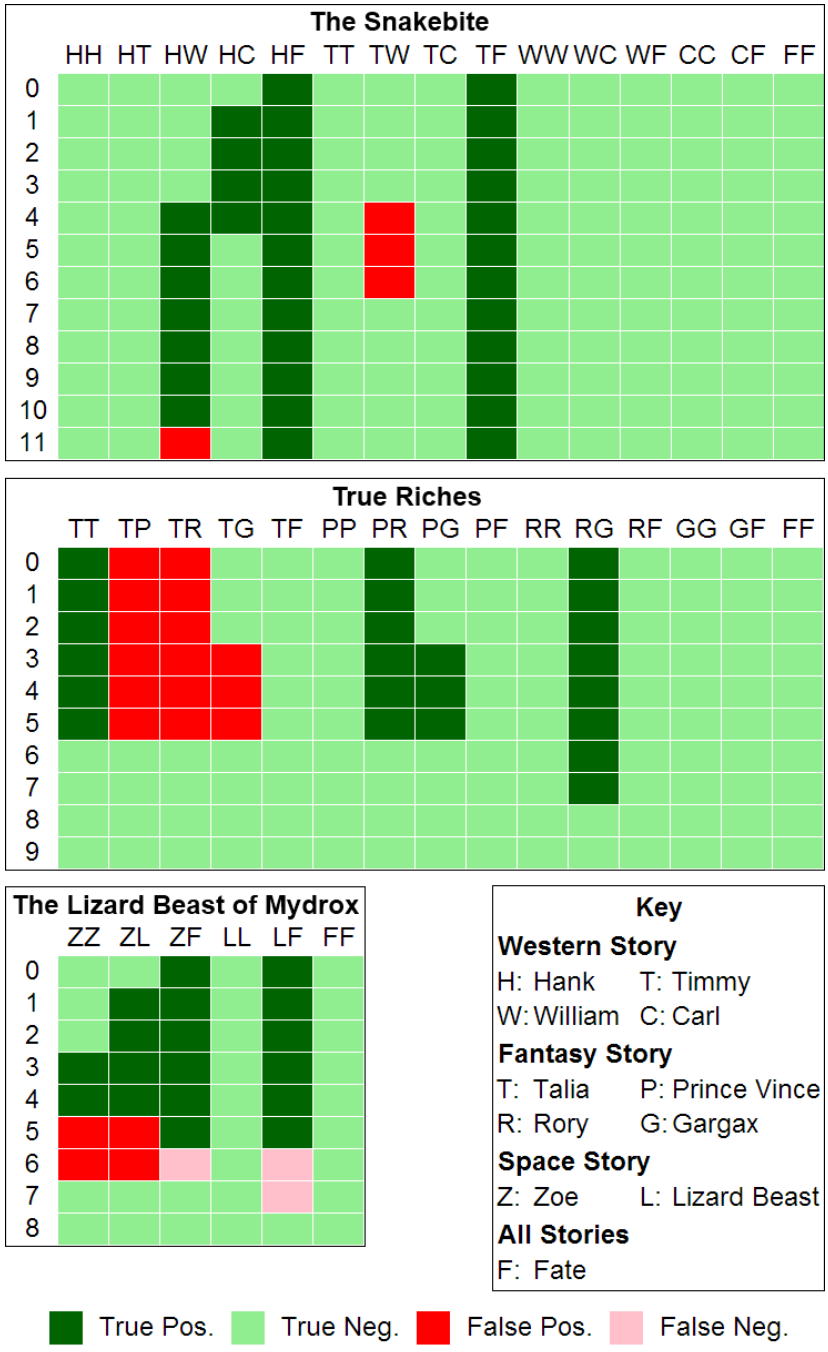


Figure 4.6: A visualization of CPOCL's performance on the second task described in Section 4.1. For each story, the vertical axis is labeled with the index of the state. The horizontal axis is every pair of characters.

their exact mechanics; rather, they think at a more abstract level where going to the general store is the same action no matter where the character is coming from.

Another potential source of confusion is that the classical planning model on which CPOCL is built does not support durative actions[2]—that is, all steps are assumed to happen immediately. A durative action must be represented as multiple actions; e.g. Timmy is bitten by a snake and then later dies from the snakebite. In the Space story, two characters intend to stay safe from natural disasters, but a volcano erupts which is unsafe for everyone in that area. According to CPOCL, this is the *end* of a conflict; Fate won because both characters failed to stay safe (which caused them to form new plans to go to safe locations). Subjects recognized these conflicts, but they reported the end as the time when each character had reached a new safe location. This seems like the most natural interpretation of the story, so CPOCL may need to be extended with research from automated scheduling to represent durative actions.

Interestingly, no false negatives were predicted for any story in the first task. Even in the second task, the only false negatives were due to the above disagreement about how long the conflict with the volcano should last, not about who was involved. In other words, the conflicts defined by CPOCL are a strict superset of the correct conflicts according to subjects. One important direction of future work will be to discover why subjects report some conflicts but not others. Certain threatened causal links are very obvious to subjects, while others (that are not formally or structurally different) seem not to be obvious at all. This may have been due to subject exhaustion—the annotation process was very taxing—or it might be explained by how subjects direct their attention while reading. Some promising research on this topic [16] is already underway which may be able to extend CPOCL in important ways.

4.2 Validating Balance, Directness, Intensity, and Resolution

A second experiment was designed to validate the four continuous dimensions of conflict—*balance*, *directness*, *intensity*, and *resolution* [80]. Subjects were given natural language descriptions of the dimension formulas and asked to rank four stories based on those descriptions.

Predicting the exact numerical value a subject will report for some dimension is difficult considering how sensitive these concepts are to subtleties of interpretation. Simply predicting high or low is easier, but success would provide less support for the CPOCL model. I attempted to reach a middle ground between these two extremes by demonstrating that the CPOCL dimension formulas can rank 4 stories in the same order as human subjects. If subjects agree on an ordering, and if that ordering agrees with the predictions made by CPOCL, it is assumed that the formulas can approximate these four dimensions of conflict.

[2] *Durative actions* is one term used in the planning literature to refer to actions with temporal extent. In the context of Markov Decision Processes, such actions are often called *options* [69].

Introduction
This story takes place in a magical kingdom ruled by a wealthy king. The king has a young son, the prince. You are just a poor farmer, but you are friends with the prince. One day, an evil sorcerer kidnaps the prince! The king offers you a reward if you can get the prince home safely.

Story A
You travel to the city.
You ask a knight to kill the sorcerer.
The knight buys a sharp sword at the market.
The knight travels to the tower.
The knight challenges the sorcerer to a fight to the death.
The sorcerer reveals that he is your father.
The knight defeats the sorcerer.
The prince travels to the city.
The king gives you a bag of gold.
The king makes you a knight.

Story B
You travel to the city.
You ask a knight to kill the sorcerer.
The knight buys a sharp sword at the market.
The knight buys a suit of armor at the market.
The knight travels to the tower.
The knight challenges the sorcerer to a fight to the death.
The sorcerer threatens to kill the prince.
The knight defeats the sorcerer.
The prince travels to the city.
The king gives you a bag of gold.

Story C
You travel to the tower.
You challenge the sorcerer to a fight to the death.
The sorcerer reveals that he is your father.
The sorcerer threatens to kill the prince.
You defeat the sorcerer.
The prince travels to the city.
You travel to the city.

Story D
You travel to the city.
You buy a sharp sword at the market.
You travel to the tower.
You challenge the sorcerer to a fight to the death.
The sorcerer reveals that he is your father.
The sorcerer and you become friends.
The sorcerer defeats you.

Figure 4.7: The four stories ranked by subjects in the experiment described in Section 4.2. Each story had the same beginning but a different middle and end.

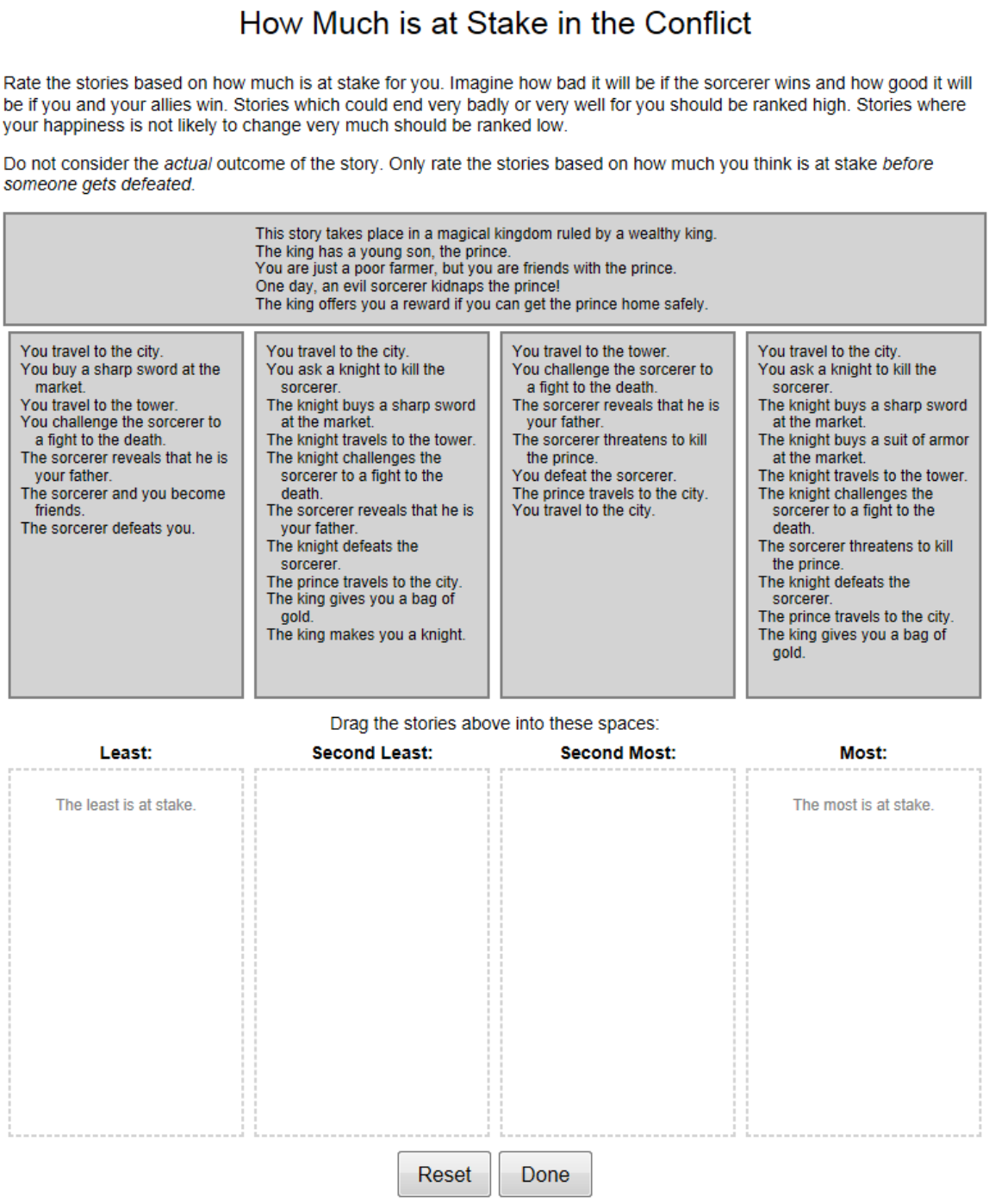


Figure 4.8: The interface which subjects used to rank the four stories in the experiment described in Section 4.2.

The study was conducted via a web interface in which subjects could drag and drop stories from an initial random order into a sorted order of their choosing. The interface is shown in Figure 4.8. Each subject ranked the same four stories for all four dimensions. Dimensions were presented to each subject in a random order. Subjects were recruited via e-mail, social networking websites, and online message boards. No compensation was offered. 30 people responded—19 males and 11 females with the most common age range being 26 to 35. No particular effort was made to avoid collecting participants who had participated in the other study described above because none of the stories, interface, or text were used in common between the studies.

The four stories used are given in Figure 4.7. Each story is narrated in second person, has the same beginning, takes place in the same domain, involves the same characters, and centers around a conflict between the subject and an evil sorcerer. The differences in the story were designed to affect their dimension values. For example, if you fight the sorcerer yourself, the conflict is more direct than if you ask the knight to fight for you. When the sorcerer threatens to kill the prince, the conflict is more intense than if he makes no threats. This experiment does not require a commitment to specific formulas for utility(a, T) and $\pi(T)$ as long as those formulas produce the predicted orderings. For example, I assume that the knight is more likely to succeed when he has a sword and armor than when he has just a sword and no armor. It is not necessary to measure the exact difference in π between the two stories.

The content of the stories was structured so that, given the orderings for each dimension predicted by the formulas, no two stories would appear at the same index for the same dimension. That is, the second most *intense* story was never ranked second for any other dimension. Subjects were not told of this constraint. It was imposed in an attempt to avoid conflating dimensions. For example, if the formulas assigned the same ordering to *balance* and *intensity* and subjects ranked stories in that order for both dimensions, it would be impossible to know whether they perceived *balance* and *intensity* as two different phenomena or if they were conflating the two.

When designing this experiment, I was faced with the difficult choice of which stories to use. Modeling existing short stories, such as fables, would demonstrate the applicability of CPOCL to stories which were not generated by a planner. However, I needed four different stories with specific combinations of the four dimension values, and I wished to control for content and length as much as possible. No such stories could be found among existing narratives. As a result, I decided to create a story domain which allowed me to craft four stories using the same setting and characters but with different values for each dimension. While this may limit the applicability of the results, it establishes a better connection between the independent and dependent variables.

Note that this experiment was *not* designed to test whether or not my definitions of the

Balance
Rate the stories based on how likely you and your allies are to win out over the sorcerer. If you expect your team to win, rate the story high. If you expect your team to lose, rate it low. Do not consider whether or not you *actually* win. Only rate the stories based on what you expected to happen *before someone gets defeated*.

Directness
Rate the stories based on how close you are to the sorcerer. There are many kinds of closeness: physical closeness, emotional closeness, familial closeness, etc. Only consider the distance between *you* and the sorcerer.

Intensity
Rate the stories based on how much is at stake for you. Imagine how bad it will be if the sorcerer wins and how good it will be if you and your allies win. Stories which could end very badly or very well for you should be ranked high. Stories where your happiness is not likely to change very much should be ranked low. Do not consider the *actual* outcome of the story. Only rate the stories based on how much you think is at stake *before someone gets defeated*.

Resolution
Rate the stories based on how much better off you are at the end. How much happier are you at the end of the story than at the beginning? Only consider how *you* have been affected. Do not consider how things *might have been*, only how they actually happened.

Figure 4.9: The descriptions of dimensions given to human subjects when sorting stories in the experiment described in Section 4.2.

dimensions correspond to those of subjects. The decision of which dimensions to model and how to define them was based on a synthesis of various narratological sources. The usefulness of these dimensions is supported by the authority of those sources, and I make no claim that these are the only dimensions worth modeling. This experiment was designed to test whether or not subject can recognize variations in my operationalization of these dimensions at the textual level. Thus, to avoid confusion from vocabulary, the dimensions were not given names in the study. Subjects were simply given the natural language descriptions of the dimensions shown in Figure 4.9.

The data collected was used to evaluate two hypotheses:

1. Subjects will rank stories similarly to one another.
2. Subjects will rank stories similarly to the CPOCL formulas.

Specifically, the formulas predicted the following orderings:

Balance: Balance measures the relative likelihood of the subject and his allies to succeed. If the subject is likely to prevail, then balance is high. If the sorcerer is is more likely to prevail, then balance is low. The formulas specify this ordering for balance (from lowest to highest):

1. **C:** The protagonist (a poor farmer) fights the sorcerer with no equipment.
2. **D:** The protagonist fights the sorcerer after buying a sword.
3. **A:** The knight (acting on behalf of the protagonist) fights the sorcerer after buying a sword.
4. **B:** The knight fights the sorcerer after buying a sword and armor.

Directness: Directness measures various kinds of closeness between the subject and the sorcerer. In stories A and B, the protagonist is interpersonally far from the sorcerer because a knight fights on his behalf. The formulas specify this ordering:

1. **B:** The protagonist and sorcerer are enemies, not related, and the knight fights for the protagonist.
2. **A:** The protagonist and sorcerer are enemies, family, and the knight fights for the protagonist.
3. **C:** The protagonist and sorcerer are enemies, family, and they fight face to face.
4. **D:** The protagonist and sorcerer are friends, family, and they fight face to face.

Intensity: Intensity measures the stakes of the conflict with the sorcerer. Before starting the study, participants are asked to make two assumptions which are relevant to the utility function: it is better to be rich than poor, and the subject values his own life higher than the lives of other characters. The formulas specify this ordering:

1. **A:** The life of the protagonist is not at stake because the knight fights the sorcerer. The life of the prince (a friend of the protagonist) is not at stake.
2. **B:** The prince's life is at stake.
3. **D:** The protagonist's life is at stake.
4. **C:** Both the protagonist's life and the prince's life are at stake.

Resolution: Resolution measures the change in utility that the subject experiences relative to the beginning of the story. The formulas specify this ordering:

1. **D:** The protagonist dies.
2. **C:** The protagonist succeeds but receives no reward.
3. **B:** The protagonist succeeds and is rewarded with money.
4. **A:** The protagonist succeeds and is rewarded with both money and knighthood.

4.2.1 Analysis

The data collected from each participant was an ordering of four stories for each dimension. The task of choosing an ordering is similar to classification, but it is important to note that two orderings can still be substantially similar even if they are not exactly identical. Capturing this degree of similarity is important, which precludes certain standard statistical tests.

For example, Cohen's or Fleiss's κ coefficient is often used to measure inter-rater reliability, but κ assumes that the raters are choosing one of several discrete categories. The orderings {A B C D} and {A B D C} would be considered two different categories even though 5 of the 6 pairwise orderings are the same in both; in other words, A comes before B in both; A comes before C in both; etc. The various edit distance metrics, such as Hamming distance [31], suffer from similar problems. The Hamming distance between {A B C D} and {D A B C} is 4, the maximum possible.

To account for similarity between responses, Kendall's τ distance [38] was used to compare orderings. τ counts the number of pairwise differences between two lists. Formally, let $\text{index}(x, S) = 1$ just when x is the first element in ordered set S, $\text{index}(x, S) = 2$ just when x is the second element in ordered set S, etc. Given two ordered sets M and N, an *inversion* is an ordered pair of elements (x, y) such that $\text{index}(x, M) < \text{index}(y, M)$ and $\text{index}(x, N) > \text{index}(y, N)$. This means that x is ordered before y in M, but x is ordered after y in N. The τ distance between two ordered sets can be expressed as $\tau(M, N)$ and is equal to the number of inversions that exist between M and N. Kendall's τ distance is symmetric,

meaning $\tau(M, N) = \tau(N, M)$. In the study of sorting algorithms, Kendall's τ is often used as a measure of "sortedness."

When comparing two orderings of length 4, the minimum τ distance is 0, which occurs when both orderings are the same. The maximum τ distance is 6, which occurs when one ordering is the reverse of the other. The τ distance between {A, B, C, D} and {D, C, B, A} is 6 because the pairs {A, B}, {A, C}, {A, D}, {B, C}, {B, D}, and {C, D} are inverted. If one fixes M and chooses N at random, assuming that all 24 permutations of the 4 stories are equally likely, then on average there will be a τ distance of 3 between M and N.

In order to evaluate the dimension formulas, it has to be determined what ordering of stories was the correct one—that is, the most popular ordering based on the data. I chose the ordering with the lowest average τ distance from each participant's ordering. For a given dimension of conflict, let $\{p_1, p_2, ..., p_n\}$ be the orderings chosen by the n participants for that dimension (here, $n = 30$). Let M be all 24 possible orderings of the 4 stories. For each of the 24 possible orderings, m, its average τ distance is:

$$\tau_{\text{avg}}(m) = \frac{\sum_{i=1}^{n} \tau(m, p_i)}{n}$$

As an example, consider m ={A B C D}, the first of the 24 permutations in M. To calculate τ_{avg} for m for the dimension of *balance*, we calculate $\tau(\{\text{A B C D}\}, p_i)$ for all 30 orderings p_i that were reported by the subjects for *balance*; then we average those 30 values. An ordering's τ_{avg} can be thought of as its average distance from each person's answer. When an ordering's τ_{avg} is low, that ordering is more popular—it agrees more with the orderings reported by the subjects. If all 30 subjects had reported the same ordering, that ordering's τ_{avg} would be 0 and the reverse ordering would have the max τ_{avg} of 6. The most popular orderings for each dimension are given in the first row of Table 4.7.

4.2.2 Inter-Subject Agreement

Before demonstrating to what extent my formulas agree with human subjects, I must first demonstrate that subjects agree amongst themselves. In other words, I wish to know how strongly the participants agree that the most popular ordering is correct.

As discussed above, there is no clear way to calculate Fleiss's κ coefficient to measure inter-subject agreement for this data. However, it is possible to express agreement by comparing the data, shown in Figure 4.11, to distributions representing agreement and disagreement, shown in Figure 4.10:

- **Perfect Agreement:** If users agreed completely with one another, they would all report the exact same ordering for a dimension.

- **Relative Agreement:** Given the subjective nature of how people perceive stories, it may be impossible to achieve perfect agreement. It is more realistic to compare against a distribution which indicates high (but not perfect) agreement. I decided to use one such distribution for analyzing the results, which is given in Figure 4.10. This distribution assumes that $\frac{2}{3}$ of the participants will choose the most popular ordering, and then the function will decay exponentially by a factor of 3. This distribution was chosen arbitrarily as representing the results I expected to see.

- **Disagreement:** If there is complete disagreement, one would expect answers to appear as if they were given at random. This would result in a uniform distribution across the 24 possible permutations for the 4 stories. That uniform distribution, when plotted as τ distance from the most popular ordering, is a roughly normal distribution (as seen in Figure 4.10).

As a null hypothesis, I assume the observed distributions for each dimension will fit the *disagreement* distribution. To evaluate this, I used Fisher's exact test, which is similar to the χ^2 test but performs better for distributions with small expected values [27]. For all four dimensions, there was a statistically significant difference between the data and the *disagreement* distribution (for *balance* $p = 0.003$, for *directness* $p < 0.001$, for *intensity* $p = 0.028$, and for *resolution* $p < 0.001$). The null hypothesis was rejected—that is, participants do not disagree.

Then I evaluated the alternative hypothesis—that users agree on the most popular ordering. For this, I employed a metric for measuring the similarity of two distributions called Bhattacharyya distance [7]. Bhattacharyya distance is 0 when two distributions are the same, and approaches 1 as the distributions become less similar. Given two discrete probability distributions p and q over domain X:

$$D_{Bhattacharyya} = -\ln\left(\sum_{x \in X} \sqrt{p(x)q(x)}\right)$$

For each dimension, I wanted to know if the distribution defined by subjects was most similar to the *perfect agreement*, *relative agreement*, or *disagreement* distribution. Table 4.6 demonstrates that the dimensions of *directness* and *resolution* were more similar to the *perfect agreement* distribution than they were to the *disagreement* distribution; however, the dimensions of *balance* and *intensity* are more similar to *disagreement* than to *perfect agreement*. However, all four dimensions are most similar to the *relative agreement* distribution. These results support the hypothesis that subjects agree amongst themselves on a correct ordering for the four dimensions, especially for *directness* and *resolution*.

Table 4.6: The Bhattacharyya distances between the observed distributions for each dimension and the *Perfect Agreement*, *Relative Agreement*, and *Disagreement* distributions given in Figure 4.10. The lowest distance is italicized for each dimension.

Dimension	**Perfect Agreement**	**Relative Agreement**	**Disagreement**
Balance	0.314	*0.108*	0.240
Directness	0.255	*0.037*	0.619
Intensity	0.465	*0.168*	0.175
Resolution	0.314	*0.040*	0.650

Table 4.7: The top 6 orderings and the bottom ordering for each dimension based the on average τ distance. The orderings predicted by the formulas are italicized.

Balance		**Directness**		**Intensity**		**Resolution**	
Ordering	τ_{avg}	**Ordering**	τ_{avg}	**Ordering**	τ_{avg}	**Ordering**	τ_{avg}
CDAB	*1.2667*	*BACD*	*0.5667*	BACD	1.7333	*DCBA*	*0.6667*
CDBA	1.6667	BADC	0.9667	BADC	1.9333	DCAB	1.2000
DCAB	1.7333	ABCD	1.3667	ABCD	2.1333	CDBA	1.4000
CADB	2.0000	BCAD	1.3667	BCAD	2.2667	DBCA	1.4000
DCBA	2.1333	ABDC	1.7667	*ABDC*	*2.3333*	CDAB	1.9333
CBDA	2.2667	BDAC	1.9000	BDAC	2.3333	DACB	1.9333
...17...	...17...	...17...	...17...	...17...	...17...	...17...	...17...
BADC	4.7333	DCAB	5.4333	DCAB	4.2667	ABCD	5.3333

4.2.3 Subject Agreement with CPOCL

For each dimension of conflict, Table 4.7 presents the 6 orderings with the lowest τ_{avg} (the top 6 best orderings for that dimension according to subjects). The orderings predicted by the formulas are italicized. For the dimensions of *balance*, *directness*, and *resolution*, the ordering predicted by the formula had the lowest τ_{avg}. For the dimension of *intensity*, the ordering predicated by the formula had the 5^{th} lowest τ_{avg}. These results support the hypothesis that participants will rank stories in the same order as my metrics. The formula for *intensity* may need to be improved based on these results to better agree with human perceptions.

4.2.4 Discussion

These initial results are promising, especially for *balance*, *directness*, and *resolution*. Several factors may have contributed to the observed disagreement.

Subjects may have misunderstood the descriptions of one or more dimensions, which were

intentionally brief and targeted at a high school reading level. I attempted to address this by running a small pilot study before the experiment, which provided valuable feedback on how to clarify the definitions. *Intensity* was the most widely misunderstood dimension during the pilot. It is also possible that subjects misunderstood the events of the story. At least one subject indicated a misunderstanding of the outcome of story D. To make the stories more G-rated, I used the template "X defeats Y" to express the `kill` action, which does not make it explicit that Y is killed. The predicted ordering for intensity is based on which characters' lives are at stake, so this may have caused confusion.

I assumed that each dimension could be measured independently of the others, but it is possible that participants perceived synergies between them. For example, if much was at stake (high *intensity*) but there was little chance that the sorcerer would prevail (low *balance*), subjects might have given the story a low ranking for *intensity*. This may explain why story C is ordered before story D in the most popular ordering for intensity. I hope to investigate how dimensions influence one another in future work.

The two dimensions that showed the least subject agreement—*balance* and *intensity*—require the subject to measure them independently of the actual outcome of the story. If the protagonist appears likely to prevail, balance should be high regardless of whether or not he or she actually prevails. At least two subjects reported difficulty ignoring their knowledge of the outcome. In future versions of this study, rather than ask subjects to ignore the ending, I intend to leave the ending out. This may help to avoid the bias introduced by foreknowledge.

4.3 Conclusion

This section has described my work to date on representing conflict in the data structures of AI plans. This work extends a previously studied model of intentional plans and incorporates operationalizations of narratological principles that can be used to describe individual conflicts. Given this means of representing conflict, I now turn to the process of generating stories using this model.

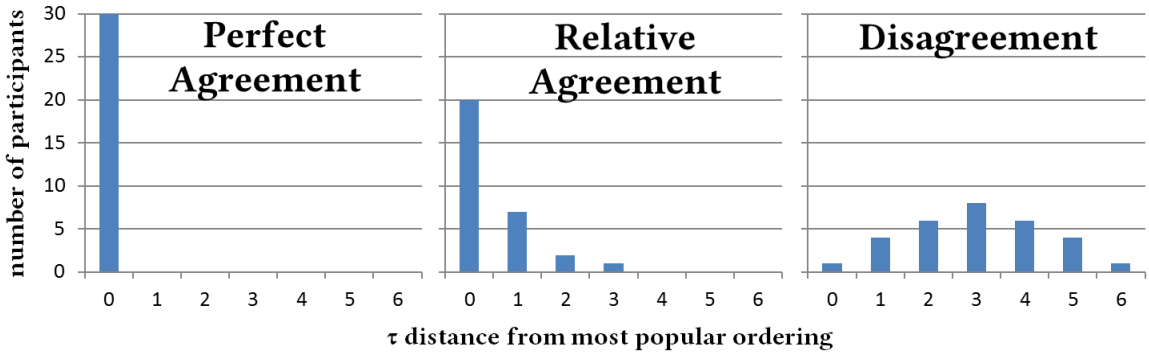


Figure 4.10: The three distributions against which the data was compared. These histograms show how many participants (y axis) chose an ordering that was some τ distance (x axis) away from the most popular ordering for each dimension.

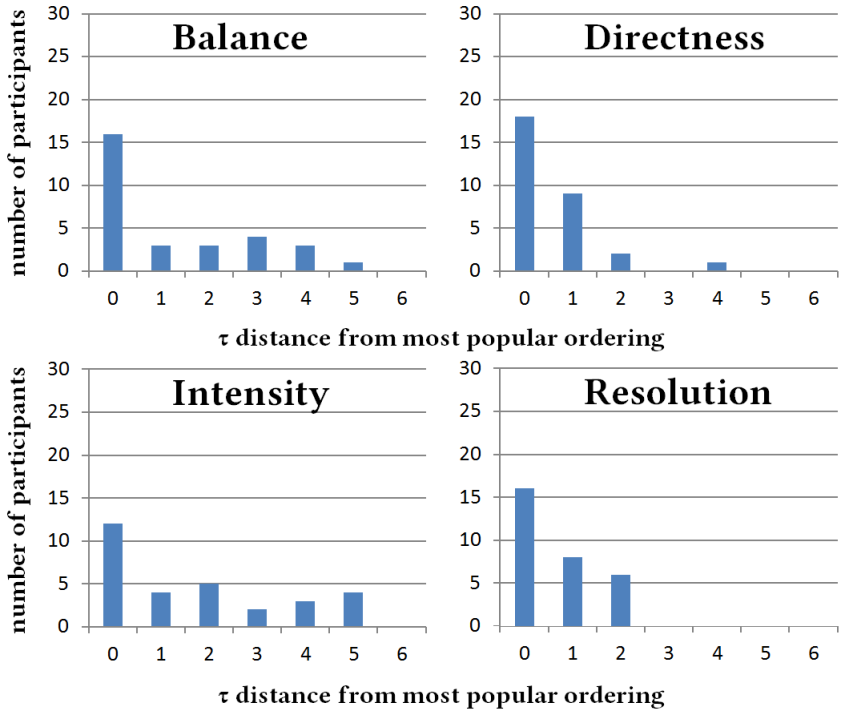


Figure 4.11: The observed distributions for each dimension. These histograms show how many participants (y axis) chose an ordering that was some τ distance (x axis) away from the most popular ordering for each dimension.

Chapter 5

Planning Algorithms Supporting Conflict

Planning algorithms can be broadly divided into plan-space search and state-space search. Both define a search space, but the structure of that space and how it is explored are different. As described in Section 2.4, the plan-space family benefits from a highly-expressive knowledge representation while the state-space family is generally faster. This section presents two algorithms that reason directly about intentionality and conflict: the plan-space CPOCL planner and the state-space Glaive planner.

The CPOCL algorithm works directly with the CPOCL knowledge representation. It begins with an empty plan which is annotated with flaws that describe what is needed to make the plan into a solution. During each iteration of the algorithm a flaw is chosen and something is added to the plan to fix the flaw. However, adding new element to the plan may in turn create new flaws. A CPOCL plan can have four kinds of flaws. The first two are drawn from the original POCL algorithm [48, 56] and the others are drawn from the IPOCL algorithm described by Reidl and Young [62]. Firstly, there may exist a precondition which is not yet satisfied. This is fixed by adding a causal link from a new or existing step to satisfy it [48]. Secondly, a causal link might be threatened without being a conflict link. This is fixed by reordering the steps or imposing variable bindings to avoid the threat [56]. Thirdly, an intention frame might not have a step to achieve its goal. This is fixed by using an existing or new step to achieve the goal [62]. Finally, an orphan step might need to be added to an intention frame. This is fixed by either adding to step to the frame or choosing not to add it [62].

Like the POCL and IPOCL algorithms on which it is built, CPOCL is a least commitment algorithm. When a new step that one or more characters must consent to is added to a CPOCL plan, it begins as non-executed step. If the step ever becomes the causal ancestor of an executed step, it too must then be marked as executed. This process ensures two things: only steps which

need to be executed are marked as executed, and no illegal causal links are created that run from a non-executed step to an executed step.

The Glaive algorithm operates very differently. Glaive begins at the current state of the problem. During each iteration of the algorithm it chooses some fully ground step whose precondition is satisfied in the current state, adds that step the plan, and updates the current state based on the effects of that step. Like the Fast-Forward algorithm on which it is built, Glaive builds a fully ground totally ordered plan until it reaches a state in which the goal is true.

To ensure that each step has an explanation in terms of the goals of its characters, Glaive also tracks causal links and all current character goals. Glaive reasons about intentional paths—chains of steps in service of a single goal—to find explanations for steps in terms of character goals. Only plans which achieve the author's goal and have no unexplained steps are solutions.

Unlike CPOCL, Glaive does not reason directly about non-executed steps. Rather, it considers every node in its search space a possible world. Recognizing partially executed and failed plans (and thus conflict) is accomplished by comparing two or more of these possible worlds to one another. A partially executed plan can appear in a solution as long as there exists some other node in the search space where that plan was completed and used to achieve some character goal. Two possible worlds can be combined to create a single story; the steps of one are simply added to the other as non-executed steps.

Glaive is a fairly simple extension of a classical state-space planner. Like other state-space planners, most of its intelligence can be found in the heuristic which estimates how far a given state is from the goal. The Fast-Forward planner evaluates a state by considering how hard it would be to reach the goal if steps never removed literals from the current state but only added them. Glaive uses this estimate, which is calculated by reasoning forward form the current state to the goal, but it also uses a second value which is calculated by reasoning backward from each character goal. Each unexplained step that exists in a plan will need to be explained before a solution can be found. Glaive uses a data structure called a goal graph to estimate how many more steps need to be taken before an unexplained step becomes explained. This additional reasoning about the intentional structure of the solution allows Glaive to calculate a more accurate estimate of how far the goal is from any given state.

5.1 The CPOCL Algorithm

Algorithm 3 gives the Conflict Partial Order Causal Link planning algorithm which produces CPOCL plans. This algorithm extends the classical POCL algorithm [82] and incorporates intentional planning similarly to the IPOCL algorithm described by Riedl and Young [62]. I assume a function MGU(p, q) which takes as parameters two predicate literals p and q and

Algorithm 3 The CPOCL (Conflict Partial Order Causal Link) planning algorithm.

CPOCL ($\Pi = \langle S, B, O, L, I\rangle$, Λ, F)

Π is a plan, initially the null plan; Λ a set of operators; F a set of flaws, initially open precondition flaws for unsatisfied preconditions of the end step and unsatisfied intention frame flaws for start step effects like `(intends c g)`.

Termination: If B or O is inconsistent, fail. If $F = \emptyset$ and Π has no orphans, return Π. If orphans exist, fail.
Plan Refinement: Choose a flaw $f \in F$. Let $F' = F - \{f\}$.
    Goal Planning: If f is open precondition flaw $f = \langle s_{need}, p\rangle$, let s_{add} be a step $\langle P, E, C\rangle$ such that $p \in E$.
        Choose s_{add} in one of two ways:
            Reuse: Choose s_{add} from S.
            New Step: Create s_{add} from an operator in Λ with effect p. Let $S' = S + \{s_{add}\}$.
                For each precondition pre of s_{add}, add new open precondition flaw $\langle s_{add}, pre\rangle$ to F'.
                Mark s_{add} as non-executed.
        Link: Create causal link $l = s_{add} \xrightarrow{p} s_{need}$. Let $L' = L + \{l\}$, $B' = B \cup \text{MGU}(e, p)$, $O' = O + \{s_{add} < s_{need}\}$.
        Execution Marking: If s_{need} is executed, mark s_{add} and all its causal ancestors as executed.
        Happening Frame: If $P = \emptyset$, create new intention frame $r = \langle fate, \emptyset, s_{add}, s_{add}, \{s_{add}\}\rangle$. Let $I' = I + \{r\}$.
        New Frames: For each effect of s_{add} like $intends(c, g)$:
            Create new intention frame $r = \langle c, g, s_{add}, \emptyset, \emptyset\rangle$. Let $I' = I + \{r\}$.
            Add new unsatisfied intention frame flaw $\langle r\rangle$ to F'.
        Intent Flaws: For each intention frame $r = \langle c, g, \sigma, m, T\rangle \in I'$:
            If $s_{add} \notin T$ and $s_{need} \in T$ and $c \in C$ for s_{add}, add new intent flaw $\langle s_{add}, r\rangle$ to F'.
    Threat Resolution: If f is threatened causal link flaw $f = \left\langle s \xrightarrow{p} u, t \right\rangle$, choose how to prevent the threat:
        Promotion: Let $O' = O' + \{t < s\}$.
        Demotion: Let $O' = O' + \{u < t\}$.
        Restriction: Add bindings to B' which cause the threatening effect of t not to unify with p.
    Satisfaction: If f is unsatisfied intention frame flaw $f = \langle r = \langle c, g, m, \emptyset, T\rangle\rangle$, let s_{sat} be a step with effect g.
        Choose s_{sat} the way s_{add} is chosen (**Reuse** or **New Step**) or by **Persistence**.
            Persistence: Make a persistence step $s_{sat} = \langle\{g\}, \{g\}, \{c\}, false\rangle$. Let $O' = O + \{s_{sat} = s_{end}\}$.
        Let $T' = T + \{s_{sat}\}$. Let $r' = \langle c, g, m, s_{sat}, T'\rangle$. Let $I' = I - \{r\} + \{r'\}$.
    Intent Planning: If f is an intent flaw $f = \langle s_{orphan}, r = \langle c, g, m, \sigma, T\rangle\rangle$, choose how to handle s_{orphan}:
        Inclusion: Let $T' = T + \{s_{orphan}\}$. Let $r' = \langle c, g, m, \sigma, T'\rangle$, $I' = I - \{r\} + \{r'\}$, $O' = O + \{m < s_{orphan}\}$.
            For each causal link $s \xrightarrow{p} s_{orphan} \in L$, if $c \in C$ for s, add new intent flaw $\langle s, r'\rangle$ to F'.
        Exclusion: Do nothing.
Threat Detection: If any casual link $l \in L'$ is threatened by step $\theta \in S'$ and l is not a conflict link,
    Add new threatened causal link flaw $\langle l, \theta\rangle$ to F'.
Recursive Invocation: Call CPOCL ($\Pi' = \langle S', B', O', L', I'\rangle$, F', Λ).

returns a set of variable bindings to make $p = q$. Line numbers in the descriptions below correspond to lines in Algorithm 3.

The POCL family of algorithms are a kind of refinement search [37]. A node in the search space of a POCL algorithm represents a partial plan. Partial plans are annotated with a set of flaws which describe how that partial plan is incomplete. These flaws are iteratively fixed until a flawless (and thus, complete) plan is found or the algorithm fails.

Definition 41 (search space, refinement planning)**.** The *search space of a refinement planner* is a directed graph. A node in the graph represents either a partial plan annotated with flaws or a complete plan. A directed edge $p_1 \xrightarrow{f} p_2$ runs from a partial plan p_1 to a plan p_2 and represents the transformation of p_1 into p_2 by repairing flaw f.

The root of the search space is the null plan:

Definition 42 (null plan)**.** A planning problem's *null plan* is the partial plan $P = \langle\{s, e\}, \emptyset, \emptyset, \emptyset, \emptyset\rangle$ where s is the problem's start step and e the problem's end step.

5.1.1 CPOCL Flaws and How They Are Repaired

A refinement planner defines a set of flaws along with ways to repair them. The process of repairing a flaw generates a new node in the search space.

Definition 43 (open precondition flaw)**.** An *open precondition flaw* indicates that some precondition of a step has not yet been met by a causal link. It is a 2-tuple $\langle s_{need}, p\rangle$, where s_{need} is some step in S and p is a precondition of s such that no causal link in L has s_{need} as its head and p as its label.

Open precondition flaws are repaired by adding a new causal link to L which has s_{need} as the head (lines 3-10). The tail of the new link can be either a step already in S (line 5) or a new step created from an operator and added to S (lines 6-8). Adding new steps to S may requires adding new open precondition flaws (line 7).

When a new step is added to the plan, it is initially marked as non-executed (line 8). If a causal link is created from a non-executed step to an executed step, the tail step and all its causal ancestors must then be marked as executed (line 10)[1]. This ensures that non-executed steps are never used to satisfy the preconditions of executed steps.

When a happening is added to the plan, it is placed in its own intention frame whose actor is Fate (line 11).

[1]A complete plan will have only those steps marked as executed that *must* occur to achieve the goal. This follows the least commitment paradigm of POCL planners. It may be possible to mark additional steps as executed without making the plan unsound, and any system using CPOCL is free to do so.

When a step with an effect like $intends(c, g)$ is added to the plan, a new intention frame is created with that step as the motivating step (lines 12-14). CPOCL must later choose a satisfying step to explain how character goal g gets fulfilled. This need translates into a flaw:

Definition 44 (unsatisfied intention frame flaw)**.** An *unsatisfied intention frame flaw* indicates that a satisfying step has not yet been chosen for an intention frame. It is a 1-tuple $\langle f \rangle$, where f is some intention frame.

After a satisfying step is chosen (lines 21-24), the frame must be populated with all the steps that the character takes in pursuit of the goal.

When a new causal link is created, it may link a step outside an intention frame ($\notin T$) to a step inside an intention frame ($\in T$). This might indicate that the outside step was taken in pursuit of the frame's goal. If so, the outside step needs to be included in the frame. This need is represented as a flaw:

Definition 45 (intent flaw)**.** An *intent flaw* occurs when a causal links $s \xrightarrow{p} u$ exists such that, for some intention frame $r = \langle c, g, m, \sigma, T \rangle$, $s \notin T$, $u \in T$, and c is a character who must consent to s. It is a 2-tuple $\langle s, f \rangle$, where s is the step which may need to be included in frame f.

Intent flaws can be solved by adding the step to the frame (lines 26-27) or by ignoring the flaw (line 28). It is necessary to consider ignoring the flaw to ensure that valid plans are not missed in the search process [62].

Definition 46 (threatened causal link flaw)**.** A *threatened causal link flaw* indicates that the condition established by a casual link may be undone before it is needed. It is a 2-tuple $\left\langle s \xrightarrow{p} u, t \right\rangle$, where $s \xrightarrow{p} u$ is a causal link in L, and t is a step in S which threatens it.

Threatened causal links are fixed by preventing the ordering $s < t < u$ (lines 18-19) or by adding bindings to B which prevents the threatening effect of t from logically unifying with $\neg p$ (line 20). Note that threatened causal link flaws are *not* added for conflict links because they do not need to be repaired (line 29). In fact, since conflict is a desirable property of stories, they probably should not be repaired.

5.1.2 Example

Recall the example domain (Figure 3.1) and problem (Figure 3.2) which have been used throughout this document. This section will describe how the example CPOCL plan that is given in Figures 3.7 and 3.6 is constructed by the CPOCL algorithm. Below is a list of steps that describe which flaw is chosen at each iteration and how it is fixed.

1. Step 0 is added to the plan as the start step. It creates 6 unsatisfied intention frames for the motivations that appear in the initial state:

- `(intends Timmy (status Timmy Healthy))`
- `(intends Hank (status Hank Healthy))`
- `(intends Hank (status Timmy Healthy))`
- `(intends Carl (has Carl Antivenom))`
- `(intends Carl (status Carl Healthy))`
- `(intends William (status William Healthy))`

2. Step 1 is added to the plan as the end step. The end step has only a single precondition: `(not (status Timmy Sick))`, for which an open precondition flaw is added.

3. Unsatisfied intention frame for motivation `(intends Carl (has Carl Antivenom))`. Satisfy the frame by creating step 2, `(persist (has Carl Antivenom))`.

4. Unsatisfied intention frame for motivation `(intends Hank (status Hank Healthy))`. Satisfy the frame by creating step 3, `(persist (status Hank Healthy))`.

5. Unsatisfied intention frame for motivation `(intends Carl (status Carl Healthy))`. Satisfy the frame by creating step 4, `(persist (status Carl Healthy))`.

6. Unsatisfied intention frame for motivation `(intends William (status William Healthy))`. Satisfy the frame by creating step 5, `(persist (status William Healthy))`.

7. Open precondition `(not (status Timmy Sick))` for step 1: Create step 6, `(die ?victim-1)`. Add causal link 6 — `(not (status Timmy Sick))`→ 1. Add bindings ⟨`?victim-1,Timmy`⟩. Create a new intention frame for Fate. Add step 6 to the intention frame for Fate.

8. Open precondition `(has Carl Antivenom)` for step 2: Add causal link 0 — `(has Carl Antivenom)`→ 2.

9. Open precondition `(status Hank Healthy)` for step 3: Add causal link 0 — `(status Hank Healthy)`→ 3.

10. Open precondition `(status Carl Healthy)` for step 4: Add causal link 0 — `(status Carl Healthy)`→ 4.

11. Open precondition `(status William Healthy)` for step 5: Add causal link 0 — `(status William Healthy)`→ 5.

12. Open precondition `(status Timmy Sick)` for step 6: Add causal link 0 — `(status Timmy Sick)`→ 6.

13. Unsatisfied intention frame for motivation `(intends Timmy (status Timmy Healthy))`. Satisfy the frame by creating step 7, `(heal ?healer ?medicine ?patient)`. Add bindings ⟨`?patient,Timmy`⟩.

14. Unsatisfied intention frame for motivation `(intends Hank (status Timmy Healthy))`. Satisfy the frame with step 7.

15. Open precondition `(status Timmy Sick)` for step 7: Add causal link 0 — `(status Timmy Sick)`→ 7.

16. Open precondition `(has Hank Antivenom)` for step 7: Create step 8, `(steal ?thief ?item ?owner ?sheriff)`. Add causal link 8 — `(has Hank Antivenom)`→ 7. Add bindings ⟨`?thief,Hank`⟩ ⟨`?healer,Hank`⟩ ⟨`?item,?medicine`⟩. Create a new intention frame for `(intends ?sheriff (status Hank Dead))`.

17. Open precondition `(not (status Hank Dead))` for step 7: Add causal link 0 — `(not (status Hank Dead))`→ 7.

18. Intent flaw for step 8 and the intention frame `(intends Hank (status Timmy Healthy))`: Add step 8 to the frame.

19. Open precondition `(has Carl Antivenom)` for step 8: Add causal link 0 — `(has Carl Antivenom)`→ 8. Add bindings ⟨`?item,Antivenom`⟩ ⟨`?owner,Carl`⟩.

20. Open precondition `(owns Carl Antivenom)` for step 8: Add causal link 0 — `(owns Carl Antivenom)`→ 8.

21. Open precondition `(status Hank Healthy)` for step 8: Add causal link 0 — `(status Hank Healthy)`→ 8.

22. Unsatisfied intention frame for motivation `(intends ?sheriff (status Hank Dead))`. Satisfy the frame by creating step 9, `(shoot ?killer ?victim-2)`. Add bindings ⟨`?victim-2,Hank`⟩ ⟨`?killer,?sheriff`⟩.

23. Open precondition `(armed William)` for step 9: Add causal link 0 — `(armed William)`→ 9. Add bindings ⟨`?killer,William`⟩.

24. Open precondition `(status William Healthy)` for step 9: Add causal link 0 — `(status William Healthy)`→ 9.

5.1.3 Differences Between IPOCL and CPOCL

Intentional planning in CPOCL is handled slightly differently than in the IPOCL algorithm described by Riedl and Young [62]. IPOCL constructs intention frames by starting with the satisfying step and later finding a motivating step. CPOCL, on the other hand, begins with the motivating step and later finds a satisfying step. This is why IPOCL has an *unmotivated intention frame flaw*, whereas CPOCL has a corresponding *unsatisfied intention frame flaw*. This change implies a different interpretation of the modal `intends` predicate. For IPOCL, a motivation *may* be used to motivate an intention frame, but it does not have to be used. This means that in a complete IPOCL plan some motivations will not have a corresponding intention frame. For CPOCL, a motivation *must* be used. In other words, when a motivation appears in the effects of a step, that agent *must* form a plan to fulfill the goal.

This different interpretation of *intends* affects the efficiency of CPOCL. When refining a partial plan by adding a new step, IPOCL must generate a new partial plan for every possible combination of characters and effects. Consider the `heal` operator in the example domain. It is possible that neither the healer nor the patient will take this step to satisfy a goal. It is also possible that just the healer will take this step to satisfy just the goal `(status ?patient Healthy)`. It is also possible that just the healer will take this step to satisfy the goals `(status ?patient Healthy)` and `(not (has ?healer ?medicine))`. The same is true for just the patient and for both he healer and the patient together. Formally, if the new step has n consenting characters and e effects, IPOCL will need to generate $2^n \cdot 2^e$ refined plans. Adding a new step with only 2 consenting characters and 3 effects to an IPOCL plan will require 32 refined plans where POCL would require only 1. This branching factor explodes quickly.

Like the original POCL algorithm, CPOCL generates only one refined plan per new step added because each motivation must create an intention frame. However, this significant decrease in CPOCL's branching factor and the resulting increase in efficiency may come at the price of expressiveness depending on the semantics of the problem. Unlike IPOCL, CPOCL will never consider adding a step to a plan just for that step's motivations. In other words, some effect of the step besides the motivation must be needed in the plan. This is equivalent to the restriction that every step in the plan must be the tail of at least one causal link—a requirement imposed by POCL but not by IPOCL. According to Riedl and Young [62], this is not necessarily a problem because the `intends` predicate is meant to signify meta-level information about a step, similar to side effects.

I point out this subtle distinction between IPOCL and CPOCL because it implies some minor differences between their search spaces. However, this does not affect the importance of the contribution of my model of conflict. The use of non-executed steps to express conflict could easily be adapted to work with IPOCL's satisfying-step-first method of building intention

frames.

5.2 The Glaive Planner

POCL plans enjoy a rich knowledge representation, but modern state-space search methods are generally much faster. Whereas a plan-space planner searches the space of partial plans seeking a complete plan, a state-space planner searches the space of world states seeking a goal state [9]. In practice, a state-space planner is also searching a space of partial plans because it must remember the steps it has taken in order to construct the solution plan once a goal state is found. The transition from one state to another during this search is accomplished by adding a single step—a fully ground instance of an operator—to the current plan. Thus, a state-space planner searches the space of totally ordered, fully-ground plans, and it constructs plans in order from start to finish.

Definition 47 (search space, state-space planning)**.** The *search space of a state-space planner* is a directed graph. A node in the graph represents a totally-ordered, fully-ground plan. A directed edge $p_1 \xrightarrow{s} p_2$ runs from a plan p_1 to a plan p_2 and represents the addition of step s to p_1 to create p_2. Such an edge may only exist if the preconditions of s are satisfied in the world state resulting from p_1.

Riedl and Young's description of intentionality [62] imposes additional constraints on the definition of a valid plan. For IPOCL and CPOCL, this increases the amount of search needed to find a solution because they must reason about new kinds of flaws in addition to all the flaws of a traditional POCL planner. However, I will demonstrate that these constraints can actually be used to speed up narrative planning for a state-space planner (in most cases).

This section describes the Glaive planner (Algorithm 4), a forward-chaining state-space heuristic search planner based on Fast-Forward that reasons about intentionality and conflict. While totally-ordered, fully-ground plans lack some of the representational richness of POCL plans, the significant speedup achieved by Glaive makes it an attractive option for narrative planning, especially for interactive systems that need to reason in real time.

5.2.1 Intentional State-Space Planning

Classical state-space planners track only the steps taken so far and the current state of the world. Glaive must also keep track of causal links[2] in order to reason about intentionality. While CPOCL tracks causal links explicitly, Glaive can track them implicitly because a plan is totally ordered and fully ground.

[2]Indeed, Glaive can be considered a Total Order Causal Link (or TOCL) planner.

Algorithm 4 The Glaive planning algorithm.

Require: A planning domain and problem.
Ensure: A totally ordered, fully-ground plan Π that is a solution to the problem, or failure.

Let $\Pi = \emptyset$ be an empty plan, and σ the initial state.
Let G be all goals which are motivated in the initial state.
Let $U = \emptyset$ be an empty set of unexplained steps.
return GLAIVE(Π, σ, G, U)

procedure GLAIVE(plan Π, state σ, goals G, unexplained steps U)
    Nondetirministically choose a potentially motivated step s with satisfied preconditions.
    Add step s to Π.
    Apply the effects of step s to state σ.
    for each character c and literal g such that $intends(c, g) \wedge \neg g$ **do** ▷ Add new goals.
        Add a new goal $\langle c, g \rangle$ to G.
    end for
    if s is not a happening **then** ▷ Keep track of new unexplained steps.
        Add s to U.
    end if
    for each goal $\langle c, g \rangle \in G$ **do** ▷ Remove explained steps from U.
        if g is an effect of s **then**
            Remove g from G.
            for each intentional path p which ends in g **do**
                for each step $t \in p$ **do**
                    if t is explained **then**
                        Remove t from U for every node in the search space.
                    end if
                end for
            end for
        end if
    end for
    if , for any node in the search space, σ is a goal state and $U = \emptyset$ **then**
        return Π for that node
    else
        return GLAIVE(Π, σ, G, U)
    end if
end procedure

* When removing an unexplained step t from U for every node in the search space, t refers to that particular instance of step t, not every instance of step t. See the discussion in Section 5.2.3 for a more detailed explanation.

Definition 48 (causal link, state-space planning)**.** In a totally ordered and fully ground plan, there exists a *causal link* $s \xrightarrow{p} u$ if and only if step s occurs before step u, step s has effect p, step u has precondition p, and no step that occurs after s and before u has the effect $\neg p$ (in other words, no step threatens the causal link).

Refinement planners like POCL, IPOCL, and CPOCL create exactly one causal link to satisfy a given step's precondition. This is because causal links are added as a way of repairing flaws, and there will only ever be one open precondition flaw per precondition. Note that Glaive may create multiple causal links that satisfy a single precondition if there are multiple earlier steps with the appropriate effect.

Since threatened causal links are the crux of conflict in CPOCL, it may seem odd that Glaive does not allow them. Section 5.2.3 gives a detailed explanation for this, but for now it suffices to say that an individual Glaive plan cannot represent conflict. Rather, conflict is discovered when comparing two Glaive plans to one another.

In addition to the current plan and current state, Glaive also explicitly tracks character goals.

Definition 49 (goal)**.** A *goal* is a two-tuple $\langle c, g \rangle$, where c is a character and g a literal that character c wishes to make true.

New goals are created when a character intends some goal which is not currently satisfied (lines 9-11). For example, at the beginning of the example problem Carl has the antivenom and he intends to have the antivenom. When Hank steals the antivenom, it causes Carl to adopt a goal to have the antivenom once again.

Glaive is an intentional planner like CPOCL, so it must ensure that every step in a solution plan is motivated and goal-oriented. It does this by reasoning about intentional paths.

Definition 50 (intentional path)**.** A *intentional path* for some character c and some goal g is an alternating sequence of n steps and n propositions $\langle s_1, p_1, s_2, p_2, ...s_n, g \rangle$ such that:

1. Character c must consent to all steps.
2. `(intends` c g`)` is true immediately before s_1 and true until step s_n.
3. Step s_n has effect g.
4. For i from 1 to $n-1$, there exists a causal link $s_i \xrightarrow{p_i} s_{i+1}$.
5. No proposition appears twice.
6. The path never contains a proposition and its negation.

An intentional path describes a sequence of steps taken by a character in service of a goal. Consider this example intentional path for Hank to achieve the goal `(status Timmy Healthy)`:

⟨`(steal Hank Antivenom Carl William)`,`(has Hank Antivenom)`,

`(heal Hank Antivenom Timmy)`,`(status Timmy Healthy)`⟩

This path can be read to mean: Hank stole the antivenom so that he could be in possession of the antivenom. He wanted to be in possession of the antivenom so that he could heal Timmy, which would achieve his goal that Timmy be healthy. There can be 0, 1, or many intentional paths for a goal. These paths may overlap to form a tree. Intentional paths are important because they allow Glaive to detect which steps have explanations in terms of character goals.

Definition 51 (explained step)**.** A step s is *explained* if and only if, for each consenting character c, there exists some intentional path p for c such that s is on p and every other step on p is also explained.

The recursive second part of this definition is important. It is not enough that a step be on some intentional path for every consenting character. Consider the example intentional path above. That path does not, by itself, provide an explanation for why Hank took those steps. There must *also* exist a second intentional path that explains why Timmy consented to `heal`. If that second path cannot be found, the first path could never happen because Timmy would not allow it to happen.

When a new step that requires the consent of one or more characters gets added to the plan, that step is put into a set of unexplained steps (lines 12-13). Glaive then checks to see if this new step satisfies any of the current character goals (line 16). If so, that goal is removed. Glaive then checks to see if the new step has created any new intentional paths. If there are new intentional paths, the steps on those paths may have become explained. If so, those steps are removed from the set of unexplained steps (lines 19-24).

Dealing with unexplained steps is one of the primary challenges when designing an intentional planner. When a step is added to a plan, it is not always possible to know what goal that step is taken in service of. The sooner an intentional planner is able to realize that an unexplained step will never have an explanation, the sooner it can prune that plan from the search space and save computational effort. Reasoning intelligently about unexplained steps is one of the ways that Glaive achieves its speed.

5.2.2 The Glaive Heuristic

The expansion of the search space during planning is guided by a heuristic. State-space planning algorithms are relatively simple compared to other families of planning algorithms; it is the

heuristics used to guide them that exhibit most of their sophistication and complexity. In fact, the names of most state-space planners are synonymous with the heuristics they define.

Definition 52 (heuristic)**.** A *state-space planning heuristic* is a function $h(\sigma)$ which, for any world state σ, returns an integer which is an estimation of the number of steps the planner will need to take before reaching a solution.

Calculating exactly how many steps are needed is as difficult as planning itself, and thus P-SPACE hard [14]. Most heuristics work by reducing the problem down to an easier (but incomplete) problem that can be solved in polynomial time, and then using the solution to that relaxed problem as an approximation of the solution to the true problem. Note that when I say "the problem," I mean the problem of starting at the current state and reaching the goal, *not* the original problem of starting at the initial state and reaching the goal.

Definition 53 (current state)**.** For a given node in a state-space search graph, the *current state* is the state after taking all the steps in the current plan.

It is important to distinguish the current state from the initial state of the planning problem. The initial state (along with the empty plan) forms the root of the search space. The heuristic estimates the distance from the current state to the goal. Because the heuristic has to be calculated for each new plan that is produced—that is, once per iteration of the planning algorithm—it is important that it be calculated efficiently. The Glaive heuristic is based on Hoffmann and Nebel's Fast Forward (or FF) [35]. I chose FF as a foundation because it strikes a balance between accuracy, complexity, and efficiency. It is more accurate than Bonet's Heuristic Search Planner [9] and uses several of the same data structures that Glaive needs to track intentionality.

Goal Graphs and Plan Graphs

In order to calculate its heuristic value, Glaive needs two kinds of data structures: goal graphs and plan graphs. Goal graphs are a new structure defined by Glaive, but plan graphs have been a popular tool for planning systems since the Graphplan algorithm of Blum and Furst [8].

Goal graphs are constructed for each character goal that might be adopted during the planning process. They contain important causal and intentional information about those goals.

Definition 54 (goal graph)**.** A *goal graph* is a directed, layered, graph composed of step nodes. It is constructed for some character c and some literal g which c wants to make true.

Definition 55 (step node, goal graph)**.** A *step node in a goal graph* for character c and goal g exists for step s at layer 0 if and only if it c must consent to s and s has an effect g. A step

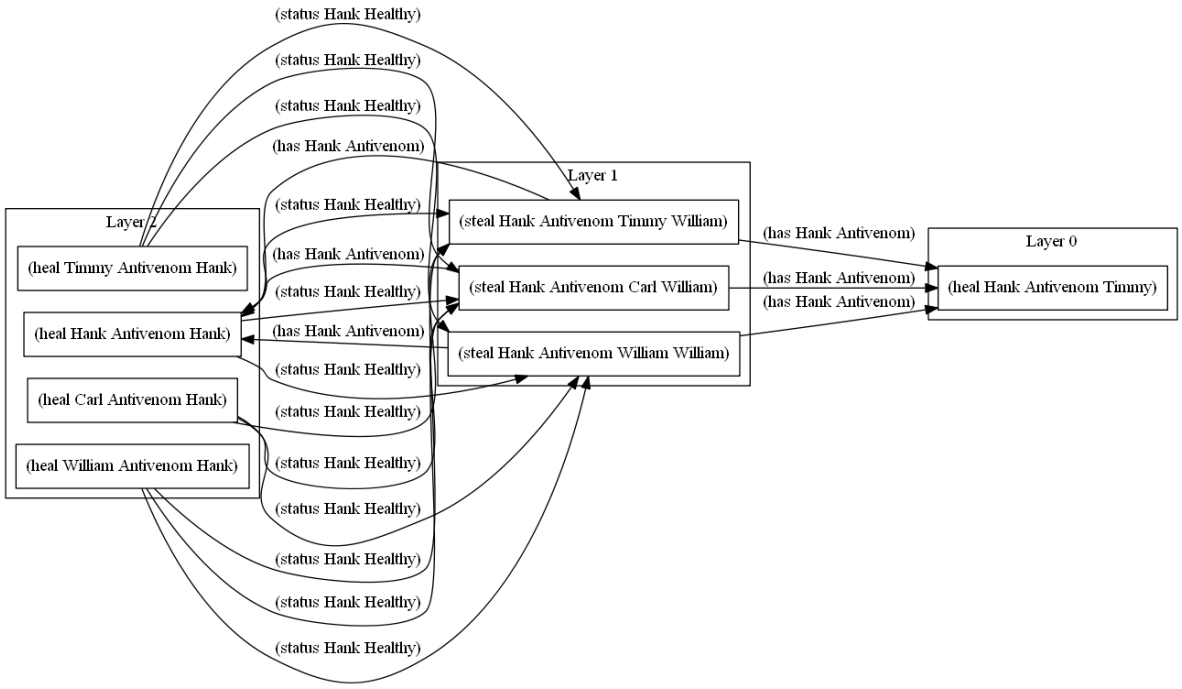


Figure 5.1: One goal graph for the example problem in Figure 3.2. This is the goal graph Glaive constructs for character Hank and goal literal `(status Timmy Healthy)`. It contains all the steps which Hank might take to achieve that goal.

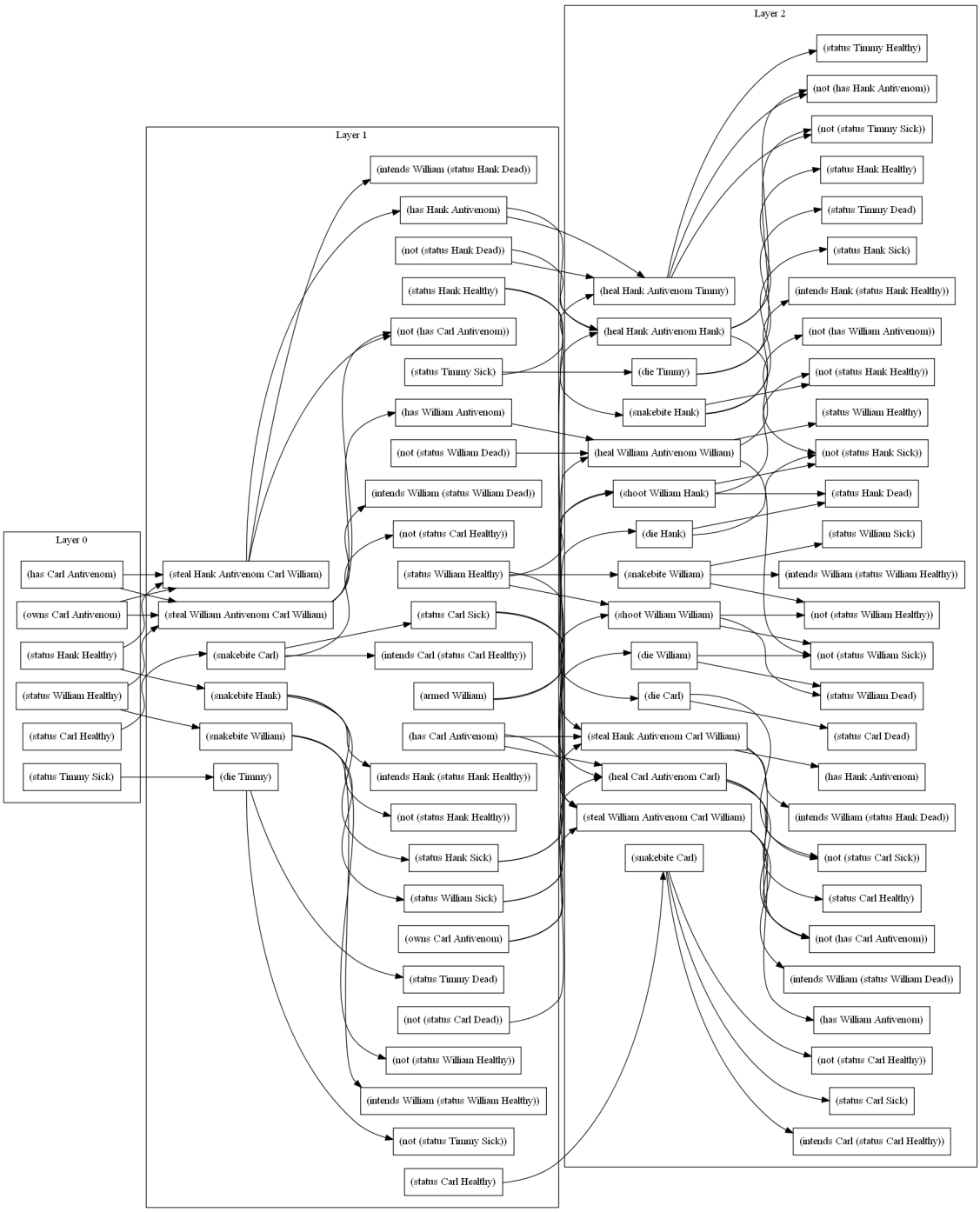


Figure 5.2: A plan graph for the example problem in Figure 3.2. This plan graph would be constructed when Glaive evaluates the root node of the state-space, so here the current state is the initial state of the planning problem. For illustration purposes, this plan graph has been extended to Layer 2. However, during the actual heuristic calculation, it would only be extended to Layer 1 because that is where the author's goal first appears.

node exists at layer $i > 0$ if and only if c must consent to s, s has an effect p, there exists a step node at layer $i - 1$ with precondition p, and no step node for that step exists at any layer $x < i$.

Definition 56 (edge, goal graph)**.** A directed edge $s \xrightarrow{p} t$ in a goal graph leads from a step node for step s to a step node for step t with label p if and only if step s has effect p and step t has precondition p.

To summarize, a goal graph for some character c and goal g can be constructed like so: It contains only steps for which c is a consenting character. Layer 0 contains every step which has g as an effect. Layer 1 contains steps which have effects that satisfy the preconditions of the steps in layer 0, except for those steps that already exist at layer 0. Layer 2 contains the steps which have effects that satisfy the preconditions of the steps in layer 1, except for those steps that already exist at earlier layers. And so on. When constructing a goal graph, new layers are added until the next layer would be empty.

A goal graph is a composite of all the possible intentional paths which might exist for a goal. Directed edges represent possible causal links between steps taken in pursuit of the goal. Layer 0 contains the possible satisfying steps for the goal, layer 1 the possible intentional parents of the satisfying steps, layer 2 the possible intentional grandparents of the satisfying steps, etc. An example goal graph is given in Figure 5.1.

We say that a goal graph is built backwards starting from the goal because the satisfying step is the last step taken. The higher the layer in which a step appears, the earlier that step must be taken in the plan. The other data structure that Glaive constructs is called a plan graph, which is built forward from the current state. A plan graph embodies a relaxed version of the planning problem that Fast Forward solves when calculating its heuristic. In this relaxed planning problem, literals are only added to the current state, never deleted. In other words, when attempting to solve the relaxed problem one can only get closer to the goals, never farther.

Definition 57 (plan graph)**.** A *plan graph* is a directed, layered, acyclic graph composed of literal nodes and step nodes.

Definition 58 (literal node, plan graph)**.** A *literal node in a plan graph* exists for a ground predicate literal p at layer 0 if and only if p is true in the current state. A literal node for literal p exists at layer $i > 0$ if and only if it exists at layer $i - 1$ or if p is the effect of a step whose node appears in layer i.

The set of literal nodes at any given level is monotonically increasing. This means that after layer 0, both a literal and its negation may appear at the same level. This is the aforementioned relaxation of the planning problem which allows it to be solved in polynomial time.

In addition to literal nodes, a plan graph also contains step nodes. Recall that Glaive keeps track of all the goals G that have been adopted by characters in the current plan.

Definition 59 (potentially motivated step)**.** Given some node in Glaive's search space, a step s is *potentially motivated* for character c if and only if it has the potential to be on an intentional path for c. A step s is *potentially motivated* (in general) if and only if it is potentially motivated for each of its consenting characters.

Goal graphs make it easy to check which steps are potentially motivated. Any step which appears in a goal graph for the goal g is potentially motivated. In other words, it might achieve g or it might contribute to achieving g. Glaive only considers potentially motivated steps when choosing what step to take next (line 6). This means its search spaces have a branching factor which is either the same as or lower than a classical state-space planner.

Definition 60 (step node, plan graph)**.** A *step node in a plan graph* exists for step s at layer i if and only if all the literal nodes for the preconditions of step s exist at layer $i-1$ and step s is potential motivated. Layer 0 has no step nodes.

The requirement that a step be potentially motivated before it appears in the plan graph is the only way in which Glaive's plan graphs are different from the plan graphs used by FF. FF only requires that a step's preconditions appear in the previous layer. Glaive adds the further constraint that the step must have some potential explanation in the future of the plan. A step which is not potentially motivated will always be unexplained, thus Glaive can save time by not considering these steps as the next step in the plan.

Definition 61 (edge, plan graph)**.** A directed edge $p \rightarrow s$ in a plan graph leads from a literal node for literal p to a step node for step s if the step node exists at layer i, the literal node exists at layer $i-1$ and p is a precondition of s. A directed edge $s \rightarrow p$ in a plan graph leads from a step node for step s to a literal node for literal p if both nodes exist at the same layer and p is an effect of s.

To summarize, a plan graph can be constructed like so: Layer 0 contains a literal for every literal in the current state and no steps. Layer 1 contains the steps whose preconditions appear in layer 0, all the literals from layer 0, plus all the literals which are effects of the steps in layer 1. Layer 2 contains the steps whose preconditions appear in layer 1, all the literals from layer 1, plus all the literals which are effects of the steps in layer 2. And so on. When constructing a plan graph, new layers are added until all the goals are present in a layer. Note that literal and step nodes increase monotonically in successive layers. An example plan graph is given in Figure 5.2. This is the graph that would be constructed when Glaive estimates how far the initial state is from the goal for the example problem given in Figure 3.2.

Note that plan graphs are built forward from the current state, which is constantly changing. This means that Glaive must construct a new plan graph each time the heuristic is evaluated. However, a goal graph is built backward from the goal and does not depend on the current state. This means that a goal graph never changes, so once it has been constructed it can be reused every time Glaive needs to reason about that goal.

Heuristic Calculation

The Glaive heuristic uses plan graphs and goal graphs to estimate how many steps need to be taken to achieve the goal from the current state. The heuristic is simply the maximum of two numbers: one derived from the plan graph and one from the goal graph.

The first value is an estimate of how many more steps must be taken to achieve the author's goal. Recall that a plan graph represents a relaxed version of the planning problem—one where steps can only add literals to the current state, never remove them. Algorithm 5 uses the plan graph to find a solution to this relaxed problem. The length of that plan (called a relaxed solution) is an estimate of how many steps must be taken in the actual problem. Algorithm 5 is the same process that Fast-Forward uses to find a relaxed solution, but with one minor addition: when a step is added to the relaxed solution, we must also add a motivation for every character who consents to that action (line 11).

The second value that Glaive uses to calculate its heuristic is derived from the goal graph. Recall that Glaive tracks a set of unexplained steps for each node in the search space. An unexplained step indicates that more steps need to be taken before an explanation can be found, and the number of those steps can be efficiently estimated as the index of the layer at which the step appears in some goal graph for that character.

Definition 62 (layer)**.** For some step s which is unexplained for character c and some goal graph γ which contains a node for step s, let the function layer(s, γ) be the integer that corresponds to the layer in γ at which the node for step s appears. For example, if we let γ be the goal graph in Figure 5.1, let the unexplained step s =`(heal Hank Antivenom Timmy)`, then layer$(s, \gamma) = 0$ because that step appears at layer 0. If we let the step s =`(steal Hank Antivenom Carl William)`, then layer$(s, \gamma) = 1$ because that step appears at layer 1.

If the character is pursuing multiple goals and the unexplained step could be directed toward more than one of those goals, Glaive chooses γ such that layer(s, γ) is minimized. Thus, one simple estimate for how many more steps need to be taken to explain an unexplained step is:

$$\text{cost}(s) = \min_{\gamma} \text{layer}(s, \gamma)$$

However, some unexplained steps may be in service of the same goal. Specifically, when one

unexplained step is the intentional parent of another unexplained step the parent can usually be ignored; this is because an explanation for the child will usually explain the parent as well. This idea can be captured formally:

Definition 63 (dominated step)**.** An unexplained step s_1 *dominates* an unexplained step s_2 if there exists an edge $s_2 \xrightarrow{p} s_1$ in any goal graph for one of the character's current goals. Any such step is said to be *dominated*.

Glaive ignores the cost of dominated unexplained steps when calculating its heuristic:

$$\text{cost}(s) = \begin{cases} 0 & \text{if } s \text{ is dominated} \\ \min_\gamma \text{layer}(s, \gamma) & \text{otherwise} \end{cases}$$

The cost of explaining all the unexplained steps U in a plan is simply the sum of the costs:

$$\text{cost}(U) = \sum_{u \in U} \text{cost}(u)$$

Like the Fast Forward heuristic, this cost function is meant to err on the side of being too low. It may sometimes overestimate and so is not admissible, however neither are the most notable state-space planning heuristics such as HSP, FF, and FD. As with these heuristics, the extra work required to ensure that cost never overestimates is too much computational effort for too small a gain in accuracy. Indeed, the problem of finding the exact number of steps needed to explain a step is itself a planning problem and thus P-SPACE hard.

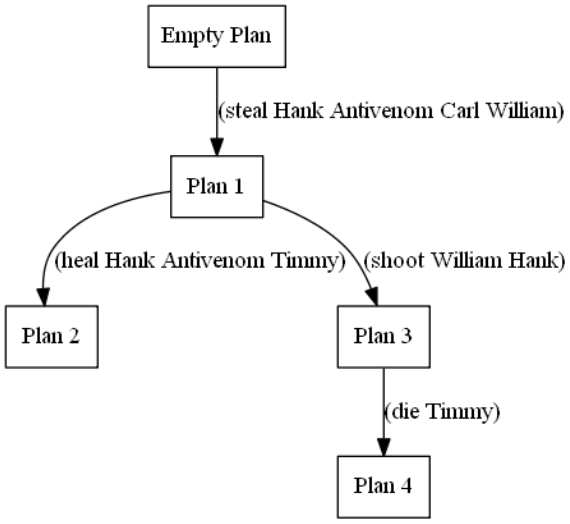
In summary, if we let $\text{FF}(\sigma)$ be Fast Forward's heuristic function, then Glaive's heuristic function is:

$$\text{h}(\sigma, U) = \max(\text{FF}(\sigma), \text{cost}(U))$$

Glaive uses the maximum of these two values (rather than, say, the sum) because it is likely that both functions are counting some of the same steps. One potential improvement to Glaive would be the ability to detect those steps and to merge the $\text{FF}(\sigma)$ and $\text{cost}(U)$ functions into a single, more accurate heuristic.

5.2.3 Possible Worlds and Conflict

Non-executed steps represent actions that a character intended to take but never actually took. They are the key to representing failed plans and conflict, but they present a significant challenge to a state-space planner. What does it mean to apply a non-executed step to the current state? Because the step never actually happens, one explanation might be that a non-executed step does not change the current state at all. However, the characters who intend that step expect



Plan 2	
`(steal Hank Antivenom Carl William)`	Hank stole antivenom from the shop, which angered sheriff William.
`(heal Hank Antivenom Timmy)`	Hank healed his son Timmy using the stolen antivenom.

Plan 4	
`(steal Hank Antivenom Carl William)`	Hank stole antivenom from the shop, which angered sheriff William.
`(shoot William Hank)`	Sheriff William shot Hank for his crime.
`(die Timmy)`	Timmy died.

Plan 2 + Plan 4	
`(steal Hank Antivenom Carl William)`	Hank stole antivenom from the shop, which angered sheriff William.
`(shoot William Hank)`	Sheriff William shot Hank for his crime.
`<heal Hank Antivenom Timmy>`	Hank intended to heal his son Timmy using the stolen antivenom.
`(die Timmy)`	Timmy died.

Figure 5.3: The search space and two possible worlds for the Western problem in Figure 3.2, along with the combination of those two possible worlds into a single story which includes a non-executed step (in angle brackets).

to see its effects and should plan as if it had been executed. To complicate things further, it may be possible that a character has a reason to expect p because of a previous non-executed step, while also having a reason to expect $\neg p$ because of a previous executed step. Allowing non-executed steps in a state-space plan would require a much richer representation of state, one which can track the beliefs of individual characters and handle these kind of inconsistencies in belief.

Glaive sidesteps this problem. Rather than representing non-executed steps, Glaive treats the entire search-space graph as set of possible worlds. Certain important narrative phenomena such as conflict can be discovered by comparing one possible world to another.

Consider Figure 5.3, which gives two solutions to the example Western problem in Figure 3.2. Both plans exist as nodes in Glaive's search space. Because both plans have the same first

step, these solution nodes share Plan 1 as a common ancestor in the search space. That first step, `(steal Hank Antivenom Carl William)`, begins as an unexplained step because there is not yet any obvious reason for why Hank would steal the antivenom. Plan 2 provides an explanation by creating two intentional paths: Hank stole the antivenom so that he could heal Timmy, and Timmy allowed himself to be healed. Thus, Plan 2 has no unexplained steps. Plan 4 does not contain any intentional paths that explain why Hank stole the antivenom. However, the existence of Plan 2 means that there exists some possible world in which Hank stealing the antivenom makes sense. Thus, we do not need to consider the first step of Plan 4 unexplained.

More generally, we say that once a step has an explanation in any node of the search space, that step also has an explanation in all nodes of the search space which are descendents of the node in which that step first appears. In Figure 5.3, `(steal Hank Antivenom Carl William)` was given an explanation in Plan 2. This means that it also has an explanation in Plan 1 (the node in which that step first appears) and in all nodes which are descendents of Plan 1. Note that this *does not* mean than every occurrence of `(steal Hank Antivenom Carl William)` is explained in every plan; only that particular occurrence of that step is explained, and that particular occurrence of that step may appear in many plans.

This definition of how a step becomes explained has important implications for Algorithm 4 on line 21. When removing an unexplained step t from the set U, the step is not only removed for that one node of the search space, but for every node in the search space which is a descendent of the first node that marked t as unexplained. This implies that once an agent's plan to achieve a goal has been completed in one possible world, any subset of that plan can appear in other possible worlds.

This method of removing unexplained steps also means that a node might become a solution when it is not the current node being considered. In Figure 5.3, Plan 4 is a solution. However, it is only a solution once Plan 2 has been discovered. If Plan 4 is discovered before Plan 2, Glaive still returns Plan 4 as a solution once Plan 2 is discovered (line 27).

Since non-executed steps are not represented directly in Glaive plans, a solution returned by Glaive will contain only executed steps. The set of solutions that Glaive produces is the same as the set of solutions that CPOCL produces when we consider only the executed steps. However, having non-executed steps in a plan is helpful for reasoning about conflict. In order to fill in the missing non-executed steps in a Glaive plan, we can combine multiple nodes in the Glaive search space into a single solution. For example, consider the combination of Plan 2 and Plan 4 in Figure 5.3. Plan 2 tells us that Hank can form a two steps plan to save his son: steal the antivenom, then heal Timmy. Only the first of these two steps occurs in Plan 4, but for the sake of representing conflict we can add the second step to Plan 4 as a non-executed step. So important narrative phenomena such as conflict can be still be reasoned about in Glaive by comparing and combining multiple possible worlds into a single representation.

5.2.4 Example

This section provides a more detailed explanation of how the two plans in Figure 5.3 are discovered by Glaive. It includes both the algorithmic process and the heuristic calculation.

As a pre-processing step, Glaive builds the goal graphs for all goals in the problem. The most important goal graph for this example can be seen in Figure 5.1, which explains the steps that Hank might take to achieve `(status Timmy Healthy)`.

Planning begins at the root of the search space: the initial state and an empty plan. The initial set of goals are all those which are motivated in the initial state but not satisfied:

- `(intends Timmy (status Timmy Healthy))`
- `(intends Hank (status Timmy Healthy))`

Glaive now evaluates the root node using its heuristic. The plan graph created for this evaluation can be seen in Figure 5.1. The relaxed solution extracted from this graph for the author's goal is a 1 step plan: `(die Timmy)`. Because the plan is empty it contains no unexplained steps, so the heuristic value is $\max(1, 0) = 1$. Note that there does exist a solution that is 1 step away, so this calculation is accurate. However, in order to provide a more helpful example of Glaive's process, this example describes the discovery of two longer solutions.

Next Glaive chooses a first step which is potentially motivated and whose preconditions are satisfied in the current state. It chooses `(steal Hank Antivenom Carl William)` and the node named Plan 1 is created in the search space. This step has the effect `(intends William (status Hank Dead))`, so a new goal ⟨`William`,`(status Hank Dead)`⟩ is added to the set of current character goals. The new step is added to the set of unexplained steps. Now Glaive evaluates the current node using its heuristic. The relaxed solution is a 1 step plan: `(heal Hank Antivenom Timmy)`. Step 1 is unexplained, it appears at Layer 1 in the goal graph in Figure 5.1, and it is not dominated, so the heuristic value is $\max(1, 1) = 1$.

Next Glaive chooses a second step which is potentially motivated and whose preconditions are satisfied in the current state. We will consider two possible choices for the second step. The first is `(shoot William Hank)`, which creates the node named Plan 3 in the search space. The new step is added to the set of unexplained steps. This step achieves the character goal ⟨`William`,`(status Hank Dead)`⟩, so that goals is removed. Also, it creates a new intentional path ⟨`(shoot William Hank)`,`(status Hank Dead)`⟩ for William, so `(shoot William Hank)` is now explained. Now Glaive evaluates the current node using its heuristic. The relaxed solution is a 1 step plan: `(die Timmy)`. The step `(steal Hank Antivenom Carl William)` has a cost of 1, so the heuristic value is $\max(1, 1) = 1$.

Glaive chooses a third step, `(die Timmy)`, which creates the node named Plan 4 in the search space. This step is a happening, so it does not need to be added to the unexplained set.

This plan achieves the author's goal, but it has an unexplained step. We will return to this node in a moment, after visiting another branch of the search space.

Now we will consider another choice for the second step, `(heal Hank Antivenom Timmy)`, which creates the node named Plan 2 in the search space. This step is added to the set of unexplained steps. This step achieves two character goals, ⟨`Hank`,`(status Timmy Healthy)`⟩ and ⟨`Timmy`,`(status Timmy Healthy)`⟩, so those goals are removed. This step creates two new intentional paths: ⟨`(steal Hank Antivenom Carl William)`,`(has Hank Antivenom)`,`(heal Hank Antivenom Timmy)`,`(status Timmy Healthy)`⟩ for Hank and ⟨`(heal Hank Antivenom Timmy)`,`(status Timmy Healthy)`⟩ for Timmy. This means that both of the currently unexplained steps are now explained. Note that the first step `(steal Hank Antivenom Carl William)` is now explained *for all of Plan 1's descendant nodes*, which includes Plan 4. This is because there now exists a story in which Hank stealing the antivenom makes sense. This node is a solution, so its heuristic value is 0. Also, Plan 4 has just become a solution thanks to this node. Both can be returned as valid plans or combined into a single plan with non-executed steps.

Conclusion

This chapter has presented two different approaches to the problem of intentional planning with non-executed steps and conflict. The plan-space CPOCL algorithm works directly with the rich CPOCL knowledge representation. The state-space Glaive algorithm improves on recent advances in heuristic search planning by incorporating the constraints of intentional planning to calculate a more accurate heuristic. The speed of Glaive has significantly increased the number of narrative planning problems which can be solved in a tractable amount of time. The next chapter presents two evaluations of Glaive: a set of computational benchmarks and its incorporation into an interactive narrative video game which reasons about conflict.

Algorithm 5 Extract a relaxed solution from a plan graph

Require: A plan graph with layers $L_0 ... L_m$ and a set of author goals G.
Ensure: A set of steps Π.

Let $\Pi = \emptyset$.
$\forall g \in G$ assign g as a goal to layer L_m.
for each layer L_i, starting at L_m going back to L_1. **do**
    for each goal g assigned to L_i **do**
        if a node for g exists at L_{i-1} **then**
            Assign g as a goal to L_{i-1}.
        else
            Choose a step s with effect g.
            Add s to Π.
            Assign preconditions of s as goals to L_{i-1}.
            Assign a motivation for s as a goal to L_{i-1}.
        end if
    end for
end for

Chapter 6

Evaluation of the Glaive Planner

The Glaive planner achieves its speedup in two ways. Firstly, it utilizes advances in fast forward-chaining state-space heuristic planning research. Secondly, it leverages the constraints of intentional planning problems to reduce its branching factor and calculate a more accurate heuristic. This chapter presents an evaluation of Glaive's computational efficiency. It then discusses how Glaive has been used to control the plot of a video game called *The Best Laid Plans* and presents an evaluation of how human players recognize intentionality and conflict in the plots generated by Glaive.

Planning is P-SPACE hard, and planning problems are often so large in scope that a formal analysis is too difficult or not helpful. The efficiency of planning algorithms has traditionally been evaluated by using a suite of benchmark test problems, so I have compiled 8 intentional and conflict planning problems from the narrative planning community and used those problems to test Glaive. To provide a basis for comparison, I have also provided the results for a version of Glaive that uses only the original Fast-Forward heuristic rather than the more intelligent Glaive heuristic. In all cases, Glaive performs the same or significantly better in terms of total time, number of nodes visited, and number of nodes expanded.

The value of a generative model combined with an efficient algorithm is the ability to use that model to reason about narratives in a real time interactive context. As a final evaluation of my work, I present a point-and-click narrative adventure game called *The Best Laid Plans* which uses Glaive to control the actions of all non-player characters. The story of this game is generated entirely at run time by the player and the narrative planner. It is constructed from atomic pieces (rather than from pre-scripted narrative fragments) according to models of intentional action and conflict. This game was evaluated relative to two other versions: a control in which the non-player characters do nothing and a scripted version in which the actions of the non-player characters is defined at design time by a human author. Based on a post-survey of 64 human subjects, players recognized intentionality and conflict in the stories produced by

Glaive more so than in the control, and there were no significant differences between Glaive and the human author.

The experiments in Chapter 4 validated that the CPOCL model reflects aspects of the human understanding of conflict as thwarted plans in static narratives. The implementation of an interactive system which uses this model to generate stories with conflict synergises with the previous evaluations to demonstrate the value of computational models of narrative. These evaluations together tell the story of how an essential feature of storytelling can be operationalized, tested, and used to automatically produce stories with properties that more closely meet the expectations of a human audience.

6.1 Computational Efficiency

The Glaive planner has been implemented in Java 7. It takes as input conflict planning domains and problems in the Planning Domain Definition Language (or PDDL) syntax, a standard in the planning research community. It has been considerably optimized and supports a number of helpful syntactic extensions, including disjunctive goals, first order quantifiers, conditional effects, and domain axioms. This section presents an evaluation of Glaive relative to its predecessor, Fast-Forward.

6.1.1 Benchmark Problems

Because planning is P-SPACE hard [14] and the search space of a planning problem so vast, the efficiency of a planning algorithm is usually evaluated by considering its performance on benchmark problems. Classical planners can draw from the wealth of benchmarks established by the bi-annual International Planning Competition, but intentional planning has received less attention and thus relatively few intentional planning problems have been studied. I have compiled a suite of 7 domains and 8 problems for testing Glaive. Details for each are given below, and the complete domains and problems can be found in Appendix A.

The size of the state space for each problem is also given below in terms of the number of literals, steps, and axioms it contains. A literal is a fully-ground atomic predicate which must be either true or false in any given state. For a problem with x literals, an upper bound on the size of the state-space is given by 2^x. In practice, the space will usually be much smaller because many literals are mutually exclusive—that is, when one is true, the other can only be false and vice versa. Recall from Chapter 3 that a step is a fully-ground instance of an operator. The number of steps provides an upper bound on the number of out-edges that a single node in the state-space can have. In practice, the number of out-edges will usually be much smaller because not every step's preconditions are satisfied in every state. An axiom is similar to a step,

except that it is not intended by any characters and it must be taken if its preconditions are fulfilled. Axioms are a form of syntactic sugar used by some planners to represent ideas such as, "If a character has a weapon, that character is armed." Because axioms must be applied, the planner does not make any non-deterministic choices regarding axioms, so they do not affect the size of the search space.

The size of each search space is given after it has been simplified. For a given domain and problem, it is possible to detect in polynomial time certain literals which will always be true or always be false and certain steps and axioms whose preconditions can never be satisfied. This is done by creating a plan graph as described in Section 5.2.2 from the initial state of the problem and extending it until it has leveled off. If a literal p does not appear in the last layer of the graph, then $\neg p$ must always be true. If a step s or axiom a does not appear in the last layer of the graph, its preconditions can never be satisfied. Note that this simplification may not remove all such literals, step, and axioms; however it does provide a fast and convenient way to reduce the size of the search space.

1. *Aladdin* - This domain and problem were originally presented by Riedl [60] for the evaluation of IPOCL, the original intentional planning algorithm. It allows the planner to tell a story similar to that of Aladdin from *One Thousand and One Nights*. The state-space for this problem contains 294 literals, 213 steps, and 165 axioms. The shortest solution to this problem is 11 steps and does not require any non-executed steps. Note that this domain also allows one character to delegate its goals to another. This has been accomplished in Glaive using axioms. The original domain and problem did not contain axioms and represented goal delegation using an extension of the POCL data structures.

2. *Heist* - This domain and problem were originally presented by Niehaus [54] for demonstrating a method of narrative generation that prompted specific inferences. It allows the planner to tell the story of a criminal in the American Old West who cheats, steals, and robs a bank among other crimes. Some syntax errors had to be corrected to use this domain, but I attempted to remain as faithful to the author's original intended narrative as possible. The state-space contains 323 literals, 1844 steps, and 0 axioms. The example solution given by Niehaus is 35 steps, but shorter solutions are possible. Though this domain was developed before the incorporation of non-executed steps into intentional planning, the domain allows for failed steps with no effects which serve a similar purpose. These failed steps were omitted from the domain because Glaive is able to reason about non-executed steps. For example, the `fail-buy-dress` action can simply be represented as a non-executed instance of `buy-dress`.

3. *Western* - This domain and problem were developed by me as one of the three story domains used for the experiment in Section 4.1, which explored how human subjects see

conflict in plan-based stories. It also takes place in the American Old West. A simplified version of this domain and problem were given in Figures 3.1 and 3.2; they have been used to provide examples throughout this document. The state-space contains 67 literals, 632 steps, and 0 axioms. The shortest solution is 7 steps, some of which must be non-executed.

4. *Fantasy* - The second domain and problem used in the experiment in Section 4.1. It takes place in a medieval kingdom and tells the story of a princess that must choose between two suitors. The state-space contains 80 literals, 46 steps, and 0 axioms. The shortest solution is 6 steps, all of which are executed.

5. *Space* - The third domain and problem used in the experiment in Section 4.1. It takes place in outer space and can tell the story of an explorer who encounters hazards from the environment and from alien lifeforms. The state-space contains 46 literals, 23 steps, and 0 axioms. The goal of this problem can be accomplished in only 2 steps, both of which are happenings. Therefore, this problem does not necessarily exercise the intentional or conflict planning features of Glaive, but it has been included for completeness.

6. *Raiders* - This domain and problem were developed by me for testing Glaive. They tell a highly-simplified version of the plot of *Indiana Jones and the Raiders of the Lost Ark*. This film was chosen because it provides an example of a narrative trope called the *McGuffin Delivery Service* in which the antagonist's plan relies on a partially-executed plan by the protagonist. The state-space contains 46 literals, 68 steps, and 6 axioms. The shortest solution is 8 steps, some of which can be non-executed.

7. *Best Laid Plans* - This domain represents the mechanics of the interactive narrative adventure game *The Best Laid Plans* (described below in Section 6.2). It tells the story of a goblin minion on a quest to retrieve a item for his master and the things that go wrong during that quest. Two problems are given for this domain: *BLP-Win*, in which the goblin succeeds in his quest, and *BLP-Die*, in which the goblin dies during his quest. The state-space contains 215 literals, 705 steps, and 632 axioms. The shortest solution to *BLP-Win* is 10 steps, all of which are executed. The shortest solution to *BLP-Die* is 11 steps, some of which must be non-executed.

6.1.2 Results

The relative speed of Glaive is demonstrated in Table 6.1, which describes the problems Glaive is able to solve and how quickly. Run times are also given for a planner dubbed Narrative Fast Forward (NFF) for comparison. Note that NFF is not Hoffmann's Fast-Forward, but rather Glaive using the Fast-Forward heuristic instead of the heuristic described in Section 5.2.2. The

Table 6.1: Glaive's performance vs. Narrative Fast-Forward's performance on 8 benchmark intentional planning problems. Time is given as the average of 10 runs.

Problem	Planner	Solution?	Time (seconds)	Nodes Visited	Nodes Expanded
Aladdin	NFF	No	288.499	67,758	2,063,927
	Glaive	Yes	0.064	12	189
Heist	NFF	No	367.924	84,854	1,654,916
	Glaive	No	351.380	115,782	1,544,048
Western	NFF	No	440.552	166,110	3,079,097
	Glaive	Yes	28.414	18,855	296,150
Fantasy	NFF	Yes	2.768	35,407	212,046
	Glaive	Yes	0.032	14	107
Space	NFF	Yes	0.005	3	15
	Glaive	Yes	0.003	3	9
Raiders	NFF	Yes	0.231	1,334	4,073
	Glaive	Yes	0.033	35	142
BLP: Win	NFF	Yes	27.384	32,262	196,576
	Glaive	Yes	0.318	109	586
BLP: Die	NFF	No	195.935	106,750	731,948
	Glaive	No	236.972	91,887	720,024

two implementations share as much code in common as possible. They both reason about possible worlds and conflict and must both meet the same narrative criteria for a solution. The only difference between the two implementations is the calculation of the heuristic, which affects how quickly they are able to find a solution. Both planners use complete A* search. For each problem, a planner is given 6 Gb of memory on an computer with an 8-core 3.5 GHz Intel Core I7 processor. Note that where Table 6.1 indicates that no solution was found, this does not mean that no solution exists, simply that the planner ran out of memory before finding a solution.

Glaive is able to solve two problems which NFF fails to solve. Both planners are able to solve four of the problems, and in all four of these cases Glaive finds a solution in less time and by visiting the same number of nodes or fewer—significantly fewer in most cases. Two problems were not solved by either planner, but Glaive visits more nodes before running out of memory. This is possible because Glaive avoids considering steps which can never possibly be motivated, so its average branching factor is lower.

Comparing Glaive to POCL-style planners is more tenuous because of the fundamental differences in the two families of algorithms, but one anecdote may provide some evidence of Glaive's relative performance. The original implementation of IPOCL, which solved the Aladdin problem, took 12.5 hours on a computer with an Intel Core2 Duo 3GHz processor,

3 Gb of memory, and 100 Gb of virtual memory. It also required a domain-specific heuristic to achieve this result, which means that a human author created the heuristic with expert knowledge that helped the planner search more efficiently. IPOCL visited 673,079 nodes and expanded 1,857,373 nodes. By contrast, Glaive uses a domain independent heuristic, takes only 64 milliseconds, visits only 12 nodes and expands only 189. This is an especially encouraging result because the solution Glaive finds is 11 steps long, meaning that Glaive visits only 1 node that is not on the direct path to the solution during the search.

6.2 Narrative Generation in *The Best Laid Plans*

6.2.1 Game Design

The Best Laid Plans is an interactive narrative point-and-click adventure game set in a medieval fantasy world. It was created with the Unity 3D engine. The player takes on the role of the quivering goblin minion of a powerful sorcerer called the Dark Overlord. The Dark Overlord has tasked the goblin to go to town, retrieve a bottle of hair tonic, and return with it to the Dark Tower. The game tells the story of how that seemingly simple quest constantly goes awry.

Mechanics

The player can visit 15 different locations, use 17 different items, interact with seven human characters and two animal characters, and cast four kinds of spells. Figure 6.1 shows the layout of the game world with important locations labeled. Ten kinds of actions are available to the player. Figure 6.2 lists nine, with the tenth being the ability to walk from place to place. All of these actions (except for *Look At*) can also be taken by the human characters. The animal characters are limited to *Walk* and *Attack*. Figure 6.3 shows a screenshot of the game that demonstrates the interface.

Story Planning and Rendering

The game alternates between two modes: "Make Your Plan" mode and "Watch Your Story Unfold" mode. Play begins in "Make Your Plan Mode," which allows the player to act out how the goblin will obtain the hair tonic. The other characters are minimally reactive in this mode, meaning that they may make simple responses to the goblin's actions but will never act of their own volition. Each action the player takes costs 1 mana, and the player is limited to 25 mana each time the game enters "Make Your Plan" mode, so the player's plan may contain at most 25 steps. Each action taken in this mode adds 1 to the player's score. The goal of the game is to minimize score, which incentivizes shorter plans over longer plans.

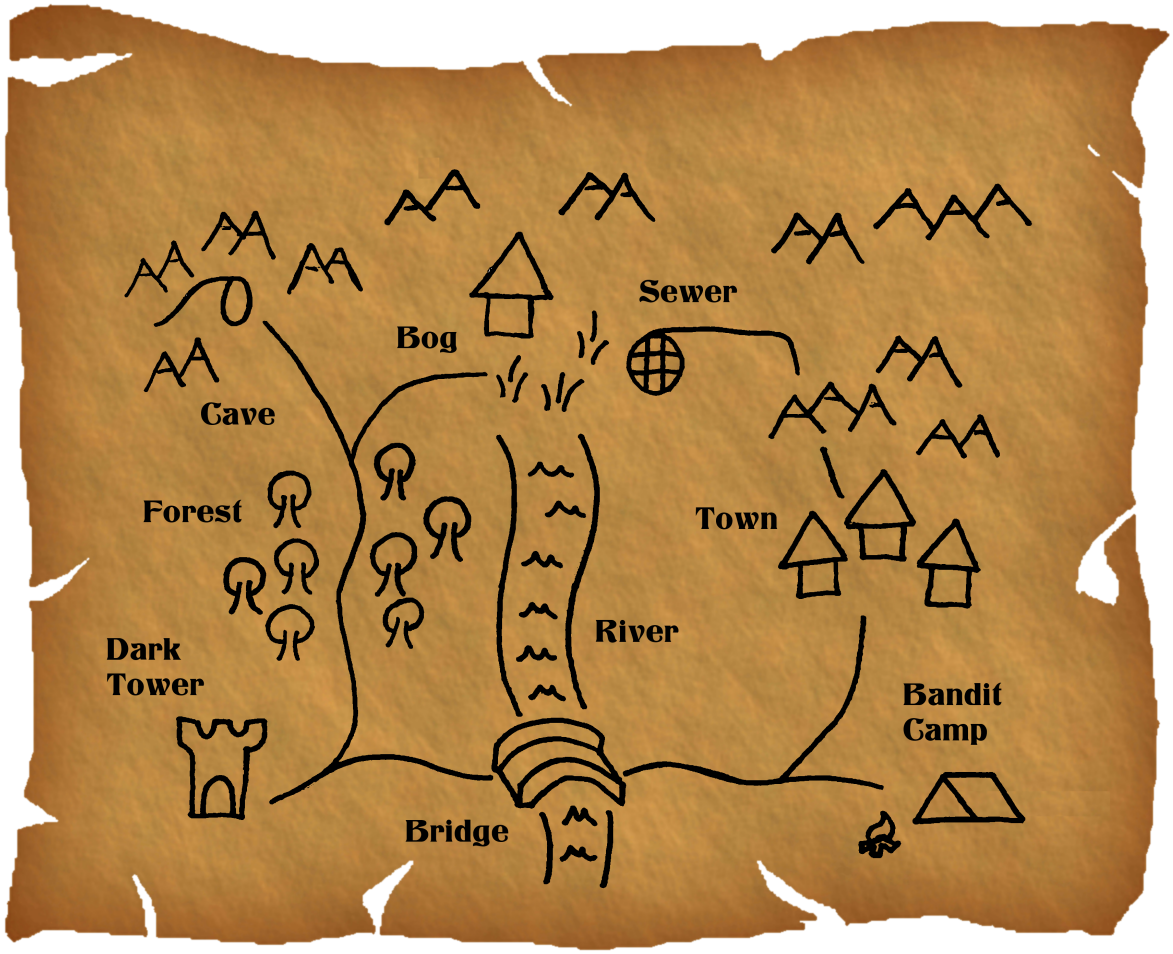


Figure 6.1: An excerpt from The Best Laid Plans instruction manual showing all the locations the player can visit during the game.

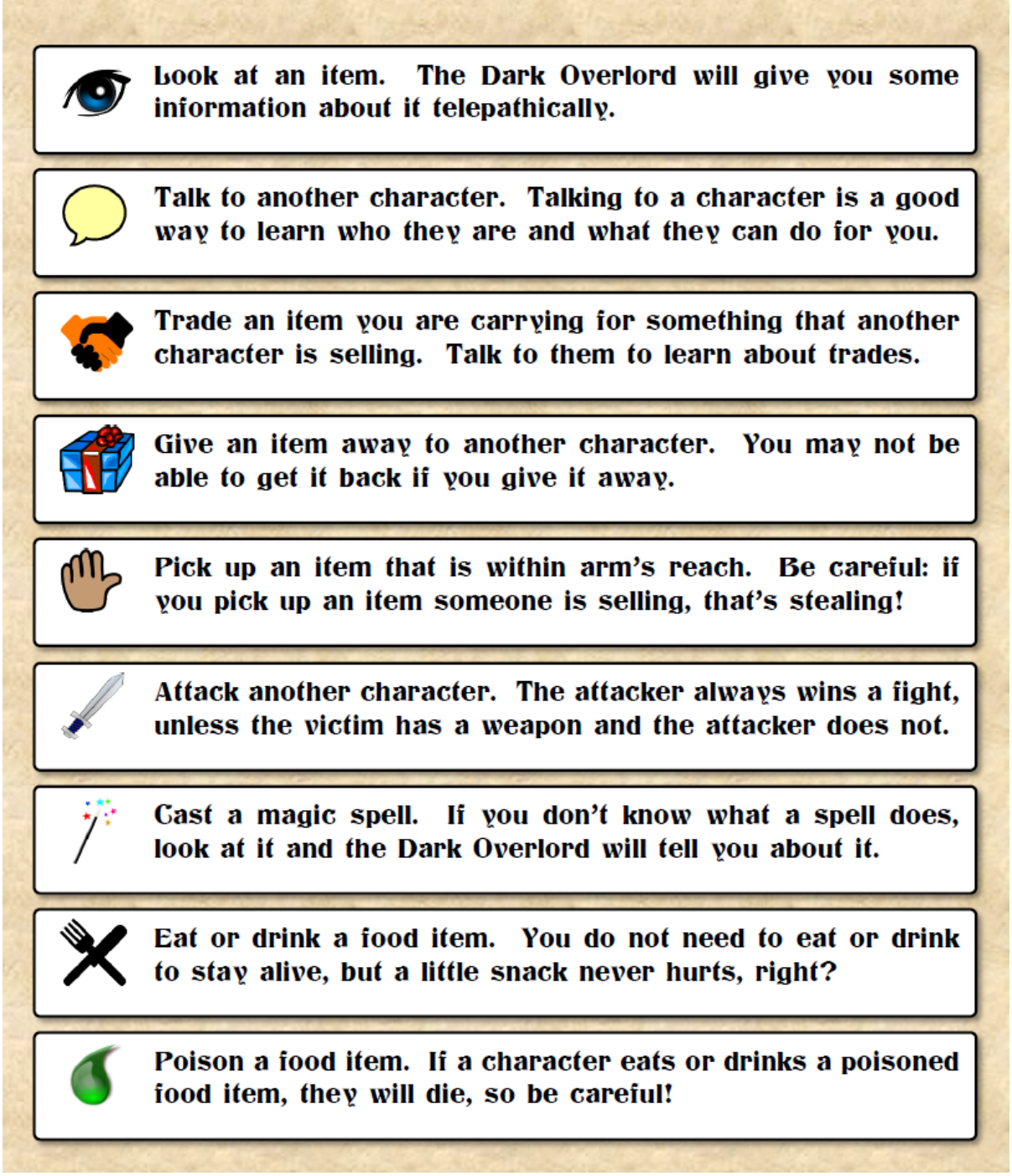


Figure 6.2: An excerpt from *The Best Laid Plans* instruction manual describing each of the actions available to the player during the game.

Figure 6.3: A screenshot demonstrating the interface for *The Best Laid Plans*. The current game mode of "Make Your Plan" is indicated at the top of the screen. The Dark Overlord views the action telepathically from the top left and occasionally comments on the goblin's actions or provides information. The player's inventory, mana, and score are shown at the bottom. The plan that the player has acted out so far is listed to the left under the portrait of the Dark Overlord. This location is the town. The characters pictured here are (from left to right) the town guard, the goblin, and the merchant. A sword lies on the market stall which can be purchased or stolen. This location leads to four others: the tavern (West), the alley (North), the potion shop (East), and the junction (South).

Figure 6.4: The client / server architecture of *The Best Laid Plans*

Once the goblin has acted out a successful plan to get the hair tonic and return to the tower, the game enters "Watch Your Story Unfold" mode. Now the player watches the goblin execute the plan that was just acted out. In this mode, other characters may act of their own volition to thwart the goblin's plan. At any point, the player can choose to abandon the current plan and return to "Make Your Plan" mode. Also, if the goblin dies, the Dark Overlord brings him back to life, rewinds time to the moment before the goblin's death, and returns the game to "Make Your Plan" mode. This process iterates until the goblin succeeds in bringing the hair tonic to the Dark Tower or gives up. To ensure that the game cannot be placed in an unwinnable state, the first action of the goblin's plan is guaranteed to occur before other characters are allowed to act.

The game has a simple client / server architecture which is visualized in Figure 6.4. The client, created in Unity 3D, allows the player to act out a plan. That plan is then sent over a socket to the Java server, which uses the Glaive planner to add actions for all the other characters to the plan. The author's goal for the planner is that the goblin die during his quest, but per the restrictions of intentional planning, this can only be accomplished if the other characters have a reason to kill the goblin. The resulting complete story is then sent back to the client where it is visualized for the player. If the goblin fails to retrieve the tonic, the player acts out a new plan. If the goblin succeeds, the player wins.

Player's First Plan	Server's First Story
The goblin walks to the crossroads.	The goblin walks to the crossroads.
The goblin walks to the bridge.	The goblin walks to the bridge.
The goblin walks to the junction.	The goblin walks to the junction.
The goblin walks to the town.	The goblin walks to the town.
The goblin walks to the potion shop.	The goblin walks to the potion shop.
The goblin trades gold to the chemist for the hair tonic.	The goblin trades gold for the chemist's hair tonic.
The goblin walks to the town.	The goblin walks to the town.
The goblin walks to the junction.	The goblin walks to the junction.
The goblin walks to the bridge.	The bandit walks from the camp to the junction.
The goblin walks to the crossroads.	The bandit attacks and kills the goblin.
The goblin walks to the Dark Tower.	(Dark Overlord undoes the last action.)

Player's Second Plan	Server's Second Story
The goblin walks to the town.	The goblin walks to the town.
The goblin steals the sword from the merchant.	The goblin steals the sword from the merchant.
The goblin walks to the junction.	The bandit walks to the town.
The goblin attacks and kills the bandit.	The town guard attacks and kills the bandit.
The goblin walks to the bridge.	The town guard attacks and kills the goblin.
The goblin walks to the crossroads.	(Dark Overlord undoes the last action.)
The goblin walks to the Dark Tower.	

Player's Third Plan	Server's Third Story
The goblin attacks and kills the town guard.	The goblin attacks and kills the town guard.
The goblin walks to the junction.	The weapons merchant picks up the bandit's sword.
The goblin walks to the bridge.	The weapons merchant attacks and kills the goblin.
The goblin walks to the crossroads.	(Dark Overlord undoes the last action.)
The goblin walks to the Dark Tower.	

Player's Fourth Plan	Server's Fourth Story
The goblin attacks and kills the merchant.	The goblin attacks and kills the merchant.
The goblin walks to the junction.	The goblin walks to the junction.
The goblin walks to the bridge.	The goblin walks to the bridge.
The goblin walks to the crossroads.	The goblin walks to the crossroads.
The goblin walks to the Dark Tower.	The goblin walks to the Dark Tower.
	(The player has won.)

Figure 6.5: Example transcript between *The Best Laid Plans* client and server (read from left to right, top to bottom).

The Best Laid Plans is designed to serve as a testbed for plan-based interactive narrative mediation techniques. The reason for the distinction between "Make Your Plan" mode and "Watch Your Story Unfold" mode is to avoid the need for a complex model of discourse. By eliciting the player's exact plan, we give both the player and the planner access to the story's fabula. Ideally, both of these modes would be merged into a single play experience which does not interrupt the player's immersion. Accomplishing this will require an accurate method of plan recognition, a large and difficult problem outside the scope of this work. This challenge will be made even more difficult by the fact that McKoon and Ratcliff demonstrate that people do not usually form complex predictions about the future when experiencing a narrative [49]. This means that players may not even make a well-formed plan for the server to recognize and mediate unless prompted to do so.

Story Planning Constraints

To ensure speed and consistency, Glaive searches the first 5,000 nodes of the state-space using complete A* search and returns the best solution discovered during that time. If no solution is found, the player's plan is returned unmodified and the player wins.

Under these constraints, Glaive almost always finds a way to thwart the goblin if one exists. This may seem counter-intuitive given that Glaive fails to solve *BLP-Die* above when given more time and memory. However, the most expensive part of this search is finding the goblin's plan to retrieve the hair tonic. Because most of this plan usually does not contribute directly to the author's goal of the goblin being dead, the Glaive heuristic has a harder time finding it than when solving *BLP-Win.* This problem is avoided in *The Best Laid Plans* by seeding the search space with the player's plan before beginning the search. In other words Glaive does not start with an empty search space, but rather with the search space that would result in discovering the player's plan (e.g. if the player's plan is 5 steps long, Glaive begins with a search space that has 5 nodes already visited). This saves the planner from needing to find a plan for the goblin, which makes sense because the planner cannot control the goblin anyway and should not spend effort trying to do so.

If multiple solutions are found during the search, Glaive uses the following domain-independent criteria to choose a best solution:

1. Prefer the story in which the player witnesses the highest number of goals achieved by other characters.

2. In the event of a tie, prefer the story in which 75% of the player's original plan is executed before the player dies (or as close to 75% as possible).

3. In the event of a tie, prefer the shorter story.

4. In the event of a tie, prefer the story in which the player witnesses the highest number of goals acted on but not achieved by other characters.

These criteria were developed through experience with *The Best Laid Plans* domain. One interesting direction of future work will be to test whether or not these criteria are effective at choosing a best plan in other story domains as well.

6.2.2 Experimental Design

A human subjects experiment was carried out to test whether or not players of *The Best Laid Plans* would recognize intentionality and conflict in the behaviors of the game's non-player characters. Three different versions of the server component were created to evaluate this:

1. *Control* - The goblin's plan is not modified by the server. This means that the first plan the player acts out always succeeds and the other characters do nothing. Technically, this can be considered equivalent to intentional planning alone—that is, intentional planning without the ability to reason about conflict—as far as the goblin is concerned. Stories in which the goblin takes at least 1 action but later dies cannot be represented as valid intentional plans because non-executed steps are needed to fill in the rest of the goblin's plan that did not succeed.

2. *Glaive* - The behaviors of the other characters are controlled by the Glaive planner as described in the preceding section.

3. *Scripted* - The behaviors of the other characters are controlled by a set of triggers written by a human author using a declarative scripting language. These triggers express concepts such as "If the goblin steals an item, he is a criminal," and "If the goblin is a criminal and he is in the same location at the guard, the guard will attack the goblin." I wish to acknowledge Rogelio E. Cardona-Rivera, a colleague with both knowledge of planning and experience in game design, for writing these triggers. The final number of trigger templates was 10, which when fully grounded are translated into 44,733 possible triggers. This version is meant to approximate the video game industry's current approach to interactive narrative: hand-authored scripts that must anticipate every important narrative situation at design time.

Participants were recruited from undergraduate and graduate Computer Science students at North Carolina State University. Most students were offered class participation credit or bonus credit by their instructors for participating, but no additional compensation was offered. Subjects were first shown a tutorial video that explained the game's interface, mechanics, and goal.

Then players were given one of the three versions of the game and asked to play it only once. After the player won or chose to quit, he or she was given a short post survey.

A total of 75 subjects participated. From those, 4 had to be removed from consideration because the game crashed. 5 had to be removed because they played the game multiple times. 1 had to be removed for not watching the tutorial video and expressing significant confusion about the interface and goal of the game. 1 had to be removed for having significant knowledge of the Glaive system before participating. The remaining 64 subjects were split nearly evenly between the three treatments:

- Control: 21 participants; 16 male, 4 female, 1 transgender; average age 22
- Glaive: 21 participants; 17 male, 4 female; average age 21
- Scripted: 22 participants; 16 male, 6 female; average age 22

The first 5 statements on the post survey were designed to test whether or not players attributed intentional behavior to the other characters in the game and whether or not they recognized conflict. Subjects were asked to rate their agreement with the following statements on a 7 point Likert scale, with 1 being "strongly disagree," and 7 being "strongly agree."

1. The other characters had good reasons for their actions.
2. The other characters were following their own goals.
3. The other characters were reacting to the things I did.
4. Some characters were trying to help me accomplish my goals.
5. Some characters were trying to prevent me from accomplishing my goals.

The following statements (also a 7 point Likert scale) were designed to measure the player's experience of agency. These statements were taken from a similar instrument used by Fendt, et. al. [25] that operationalized Murray's definition of agency. Murray defines agency as the player's experience of satisfying power to take meaningful action and see the results of his or her decisions and choices [52].

6. The actions I took were meaningful in the context of the story.
7. My actions were important to the progression of the story.
8. I was able to see the results of my actions.
9. The story would have been different if I had taken different actions.

Table 6.2: Questions on *The Best Laid Plans* post survey which showed a significant difference between treatments.

Question	Hypothesis	p Value
The other characters were following their own goals.	Glaive > Control	0.006
	Scripted > Control	0.000
The other characters were reacting to the things I did.	Glaive > Control	0.005
	Scripted > Control	0.006
Some characters were trying to prevent me from accomplishing my goals.	Glaive > Control	0.000
	Scripted > Control	0.000
This game was challenging.	Glaive > Control	0.025

10. I had control over aspects of the story that I wanted control over.

The final set of statements (also a 7 point Likert scale) was designed to measure player engagement.

11. The story in this game was interesting.

12. This game was challenging.

13. I enjoyed playing this game.

14. I would play this game again.

The general hypothesis is that Glaive and Scripted will outperform Control, but that there will be no significant difference between Glaive and Scripted. Specifically, participants will rate all questions except question 4 higher for Glaive and for Scripted than for Control. There will be no significant differences in how participants rate question 4 across the three treatments. The first set of questions is the most important for validating that the narratives generated by Glaive successfully communicate intentionality and conflict. The second and third sets of questions are of secondary importance.

6.2.3 Results

The Wilcoxon rank-sum test [42] was used to compare the responses to each statement pairwise between treatments. It is a non-parametric test for independent samples of ordinal data. This test not only allows the null hypothesis to be rejected but can also confirm an alternative hypothesis that one statement has higher agreement than the other. For each statement and for each pair of treatments, this test was used to see if players agreed with the statement significantly more for one treatment than in the other. For example, Glaive can be compared to Control for statement 1. The null hypothesis is that both sets of participants agreed with

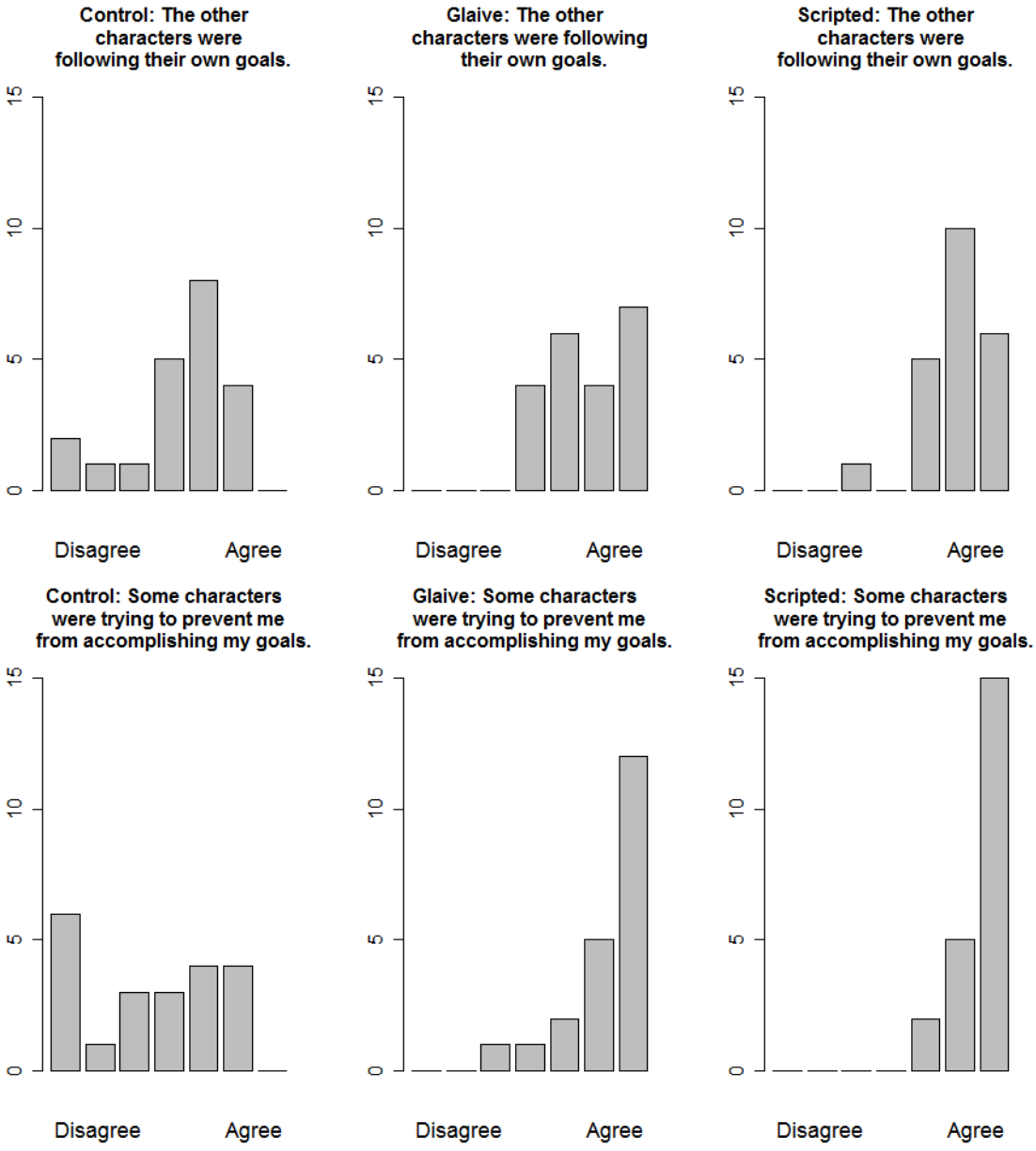


Figure 6.6: Responses to two questions designed to measure the perception of intentionality and conflict on *The Best Laid Plans* post survey.

the statement to the same extent. The alternative hypothesis is that participants who played Glaive agreed more with the statement than participants who played Control.

Four statements on the survey showed significant differences between treatments ($p < 0.05$). They are given in Table 6.2 along with the alternative hypothesis and p value. The first set of statements, which measures the player's perception of intentionality and conflict, are the most important. 4 of the 5 statements in that set showed the expected behavior. Participants who played Glaive and Scripted attributed intentionality to the non-player characters in the game, as demonstrated by the fact that they more strongly agreed with the statements "The other characters were following their own goals," and "The other characters were reacting to the things I did." Participants who played Glaive and Scripted recognized conflict in the story, as demonstrated by the fact that they more strongly agreed with the statement "Some characters were trying to prevent me from accomplishing my goals," and neither agreed nor disagreed more strongly with the statement "Some characters were trying to help me accomplish my goals." The responses to two of these statements are visualized in Figure 6.6. Statement 1, "The other characters had good reasons for their actions," showed no significant difference between treatments, which is counter to the hypothesis.

There were no significant differences between treatments for any of the statements in the second set that measures agency. For the third set of statements, designed to measure engagement, participants who played Glaive more strongly agreed with the statement, "This game was challenging," than those who played Control. There was no significant difference between Scripted and Control for this statement.

6.2.4 Discussion

These results are encouraging. While more significant differences would have been preferred, no statements confirmed the opposite alternative hypothesis. In other words, while it would have been ideal for Glaive to outperform Control on the statement, "I enjoyed playing this game," at least the Control did not outperform Glaive.

The most important statements demonstrated the expected behavior. Players attributed goal-directed behavior to the game's non-player characters and recognized that the actions of those characters were thwarting their plans (that is, the plans of the goblin). The fact that Glaive outperformed Control on the statement, "This game was challenging," suggests that players experienced a connection between the conflicts in the narrative and the mechanics of the game.

It is interesting that, despite recognizing goal-directed behavior, there was no difference for the statement, "The other characters had good reasons for their actions." McKoon and Ratcliff's findings [49] may suggest an explanation; players do not make inferences about what

a character will do, but they will retroactively seek an explanation for a character's behavior once it is observed. For example, Control players may not necessarily expect the town guard to respond to the goblin stealing an item, so when he does nothing it does not prevent them from agreeing with that statement. However, when Glaive and Scripted players see the town guard attack after the goblin has stolen an item, that motivation is retroactively assigned to the guard's behavior. If so, this suggests a directive for narrative design: no action is better than actions which cannot be explained.

It is also surprising that participants did not experience a difference in agency between the treatments. In other words, players seem to feel the same high satisfying power in a game where the other characters do nothing as in a game where the other character act intelligently, as visualized in Figure 6.7. This may suggest that the perception of agency is minimally influenced by the behavior of non-player characters. A more likely explanation is that this uniformity of response reflects Murray's player-centric definition of agency [52] that was operationalized in the post survey. Mateas [45] defines agency differently, claiming that there must exist a balance between material affordances (what a player can do in a game) and formal affordances (what the game motivates the player to do). Murray's definition focus mainly on material affordances, so it would be interesting to investigate the formal affordances of *The Best Laid Plans* to see if more insight can be gained about the role of non-player characters in the perception of agency.

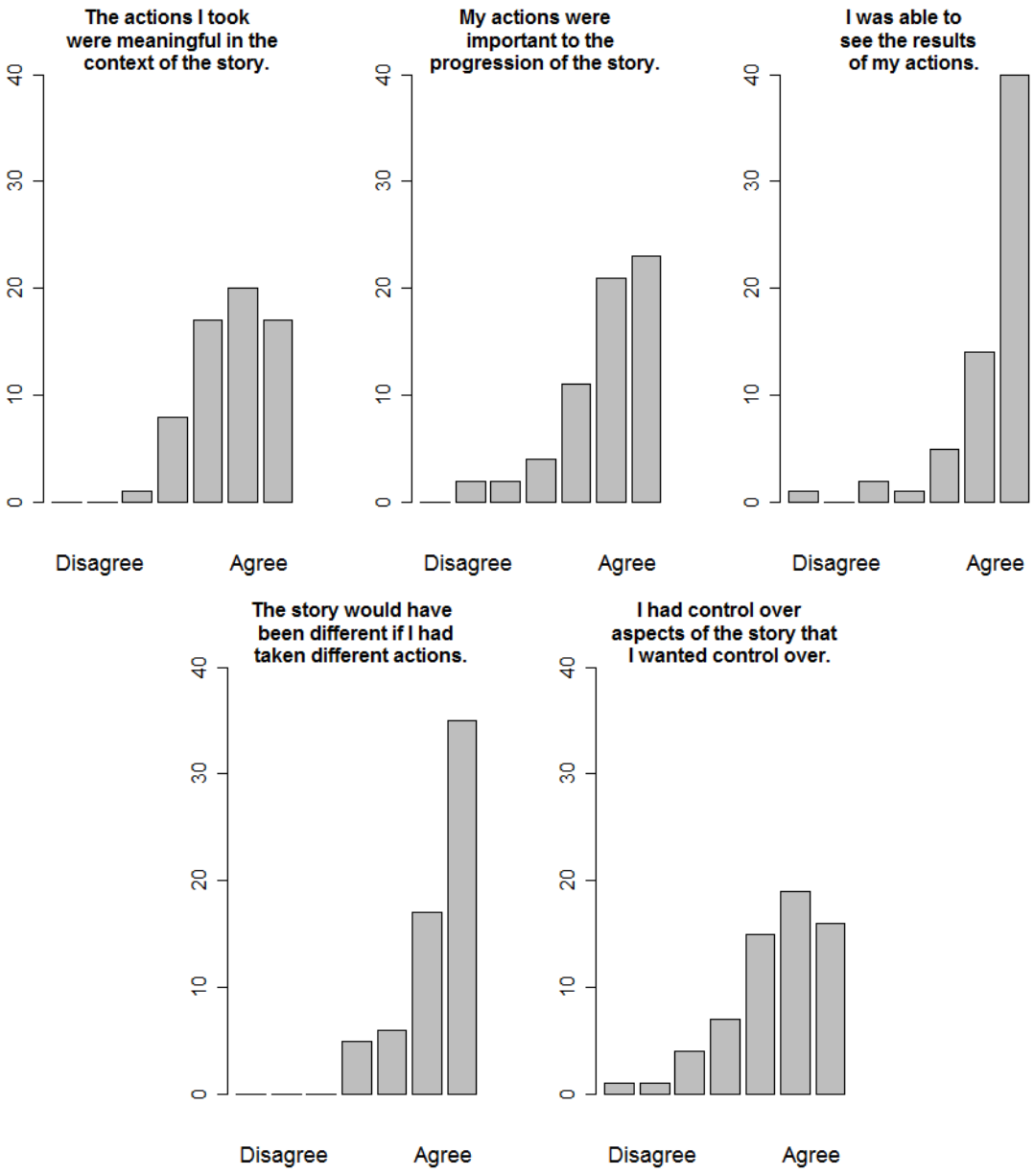


Figure 6.7: Responses to five questions designed to measure agency on *The Best Laid Plans* post survey. No significant differences were observed between treatments.

Chapter 7

Conclusion

7.1 Summary

The research I describe here demonstrates how the essential narrative phenomenon of conflict can be operationalized as the thwarted plans of intentional agents. Here I define the Conflict Partial Order Causal Link model for expressing stories. This model contains an explicit representation of character intentionality and failed subplans. By representing non-executed steps, a CPOCL plan can contain a complete description of how an agent intended to achieve a goal even if its subplan for that goal failed. This extension allows a plan to retain important threatened causal links, which correspond to conflict, while still guaranteeing that the plan will achieve the author's goal. It constrains the definition of a valid plan to be those in which every character's actions are clearly motivated and goal-oriented but not every character's subplan succeeds. Non-executed steps in a CPOCL plan also provide a representation of alternate worlds where different subplans succeed or fail. Given a means of measuring agent utility and the likelihood of plan success, reasoning about these alternate worlds allows the formalization of dimensions that can be used to distinguish one conflict from another, such as *balance, directness*, *intensity*, and *resolution*.

One attractive feature of plan-based models is the ability to generate them using a planner. This research presents the plan-space search CPOCL algorithm which directly manipulates the CPOCL representation, though the speed of POCL-based algorithms limits their applicability in real time systems. The Glaive algorithm demonstrates how advances in state-space heuristic search can be applied to a narrative planning algorithm to achieve a significant speedup. The constraint that steps in an intentional plan must be goal directed allows Glaive to reduce its branching factor and calculate a more accurate heuristic by reasoning not just forward from an initial state but also backwards from a goal state. To reason about failed plans and conflict, Glaive treats the entire search space as a meaningful data structure of possible worlds. Any

node where a character's goal is achieved can inform the other nodes in the search space by providing an explanation for the steps that agent took to achieve that goal. Different nodes in the search space can be compared, which equates to a comparison of possible words for discovering valuable information, such as how plans conflict and thwart one another.

This work has been empirically evaluated in a series of human subjects experiments. The first demonstrated that a human audience can effectively recognize threatened causal links in a CPOCL plan. When given static text story fabulae and asked to report instances of characters thwarting the plans of others, participants agreed which conflicts to report and their reports corresponded to threatened causal links in CPOCL plans. The second experiment demonstrated that when given simple definitions for the dimensions of *balance, directness, intensity*, and *resolution* that a human audience agreed on which stories had high and low values for these dimensions and that their responses mirrored the rankings defined by a set of formulas meant to operationalize those concepts. The third evaluation demonstrated that Glaive can be used to control the plot of *The Best Laid Plans* in real time. Players recognized when other agents in the game were thwarting their plans. They noticed these conflicts more than in a control where the other agents took no action and reported no significant differences between the plots produced by Glaive and those specified by a human author at design time.

7.2 Limitations and Future Work

Though this work has endeavored to present the complete lifecycle of a computational model of narrative, there are many ways in which it still needs to be extended. One of the key limitations of this work is that it only provides a model of how conflict can be represented in stories, but not a theory of when conflict should be included. Chapter 3 described how the CPOCL model subsumes the IPOCL model by representing stories in which one or more character plans fail. However, the solution space for a conflict planning problem may include solutions which contain no conflict if such a story is possible. This raises some important questions about the use of conflict as a rhetorical device.

Consider a simple example: Character 1 has cake but wants money. Character 2 has money but wants cake. One solution to this problem is that both characters simply trade, but this story contains no conflict. Should this be included as a solution to the problem or not? This is a question of aesthetics, and the answer depends on the communicative goals of the author. Conflict has been identified as a key element of interesting stories, but does that imply that this solution should be excluded from the set of possible solutions or that the story domain needs to be revised to tell a more interesting space of stories? How much conflict is too little, and how much is too much? Stories have been criticized for having no conflict at all, but rarely are they criticized for having too much. This makes it difficult to map out the boundaries of this aesthetic

space based on the existing corpus of human stories. Because of this difficulty, and because of the disagreement that exists between critics, my work has focused on empowering authors to represent and reason computationally about important narrative tools but has avoided a prescriptive theory of how and when to use those tools. If an aesthetic theory of when conflict should be created, when it should be avoided, and what values it should have for the seven dimensions can be developed, that theory will be a valuable extension to this work.

Perhaps the most promising direction for future development will be incorporating discourse-level reasoning about conflict into this model. All aspects of this work have assumed that the audience has a god's eye view of the story fabula. This has imposed significant constraints on the evaluation, e.g. the immersion-breaking distinction between "Make Your Plan" mode and "Watch Your Story Unfold" mode in *The Best Laid Plans*. The results of the first experiment demonstrated that CPOCL represents a superset of the conflicts that a human audience notices in a story, and one promising explanation for this inaccuracy is the model's inability to reason about what aspects of the narrative the audience pays most attention to when consuming a story.

A number of important discourse-level problems will need to be solved to transcend these fabula-only constraints. Firstly, we need to know to what extent an audience infers the plans of characters in a narrative. Conflict is inherently tied to some representation of a possible world in which a plan might have succeeded but instead failed. What is needed to communicate the details of that possible world to an audience? Surely there is a more elegant solution than the thought bubbles used in the first experiment to communicate every agent's exact plan to the audience. The problem of communicating the author's future plans to the audience is an interesting one, but an even more challenging problem exists in the opposite direction. If the model is to be used in an interactive setting, the author needs some means of predicting what the audience is planning. This might be accomplished though plan recognition. Glaive uses the specific constraints of narrative to speed up the generation process, and these same constraints might also be used to improve plan recognition in a narrative context. The data collected from the evaluation of *The Best Laid Plans* will serve as an excellent starting point for this investigation. In addition to plan recognition, we might also develop more sophisticated ways of eliciting the audience's plan and/or prompting the audience to form one. This presents a number of interesting challenges for research in interactive narrative mediation and discourse.

There are a number of computational improvements that can be made to the narrative generation process. Fast-Forward made a good foundation for Glaive due to its simplicity, but the Fast-Downward planner [32] is significantly faster and more memory efficient. It is likely that another order of magnitude speedup can be achieved by incorporating narrative constraints into this considerably more complex but more accurate heuristic. Perhaps more attractive than Fast-Downward's speed is that its method of problem decomposition has important parallels to

the way that a complete story can be decomposed into chains of intentional actions that make up the subplans of each agent. This suggests that Fast-Downward can make both computational and cognitive improvements to the narrative planning process. While it is satisfactory that Glaive's generation of conflict in *The Best Laid Plans* is not significantly different from that of a human author, the eventual goal of this research is to surpass a human author—not necessarily in the quality of the stories produced but in the size of the domain it can handle. By continuing to speed up the narrative generation process, it should be possible to control a narrative at run time in a domain which is too large for any human author to anticipate at design time.

There are many ways in which this model could be further extended, and there are also aspects of this work which should bear fruit for other computational models. One promising element of Glaive, aside from its speed, is its elegant representation of the entire narrative search space as a database of possible worlds. Possible worlds semantics has been used to analyze more aspects of narrative than just conflict [65] and has a solid formal basis in modal logic [39]. Possible worlds might be used, for example, to reason intelligently about how inaccurate and incomplete information affect the behavior of agents in a story world. One can imagine numerous ways that knowing which worlds are possible and which are impossible will benefit interactive narrative mediation.

7.3 Closing Remarks

The research program that investigates computational models of narrative has brought together narratology, computer science, and cognitive science into a symbiotic system. Each makes contributions to and prompts further investigation by the other. This work provides one example of how narratologists defined conflict in terms of thwarted plans, that definition was operationalized into a plan-based model that leverages artificial intelligence techniques, and finally that model was evaluated in a human context with both static and interactive narratives. I hope this work has made at least a small contribution to all three disciplines and has generated more questions than it has answered. I believe it has shed some light on the complex mental machinery that we use to construct and understand stories and has advanced the science of building artificial machinery to mirror these tasks. Eventually, I believe it will serve as one small step toward a fundamentally new kind of interactive narrative experience in which authorship can truly be said to be distributed between the author, the audience, and the machine.

References

[1] H. P. Abbott. *The Cambridge introduction to narrative*. Cambridge University Press, 2008.

[2] R. Aylett, M. Vala, P. Sequeira, and A. Paiva. Fearnot!–an emergent narrative approach to virtual dramas for anti-bullying education. In *Proceedings of the 4th international conference on virtual storytelling: using virtual reality technologies for storytelling*, pages 202–205, 2007.

[3] B.C. Bae and R. Young. A use of flashback and foreshadowing for surprise arousal in narrative using a plan-based approach. In *Proceedings of the International Conference on Interactive Digital Storytelling*, pages 156–167. Springer, 2008.

[4] B.C. Bae and R. Young. Suspense? surprise! or how to generate stories with surprise endings by exploiting the disparity of knowledge between a story's reader and its characters. In *International Conference on Interactive Digital Storytelling*, pages 304–307. Springer, 2009.

[5] M. Bal. *Narratology: Introduction to the theory of narrative*. University of Toronto Press, 1997.

[6] Heather Barber and Daniel Kudenko. Dynamic generation of dilemma-based interactive narratives. In *Proceedings of The 3rd AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, pages 2–7, 2007.

[7] A. Bhattacharyya. On a measure of divergence between two statistical populations defined by their probability distributions. *Bulletin of the Calcutta Mathematical Society*, 35(99-109):4, 1943.

[8] A. L. Blum and M. L. Furst. Fast planning through planning graph analysis. *Artificial Intelligence*, 90(1-2):281–300, 1997.

[9] B. Bonet and H. Geffner. Planning as heuristic search. *Artificial Intelligence*, 129(1-2):5–33, 2001.

[10] M. Booth. The AI systems of Left 4 Dead. In *Keynote, Artificial Intelligence and Interactive Digital Storytelling*, 2009.

[11] M. Bratman. *Intentions, Plans, and Practical Reason*. Harvard University Press, 1987.

[12] Selmer Bringsjord and David Ferrucci. *Artificial intelligence and literary creativity: Inside the mind of BRUTUS, a storytelling machine*. Psychology Press, 1999.

[13] C. Brooks and R. P. Warren. *Understanding fiction*. Prentice Hall, 1979.

[14] Tom Bylander. Complexity results for planning. In *International Joint Conference on Artificial Intelligence*, volume 10, pages 274–279, 1991.

[15] J.G. Carbonell. Counterplanning: A strategy-based model of adversary planning in real-world situations. *Artificial Intelligence*, 16(3):295–329, 1981.

[16] R. E. Cardona-Rivera, B. A. Cassell, S. G. Ware, and R. M. Young. Indexter: A computational model of the event-indexing situation model for characterizing narratives. In *Proceedings of the Computational Models of Narrative Workshop*, pages 34–43, 2012.

[17] N. Carroll. *Theorizing the moving image*. Cambridge University Press, 1996.

[18] M. Cavazza, F. Charles, and S. J. Mead. Character-based interactive storytelling. *Intelligent Systems*, pages 17–24, 2002.

[19] Fred Charles, Miguel Lozano, Steven J Mead, Alicia Fornes Bisquerra, and Marc Cavazza. Planning formalisms and authoring in interactive storytelling. In *Proceedings of the 1st International Conference on Technologies for Interactive Digital Storytelling and Entertainment*, volume 3, 2003.

[20] Yun-Gyung Cheong, Rilla Khaled, Corrado Grappiolo, Joana Campos, Carlos Martinho, Gordon PD Ingram, Ana Paiva, and Georgios Yannakakis. A computational approach towards conflict resolution for serious games. In *Proceedings of the 6th International Conference on the Foundations of Digital Games*, pages 15–22, 2011.

[21] A. J. Coles, A. I. Coles, A. Garca Olaya, S. Jimnez, C. Linares Lopez, S. Sanner, and S. Yoon. A survey of the seventh international planning competition. *AI Magazine*, 2011.

[22] C. Crawford. *Chris Crawford on game design*. New Riders, 2003.

[23] Ansen Dibell. *Elements of Fiction Writing: Plot*. Writers Digest, 1988.

[24] L. Egri. *The art of dramatic writing*. Wildside, 1988.

[25] Matthew William Fendt, Brent Harrison, Stephen G. Ware, Rogelio E. Cardona-Rivera, and David L. Roberts. Achieving the illusion of agency. In *Proceedings of the 5th International Conference on Interactive Digital Storytelling*, pages 114–125, 2012.

[26] R. Fikes and N. J. Nilsson. STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 2(3/4):189–208, 1971.

[27] J. L. Fleiss, B. Levin, and M. C. Paik. *Statistical Methods for Rates and Proportions*. John Wiley & Sons, 3 edition, 2003.

[28] R. J. Gerrig. *Experiencing narrative worlds: On the psychological activities of reading*. Yale University Press, 1993.

[29] Pablo Gervás. Computational approaches to storytelling and creativity. *AI Magazine*, 30(3):49, 2009.

[30] J. Gratch. Émile: Marshalling passions in training and education. In *Proceedings of the 4th International Conference on Autonomous Agents*, pages 325–332, 2000.

[31] R. W. Hamming. Error detecting and error correcting codes. *Bell System Technical Journal*, 29(2):147–160, 1950.

[32] M. Helmert. The fast downward planning system. *Journal of Artificial Intelligence Research*, 26(1):191–246, 2006.

[33] D. Herman. *Story logic*. University of Nebraska Press, 2002.

[34] D. Herman, M. Jahn, and M. L. Ryan. *Routledge encyclopedia of narrative theory*. Routledge, 2005.

[35] Jörg Hoffmann and Bernhard Nebel. The ff planning system: Fast plan generation through heuristic search. *arXiv preprint arXiv:1106.0675*, 2011.

[36] D. Howard and E. Mabley. *The tools of screenwriting: A writer's guide to the craft and elements of a screenplay*. St. Martin's Griffin, 1995.

[37] S. Kambhampati, C. A. Knoblock, and Q. Yang. Planning as refinement search: A unified framework for evaluating design tradeoffs in partial-order planning. *Artificial Intelligence*, 76(1-2):167–238, 1995.

[38] M. G. Kendall. *Rank correlation methods.* Griffin, 1948.

[39] Saul A Kripke. Semantical considerations on modal logic. *Acta Philosophica Fennica*, 16:83–94, 1963.

[40] J. R. Landis and G. G. Koch. The measurement of observer agreement for categorical data. *Biometrics*, pages 159–174, 1977.

[41] Michael Lebowitz. Story-telling as planning and learning. *Poetics*, 14(6):483–502, 1985.

[42] Erich Leo Lehmann and Howard J. M. D'Abrera. *Nonparametrics: statistical methods based on ranks*. Springer, 2006.

[43] B. Li and M. O. Riedl. An offline planning approach to game plotline adaptation. In *Proceedings of Artificial Intelligence for Interactive Digital Entertainment*, 2010.

[44] S.C. Marsella and J. Gratch. Ema: A process model of appraisal dynamics. *Cognitive Systems Research*, 10(1):70–90, 2009.

[45] Michael Mateas. A preliminary poetics for interactive drama and games. *Digital Creativity*, 12(3):140–152, 2001.

[46] Michael Mateas. *Interactive drama, art and artificial intelligence*. PhD thesis, Carnegie Mellon University, 2002.

[47] Michael Mateas and Andrew Stern. Façade: An experiment in building a fully-realized interactive drama. In *Game Developers Conference*, pages 4–8, 2003.

[48] D. McAllester and D. Rosenblitt. Systematic nonlinear planning. In *Proceedings of the 9th National Conference on Artificial Intelligence*, pages 634–639, 1991.

[49] Gail McKoon and Roger Ratcliff. Inference during reading. *Psychological review*, 99(3):440, 1992.

[50] Ben Medler, Joe Fitzgerald, and Brian Magerko. Using conflict theory to model complex societal interactions. In *Proceedings of Future Play*, pages 65–72, 2008.

[51] J. R. Meehan. Tale-spin, an interactive program that writes stories. In *Proceedings of the 5th International Joint Conference on Artificial Intelligence*, pages 91–98, 1977.

[52] Janet Horowitz Murray. *Hamlet on the holodeck: The future of narrative in cyberspace*. Simon and Schuster, 1997.

[53] A. Newell and H. A. Simon. GPS, a program that simulates human thought. *Computers and Thought*, pages 279–293, 1961.

[54] J. M. Niehaus. *Cognitive models of discourse comprehension for narrative generation*. PhD thesis, North Carolina State University, 2009.

[55] F. Peinado and P. Gervás. Evaluation of automatic generation of basic stories. *New Generation Computing*, 24(3):289–302, 2006.

[56] Mark A Peot and David E Smith. Threat-removal strategies for partial-order planning. In *Proceedings of the eleventh national conference on Artificial Intelligence*, volume 93, pages 492–499, 1993.

[57] R. P. É. Ý. Pérez and M. Sharples. MEXICA: A computer model of a cognitive account of creative writing. *Journal of Experimental & Theoretical AI*, 13(2):119–139, 2001.

[58] J. Porteous and M. Cavazza. Controlling narrative generation with planning trajectories: The role of constraints. In *International Conference on Interactive Digital Storytelling*, pages 234–245, 2009.

[59] G. Prince. *A dictionary of narratology*. University of Nebraska Press, 2003.

[60] M. O. Riedl. *Narrative planning: Balancing plot and character*. PhD thesis, North Carolina State University, 2004.

[61] M. O. Riedl and R. M. Young. Story planning as exploratory creativity: Techniques for expanding the narrative search space. *New Generation Computing*, 24(3):303–323, 2006.

[62] M. O. Riedl and R. M. Young. Narrative planning: balancing plot and character. *Journal of Artificial Intelligence Research*, 39:217–268, 2010.

[63] Mark O Riedl and Vadim Bulitko. Interactive narrative: An intelligent systems approach. *AI Magazine*, 34(1), 2013.

[64] David L Roberts and Charles L Isbell. A survey and qualitative analysis of recent advances in drama management. *International Transactions on Systems Science and Applications, Special Issue on Agent Based Systems for Human Learning*, 4(2):61–75, 2008.

[65] M. L. Ryan. *Possible worlds, artificial intelligence, and narrative theory.* Indiana University, 1991.

[66] Roger C Schank. *Tell me a story: Narrative and intelligence.* Northwestern University Press, 1995.

[67] N. M. Sgouros. Dynamic generation, management and resolution of interactive plots. *Artificial Intelligence*, 107(1):29–62, 1999.

[68] T. C. Smith and I. H. Witten. A planning mechanism for generating story text. *Literary and Linguistic Computing*, 2(2):119–126, 1987.

[69] Richard S Sutton, Doina Precup, and Satinder Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112(1):181–211, 1999.

[70] N. Szilas. Interactive drama on computer: beyond linear narrative. In *Association for the Advancement of Artificial Intelligence Fall Symposium on Narrative Intelligence*, volume 144, 1999.

[71] Nicolas Szilas. Idtension: a narrative engine for interactive drama. In *Proceedings of the 1st Conference on Technologies for Interactive Digital Storytelling and Entertainment*, volume 3, pages 187–203, 2003.

[72] J. M. Thomas and R. M. Young. Using task-based modeling to generate scaffolding in narrative-guided exploratory learning environments. In *Proceedings of the 14th International Conference on Artificial Intelligence in Education*, pages 107–114, 2009.

[73] K.W. Thomas and M.D. Dunnette. *Handbook of industrial and organizational psychology.* Rand McNally, 1976.

[74] M. Tierno. *Aristotle's poetics for screenwriters: storytelling secrets from the greatest mind in Western civilization.* Hyperion, 2002.

[75] Tzvetan Todorov. *Introduction to poetics.* University of Minnesota Press, 1981.

[76] Tom Trabasso and Linda L Sperry. Causal relatedness and importance of story events. *Journal of Memory and language*, 24(5):595–611, 1985.

[77] Tom Trabasso and Paul Van Den Broek. Causal thinking and the representation of narrative events. *Journal of memory and language*, 24(5):612–630, 1985.

[78] E. Vale. *The technique of screenplay writing.* Souvenir Press, 1973.

[79] S. G. Ware and R. M. Young. Validating a Plan-Based Model of Narrative Conflict. In *Proceedings of the 7th International Conference on the Foundations of Digital Games*, pages 220–227, 2012.

[80] Stephen G Ware, R Michael Young, Brent Harrison, and David L Roberts. Four quantitative metrics describing narrative conflict. In *Proceedings of the 5th International Conference on Interactive Digital Storytelling*, pages 18–29, 2012.

[81] D. S. Weld. An introduction to least commitment planning. *AI Magazine*, 15(4):27–61, 1994.

[82] Mike Williamson and Steve Hanks. Flaw selection strategies for value-directed planning. In *Proceedings of the 3rd International Conference on Artificial Intelligence Planning Systems*, volume 23, pages 244–252, 1996.

[83] G. N. Yannakakis. How to model and augment player satisfaction: A review. In *1st Workshop on Child, Computer and Interaction*, 2008.

[84] R. M. Young. Notes on the use of plan structures in the creation of interactive plot. In *Proceedings of the Association for the Advancement of Artificial Intelligence Fall Symposium on Narrative Intelligence*, pages 164–167, 1999.

[85] R.M. Young and M.O. Riedl. Integrating plan-based behavior generation with game environments. In *Conference on Advances in Computer Entertainment Technology*, pages 370–370. ACM, 2005.

[86] R.M. Young, M.O. Riedl, M. Branly, A. Jhala, R.J. Martin, and C.J. Saretto. An architecture for integrating plan-based behavior generation with interactive game environments. *Journal of Game Development*, 1(1):51–70, 2004.

[87] F. Zambetta, A. Nash, and P. Smith. Two families: dynamical policy models in interactive storytelling. In *Proceedings of the Australasian Conference on Interactive Entertainment*, pages 1–8. RMIT University, 2007.

Appendix

Appendix A

Planning Domains and Problems

This appendix contains the intentional and conflict planning domains and problems used to evaluate the computational efficiency of the Glaive planner. They are written in the Planning Domain Definition Language, or PDDL.

A.1 Aladdin

```
;;;
;;; A domain for telling the story of Aladdin from 1001 Nights
;;; Created by Mark O. Riedl for his dissertation
;;; Ported to PDDL 3 and modified to use the 'delegated' modality by Stephen G. Ware
;;;
(define (domain aladdin)
  (:requirements :adl :domain-axioms :expression-variables :intentionality :delegation)
  (:types character thing place - object
          male female monster - character
          knight king - male
          genie dragon - monster
          magic-lamp - thing)
  (:predicates (alive ?character - character)
               (scary ?monster - monster)
               (beautiful ?character - character)
               (confined ?character - character)
               (single ?character - character)
               (married ?character - character)
               (at ?character - character ?place - place)
               (in ?genie - genie ?magic-lamp - magic-lamp)
               (has ?character - character ?thing - thing)
               (loyal-to ?knight - knight ?king - king)
               (controls ?character - character ?genie - genie)
               (loves ?lover - character ?love-interest - character)
               (married-to ?character1 - character ?character2 - character))

  ;; A character moves from one location to another.
  (:action go
    :parameters   (?character - character ?from - place ?to - place)
```

```
:precondition (and (not (= ?from ?to))
                        (alive ?character)
                        (at ?character ?from))
:effect       (and (not (at ?character ?from))
                        (at ?character ?to))
    :agents       (?character))

  ;; A character slays a monster.
  (:action slay
    :parameters   (?knight - knight ?monster - monster ?place - place)
    :precondition (and (alive ?knight)
                        (at ?knight ?place)
                        (alive ?monster)
                        (at ?monster ?place))
    :effect       (not (alive ?monster))
    :agents       (?knight))

  ;; One character takes an item from the corpse of another.
  (:action pillage
    :parameters   (?pillager - character ?victim - character ?thing - thing ?place - place)
    :precondition (and (alive ?pillager)
                        (at ?pillager ?place)
                        (not (alive ?victim))
                        (at ?victim ?place)
                        (has ?victim ?thing))
    :effect       (and (not (has ?victim ?thing))
                        (has ?pillager ?thing))
    :agents       (?pillager))

  ;; One character gives an item to another.
  (:action give
    :parameters   (?giver - character ?recipient - character ?thing - thing ?place - place)
    :precondition (and (not (= ?giver ?recipient))
                        (alive ?giver)
                        (at ?giver ?place)
                        (has ?giver ?thing)
                        (alive ?recipient)
                        (at ?recipient ?place))
    :effect       (and (not (has ?giver ?thing))
                        (has ?recipient ?thing))
    :agents       (?giver))

  ;; A character summons a genie from a magic lamp.
  (:action summon
    :parameters   (?character - character ?genie - genie ?lamp - magic-lamp ?place - place)
    :precondition (and (alive ?character)
                        (at ?character ?place)
                        (has ?character ?lamp)
                        (alive ?genie)
                        (in ?genie ?lamp))
    :effect       (and (not (confined ?genie))
                        (not (in ?genie ?lamp))
                        (at ?genie ?place)
                        (controls ?character ?genie))
    :agents       (?character))
```

```
;; A genie causes one character to fall in love with another.
(:action love-spell
  :parameters   (?genie - genie ?target - character ?lover - character)
  :precondition (and (not (= ?target ?lover))
                     (not (= ?genie ?target))
                     (not (= ?genie ?lover))
                     (alive ?genie)
                     (not (confined ?genie))
                     (alive ?target)
                     (alive ?lover)
                     (not (loves ?target ?lover)))
  :effect       (and (loves ?target ?lover)
                     (intends ?target (married-to ?target ?lover)))
  :agents       (?genie))

;; Two characters who are in love get married.
(:action marry
  :parameters   (?groom - male ?bride - female ?place - place)
  :precondition (and (alive ?groom)
                     (at ?groom ?place)
                     (loves ?groom ?bride)
                     (alive ?bride)
                     (at ?bride ?place)
                     (loves ?bride ?groom))
  :effect       (and (not (single ?groom))
                     (married ?groom)
                     (married-to ?groom ?bride)
                     (not (single ?bride))
                     (married ?bride)
                     (married-to ?bride ?groom))
  :agents       (?groom ?bride))

;; One character falls in love with another.
(:action fall-in-love
  :parameters   (?male - male ?female - female ?place - place)
  :precondition (and (alive ?male)
                     (single ?male)
                     (at ?male ?place)
                     (not (loves ?male ?female))
                     (alive ?female)
                     (beautiful ?female)
                     (single ?female)
                     (at ?female ?place))
  :effect       (and (loves ?male ?female)
                     (intends ?male (married-to ?male ?female))))

;; A character delegates a goal to a subordinate.
(:action order
  :parameters   (?king - king ?knight - knight ?place - place ?objective - expression)
  :precondition (and (alive ?king)
                     (at ?king ?place)
                     (alive ?knight)
                     (at ?knight ?place)
                     (loyal-to ?knight ?king))
```

```
    :effect       (and (intends ?knight ?objective)
                       (delegated ?king ?objective ?knight))
    :agents       (?king))

  ;; A character delegates a goal to a genie.
  (:action command
    :parameters   (?character - character ?genie - genie ?lamp - magic-lamp
                   ?objective - expression)
    :precondition (and (not (= ?character ?genie))
                       (alive ?character)
                       (has ?character ?lamp)
                       (controls ?character ?genie)
                       (alive ?genie))
    :effect       (and (intends ?genie ?objective)
                       (delegated ?character ?objective ?genie))
    :agents       (?character))

  ;; A monster appears threatening.
  (:action appear-threatening
    :parameters   (?monster - monster ?character - character ?place - place)
    :precondition (and (not (= ?monster ?character))
                       (scary ?monster)
                       (at ?monster ?place)
                       (at ?character ?place))
    :effect       (intends ?character (not (alive ?monster)))))

;;;
;;; A problem for felling the story of Aladdin from 1001 Nights
;;; Created by Mark O. Riedl for his dissertation
;;; Ported to PDDL 3 by Stephen G. Ware
;;;
(define (problem aladdin)
  (:domain aladdin)
  (:objects hero - knight
            king - king
            jasmine - female
            dragon - dragon
            genie - genie
            castle mountain - place
            lamp - magic-lamp)
  (:init (alive hero)
         (single hero)
         (at hero castle)
         (loyal-to hero king)
         (alive king)
         (single king)
         (at king castle)
         (alive jasmine)
         (beautiful jasmine)
         (single jasmine)
         (at jasmine castle)
         (alive dragon)
         (scary dragon)
         (at dragon mountain)
         (has dragon lamp)
```

```
        (alive genie)
        (scary genie)
        (confined genie)
        (in genie lamp))
  (:goal (and (not (alive genie))
              (married-to king jasmine))))

;;;
;;; Solution discovered by Glaive
;;; 12 nodes visited; 189 nodes generated
;;;
(define (plan aladdin-solution)
  (:problem aladdin)
  (:steps (fall-in-love king jasmine castle)
          (order king hero castle (loves jasmine king))
          (go hero castle mountain)
          (slay hero dragon mountain)
          (pillage hero dragon lamp mountain)
          (summon hero genie lamp mountain)
          (command hero genie lamp (loves jasmine king))
          (love-spell genie jasmine king)
          (marry king jasmine castle)
          (appear-threatening genie hero mountain)
          (slay hero genie mountain)))
```

A.2 Heist

```
;;;
;;; A domain for telling a story about an evil bank robber in the Wild West
;;; Originally created by James Niehaus for his dissertation
;;; Adapted and corrected by Stephen G. Ware
;;;
(define (domain heist)
  (:requirements :adl :intentionality)
  (:types mobile bystanders seller pawn-broker - person
          evil sheriff guard - mobile
          bank bar store alley - place
          big-money mother-lode - money
          gun cuffs small-goods valuable money - thing
  horse - valuable
          poker-game)
  (:predicates (connected ?from - place ?to - place)
               (alley-of ?alley - alley ?place - place)
               (at ?thing - thing ?place - place)
               (at ?person - person ?place - place)
               (at ?poker-game - poker-game ?place - place)
               (has ?person - person ?thing - thing)
               (has ?bank - bank ?money - money)
               (open ?store - store)
               (forsale ?thing - thing ?store - store)
               (hidden ?person - person)
               (drunk ?person - person)
               (sleeping ?person - person)
               (in-cuffs ?person - person ?cuffs - cuffs)
```

```
            (free-with-money ?person - person)
            (friendly ?friend - person ?person - person)
            (blocking ?person - person ?alley - alley)
            (guard-of ?person - person ?place - place)
            (guarded ?place - place)
            (bet-at ?money - money ?poker-game - poker-game)
            (held-up ?person - person ?bank - bank)
            (arrested ?sheriff - sheriff ?person - person))

;; Pick something up.
(:action pick-up
  :parameters   (?person - person ?thing - thing ?place - place)
  :precondition (and (at ?person ?place)
                     (at ?thing ?place))
  :effect       (and (not (at ?thing ?place))
                     (has ?person ?thing))
  :agents       (?person))

;; Pick up and holster a gun.
(:action holster-gun
  :parameters   (?person - person ?gun - gun ?place - place)
  :precondition (and (at ?person ?place)
                     (at ?gun ?place))
  :effect       (and (not (at ?gun ?place))
                     (has ?person ?gun))
  :agents       (?person))

;; Withdraw some money from the bank.
(:action withdraw-money
  :parameters   (?person - person ?bank - bank ?money - money)
  :precondition (and (at ?person ?bank)
                     (has ?bank ?money))
  :effect       (and (not (has ?bank ?money))
                     (has ?person ?money))
  :agents       (?person))

;; Open a store for business.
(:action open
  :parameters   (?person - person ?store - store)
  :precondition (at ?person ?store)
  :effect       (open ?store))

;; Sell some small goods.
(:action sell
  :parameters   (?person - person ?buyer - person ?thing - small-goods ?money - money
                 ?place - place)
  :precondition (and (at ?person ?place)
                     (has ?person ?thing)
                     (at ?buyer ?place)
                     (has ?buyer ?money))
  :effect       (and (not (has ?person ?thing))
                     (has ?person ?money)
                     (not (has ?buyer ?money))
                     (has ?buyer ?thing))
  :agents       (?person))
```

```
;; Buy a dress (or other good) from a store.
(:action buy-dress
  :parameters   (?person - person ?thing - valuable ?store - store ?money - money)
  :precondition (and (open ?store)
                     (forsale ?thing ?store)
                     (at ?person ?store)
                     (has ?person ?money))
  :effect       (and (has ?person ?thing)
                     (not (forsale ?thing ?store))
                     (not (has ?person ?money)))
  :agents       (?person))

;; Fail to buy a dress (or other good) from a store because of no money.
(:action fail-buy-dress
  :parameters   (?person - person ?thing - valuable ?store - store ?money - money)
  :precondition (and (open ?store)
                     (forsale ?thing ?store)
                     (at ?person ?store)
                     (not (has ?person ?money)))
  :agents       (?person))

;; kick someone out of the way.
(:action kick-out-of-way
  :parameters   (?person - evil ?roadblock - person ?alley - alley ?place - place)
  :precondition (and (at ?person ?place)
                     (at ?roadblock ?place)
                     (blocking ?roadblock ?alley))
  :effect       (not (blocking ?roadblock ?alley))
  :agents       (?person))

;; Hatch a plan to rob the bank.
(:action hatch-plan
  :parameters   (?person - evil ?gun - gun ?horse - horse ?bank - bank
                 ?mother-lode - mother-lode)
  :precondition (has ?bank ?mother-lode)
  :effect       (and (not (has ?person ?mother-lode))
                     (intends ?person (has ?person ?gun))
                     (intends ?person (has ?person ?horse))
                     (intends ?person (has ?person ?mother-lode))
                     (intends ?person (free-with-money ?person))))

;; Hide in an alley.
(:action hide-in-dark-alley
  :parameters   (?person - evil ?alley - alley)
  :precondition (at ?person ?alley)
  :effect       (hidden ?person)
  :agents       (?person))

;; Pickpocket.
(:action pickpocket
  :parameters   (?person - evil ?mark - person ?money - money ?place - place
                 ?alley - alley)
  :precondition (and (alley-of ?alley ?place)
                     (at ?person ?alley)
```

```
                    (at ?mark ?place)
                    (hidden ?person)
                    (has ?mark ?money))
  :effect       (and (has ?person ?money)
                    (not (has ?mark ?money))
                    (not (hidden ?person)))
  :agents       (?person))

;; Move.
(:action move-once
  :parameters   (?person - mobile ?from - place ?to - place)
  :precondition (and (connected ?from ?to)
                    (at ?person ?from))
  :effect       (and (at ?person ?to)
                    (not (at ?person ?from)))
  :agents       (?person))

;; Buy drinks for (and get drunk).
(:action buy-drinks-for
  :parameters   (?person - person ?drinker - person ?money - money ?place - bar)
  :precondition (and (at ?person ?place)
                    (at ?drinker ?place)
                    (has ?person ?money))
  :effect       (and (friendly ?drinker ?person)
                    (drunk ?drinker))
  :agents       (?person))

;; Cheat at a poker game (put up some money).
(:action cheat-at-poker
  :parameters   (?person - evil ?poker - poker-game ?money - money ?winnings - money
                 ?place - place)
  :precondition (and (at ?person ?place)
                    (at ?poker ?place)
                    (has ?person ?money)
                    (bet-at ?winnings ?poker))
  :effect       (has ?person ?winnings)
  :agents       (?person))

;; Leave with.
(:action escort-drunk-friend
  :parameters   (?person - person ?friend - person ?from - place ?to - place)
  :precondition (and (connected ?from ?to)
                    (at ?person ?from)
                    (at ?friend ?from)
                    (drunk ?friend)
                    (friendly ?friend ?person))
  :effect       (and (not (at ?person ?from))
                    (at ?person ?to)
                    (not (at ?friend ?from))
                    (at ?friend ?to))
  :agents       (?person))

;; Lay to rest in alley.
(:action lay-to-rest-in-alley
  :parameters   (?person - person ?friend - guard ?place - alley ?bank - bank)
```

```
  :precondition (and (at ?person ?place)
                     (at ?friend ?place)
                     (drunk ?friend)
                     (friendly ?friend ?person)
                     (guard-of ?friend ?bank))
  :effect       (and (sleeping ?friend)
                     (not (guarded ?bank)))
  :agents       (?person))

;; Take item off sleeping person,
(:action take-thing-off-sleeper
  :parameters   (?person - person ?sleeper - person ?thing - thing ?place - alley)
  :precondition (and (at ?person ?place)
                     (at ?sleeper ?place)
                     (sleeping ?sleeper)
                     (has ?sleeper ?thing))
  :effect       (and (has ?person ?thing)
                     (not (has ?sleeper ?thing)))
  :agents       (?person))

;; Pawn a valuable for money.
(:action pawn-valuable
  :parameters   (?person - person ?pawn-broker - pawn-broker ?thing - valuable
                 ?place - place ?big-money - big-money)
  :precondition (and (at ?person ?place)
                     (at ?pawn-broker ?place)
                     (has ?person ?thing)
                     (has ?pawn-broker ?big-money))
  :effect       (and (not (has ?person ?thing))
                     (has ?pawn-broker ?thing)
                     (has ?person ?big-money))
  :agents       (?person))

;; Buy a horse.
(:action buy-valuable
  :parameters   (?person - person ?seller - seller ?thing - valuable ?place - place
                 ?big-money - big-money)
  :precondition (and (has ?seller ?thing)
                     (at ?person ?place)
                     (at ?seller ?place)
                     (has ?person ?big-money))
  :effect       (and (has ?person ?thing)
                     (not (has ?seller ?thing))
                     (not (has ?person ?big-money)))
  :agents       (?person))

;; Ride a horse to a location.
(:action ride-horse-to
  :parameters   (?person - mobile ?horse - horse ?from - place ?to - place)
  :precondition (and (connected ?from ?to)
                     (at ?person ?from)
                     (at ?horse ?from)
                     (has ?person ?horse))
  :effect       (and (at ?person ?to)
                     (not (at ?person ?from))
```

```
                    (at ?horse ?to)
                    (not (at ?horse ?from)))
    :agents       (?person))

  ;; Hold up a bank.
  (:action hold-up-bank
    :parameters   (?person - evil ?gun - gun ?bank - bank ?sheriff - sheriff)
    :precondition (and (at ?person ?bank)
                    (has ?person ?gun)
                    (not (guarded ?bank)))
    :effect       (and (held-up ?person ?bank)
                    (intends ?sheriff (arrested ?sheriff ?person)))
    :agents       (?person))

  ;; Collect money.
  (:action collect-money-from-heist
    :parameters   (?person - evil ?bank - bank ?mother-lode - mother-lode)
    :precondition (and (at ?person ?bank)
                    (held-up ?person ?bank)
                    (has ?bank ?mother-lode))
    :effect       (and (has ?person ?mother-lode)
                    (not (held-up ?person ?bank))
                    (not (has ?bank ?mother-lode)))
    :agents       (?person))

  ;; Getaway with stolen money.
  (:action getaway-with-money
    :parameters   (?person - evil ?mother-lode - mother-lode ?horse - horse
                   ?place - place ?dest - place)
    :precondition (and (connected ?place ?dest)
                    (at ?person ?place)
                    (at ?horse ?place)
                    (has ?person ?mother-lode))
    :effect       (and (not (at ?person ?place))
                    (not (at ?horse ?place))
                    (free-with-money ?person)
                    (at ?person ?dest))
    :agents       (?person))

  ;; Arrest.
  (:action arrest
    :parameters   (?criminal - evil ?sheriff - sheriff ?place - place
                   ?cuffs - cuffs ?money - money)
    :precondition (and (at ?sheriff ?place)
                    (at ?criminal ?place)
                    (has ?sheriff ?cuffs)
                    (has ?criminal ?money))
    :effect       (and (arrested ?sheriff ?criminal)
                    (in-cuffs ?criminal ?cuffs)
                    (has ?sheriff ?money)
                    (not (has ?criminal ?money)))
    :agents       (?sheriff)))

;;;
;;; A problem for telling a story about an evil bank robber in the Wild West
```

```
;;; Originally created by James Niehaus for his dissertation
;;; Adapted and corrected by Stephen G. Ware
;;;
(define (problem heist)
  (:domain heist)
  (:objects ;; People
            robbie - evil
            tom - sheriff
            sally - mobile
            barney - guard
            horse-seller - seller
            pawn-broker - pawn-broker
            jill - person
            anne - person
            child - person
            ;; Places
            bank - bank
            main-street - place
            saloon - bar
            dress-shop - place
            sheriffs-office - place
            sallys-home - place
            dark-alley - alley
            barber-shop - place
            barneys-room - place
            general-store - store
            out-of-town - place
            ;; Things
            mother-lode - mother-lode
            six-shooter - gun
            dress-money - money
            tomato-money - money
            poker-money - money
            locket-money - big-money
            brown-horse - horse
            white-horse - horse
            locket - valuable
            handcuffs - cuffs
            tomatoes - small-goods
            blue-dress - valuable
            poker-game - poker-game)
  (:init ;; Map
         (connected bank main-street)
         (connected main-street bank)
         (connected saloon main-street)
         (connected main-street saloon)
         (connected dress-shop main-street)
         (connected main-street dress-shop)
         (connected sheriffs-office main-street)
         (connected main-street sheriffs-office)
         (connected sallys-home main-street)
         (connected main-street sallys-home)
         (connected dark-alley main-street)
         (connected main-street dark-alley)
         (alley-of dark-alley main-street)
```

```
        (connected barber-shop main-street)
        (connected main-street barber-shop)
        (connected sheriffs-office barber-shop)
        (connected barber-shop sheriffs-office)
        (connected barneys-room saloon)
        (connected general-store main-street)
        (connected main-street general-store)
        (connected barber-shop out-of-town)
        (connected bank out-of-town)
        ;; Where things are.
        (has bank mother-lode)
        (has bank dress-money)
        (has barney six-shooter)
        (has horse-seller brown-horse)
        (has tom white-horse)
        (at white-horse barber-shop)
        (has pawn-broker locket-money)
        (has robbie locket)
        (at handcuffs sheriffs-office)
        (has sally tomatoes)
        (has anne tomato-money)
        (forsale blue-dress general-store)
        (at poker-game saloon)
        (bet-at poker-money poker-game)
        ;; Locations
        (at robbie main-street)
        (at sally main-street)
        (at tom sheriffs-office)
        (at barney barneys-room)
        (at horse-seller main-street)
        (at brown-horse main-street)
        (at pawn-broker main-street)
        (at jill general-store)
        (at anne main-street)
        (at child main-street)
        ;; Misc.
        (guard-of barney bank)
        (blocking child dark-alley)
        ;; Intentions
        (intends robbie (has robbie poker-money))
        (intends sally (has sally blue-dress))
        (intends barney (at barney saloon)))
  (:goal (arrested tom robbie)))

;;;
;;; Example solution adapted from James Niehaus's dissertation
;;;
(define (plan heist-solution)
  (:problem heist)
  (:steps (hatch-plan robbie six-shooter brown-horse bank mother-lode)
          (open jill general-store)
          (move-once sally main-street bank)
          (withdraw-money sally bank dress-money)
          (move-once sally bank main-street)
          (sell sally anne tomatoes tomato-money main-street)
```

```
        (move-once robbie main-street dark-alley)
        (hide-in-dark-alley robbie dark-alley)
        (move-once barney barneys-room saloon)
        (pickpocket robbie sally dress-money main-street dark-alley)
        (move-once robbie dark-alley main-street)
        (move-once robbie main-street saloon)
        (buy-drinks-for robbie barney dress-money saloon)
        (escort-drunk-friend robbie barney saloon main-street)
        (escort-drunk-friend robbie barney main-street dark-alley)
        (lay-to-rest-in-alley robbie barney dark-alley bank)
        (take-thing-off-sleeper robbie barney six-shooter dark-alley)
        (move-once robbie dark-alley main-street)
        (move-once robbie main-street saloon)
        (cheat-at-poker robbie poker-game dress-money poker-money saloon)
        (move-once robbie saloon main-street)
        (pawn-valuable robbie pawn-broker locket main-street locket-money)
        (buy-valuable robbie horse-seller brown-horse main-street locket-money)
        (ride-horse-to robbie brown-horse main-street bank)
        (hold-up-bank robbie six-shooter bank tom)
        (collect-money-from-heist robbie bank mother-lode)
        (getaway-with-money robbie mother-lode brown-horse bank out-of-town)
        (pick-up tom handcuffs sheriffs-office)
        (move-once tom sheriffs-office barber-shop)
        (ride-horse-to tom white-horse barber-shop out-of-town)
        (arrest robbie tom out-of-town handcuffs mother-lode)))
```

A.3 Western

```
;;;
;;; A domain for modeling stories in the Wild West
;;; Created by Stephen G. Ware
;;; Originally used for validating CPOCL narrative structure
;;;
(define (domain western)
  (:requirements :adl :intentionality)
  (:types ; People and animals are living things.
          person animal - living
          ; Animals are items that can be owned.
          animal - item
          ; Some items are valuable.
          valuable - item
          ; Places exist.
          place
          ; Sicknesses exist
          sickness)
  (:constants ; A place to imprison criminals
              jailhouse - place
              ; The "sickness" of being bitten by a poisonous snake
              snakebite - sickness)
  (:predicates ; A person is alive.
               (alive ?person - person)
               ; A person is not restrained.
               (free ?person - person)
               ; A person is the sheriff.
```

```
            (sheriff ?person - person)
            ; A person or thing is at a place.
            (at ?object - object ?place - place)
            ; An item belongs to a person.
            (belongsto ?item - item ?person - person)
            ; A person has an item.
            (has ?person - person ?item - item)
            ; A person is sick with some kind of sickness.
            (sick ?person - person ?sickness - sickness)
            ; An item can cure a sickness.
            (cures ?item - item ?sickness - sickness)
            ; One person loves another.
            (loves ?lover - person ?love - person))

; A character gets bitten by a rattlesnake and becomes sick.
(:action snakebite
  :parameters   (?victim - person)
  :precondition (alive ?victim)
  :effect       (and (sick ?victim snakebite)
                     (intends ?victim (not (sick ?victim snakebite)))
                     (forall (?p - person)
                             (when (loves ?p ?victim)
                                   (intends ?p (not (sick ?victim snakebite)))))))

; A character dies of dies of some sickness.
(:action die
  :parameters   (?person - person ?sickness - sickness)
  :precondition (and (alive ?person)
                     (sick ?person ?sickness))
  :effect       (not (alive ?person)))

; A character travels from one location to another.
(:action travel
  :parameters   (?person - person ?from - place ?to - place)
  :precondition (and (alive ?person)
                     (free ?person)
                     (at ?person ?from))
  :effect       (and (at ?person ?to)
                     (not (at ?person ?from)))
  :agents       (?person))

; A character forces a tied up character to move from one place to another.
(:action forcetravel
  :parameters   (?person - person ?victim - person ?from - place ?to - place)
  :precondition (and (alive ?person)
                     (free ?person)
                     (at ?person ?from)
                     (alive ?victim)
                     (not (free ?victim))
                     (at ?victim ?from))
  :effect       (and (at ?person ?to)
                     (not (at ?person ?from))
                     (at ?victim ?to)
                     (not (at ?victim ?from)))
  :agents       (?person))
```

```
; One character gives an item to another.
(:action give
  :parameters   (?giver - person ?receiver - person ?item - item ?place - place)
  :consent      (?giver ?receiver)
  :precondition (and (alive ?giver)
                     (free ?giver)
                     (at ?giver ?place)
                     (has ?giver ?item)
                     (alive ?receiver)
                     (free ?receiver)
                     (at ?receiver ?place))
  :effect       (and (has ?receiver ?item)
                     (not (has ?giver ?item))
                     (when (belongsto ?item ?giver)
                           (belongsto ?item ?receiver)))
  :agents       (?giver))

; One character ties up another.
(:action tieup
  :parameters   (?person - person ?victim - person ?place - place)
  :precondition (and (alive ?person)
                     (free ?person)
                     (at ?person ?place)
                     (alive ?victim)
                     (at ?victim ?place))
  :effect       (and (not (free ?victim))
                     (intends ?victim (free ?victim)))
  :agents       (?person))

; One character unties another.
(:action untie
  :parameters   (?person - person ?victim - person ?place - place)
  :precondition (and (alive ?person)
                     (free ?person)
                     (at ?person ?place)
                     (alive ?victim)
                     (not (free ?victim))
                     (at ?victim ?place))
  :effect       (free ?victim)
  :agents       (?person))

; One character takes an item from a tied up character.
(:action take
  :parameters   (?taker - person ?item - item ?victim - person ?place - place)
  :precondition (and (not (= ?taker ?victim))
                     (alive ?taker)
                     (free ?taker)
                     (at ?taker ?place)
                     (alive ?victim)
                     (not (free ?victim))
                     (at ?victim ?place)
                     (has ?victim ?item))
  :effect       (and (has ?taker ?item)
                     (not (has ?victim ?item))
```

```
                    (when (belongsto ?item ?victim)
                          (and (intends ?victim (has ?victim ?item))
                               (forall (?s - person)
                                       (when (sheriff ?s)
                                             (intends ?s (and (at ?taker jailhouse)
                                                              (not (free ?taker))
                                                              (has ?victim ?item)
                                                              (free ?victim))))))))
    :agents       (?taker))

  ; One character uses medicine to heal a sick character.
  (:action heal
    :parameters   (?healer - person ?patient - person ?sickness - sickness
                   ?medicine - item ?place - place)
    :precondition (and (cures ?medicine ?sickness)
                       (alive ?healer)
                       (free ?healer)
                       (at ?healer ?place)
                       (has ?healer ?medicine)
                       (alive ?patient)
                       (at ?patient ?place)
                       (sick ?patient ?sickness))
    :effect       (and (not (sick ?patient ?sickness))
                       (not (has ?healer ?medicine)))
    :agents       (?healer ?patient)))

;;;
;;; Rancher Hank responds to his son being bitten by a snake.
;;; Created by Stephen G. Ware
;;; Originally used for validating CPOCL narrative structure
;;;
(define (problem western)
  (:domain western)
  ; All the people, places, and things in the problem
  (:objects
     ; A saloon
     saloon - place
     ; Hank's ranch
     ranch - place
     ; The town general store
     generalstore - place
     ; Hank, a cattle rancher
     hank - person
     ; Timmy, Hank's son.
     timmy - person
     ; Will, the sheriff
     will - person
     ; Carl, the shopkeeper.
     carl - person
     ; Antivenom to cure a snakebite
     antivenom - item)
  ; Initial state of the world
  (:init
     ; Hank lives on his ranch and loves his son.
     (alive hank)
```

```
    (free hank)
    (at hank ranch)
    (loves hank timmy)
    ; Timmy is Hank's son, and also lives at the ranch.
    (alive timmy)
    (free timmy)
    (at timmy ranch)
    (loves timmy hank)
    ; Will is the sheriff.  He is in the saloon, plotting the downfall of lawbreakers.
    (alive will)
    (free will)
    (at will saloon)
    ; Carl is the manager of the town general store.
    (alive carl)
    (free carl)
    (at carl generalstore)
    (has carl antivenom)
    ; Antivenom cures a snakebite.
    (cures antivenom snakebite))
  ; Goal state: Timmy is dead and Hank is tied up.
  (:goal (and (not (alive timmy))
              (not (free hank)))))

;;;
;;; Solution discovered by Glaive
;;; 18,855 nodes visited; 296,150 nodes generated
;;;
(define (plan western-solution)
  (:problem western)
  (:steps (snakebite timmy)
          (tieup timmy hank ranch)
          (non-executed (forcetravel timmy hank ranch generalstore))
          (die timmy snakebite)
          (snakebite hank)
          (non-executed (tieup timmy carl generalstore))
          (non-executed (take timmy antivenom carl generalstore))
          (non-executed (heal timmy timmy snakebite antivenom generalstore))))
```

A.4 Fantasy

```
;;;
;;; A domain for modeling stories in a magical kingdom
;;; Created by Stephen G. Ware
;;; Originally used for validating CPOCL narrative structure
;;;
(define (domain fantasy)
  (:requirements :adl :intentionality)
  (:types ; Person and monster are a types of creature.
          person monster - creature
          ; Items exist.
          valuable - item
          ; Places exist.
          place)
  (:predicates ; A creature is alive.
```

```
            (alive ?creature - creature)
            ; A person is single.
            (single ?person - person)
            ; A creature is rich.
            (rich ?creature - creature)
            ; A creature is happy.
            (happy ?creature - creature)
            ; A creature is hungry.
            (hungry ?creature - creature)
            ; An object is at a place.
            (at ?object - object ?place - place)
            ; A creature has an item.
            (has ?creature - creature ?item - item)
            ; An item belongs to a creature
            (belongsto ?item - item ?creature - creature)
            ; One creature loves another.
            (loves ?lover - creature ?love - creature)
            ; One person has proposed to another.
            (hasproposed ?proposer - person ?proposee - person)
            ; One person has accepted another's proposal.
            (hasaccepted ?person1 - person ?person2 - person)
            ; Two people are married.
            (marriedto ?person1 - person ?person2 - person))

;; A creature travels from one place to another.
(:action travel
  :parameters   (?creature - creature ?from - place ?to - place)
  :precondition (and (alive ?creature)
                     (at ?creature ?from))
  :effect       (and (at ?creature ?to)
                     (not (at ?creature ?from)))
  :agents       (?creature))

;; One person proposes to another.
(:action propose
  :parameters   (?proposer - person ?proposee - person ?place - place)
  :precondition (and (alive ?proposer)
                   (at ?proposer ?place)
                     (alive ?proposee)
                     (at ?proposee ?place)
                     (loves ?proposer ?proposee))
  :effect       (hasproposed ?proposer ?proposee)
  :agents       (?proposer))

;; One person accepts another's proposal.
(:action accept
  :parameters   (?accepter - person ?proposer - person ?place - place)
  :precondition (and (alive ?accepter)
                   (at ?accepter ?place)
                     (alive ?proposer)
                     (at ?proposer ?place)
                     (hasproposed ?proposer ?accepter))
  :effect       (hasaccepted ?accepter ?proposer)
  :agents       (?accepter))
```

```
;; Two people marry.
(:action marry
  :parameters   (?groom - person ?bride - person ?place - place)
  :precondition (and (alive ?groom)
                     (at ?groom ?place)
                     (hasproposed ?groom ?bride)
                     (single ?groom)
                     (alive ?bride)
                     (at ?bride ?place)
                     (hasaccepted ?bride ?groom)
                     (single ?bride))
  :effect       (and (marriedto ?groom ?bride)
                     (marriedto ?bride ?groom)
                     (not (single ?groom))
                     (not (single ?bride))
                     (forall (?v - valuable)
                             (when (has ?groom ?v)
                                   (rich ?bride)))
                     (when (loves ?groom ?bride)
                           (happy ?groom))
                     (when (loves ?bride ?groom)
                           (happy ?bride)))
  :agents       (?groom ?bride))

;; A creature steals an object from another creature.
(:action steal
  :parameters   (?thief - creature ?victim - creature ?item - item ?place - place)
  :precondition (and (not (= ?thief ?victim))
                     (alive ?thief)
                     (at ?thief ?place)
                     (at ?item ?place)
                     (belongsto ?item ?victim))
  :effect       (and (has ?thief ?item)
                     (when (at ?victim ?place)
                     (intends ?victim (has ?victim ?item)))
                     (when (forall (?v - valuable)
                                   (not (has ?victim ?v)))
                     (not (rich ?victim))))
  :agents       (?thief))

;; A creature becomes hungry.
(:action get-hungry
  :parameters   (?creature - creature)
  :precondition (not (hungry ?creature))
  :effect       (and (hungry ?creature)
                     (intends ?creature (not (hungry ?creature))))
  :agents       (?creature))

;; A monster eats another creature.
(:action eat
  :parameters   (?monster - monster ?creature - creature ?place - place)
  :precondition (and (alive ?monster)
                     (at ?monster ?place)
                     (hungry ?monster)
                     (alive ?creature)
```

```
                    (at ?creature ?place))
    :effect       (and (not (hungry ?monster))
                    (not (alive ?creature))
                    (not (rich ?creature))
                    (not (happy ?creature)))
    :agents       (?monster)))

;;;
;;; A fair maiden is faced with two marriage proposals.
;;; Created by Stephen G. Ware
;;; Originally used for validating CPOCL narrative structure
;;;
(define (problem fantasy)
  (:domain fantasy)
  (:objects ;; People
            talia - person
            rory - person
            vince - person
            gargax - monster
            ;; Places
            village - place
            cave - place
            ;; Things
            money - valuable
            treasure - valuable)
  (:init (alive talia)
         (at talia village)
         (single talia)
         (loves talia rory)
         (alive vince)
         (at vince village)
         (has vince money)
         (rich vince)
         (single vince)
         (loves vince talia)
         (alive rory)
         (at rory village)
         (single rory)
         (loves rory talia)
         (alive gargax)
         (at gargax cave)
         (at treasure cave)
         (belongsto treasure gargax)
         (rich gargax)
         (intends talia (alive talia))
         (intends talia (rich talia))
         (intends talia (happy talia))
         (intends vince (alive vince))
         (intends vince (rich vince))
         (intends vince (happy vince))
         (intends rory (alive rory))
         (intends rory (happy rory))
         (intends gargax (alive gargax))
         (intends gargax (rich gargax)))
  (:goal (and (happy talia)
```

```
              (rich talia)
              (alive vince))))

;;;
;;; Solution discovered by Glaive
;;; 14 nodes visited; 107 nodes generated
;;;
(define (plan fantasy-solution)
  (:problem fantasy)
  (:steps (propose rory talia village)
          (accept talia rory village)
          (travel rory village cave)
          (steal rory gargax treasure cave)
          (travel rory cave village)
          (marry rory talia village)))
```

A.5 Space

```
;;;
;;; A domain for modeling encounters with extraterrestrial guardians of volatile planets
;;; Created by Stephen G. Ware
;;; Originally used for validating CPOCL narrative structure
;;;
(define (domain space)
  (:requirements :adl :intentionality)
  (:types ; A creature is any living thing
          creature
          ; There are two types of places: landforms and ships.
          landform ship - place)
  (:predicates ; A creature is alive.
               (alive ?creature - creature)
               ; A creature is stunned.
               (stunned ?creature - creature)
               ; A place is habitable.
               (habitable ?place - place)
               ; A place is safe.
               (safe ?place - place)
               ; A creature is safe.
               (safe ?creature - creature)
               ; A landform is erupting lava.
               (erupting ?landform - place)
               ; An creature is at a place.
               (at ?creature - creature ?place - place)
               ; Two creatures are fighting.
               (fighting ?creature1 - creature ?creature2 - creature)
               ; Two creatures are friends.
               (friends ?creature1 - creature ?creature2 - creature)
               ; A creature is captain of a ship
               (captain ?creature - creature ?ship - ship)
               ; A creature is a guardian of a place.
               (guardian ?creature - creature ?place - place))

  ;; A creature walks from one landform to another.
  (:action walk
```

```
  :parameters   (?creature - creature ?from - landform ?to - landform)
  :precondition (and (not (= ?from ?to))
                     (alive ?creature)
                     (not (stunned ?creature))
                     (at ?creature ?from)
                     (habitable ?to)
                     (safe ?to)
                     (forall (?c - creature)
                             (and (not (fighting ?creature ?c))
                                  (not (fighting ?c ?creature)))))
  :effect       (and (at ?creature ?to)
                     (not (at ?creature ?from))
                     (when (not (safe ?creature))
                           (safe ?creature)))
  :agents       (?creature))

;; A creature teleports from a ship to a place.
(:action teleport-from-ship
  :parameters   (?creature - creature ?from - ship ?to - place)
  :precondition (and (not (= ?from ?to))
                     (alive ?creature)
                     (not (stunned ?creature))
                     (at ?creature ?from)
                     (habitable ?to)
                     (safe ?to)
                     (captain ?creature ?from))
  :effect       (and (at ?creature ?to)
                     (not (at ?creature ?from))
                     (when (not (safe ?creature))
                           (safe ?creature))
                     (forall (?c - creature)
                             (and (not (fighting ?creature ?c))
                                  (not (fighting ?c ?creature))
                                  (when (guardian ?c ?to)
                                        (intends ?c (not (alive ?creature)))))))
  :agents       (?creature))

;; A creature teleports from a place to a ship.
(:action teleport-to-ship
  :parameters   (?creature - creature ?from - place ?to - ship)
  :precondition (and (not (= ?from ?to))
                     (alive ?creature)
                     (not (stunned ?creature))
                     (at ?creature ?from)
                     (habitable ?to)
                     (safe ?to)
                     (captain ?creature ?to))
  :effect       (and (at ?creature ?to)
                     (not (at ?creature ?from))
                     (when (not (safe ?creature))
                           (safe ?creature))
                     (forall (?c - creature)
                             (and (not (fighting ?creature ?c))
                                  (not (fighting ?c ?creature)))))
  :agents       (?creature))
```

```
;; One creature starts a fight with another.
(:action attack
  :parameters   (?attacker - creature ?victim - creature ?place - place)
  :precondition (and (alive ?attacker)
                     (not (stunned ?attacker))
                     (at ?attacker ?place)
                     (alive ?victim)
                     (not (stunned ?victim))
                     (at ?victim ?place))
  :effect       (and (fighting ?attacker ?victim)
                     (intends ?victim (not (fighting ?attacker ?victim))))
  :agents       (?attacker))

;; One creatures kills another to end a fight.
(:action kill
  :parameters   (?killer - creature ?victim - creature)
  :precondition (and (alive ?killer)
                     (not (stunned ?killer))
                     (alive ?victim)
                     (or (fighting ?killer ?victim)
                         (fighting ?victim ?killer)))
  :effect       (and (not (alive ?victim))
                     (when (fighting ?killer ?victim)
                           (not (fighting ?killer ?victim)))
                     (when (fighting ?victim ?killer)
                           (not (fighting ?victim ?killer))))
  :agents       (?killer))

;; One creatures stuns another to end a fight.
(:action stun
  :parameters   (?stunner - creature ?victim - creature)
  :precondition (and (alive ?stunner)
                     (not (stunned ?stunner))
                     (alive ?victim)
                     (not (stunned ?victim))
                     (or (fighting ?stunner ?victim)
                         (fighting ?victim ?stunner)))
  :effect       (and (stunned ?victim)
                     (when (fighting ?stunner ?victim)
                           (not (fighting ?stunner ?victim)))
                     (when (fighting ?victim ?stunner)
                           (not (fighting ?victim ?stunner))))
  :agents       (?stunner))

;; A stunned creature breaks free.
(:action break-free
  :parameters   (?victim - creature)
  :precondition (and (alive ?victim)
                     (stunned ?victim))
  :effect       (not (stunned ?victim))
  :agents       (?victim))

;; One creature makes peace with another.
(:action make-peace
  :parameters   (?peacemaker - creature ?creature - creature ?place - place)
```

```
    :precondition (and (alive ?peacemaker)
                       (not (stunned ?peacemaker))
                       (at ?peacemaker ?place)
                       (alive ?creature)
                       (at ?creature ?place)
                       (not (fighting ?peacemaker ?creature))
                       (not (fighting ?creature ?peacemaker)))
    :effect       (and (friends ?peacemaker ?creature)
                       (friends ?creature ?peacemaker))
    :agents       (?peacemaker))

  ;; A volcano begins to errupt.
  (:action begin-erupt
    :parameters   (?landform - landform)
    :effect       (and (erupting ?landform)
                       (forall (?c - creature)
                               (when (at ?c ?landform)
                                     (and (not (safe ?c))
                                          (intends ?c (safe ?c)))))))

  ;; A volcano errupts.
  (:action erupt
    :parameters   (?landform - landform)
    :precondition (erupting ?landform)
    :effect       (and (not (habitable ?landform))
                       (not (erupting ?landform))
                       (forall (?c - creature)
                               (when (at ?c ?landform)
                                     (not (alive ?c)))))))

;;;
;;; An intergalactic explorer encounters environmental and sentient hazards
;;; Created by Stephen G. Ware
;;;
(define (problem space)
  (:domain space)
  (:objects ;; People
            zoe - creature
            lizard - creature
            ;; Places
            ship - ship
            cave - place
            surface - landform)
  (:init (habitable ship)
         (safe ship)
         (habitable surface)
         (safe surface)
         (habitable cave)
         (safe cave)
         (alive zoe)
         (safe zoe)
         (at zoe ship)
         (captain zoe ship)
         (alive lizard)
         (safe lizard)
```

```
         (at lizard cave)
         (guardian lizard surface)
         (intends zoe (friends zoe lizard))
         (intends zoe (safe zoe))
         (intends zoe (alive zoe))
         (intends lizard (safe lizard))
         (intends lizard (alive lizard)))
  (:goal (not (habitable surface))))

;;;
;;; Solution discovered by Glaive
;;; 3 nodes visited; 9 nodes generated
;;;
(define (plan space-solution)
  (:problem space)
  (:steps (begin-erupt surface)
          (erupt surface)))
```

A.6 Raiders

```
;;;
;;; A highly simplified version of the actions in Indiana Jones and the Raiders of the Lost
;;; Ark
;;; Created by Stephen G. Ware
;;; Originally used to test the Glaive Narrative Planner
;;;
(define (domain raiders)
  (:requirements :adl :domain-axioms :intentionality)
  (:types character place - object
          weapon - item)
  (:constants ark - item)
  (:predicates (open ark)
               (alive ?character - character)
               (armed ?character - character)
               (burried ?item - item ?place - place)
               (knows-location ?character - character ?item - item ?place - place)
               (at ?character - character ?place - place)
               (has ?character - character ?item - item))

  ;; A character travels from one place to another.
  (:action travel
    :parameters   (?character - character ?from - place ?to - place)
:precondition (and (not (= ?from ?to))
                       (alive ?character)
                       (at ?character ?from))
:effect       (and (not (at ?character ?from))
                       (at ?character ?to))
    :agents       (?character))

  ;; A character excavates an item.
  (:action excavate
    :parameters   (?character - character ?item - item ?place - place)
:precondition (and (alive ?character)
                       (at ?character ?place)
```

```
                      (burried ?item ?place)
                      (knows-location ?character ?item ?place))
:effect       (and (not (burried ?item ?place))
                      (has ?character ?item))
    :agents       (?character))

  ;; One character gives an item to another.
  (:action give
    :parameters   (?giver - character ?item - item ?receiver - character ?place - place)
:precondition (and (not (= ?giver ?receiver))
                      (alive ?giver)
                      (at ?giver ?place)
                      (has ?giver ?item)
                      (alive ?receiver)
                      (at ?receiver ?place))
:effect       (and (not (has ?giver ?item))
                      (has ?receiver ?item))
    :agents       (?giver ?receiver))

  ;; One character kills another.
  (:action kill
    :parameters   (?killer - character ?weapon - weapon ?victim - character ?place - place)
    :precondition (and (alive ?killer)
                      (at ?killer ?place)
                      (has ?killer ?weapon)
                      (alive ?victim)
                      (at ?victim ?place))
    :effect       (not (alive ?victim))
    :agents       (?killer))

  ;; One character takes an item from another at weapon-point.
  (:action take
    :parameters   (?taker - character ?item - item ?victim - character ?place - place)
:precondition (and (not (= ?taker ?victim))
                      (alive ?taker)
                      (at ?taker ?place)
                      (or (not (alive ?victim))
                          (and (armed ?taker)
                               (not (armed ?victim))))
                      (at ?victim ?place)
                      (has ?victim ?item))
:effect       (and (not (has ?victim ?item))
                      (has ?taker ?item))
    :agents       (?taker))

  ;; A character opens the Ark.
  (:action open-ark
    :parameters   (?character - character)
:precondition (and (alive ?character)
                      (has ?character ark))
:effect       (and (open ark)
                      (not (alive ?character)))
    :agents       (?character))

  ;; The Ark closes.
```

```
  (:action close-ark
:precondition (open ark)
:effect       (not (open ark)))

  ;; When a character has a weapon, they are armed.
  (:axiom
    :vars    (?character - character)
    :context (and (not (armed ?character))
                  (exists (?w - weapon)
                          (has ?character ?w)))
    :implies (armed ?character))

  ;; When a character does not have a weapon, they are unarmed.
  (:axiom
    :vars    (?character - character)
    :context (and (armed ?character)
                  (forall (?w - weapon)
                          (not (has ?character ?w))))
    :implies (not (armed ?character))))

;;;
;;; A highly simplified problem for Indiana Jones and the Raiders of the Lost Ark
;;; Created by Stephen G. Ware
;;;
(define (problem raiders)
  (:domain indiana-jones-ark)
  (:objects indiana nazis army - character
            usa tanis - place
            gun - weapon)
  (:init (burried ark tanis)
         (alive indiana)
         (at indiana usa)
         (knows-location indiana ark tanis)
         (intends indiana (alive indiana))
         (intends indiana (has army ark))
         (alive army)
         (at army usa)
         (intends army (alive army))
         (intends army (has army ark))
         (alive nazis)
         (at nazis tanis)
         (intends nazis (alive nazis))
         (intends nazis (open ark))
         (has nazis gun))
  (:goal (and (at army usa)
              (has army ark)
              (not (alive nazis)))))

;;;
;;; Solution discovered by Glaive
;;; 35 nodes visited; 142 nodes expanded
;;;
(define (plan raiders-solution)
  (:problem raiders)
  (:steps (travel indiana usa tanis)
```

```
         (excavate indiana ark tanis)
         (travel indiana tanis usa)
         (non-executed (give indiana ark army usa))
         (travel nazis tanis usa)
         (take nazis ark indiana usa)
         (open-ark nazis)
         (take army ark nazis usa)))
```

A.7 The Best Laid Plans

```
;;;
;;; All the actions that characters can take in the interactive narrative adventure
;;; game The Best Laid Plans
;;; Created by Stephen G. Ware
;;; Originally used for the automatic planning and replanning of stories in The Best
;;; Laid Plans
;;;
(define (domain best-laid-plans)
  (:requirements :adl :domain-axioms :intentionality)
  (:types humanoid - character
          predator - character
          protagonist merchant guard - humanoid
          location - object
          money weapon light food poison spell - item
          light_spell alchemy_spell love_spell kill_spell teleport_spell - spell)
  (:predicates (alive ?character - character)
               (armed ?character - character)
               (dark ?location - location)
               (poisoned ?food - food)
               (path ?from - location ?to - location)
               (at ?character - character ?location - location)
               (at ?item - item ?location - location)
               (knows-location ?character - character ?location - location)
               (has ?character - character ?item - item)
               (buying ?merchant - merchant ?item - item)
               (selling ?merchant - merchant ?item - item)
               (summons ?light_spell - light_spell ?light - light)
               (summons ?alchemy_spell - alchemy_spell ?money - money)
               (leads_to ?teleport_spell - teleport_spell ?location - location)
               (hungry ?character)
               (citizen ?character - character)
               (guard ?character - character)
               (criminal ?character - character)
               (owned ?item - item)
               (owns ?character - humanoid ?item - item))

  ;; A character walks from one place to another.
  (:action walk
    :parameters   (?character - character ?from - location ?to - location)
    :precondition (and (not (= ?from ?to))
                       (alive ?character)
                       (at ?character ?from)
                       (path ?from ?to)
                       (knows-location ?character ?to)
```

```
                    (or (not (dark ?to))
                        (exists (?l - light)
                                (has ?character ?l))))
  :effect       (and (not (at ?character ?from))
                    (at ?character ?to))
  :agents       (?character))

;; One character attacks another.
(:action attack
  :parameters   (?character - character ?target - character ?location - location)
  :precondition (and (not (= ?character ?target))
                    (alive ?character)
                    (at ?character ?location)
                    (alive ?target)
                    (at ?target ?location)
                    (or (not (guard ?character))
                        (criminal ?target)))
  :effect       (and (when (and (armed ?character)
                                (armed ?target))
                           (not (alive ?target)))
                    (when (and (armed ?character)
                                (not (armed ?target)))
                           (not (alive ?target)))
                    (when (and (not (armed ?character))
                                (armed ?target))
                           (not (alive ?character)))
                    (when (and (not (armed ?character))
                                (not (armed ?target)))
                           (not (alive ?target)))
                    (when (citizen ?target)
                           (criminal ?character)))
  :agents       (?character))

;; A character poisons some food.
(:action poison
  :parameters   (?character - protagonist ?poison - poison ?food - food)
  :precondition (and (alive ?character)
                    (has ?character ?poison)
                    (has ?character ?food))
  :effect       (and (not (has ?character ?poison))
                    (poisoned ?food))
  :agents       (?character))

;; A character eats some food.
(:action eat
  :parameters   (?character - humanoid ?food - food)
  :precondition (and (alive ?character)
                    (has ?character ?food))
  :effect       (and (not (has ?character ?food))
                    (not (hungry ?character))
                    (when (poisoned ?food)
                           (not (alive ?character))))
  :agents       (?character))

;; A character picks up an item.
```

```
(:action pickup
  :parameters   (?character - humanoid ?item - item ?location - location)
  :precondition (and (alive ?character)
                     (at ?character ?location)
                     (at ?item ?location)
                     (or (not (guard ?character))
                         (not (owned ?item))))
  :effect       (and (not (at ?item ?location))
                     (has ?character ?item))
  :agents       (?character))

;; One character trades an item for another item that another character is selling.
(:action trade
  :parameters   (?character - protagonist ?character_item - item ?seller - merchant
                 ?seller_item - item ?location - location)
  :precondition (and (alive ?character)
                     (at ?character ?location)
                     (has ?character ?character_item)
                     (alive ?seller)
                     (at ?seller ?location)
                     (at ?seller_item ?location)
                     (owns ?seller ?seller_item)
                     (buying ?seller ?character_item)
                     (selling ?seller ?seller_item))
  :effect       (and (not (has ?character ?character_item))
                     (has ?seller ?character_item)
                     (not (at ?seller_item ?location))
                     (has ?character ?seller_item)
                     (not (owns ?seller ?seller_item))
                     (owns ?character ?seller_item)
                     (not (buying ?seller ?character_item))
                     (not (selling ?seller ?seller_item)))
  :agents       (?character))

;; One character gives an item to another.
(:action give
  :parameters   (?character - protagonist ?item - item ?receiver - humanoid
                 ?location - location)
  :precondition (and (not (= ?character ?receiver))
                     (alive ?character)
                     (at ?character ?location)
                     (has ?character ?item)
                     (alive ?receiver)
                     (at ?receiver ?location))
  :effect       (and (not (has ?character ?item))
                     (has ?receiver ?item))
  :agents       (?character))

;; A character casts the Light spell.
(:action cast_light_spell
  :parameters   (?character - protagonist ?spell - light_spell ?light - light)
  :precondition (and (alive ?character)
                     (has ?character ?spell)
                     (summons ?spell ?light))
  :effect       (and (not (has ?character ?spell))
```

```
                    (has ?character ?light))
  :agents       (?character))

;; A character casts the Alchemy spell.
(:action cast_alchemy_spell
  :parameters   (?character - protagonist ?spell - alchemy_spell ?item - item ?money - money)
  :precondition (and (alive ?character)
                    (has ?character ?spell)
                    (has ?character ?item)
                    (summons ?spell ?money))
  :effect       (and (not (has ?character ?spell))
                    (not (has ?character ?item))
                    (has ?character ?money))
  :agents       (?character))

;; A character casts the Kill spell.
(:action cast_kill_spell
  :parameters   (?character - humanoid ?spell - kill_spell ?target - character
                 ?location - location)
  :precondition (and (alive ?character)
                    (has ?character ?spell)
                    (at ?character ?location)
                    (alive ?target)
                    (at ?target ?location))
  :effect       (and (not (has ?character ?spell))
                    (not (alive ?target)))
  :agents       (?character))

;; A character casts the Teleport spell.
(:action cast_teleport_spell
  :parameters   (?character - protagonist ?spell - teleport_spell ?from - location
                 ?to - location)
  :precondition (and (alive ?character)
                    (has ?character ?spell)
                    (at ?character ?from)
                    (leads_to ?spell ?to))
  :effect       (and (not (has ?character ?spell))
                    (not (at ?character ?from))
                    (at ?character ?to))
  :agents       (?character))

;; When a character dies, they drop all their items.
(:axiom
  :vars         (?character - humanoid ?item - item ?location - location)
  :context      (and (not (alive ?character))
                    (at ?character ?location)
                    (has ?character ?item))
  :implies      (and (not (has ?character ?item))
                    (at ?item ?location)))

;; When a character dies, they no longer own their items.
(:axiom
  :vars         (?character - humanoid ?item - item)
  :context      (and (not (alive ?character))
                    (owns ?character ?item))
```

```
  :implies      (not (owns ?character ?item)))

;; When an item is owned by anyone, it is owned.
(:axiom
  :vars         (?item - item)
  :context      (and (not (owned ?item))
                     (exists (?c - humanoid)
                             (owns ?c ?item)))
  :implies      (owned ?item))

;; When an item is not owned by anyone, it is not owned.
(:axiom
  :vars         (?item - item)
  :context      (and (owned ?item)
                     (forall (?c - humanoid)
                             (not (owns ?c ?item))))
  :implies      (not (owned ?item)))

;; When a character has a weapon, they are armed.
(:axiom
  :vars         (?character - character)
  :context      (and (not (armed ?character))
                     (exists (?w - weapon)
                             (has ?character ?w)))
  :implies      (armed ?character))

;; When a character has no weapon, they are not armed.
(:axiom
  :vars         (?character - character)
  :context      (and (armed ?character)
                     (forall (?w - weapon)
                             (not (has ?character ?w))))
  :implies      (not (armed ?character)))

;; When a character has a stolen item, the owner attacks.
(:axiom
  :vars         (?character - humanoid ?item - item ?owner - humanoid)
  :context      (and (not (= ?character ?owner))
                     (alive ?character)
                     (has ?character ?item)
                     (alive ?owner)
                     (owns ?owner ?item))
  :implies      (and (intends ?owner (not (alive ?character)))
                     (when (citizen ?owner)
                           (criminal ?character))))

;; A guard wants to kill criminals.
(:axiom
  :vars         (?character - humanoid ?guard - guard)
  :context      (and (alive ?character)
                     (criminal ?character)
                     (alive ?guard))
  :implies      (intends ?guard (not (alive ?character))))

;; A predator wants to attack nearby humans.
```

```
  (:axiom
    :vars         (?predator - predator ?character - humanoid ?location - location)
    :context      (and (alive ?predator)
                       (at ?predator ?location)
                       (alive ?character)
                       (at ?character ?location))
    :implies      (intends ?predator (not (alive ?character)))))

;;;
;;; The characters, objects, and things in the interactive narrative adventure game The
;;; Best Laid Plans
;;; Goal: goblin gets the tonic and returns to the tower
;;; Created by Stephen G. Ware
;;;
(define (problem win)
  (:domain best-laid-plans)
  (:objects ; Characters
            goblin - protagonist
            bandit - humanoid
            guard - guard
            merchant - merchant
            chemist - merchant
            barkeep - merchant
            crocodile - predator
            witch - merchant
            troll - humanoid
            wolf - predator
            ; Locations
            tower - location
            crossroads - location
            bridge - location
            junction - location
            camp - location
            town - location
            shop - location
            tavern - location
            alley - location
            sewer - location
            bog - location
            clearing - location
            caveLedge - location
            cave - location
            forest - location
            ; Items
            tonic - item
            ; Money
            money_1 - money
            money_2 - money
            spell_money - money
            ; Weapons
            sword_1 - weapon
            sword_2 - weapon
            sword_3 - weapon
            ; Lights
            torch - light
```

```
          spell_torch_1 - light
          spell_torch_2 - light
          ; Food
          ale - food
          ; Poison
          nightshade - poison
          ; Spells
          light_spell_1 - light_spell
          light_spell_2 - light_spell
          alchemy_spell_1 - alchemy_spell
          kill_spell_1 - kill_spell
          teleport_spell_1 - teleport_spell)
(:init ; Map
       (path tower crossroads) (path crossroads tower)
       (path crossroads bridge) (path bridge crossroads)
       (path bridge junction) (path junction bridge)
       (path junction camp) (path camp junction)
       (path junction town) (path town junction)
       (path town shop) (path shop town)
       (path town tavern) (path tavern town)
       (path town alley) (path alley town)
       (dark sewer)
       (path alley sewer) (path sewer alley)
       (path sewer bog) (path bog sewer)
       (path bog clearing) (path clearing bog)
       (path clearing caveLedge) (path caveLedge clearing)
       (dark cave)
       (path caveLedge cave) (path cave caveLedge)
       (path clearing forest) (path forest clearing)
       (path forest crossroads) (path crossroads forest)
       ; Goblin
       (alive goblin)
       (knows-location goblin tower)
       (knows-location goblin crossroads)
       (knows-location goblin bridge)
       (knows-location goblin junction)
       (knows-location goblin camp)
       (knows-location goblin town)
       (knows-location goblin shop)
       (knows-location goblin tavern)
       (knows-location goblin alley)
       (knows-location goblin sewer)
       (knows-location goblin bog)
       (knows-location goblin clearing)
       (knows-location goblin caveLedge)
       (knows-location goblin cave)
       (knows-location goblin forest)
       (at goblin tower)
       (has goblin money_1)
       (intends goblin (alive goblin))
       (intends goblin (and (at goblin tower)
                            (has goblin tonic)))
       ; Bandit
       (alive bandit)
       (knows-location bandit camp)
```

```
(knows-location bandit junction)
(knows-location bandit bridge)
(knows-location bandit town)
(at bandit camp)
(has bandit sword_1)
(owns bandit sword_1)
(at money_2 camp)
(owns bandit money_2)
(criminal bandit)
(intends bandit (alive bandit))
(intends bandit (has bandit tonic))
; Guard
(alive guard)
(knows-location guard town)
(knows-location guard shop)
(at guard town)
(has guard sword_2)
(owns guard sword_2)
(citizen guard)
(guard guard)
(hungry guard)
(intends guard (alive guard))
(intends guard (not (hungry guard)))
; Merchant
(alive merchant)
(knows-location merchant town)
(at merchant town)
(buying merchant money_1)
(buying merchant money_2)
(buying merchant spell_money)
(selling merchant sword_3)
(at sword_3 town)
(owns merchant sword_3)
(citizen merchant)
(intends merchant (alive merchant))
; Chemist
(alive chemist)
(knows-location chemist shop)
(at chemist shop)
(buying chemist money_1)
(buying chemist money_2)
(buying chemist spell_money)
(selling chemist tonic)
(owns chemist tonic)
(at tonic shop)
(selling chemist light_spell_1)
(owns chemist light_spell_1)
(at light_spell_1 shop)
(selling chemist alchemy_spell_1)
(owns chemist alchemy_spell_1)
(at alchemy_spell_1 shop)
(selling chemist teleport_spell_1)
(owns chemist teleport_spell_1)
(at teleport_spell_1 shop)
(citizen chemist)
```

```
        (intends chemist (alive chemist))
        ; Barkeep
        (alive barkeep)
        (knows-location barkeep tavern)
        (at barkeep tavern)
        (buying barkeep money_1)
        (buying barkeep money_2)
        (buying barkeep spell_money)
        (selling barkeep ale)
        (owns barkeep ale)
        (at ale tavern)
        (citizen barkeep)
        (intends barkeep (alive barkeep))
        ; Crocodile
        (alive crocodile)
        (knows-location crocodile sewer)
        (knows-location crocodile bog)
        (knows-location crocodile alley)
        (at crocodile sewer)
        (intends crocodile (alive crocodile))
        ; Witch
        (alive witch)
        (knows-location witch bog)
        (at witch bog)
        (buying witch nightshade)
        (selling witch light_spell_2)
        (owns witch light_spell_2)
        (at light_spell_2 bog)
        (selling witch kill_spell_1)
        (owns witch kill_spell_1)
        (at kill_spell_1 bog)
        (intends witch (alive witch))
        ; Troll
        (alive troll)
        (knows-location troll cave)
        (at troll cave)
        (intends troll (alive troll))
        ; Wolf
        (alive wolf)
        (knows-location wolf clearing)
        (knows-location wolf forest)
        (knows-location wolf bog)
        (at wolf clearing)
        (intends wolf (alive wolf))
        ; Items
        (at torch camp)
        (at nightshade forest)
        ; Spells
        (summons light_spell_1 spell_torch_1)
        (summons light_spell_2 spell_torch_2)
        (summons alchemy_spell_1 spell_money)
        (leads_to teleport_spell_1 bridge))
(:goal (and (at goblin tower)
            (has goblin tonic))))
```

```
;;;
;;; Solution discovered by Glaive
;;; 109 nodes visited; 586 nodes generated
;;;
(define (plan win-solution)
  (:problem win)
  (:steps (walk goblin tower crossroads)
          (walk goblin crossroads bridge)
          (walk goblin bridge junction)
          (walk goblin junction town)
          (walk goblin town shop)
          (pickup goblin tonic shop)
          (pickup goblin teleport_spell_1 shop)
          (cast_teleport_spell goblin teleport_spell_1 shop bridge)
          (walk goblin bridge crossroads)
          (walk goblin crossroads tower)))

;;;
;;; The characters, objects, and things in the interactive narrative adventure game The Best
;;; Laid Plans
;;; Goal: goblin is dead
;;; Created by Stephen G. Ware
;;;
(define (problem die)
  (:domain best-laid-plans)
  (:objects ; Characters
            goblin - protagonist
            bandit - humanoid
            guard - guard
            merchant - merchant
            chemist - merchant
            barkeep - merchant
            crocodile - predator
            witch - merchant
            troll - humanoid
            wolf - predator
            ; Locations
            tower - location
            crossroads - location
            bridge - location
            junction - location
            camp - location
            town - location
            shop - location
            tavern - location
            alley - location
            sewer - location
            bog - location
            clearing - location
            caveLedge - location
            cave - location
            forest - location
            ; Items
            tonic - item
            ; Money
```

```
          money_1 - money
          money_2 - money
          spell_money - money
          ; Weapons
          sword_1 - weapon
          sword_2 - weapon
          sword_3 - weapon
          ; Lights
          torch - light
          spell_torch_1 - light
          spell_torch_2 - light
          ; Food
          ale - food
          ; Poison
          nightshade - poison
          ; Spells
          light_spell_1 - light_spell
          light_spell_2 - light_spell
          alchemy_spell_1 - alchemy_spell
          kill_spell_1 - kill_spell
          teleport_spell_1 - teleport_spell)
(:init ; Map
       (path tower crossroads) (path crossroads tower)
       (path crossroads bridge) (path bridge crossroads)
       (path bridge junction) (path junction bridge)
       (path junction camp) (path camp junction)
       (path junction town) (path town junction)
       (path town shop) (path shop town)
       (path town tavern) (path tavern town)
       (path town alley) (path alley town)
       (dark sewer)
       (path alley sewer) (path sewer alley)
       (path sewer bog) (path bog sewer)
       (path bog clearing) (path clearing bog)
       (path clearing caveLedge) (path caveLedge clearing)
       (dark cave)
       (path caveLedge cave) (path cave caveLedge)
       (path clearing forest) (path forest clearing)
       (path forest crossroads) (path crossroads forest)
       ; Goblin
       (alive goblin)
       (knows-location goblin tower)
       (knows-location goblin crossroads)
       (knows-location goblin bridge)
       (knows-location goblin junction)
       (knows-location goblin camp)
       (knows-location goblin town)
       (knows-location goblin shop)
       (knows-location goblin tavern)
       (knows-location goblin alley)
       (knows-location goblin sewer)
       (knows-location goblin bog)
       (knows-location goblin clearing)
       (knows-location goblin caveLedge)
       (knows-location goblin cave)
```

```
(knows-location goblin forest)
(at goblin tower)
(has goblin money_1)
(intends goblin (alive goblin))
(intends goblin (and (at goblin tower)
                     (has goblin tonic)))
; Bandit
(alive bandit)
(knows-location bandit camp)
(knows-location bandit junction)
(knows-location bandit bridge)
(knows-location bandit town)
(at bandit camp)
(has bandit sword_1)
(owns bandit sword_1)
(at money_2 camp)
(owns bandit money_2)
(criminal bandit)
(intends bandit (alive bandit))
(intends bandit (has bandit tonic))
; Guard
(alive guard)
(knows-location guard town)
(knows-location guard shop)
(at guard town)
(has guard sword_2)
(owns guard sword_2)
(citizen guard)
(guard guard)
(hungry guard)
(intends guard (alive guard))
(intends guard (not (hungry guard)))
; Merchant
(alive merchant)
(knows-location merchant town)
(at merchant town)
(buying merchant money_1)
(buying merchant money_2)
(buying merchant spell_money)
(selling merchant sword_3)
(at sword_3 town)
(owns merchant sword_3)
(citizen merchant)
(intends merchant (alive merchant))
; Chemist
(alive chemist)
(knows-location chemist shop)
(at chemist shop)
(buying chemist money_1)
(buying chemist money_2)
(buying chemist spell_money)
(selling chemist tonic)
(owns chemist tonic)
(at tonic shop)
(selling chemist light_spell_1)
```

```
(owns chemist light_spell_1)
(at light_spell_1 shop)
(selling chemist alchemy_spell_1)
(owns chemist alchemy_spell_1)
(at alchemy_spell_1 shop)
(selling chemist teleport_spell_1)
(owns chemist teleport_spell_1)
(at teleport_spell_1 shop)
(citizen chemist)
(intends chemist (alive chemist))
; Barkeep
(alive barkeep)
(knows-location barkeep tavern)
(at barkeep tavern)
(buying barkeep money_1)
(buying barkeep money_2)
(buying barkeep spell_money)
(selling barkeep ale)
(owns barkeep ale)
(at ale tavern)
(citizen barkeep)
(intends barkeep (alive barkeep))
; Crocodile
(alive crocodile)
(knows-location crocodile sewer)
(knows-location crocodile bog)
(knows-location crocodile alley)
(at crocodile sewer)
(intends crocodile (alive crocodile))
; Witch
(alive witch)
(knows-location witch bog)
(at witch bog)
(buying witch nightshade)
(selling witch light_spell_2)
(owns witch light_spell_2)
(at light_spell_2 bog)
(selling witch kill_spell_1)
(owns witch kill_spell_1)
(at kill_spell_1 bog)
(intends witch (alive witch))
; Troll
(alive troll)
(knows-location troll cave)
(at troll cave)
(intends troll (alive troll))
; Wolf
(alive wolf)
(knows-location wolf clearing)
(knows-location wolf forest)
(knows-location wolf bog)
(at wolf clearing)
(intends wolf (alive wolf))
; Items
(at torch camp)
```

```
        (at nightshade forest)
        ; Spells
        (summons light_spell_1 spell_torch_1)
        (summons light_spell_2 spell_torch_2)
        (summons alchemy_spell_1 spell_money)
        (leads_to teleport_spell_1 bridge))
  (:goal (not (alive goblin))))

;;;
;;; Example solution generated by hand
;;;
(define (plan die-solution)
  (:problem die)
  (:steps (walk goblin tower crossroads)
          (walk goblin crossroads bridge)
          (walk goblin bridge junction)
          (walk goblin junction town)
          (walk goblin town shop)
          (pickup goblin tonic shop)
          (attack chemist goblin shop)
          (non-executed (pickup goblin teleport_spell_1 shop))
          (non-executed (cast_teleport_spell goblin teleport_spell_1 shop bridge))
          (non-executed (walk goblin bridge crossroads))
          (non-executed (walk goblin crossroads tower))))
```