Toward a Computational Model of Narrative Conflict

Stephen G. Ware

May 24, 2011

Abstract

Conflict is an essential element of interesting stories, and while this has been acknowledged in the fields of Narratology and Artificial Intelligence, no formal model of conflict yet exists. I survey work in these areas to arrive at a definition: conflict occurs when a goal-seeking agent's plan is thwarted—it cannot succeed because the conditions of the world prevent one step in the plan from being carried out. These difficulties arise from the plans of the agent itself, the actions of other goal-seeking agents, or the environment. Individual conflicts can be further analyzed based on seven dimensions: participants, subject, duration, directness, intensity, balance, and resolution.

This definition of conflict as thwarted plans meshes nicely with existing research that uses AI planning for story generation. However, traditional planners make every effort to remove conflict from plans. I present CPOCL, a Conflict Partial Order Causal Link planner which allows narrative conflict to arise in a plan without destroying causal soundness. This model and algorithm are intended as a foundation for research into conflict-based story generation.

Contents

1 Introduction

Consider this reimagining of the classic fairy tale *Little Red Riding Hood*:

> One day, Little Red Riding Hood's mother gave her a basket of cakes and told her to deliver them to her elderly grandmother who lived in another village. Red Riding Hood set off through the forest and soon arrived at her grandmother's cottage. She delivered the cakes, and everyone lived happily ever after.

This story is unlikely to top the bestsellers list. Why? It has characters, setting, plot, and even a happy ending. Something essential is missing, and that something is *conflict*. There is no antagonist to oppose Red Riding Hood and generate narrative friction. Put simply, the story is boring.

Conflict is a key component of interesting stories. The *Routledge Encyclopedia of Narrative Theory* [20], *A Dictionary of Narratology* [33], 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" [20], while Brooks and Warren even tell us that "story means conflict" [8].

Psychologists studying narrative claim that conflict engages the audience of a story. It causes the audience to ask questions and form expectations about the outcome, which propels them forward through the plot [16, 1]. Readers also experience *anomalous suspense*, which is a form of engagement that occurs despite knowing the outcome of the story [16, 9, 1].

Other scholars analyzing computational storytelling have come to similar conclusions about the centrality of conflict [29, 39, 43, 41, 12, 4, 28]. Szilas, creator of the IDtension narrative system, declares that "the notion of conflict is the core of the drama" [44]. Crawford argues that a narrative game is impossible without a thorough analysis of conflict [12]. Screenwriting handbooks also highlight the importance of a story's central struggle [13, 48, 21, 9, 47].

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. [47, p. 164]

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.

This paper describes one such model of narrative conflict suitable for automatic story generation systems. I begin by exploring the phenomenon of conflict in narrative and arriving at a formal definition which can be represented in the data structures of AI planning systems. I then describe a planning algorithm, which is based on previous work in computational narrative, that permits conflict in stories. I conclude with a discussion of the limitations and future directions of this research.

2 What is Conflict?

There is debate amongst narratologists about whether conflict is a necessary condition for narrative, but the fringe cases here are mainly interesting as a philosophical exercise. The kinds of stories that authors want to generate *do* contain some kind of struggle, so for pragmatic purposes I side with Prince and others in saying that it is necessary [33]. Herman goes a step further in saying that conflict is actually *constitutive* of narrative. He claims that it is not simply an ingredient, but the very heart and soul of a story [20]:

> The difficulties participants experience in trying to accomplish plans [...] confer on sequences the noteworthiness of tellability distinguishing a story from a stereotype. Action structures are mental models of such participant-oriented patterns of effort, conflict, trouble, and, in some cases at least, resolution of conflict and overcoming trouble. [19, p. 90]

In other words, conflict is what differentiates a story from a sequence of events.

The structure of a story usually corresponds to the structure of one or more of its conflicts. Chapters or scenes usually end when subplots finish (or right before they finish, so as to entice the audience into the next part). A story's climax is often the resolution of its central contest. Conflict can be more than just a feature of narrative; it can be the very skeleton. In the words of H. Porter Abbott, "Conflict structures narrative" [1]. Even storytellers who believe that character is paramount acknowledge that conflict is an essential part of developing character [13]. Authors wishing to generate effective narrative experiences must understand this essential phenomenon. This section provides a brief survey of what conflict is, how it can be classified, and why it is important.

2.1 Defining Conflict: How Are All Conflicts the Same?

Conflict can be as obvious as the boxing match in *Rocky*, or as subtle as the political intrigue in *The Manchurian Candidate*. It can come from the plot of an enemy, the difficulties of the environment, or even from the warring forces within the protagonist's own mind.

All kinds of conflict involve the formation and thwarting (or attempted thwarting) of a plan [8, 19, 33, 1]. Some character sets a goal and works toward it, only to encounter difficulties along the way which help to make the story interesting. These difficulties may be foreseen or surprising. The antagonistic forces may succeed or fail. These details identify specific stories, but at the heart of it all is planning and the thwarting of plans. *Conflict* occurs when a goal-seeking agent forms a plan which has the potential to fail due to the actions of an opposing force, which is said to thwart the plan. *Internal conflict* occurs when the opposing force is the agent itself. *External conflict* occurs when the opposing force is another agent or the environment.

This is a very broad definition that is intended to encompass every narrative conflict. It applies to obvious cases, like when the Roadrunner triggers one of Wile E. Coyote's traps prematurely. It also applies to more subtle cases like the story of Robinson Crusoe, where the sea and the harsh Caribbean environment thwart Robinson's plan to reach Brazil and eventually even his plan to survive.

To ensure the definition is not overly general, it is also important to clarify what it does not cover. A thwarted plan is not equivalent to an unfulfilled plan. In the Red Riding Hood story, Red begins at her house but must reach her grandmother's cottage before making her delivery. At the start of the story, it would be overzealous to say that Red is in conflict with her environment or with the force of time simply because she has not yet reached her destination. She has not finished carrying out her plan, but there is no reason to believe that it will be thwarted. Her plan to travel from one place to another is not a source of conflict.

Conflict is a source of narrative tension, but it is not the only source, and not all sources of tension are conflicts. In *Romeo and Juliet*, the Montagues and Capulets hate one another. This tension is felt when members of both sides meet in the streets of Verona in the first scene, but feelings of ire do not a story make. "Two men meet and hate one another," is a story which contains no conflict, only tension. The ensuing street brawl is required. A plan for some kind of action (in other words, a plot) is essential [13]. Conflict, as defined here, is a property of plots, not of situations.

2.2 Classifying Conflict: How Do Conflicts Differ?

Conflict comes in many forms, so it is important to consider the various criteria by which individual kinds of conflict are distinguished. This section introduces seven important dimensions which will be formally defined in the next section.

1. Participants - the opposing forces between whom the conflict occurs
2. Subject - the world condition which is preventing the progress of the story
3. Duration - the length of time during which the forces are in conflict
4. Directness - the closeness (physical, emotional, interpersonal, etc.) of the participants
5. Intensity - the difference between a participant's utility if it succeeds and its utility if it fails
6. Balance - how evenly matched the participants are
7. Resolution - the outcome of the conflict and its effects on the participants

This list was compiled from several sources (especially [20, 12, 13, 45]), but its completeness is an open question. There is no definitive narratological study of conflict on which to rely, so I have attempted to distill as many important features as possible from those sources which do make mention of it. The next section presents a formalization of planning, thwarted plans, conflict, and the seven dimensions which can be used to classify it.

3 Modeling Story and Conflict

This section describes a formal model of conflict in stories based on the data structures of AI planning. It is important to note that, while representations and algorithms for planning evolved simultaneously, they can be considered independently. Indeed, *plan data structures* are quite useful as models of story, but *planning algorithms* are often quite bad at producing the stories we wish to generate. This section is devoted to the model. Producing the model is described in the next section.

3.1 AI Plans as Story Fabulae

Rimmon-Kenan [37] divides narrative into three distinct levels: fabula, narration, and text. The *fabula* is a complete chronological sequence of actions involving characters, locations, and objects which describe precisely what occurs in the hypothetical story world. A *narration* is a communication of a fabula. It presents only a subset of the actions represented in the fabula and often in non-chronological order to achieve various effects such as surprise or suspense. A *text* is the actual artifact that conveys a story—book, radio, cinema, etc.—and includes the various devices and limitations specific to each medium. Other narratologists make similar distinctions between the *story* and its *discourse*, which is the telling of the story [39, 3].

There is a precedent of using AI plans to represent the fabula layer of stories. Young, Cavazza, Riedl and their collaborators have produced numerous systems that rely on classical planning (selected examples include [10], [54], [53], [35], [2], [32], [25]). Other story generating systems have used representations which, though not equivalent to classical planning, bare a close resemblance (e.g. [31], [26]).

Partial order causal link (POCL) plans are a popular choice for representing stories because they explicitly model the events of a story along with the casual and temporal relationships between them [52]. These are the key ingredients of a fabula [3]. POCL plans can also be easily translated into psychological models, especially QUEST knowledge structures, which formally represent various features of stories [11]. QUEST knowledge structures and their associated search methods mirror the question answering performance of human readers, allowing researchers to test understanding of narrative concepts [17].

In short, modeling stories as AI plans allows us to live in three worlds simultaneously. They can model concepts inspired by narratological research which can then be represented and manipulated on machines and which can then be evaluated with psychological models of story understanding.

3.2 Conflict as Threatened Causal Links

The last few decades have produced a rich body of work on classical least commitment partial order causal link refinement planning. Definitions of essential concepts are repeated here. Additional detail is available in surveys by Weld [50, 51].

3.2.1 Plans and Their Parts

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 [30].

Definition 1. 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. The **initial state** completely describes a story world before the beginning of a plan. The **goal state** is a literal or conjunction of literals which must be true when the plan has finished.

The various objects in the world, such as characters, items, and places, are represented as logical constants.

Consider a very simple story that takes place in a house with two rooms: a bedroom and a kitchen. The rooms are separated by a locked door. You are in the bedroom but wish to be in the kitchen.

Initial State:	`(and (in you bedroom) (near you door) (locked door))`
Goal State:	`(in you kitchen)`

The actions which materialize between the initial and goal states make up the plan. Actions are created from templates.

Definition 2. 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 [15]. The literals in an operator's preconditions and effects can use variables as terms. These variables get bound to individual constants.

Definition 3. A **variable binding** is an equality $v = c$ where v is a variable and c is a constant which can be used in place of the variable when describing a world's state.

Here are two example operators which use variables in their descriptions.

Name:	`(unlock ?who ?what)`
Preconditions:	`(and (near ?who ?what) (locked ?what))`
Effects:	`(not (locked ?what))`
Name:	`(walk ?who ?from ?to)`
Preconditions:	`(and (in ?who ?from) (not (locked door)))`
Effects:	`(and (in ?who ?to) (not (in ?who ?from)))`

Definition 4. An instance of an operator, called a **step**, represents an actual action that is taken in a plan.

Here are two example steps instantiated from the operators above. The variable bindings for the first step are `?who=you` and `?what=door`. The variable bindings for the second step are `?who=you`, `?from=bedroom`, and `?to=kitchen`.

`(unlock you door)`	"You unlock the door."
`(walk you bedroom kitchen)`	"You walk from the bedroom to the kitchen."

Steps in a plan are partially ordered with respect to time [40].

Definition 5. An **ordering** over two steps is denoted $s < u$, where s and u are steps in the plan and s must come before u.

A plan needs to guarantee that, for each step, all of that step's preconditions are true before the step is carried out [27]. A precondition can be true in the initial state or made true by the effect of an earlier step.

Definition 6. A **causal link** is denoted $s \xrightarrow{p} u$, where s is a step with some effect p, u is a step with some precondition p, and $s < u$. A casual link explains how a precondition of a step is satisfied. In other words, p is true for u because s made it so. Given a step u, we say that u's **causal parents** are all steps s such that there exists a causal link $s \xrightarrow{p} u$. A step's **causal ancestors** are its causal parents in the transitive closure of the parent relation. If a plan is imagined as a directed acyclic graph which has steps as nodes and causal links as edges, s is the causal ancestor of u if there exists a path from s to u.

For example, a causal link can exist between the `(not (locked door))` effect of the `unlock` step and the `(not (locked door))` precondition of the `walk` step.

These definitions allow us to arrive at descriptions of the problems and solutions encountered in the field of classical planning:

Definition 7. A **planning domain** is a set of operators, Λ, which describes all the actions that can be taken in a story world. A **planning problem** is a two-tuple, $\langle I, G \rangle$, composed of an initial state I and a goal state G. Given a planning problem, a **planner** attempts to find a set of steps (instantiated from the operators in Λ) that lead from the initial state I to a state in which all the literals in G are true.

The artifact produced by a planner is a plan:

Definition 8. A **plan** is a four-tuple $\langle S, B, O, L \rangle$ where S is a set of steps, B a set of variable bindings, O a set of orderings, and L a set of causal links. A **complete plan** is guaranteed[1] to transform the world from its initial state to its goal state for every valid total ordering of O. Any plan which is not complete is called a **partial plan**.

[1] Classical planning assumes the world is fully-observable, deterministic, static, and discrete.

3.2.2 Intentionality In Plans

Classical planners were originally used to solve real world problems of a logistical nature like robot movement [15]. When they were eventually applied to narrative problems, many deficiencies in classical planning became apparent.

One problem was that characters failed to act on their own intentions. Since a planner is a means-ends analysis tool, it is likely to construct a plan in which the protagonist and antagonist work together to reach the author's desired goal state regardless of personal motivation. In other words, the characters only act to achieve the goals of the author, not their character goals. This problem was addressed with *intentional planning* [34].

Definition 9. An **intention** is a modal predicate of the form (`intends` a g) where a is an actor and g is a literal that actor a wishes to be true. Intentions can only appear in the effects of an operator [34]. Intentions are the only modal predicates allowed in this model.

Operators in an intentional planning domain are annotated with a set of actor who must all consent to carry out a step.

Definition 10. An **intentional operator** is defined as a three-tuple $\langle P, E, A\rangle$ where P is a set of preconditions, E is a set of effects, and A is a set of **actors**, or logical terms which represent characters in the story world. The actors in A must all consent to carry out any step which is an instance of the operator. An **intentional step** is an instance of an intentional operator. An step for which $A = \emptyset$ is called a **happening**. These actions represent events like accidents or the forces of nature which have no attributable agent performing them.

Expanding on the previous example, here are two intentional operators. The first is a happening. The second requires consent from the person who walks.

Name:	`(get-hungry ?who)`
Actors:	*none*
Preconditions:	*none*
Effects:	`(intends ?who (in ?who kitchen))`

Name:	`(walk ?who ?from ?to)`
Actors:	`?who`
Preconditions:	`(and (in ?who ?from) (not (locked door)))`
Effects:	`(and (in ?who ?to) (not (in ?who ?from)))`

One example plan to achieve the goal of being in the kitchen using the above intentional operators would be:

`(get-hungry you)`	"You become hungry."
`(unlock you door)`	"You unlock the door."
`(walk you bedroom kitchen)`	"You walk from the bedroom to the kitchen."

In addition to the information produced by classical planners, intentional planners describe what steps caused characters to adopt their individual goals and what steps achieved those goals. For every step in an intentional plan, we can identify not only why it was added to the plan, but also why the characters involved chose to take the step.

Definition 11. A **motivating step** is an intentional step in a plan which causes an actor to adopt a goal. It has as one of its effects an intention—a modal predicate of the form (`intends` a g) where a is an actor and g a literal which a wishes to make true. A **satisfying step** is an intentional step in the plan which achieves some actor goal. It must have the literal g as one of its effects.

The `get-hungry` step is a motivating step because it causes you to adopt the goal of being in the kitchen. That goal gets satisfied by the `walk` step. A motivating and satisfying step form the beginning and end of a character's subplan to achieve a personal goal. This notion is captured in a structure called an intention frame.

Definition 12. An **intention frame** is defined as a five-tuple $\langle a, g, m, \sigma, T \rangle$ where a is some actor, g is some fact that a wishes to make true, m is the motivating step which caused a to intend g, σ is the satisfying step by which a achieves g, and T is a set of intentional steps such that $\sigma \in T$, and for each intentional step $s_i = \langle P_i, E_i, A_i \rangle \in S$, s_i is a causal ancestor of σ, and $a \in A_i$ (that is, a must be one of the consenting actors for every intentional step in T).

Simply put, an intention frame describes what step caused an actor to adopt a goal, the steps that actor took to achieve the goal, and the step which finally achieved the goal.

Definition 13. An **intentional plan** is a five-tuple $P = \langle S, B, O, L, I \rangle$ where S is a set of intentional steps, B, O, and L are defined as for a plan, and I is a set of intention frames.

The plan above would have one intention frame. It describes why you wanted to be in the kitchen and how you went about arriving at the kitchen.

Actor:	`you`
Goal:	`(in you kitchen)`
Motivating Step:	`(get-hungry you)`
Satisfying Step:	`(walk you bedroom kitchen)`
T:	`(unlock you door)`
	`(walk you bedroom kitchen)`

Henceforth, the steps in an intention frame (this is, T) are referred to as one actor's *subplan* to achieve some goal.

3.2.3 Failed Plans

Riedl and Young clearly identified one significant limitation of intentional planning: the inability to represent plans which fail [34]. In most stories, the subplans of some actors will not succeed or will only partially succeed. I address this problem by extending the planning model to represent steps which are intended but never carried out.

Definition 14. An **executable step** is defined as a four-tuple[2] $\langle P, E, A, x \rangle$, where P is a set of preconditions, E is a set of effects, A is a set of consenting actors, and x is a boolean flag. When $x = true$, the step is an **executed step**, which will occur during the story. When $x = false$, the step is a **non-executed step**, which will not occur during the story. A step which is a happening must be a real step—that is, $x = true$.

This notion of *execution* is not temporal; it should not be confused with similar notions from the subfield of mixed planning and acting. An executed step can be thought of as a "to be executed" step, one which will eventually occur in the story. A non-executed step can be thought of as a "not to be executed" step, one which will never occur in the story.

Non-executed steps do not affect the world state, so their effects cannot satisfy the preconditions of executed steps. If a causal link $s \xrightarrow{p} u$ exists and s is non-executed, then necessarily u is also non-executed.

Definition 15. A **conflict plan** is a five-tuple $\langle S, B, O, L, I \rangle$ where S is a set of executable steps, and B, O, L, and I are defined as for an intentional plan.

This augmentation for representing failed subplans only makes sense in the context of intentional planning. In a narrative setting, intentionality and failure are important traits. A non-executed step which appears in a frame of intention represents an action that some actor wanted to take but was unable to. Non-executed steps are important because they preserve the subplans of all actors (in order to explain how an actor intended to achieve a goal) while allowing only certain subplans to succeed.

[2] Until now, an operator and a step have had the same components, and a step has simply been an instance of an operator. This definition adds to an executable step a fourth component which is not present in an operator.

3.2.4 Threatened Casual Links and Conflict

Fortunately, classical planners already contain a first class representation of conflict called a *threatened casual link*. Unfortunately, they seek to eliminate all such conflicts in order to ensure that a plan is a solution to its planning problem. The crux of this research is to find a way of preserving the information encoded in threatened causal links without destroying the causal soundness of a plan.

Definition 16. A causal link $s \xrightarrow{p} u$ is said to be a **threatened causal link** when these exists a step t which has as an effect $\neg p$ and which can be ordered between s and u. In other words, t threatens $s \xrightarrow{p} u$ when $s < t < u$ is one possible total ordering of the steps s, t, and u based on the ordering constraints in O. The threatening step t undoes some fact p established by s, leaving one or more preconditions of u unsatisfied.

Classical planners seek to eliminate threatened causal links because they indicate causal inconsistencies in a plan. However, when considering non-executed steps, certain threatened causal links need to be reconsidered.

Definition 17. A **conflict** is a four-tuple $\left\langle a_1, a_2, s \xrightarrow{p} u, t \right\rangle$ such that:

- a_1 and a_2 are actors (possibly the same)
- there exists a causal link $s \xrightarrow{p} u$ threatened by step t
- u belongs to an intention frame whose actor is a_1
- t belongs to a different intention frame whose actor is a_2
- either t or u (or both) are non-executed

In other words, a conflict exists when some action threatens an agent's subplan, and either that action is non-executed or the subplan does not proceed past the threat.

These conflicts represent *narrative* problems, not *logical* problems. The subplan *could* succeed or the threatening action *could* succeed, but they cannot *both* succeed, so one of them must fail. This definition captures the narratological notion of conflict described in the previous section.

Two cases have been left out in the definition above: namely when u or t is a happening. These cases correspond to conflicts with the environment. In the interest of keeping definitions simple, we can place all happenings into a special frame of intention which has the environment as its actor. This avoids the need to define special cases for every situation involving happenings.

3.2.5 Example Fabula: *The Princess Bride*

Figure 1 provides an example fabula to illustrate the concepts discussed in this section. The story is a highly-simplified rendition of the 1987 film *The Princess Bride*.

The story begins when the ambitious Prince Humperdinck of Florin secretly hires a troupe of bandits to kidnap and kill his fiance, implicating troops from the rival kingdom of Guilder. The girl is a beautiful maiden named Buttercup who is much beloved by the people of Florin. The murder of Buttercup will, Humperdinck hopes, outrage the people and allow him to declare war on Gilder. However, the kidnapping is thwarted by a young boy named Westley, an old lover of Buttercup's who hopes to marry her. When Humperdinck learns that the kidnapping has failed, he apprehends Buttercup and makes plans to kill her himself. This plan too is eventually thwarted by Westley. Humperdinck is exiled from the kingdom, and the two young lovers eventually marry and live happily ever after.

Steps in the plan are represented as boxes—executed steps with a solid border and non-executed steps with a dashed border. Intention frames are drawn around subplans with dashed lines. Note that one step can be a member of multiple frames. For example, the first step in Humperdinck's top subplan is "Bandits kidnap Buttercup," and the first step in the bandits' plan is "Kidnap buttercup." These two boxes represent the same step which is intended by multiple actors.

Causal links are represented by vertical arrows and labeled by the facts which they establish. Conflicts are represented by horizontal red arrows which indicate when the effect of some step threatens a causal link. Notable examples of each of the seven dimensions of conflict (discussed next) are also highlighted.

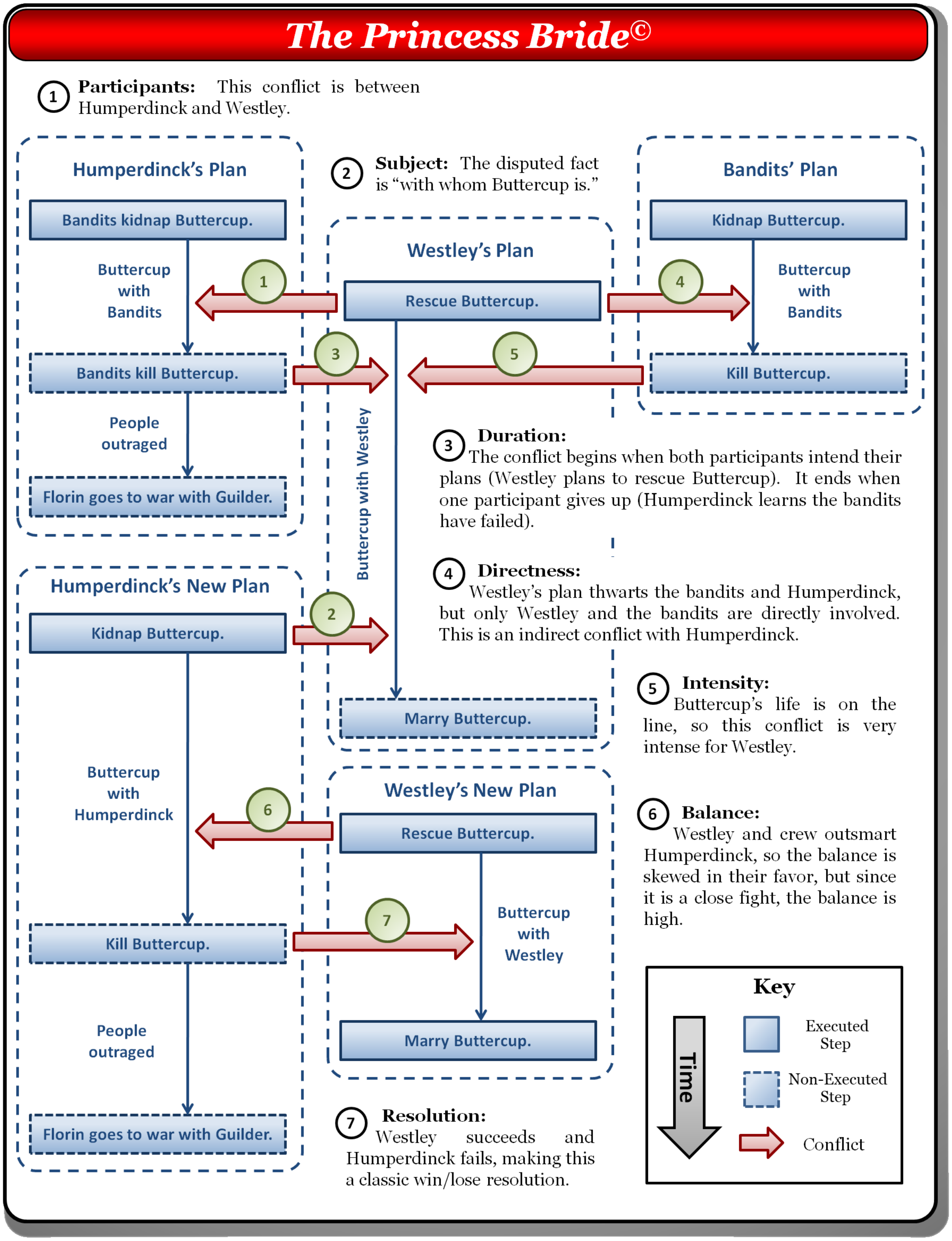


Figure 1: A highly simplified fabula for *The Princess Bride*

3.3 Measuring Dimensions of Conflict

Section 2.2 introduced seven dimensions which differentiate one conflict from another. The following seven sections provide formal definitions of those terms based on the fabula structure described previously. A summary is provided in table 2.

The first three dimensions—participants, subject, and duration—have discrete values which can be directly extracted from a fabula. The other four—directness, intensity, balance, and resolution—correspond to more abstract ideas which are harder to measure. Since there are no agreed-upon definitions for these terms, I provide formulas which give continuous real number values.

The descriptions below assume that some conflict $c = \left\langle a_1, a_2, s \xrightarrow{p} u, t \right\rangle$ exists between two frames of intention $f_1 = \langle a_1, g_1, m_1, \sigma_1, T_1 \rangle$ and $f_2 = \langle a_2, g_2, m_2, \sigma_2, T_2 \rangle$ such that $u \in T_1$ and $t \in T_2$. In words, there is a conflict between actors a_1 and a_2. One of a_2's actions threatens a causal link in a_1's subplan.

Each dimension is stated as a formula. All take at least one parameter, a conflict c.

Some dimensions have different values depending on the point of view from which they are measured: a_1 or a_2. These dimensions come in two forms. A dimension prefixed with an o means "overall." An overall dimension gives the value that would be reported by a disinterested third party observer. Dimensions prefixed with an i mean "individual." These formulas take a second parameter, an actor a, from whose point of view the value is being measured. For example, o_intensity(c) denotes "the overall intensity of conflict c," whereas i_intensity(c, a_1) denotes "the intensity of conflict c from actor a_1's point of view."

These formulas are intended only as preliminary estimates of narrative phenomenon. Readers familiar with economics, game theory, or psychology may find these formulas naive, and I admit that each one could be improved in many ways. What I present here are intentionally simple formulas which represent important features of conflict and are intended to serve as a foundation for further development.

Formula	Range	Meaning
participants(c)	two actors	the participants involved in the conflict
subject(c)	a literal	the disputed fact
duration(c)	an integer	the length of time the conflict lasts
o_directness(c)	a real number between 0 and 1	how close the actors are to one another
i_directness(c, a)	a real number between 0 and 1	how close a is to the other participant
o_intensity(c)	a real number between 0 and 1	how dramatically the conflict will influence the actors
i_intensity(c, a)	a real number between 0 and 1	how dramatically the conflict will influence a
o_balance(c)	a real number between 0 and 1	how evenly matched the participants are
i_balance(c, a)	a real number between 0 and 1	how evenly matched a is to the other participant
o_resoluiton(c)	a real number between -1 and 1	the overall change in utility after the conflict
i_resolution(c, a)	a real number between -1 and 1	a's change in utility after the conflict

Figure 2: A summary of the 7 dimensions of conflict

Function	Range	Meaning
index(s)	an integer ≥ 0	the time index of step s in a total ordering of O
closeness$_i(a_1, a_2)$	a real number between 0 and 1	how close a_1 is to a_2 in some context i
utility(a, P)	a real number between 0 and 1	how satisfied actor a is with the world after steps P
$\pi(P)$	a real number between 0 and 1	how likely steps P are to succeed as intended

Figure 3: A summary of other important functions related to measuring the dimensions of conflict

3.3.1 Participants

The participants of the conflict are the two actors a_1 and a_2. Either of these actors may be the environment.

$$\text{participants}(c) = \{a_1, a_2\}$$

Note that a_1 and a_2 may be the same actor (though u and t must still be in different intention frames). This represents internal conflict—when one character forms two conflicting subplans. Because happenings must be executed steps, it is impossible for a conflict to arise between the environment and itself.

This formula restricts all conflicts to having two participants, but in a narrative, it is often helpful to describe conflicts with multiple participants. The formulas can accommodate this in one of two ways.

Firstly, some groups can be represented as a single logical constant. For example, we can describe a war between Florin and Guilder without naming all the citizens of each nation. In the *Princess Bride* example, the bandits are represented collectively as one agent. Secondly, conflicts between three or more participants which cannot be reduced to two sides can be reduced to pairwise conflicts. If a conflict exists between actors a, b, and c, then a conflict exists between a and b, a and c, and b and c. A more robust method for handling conflicts with multiple participants is planned for future work.

3.3.2 Subject

The subject of a conflict is the contested fact p that actor a_1 is trying to make true and which t potentially makes false.

$$\text{subject}(c) = p$$

Conflicts between two actors often arise in complementary pairs. Take, for example, the conflict between Westley and the bandits in at the beginning of *The Princess Bride*. The bandits want to kill Buttercup, so they plan to kidnap her. Westley wishes to marry Buttercup, so he plans to rescue her. There are two conflicts here: one between the bandits and Westley with subject `(with Buttercup Bandits)` and one between Westley and the bandits with subject `(with Buttercup Westley)`. Westley and the bandits threaten each other's subplans, resulting in two conflicts.

3.3.3 Duration

The duration of a conflict is the span of time during which both participants intend their incompatible subplans. It begins once both subplans have been formed and ends once one subplan either succeeds or is abandoned.

The partially ordered nature of plans presents several obstacles to measuring time. Firstly, classical plan representations do not include an explicit measure of time. Secondly, actions might occur simultaneously. Thirdly, a partial ordering represents many possible total orderings, so durations may differ depending on the total ordering chosen. These problems are addressed by the subfield of scheduling [38], and a truly robust solution will need to incorporate these principles.[3]

In the interest of simplicity, assume that a total ordering has been chosen for all steps. Thus, the story can be represented as a sequence of states. The first is the initial state. The second state is the state of the world after the first step has taken place. The third is the state after the second step, and so on. In each state, actors are either in conflict or not. Thus, duration can be measured as the number of states in which a_1 and a_2 both intend their conflicting subplans.

Let $\text{index}(s) = x$ just when step s is the x^{th} executed step in the chosen total ordering. If step s is the first executed step, $\text{index}(s) = 1$. If s is the second executed step, $\text{index}(s) = 2$, etc. A conflict begins once both actors intend their subplans—that is, after the motivating step with the higher index. It lasts until one subplan succeeds or fails, and generally this will be either step t or step u. However, it is possible that additional conflicts exist that cause both subplans to fail even earlier. In this case, the end of the conflict occurs after the last executed step in either T_1 (which includes u) or T_2 (which includes t).

[3] These assumptions made about representing time are perhaps not as impoverished as they seem. It does not capture the fact that some steps are carried out almost instantly while others may take years. However, when a story is actually presented to an audience, real time gets expanded or contracted in a similar way. The Hundred Years War might be described in a paragraph while a single kiss is expounded upon over an entire chapter. The number of steps that occur in a fabula correspond roughly with the amount of change that happens in a story world, and thus might be a more useful measure of duration than real world time. Imposing a total ordering also prevents steps from being carried out simultaneously. This is not realistic, but when a story is presented its events must be narrated in some sequence, even if it is understood that they are occurring simultaneously.

$$\begin{aligned}
\text{start}(c) &= \max\left(\text{index}(m_1), \text{index}(m_2)\right) \\
\text{end}(c) &= \max_i\left(i = index(s)\ \forall s \in T_1 \cup T_2 \text{ such that } s \text{is executed}\right) \\
\text{duration}(c) &= \text{end}(c) - \text{start}(c) + 1
\end{aligned}$$

One total ordering of the executed steps in figure 1 can be represented like so:

state 0 (initial state)
  index 1 - Bandits kidnap Buttercup.
state 1
  index 2 - Westley rescues Buttercup.
state 2
  index 3- Humperdinck kidnaps Buttercup.
state 3
  index 4 - Westley rescues Buttercup.
state 4
  index 5 - Westley marries Buttercup.
state 5 (goal state)

Humperdinck's first subplan exists in state 0, but Westley does not form a subplan to rescue her until state 1. Thus, the first conflict between Humperdinck and Westley begins at state 1. Humperdinck has to abandon his subplan in state 2 because the bandits have failed, so the conflict ends at state 2. This conflict spans 2 states, so its duration is 2.

3.3.4 Directness

> *The most direct conflict places the two combatants face to face at arm's length; whether they are punching each other or insulting each other, their conflict is about as direct as it can get. But conflict need not be so simple; recourse to indirection can often yield more interesting possibilities.* [12, p. 59]

The directness of a conflict is a measure of how closely the participants are related. There are many kinds of closeness which might be measured depending on the domain: geospacial, friendship, family relation, etc., so o_directness(c, a) is simply the average of n kinds of *closeness*:

$$0 \leq \text{i\_directness}(c, a_1) = \frac{\sum_{i=1}^{n} \text{closeness}_i(a_1, a_2)}{n} \leq 1$$

Since directness can change over the course of a subplan, it should be measured at the last moment of the conflict. As noted in the previous section, this is usually index(t) or index(u), but might occur earlier if some other conflict prevents both t and u from being carried out. Formally, directness should be measured during state $\max_i(i = \text{index}(s)\ \forall s \in T_1 \cup T_2$ such that $s$is executed).

We are only concerned with representing *fabula* here, which makes no distinctions between which characters or events are more important than others. Therefore, we assume that all kinds of closeness are equally important and that all characters are equally important. At the level of *narration*, certain story elements will be given preference and this formula becomes a weighted average. A family drama, for instance, would weight familial closeness heavily. We leave the translation of fabula to narration for future work; thus every story element is assumed to have equal value.

Various kinds of closeness can be captured with binary predicates, i.e. friends(a_1, a_2) is true when a_1 considers a_2 a friend. It is also possible to model more complex notions like physical closeness and *interpersonal closeness*, which occurs when one actor uses others to accomplish its goals. When Humperdinck hires the bandits to capture Buttercup, the bandits form a subplan to kidnap her. Buttercup is in conflict with both the bandits and Humperdinck, but while she is interpersonally close to the bandits, she is interpersonally far from Humperdinck.

Note that in some cases, i_directness$(c, a_1) \neq$ i_directness(c, a_2). This might happen if a_1 considers a_2 a friend, but the feeling is not mutual. This emphasizes the need to measure dimensions based on individual point of view.

Since we have assumed that all story elements carry equal importance, the overall directness of a conflict is simply the average of its participants' individual values:

$$\text{o\_directness}(c) = \frac{\text{i\_directness}(c, a_1) + \text{i\_directness}(c, a_2)}{2}$$

Again, this would be a weighted average if measured at the *narration* level.

3.3.5 Intensity

> *There is no narratively remarkable conflict involved— [...] to qualify as a narrative, the action sequence would have to highlight not just the way the world would have been had X not brushed her teeth but also the* riskiness *of her allowing it to take the alternative course, what with Tooth Decay and Recession of the Gums lurking in the wings.* [19, p. 76]

Intensity measures how much is at stake—how dramatically the effects of a conflict will influence the actors involved. For this we must introduce $0 \leq \text{utility}(a, P) \leq 1$, which denotes how satisfied some actor a is with the state of the story world after some sequence of steps P has been carried out. An actor's satisfaction in the initial state of the story is denoted $\text{utility}(a, \emptyset)$. If $\{e_1, e_2, ..., e_x\}$ is all the executed steps in a story, an actor's utility after the n^{th} step is $\text{utility}(a, \{e_1, e_2, ..., e_n\})$. Including non-executed steps in P allows us to measure utility in a hypothetical world.

One simple formula for measuring this is captured by the difference between how happy an actor will be if he succeeds and how sad he will be if he fails. However, quantifying how badly things can go wrong is difficult because it requires one to imagine any number of ways that a subplan could fail. The search space of plans includes many more failed plans than successful ones, usually infinitely more. Luckily, when dealing with conflict, one important alternative outcome is already given in the subplan of the opposing actor. This provides a simple but useful heuristic for measuring how badly a subplan can fail. Thus, the intensity of a conflict for actor a can be estimated as his utility if his subplan succeeds minus his utility if his opponent's subplan succeeds.

Consider the intensity of conflict c from a_1's point of view. Let E be the set of executed steps occurring before $\max(\text{index}(m_1), \text{index}(m_2))$. E can be thought of as the story up until the conflict begins. The *hypothetical* story in which the opponent's subplan succeeds is made up of steps $E \cup T_2$ (the story so far, plus the steps in a_2's subplan). The formula for intensity is thus:

$$\begin{array}{rlcll} & \textit{high} & = & \max(\text{utility}(a_1, E), \text{utility}(a_1, E \cup T_1)) & \\ & \textit{low} & = & \min(\text{utility}(a_1, E), \text{utility}(a_1, E \cup T_2)) & \\ 0 \leq & \text{i\_intensity}(c, a_1) & = & \textit{high} - \textit{low} & \leq 1 \end{array}$$

The value for *high* is constrained to be a_1's current utility or better, and low is constrained to be a_1's current utility or worse. These constraints prevent negative intensity in the rare cases where a_1 would be better off if its subplan failed or if its opponent's subplan succeeded.

Usually, agents follow subplans that will result in higher utility. If an agent's subplan fails due to a threat from an opponent, this will often lead to a decrease (or no change) in utility. In these cases, intensity is non-zero. Take for example the "low risk, high reward" scenario where a_1's subplan will result in a_1 being much better off and a_2's subplan thwarts a_1 (but does not cause any decrease in utility). Here, intensity is non-zero, but still lower than a "high risk, high reward" scenario in which a_2's subplan would decrease a_1's utility.

To use a more concrete example, consider a subplot from *The Princess Bride*. In the prologue, it is revealed that Westley and Buttercup were in love as children. Westley disappeared unexpectedly and was believed killed by The Dread Pirate Roberts. In truth, Westley became The Dread Pirate Roberts, and uses this persona to rescue Buttercup from the bandits. Westley intends to reveal his true identity to Buttercup, and this is a "high risk, high reward scenario." If he succeeds, Buttercup will love him all the more for rescuing her. If he fails, she will hate him all the more since she believes him to be the pirate who murdered her love. Westley's subplan has a high intensity because he stands to benefit greatly from its success but to suffer greatly from its failure.

Like directness, the overall intensity of a conflict is simply the average of the individual participant values:

$$\text{o\_intensity}(c) = \frac{\text{i\_intensity}(c, a_1) + \text{i\_intensity}(c, a_2)}{2}$$

3.3.6 Balance

> *We see real, rising conflict when the antagonists are evenly matched. There is no thrill in watching a strong, skillful man fighting a sickly, awkward one. When two people are evenly matched, whether in the prize ring or on the stage, each is forced to utilize all that is in him.* [13, p. 132]

Balance measures the relative likelihood that each participant in the conflict will prevail. This implies that plans have the potential to fail, which is allowed via non-executed steps. It also implies that certain plans are more or less likely to fail, which transcends classical planning.

A common extension to allow uncertainty in planning is to give operators multiple sets of effects, each with an associated probability that dictates the likelihood of that outcome [6]. This representation lends itself nicely to many interactive narrative settings (where success might be dependent on a dice roll) but is not strictly necessary for our model. In the interest of generality, we introduce the function $0 \leq \pi(P) \leq 1$ to denote the probability that a sequence of steps P will succeed as intended. If the actions in a_1's subplan are ones which are likely to succeed, then $\pi(T_1)$ is high, and vice versa.

Two conflicting subplans might be independent events, meaning $\pi(T_1) + \pi(T_2) \neq 1$, but balance is a dependent notion. Thus, the formula for a_1's individual balance is the likelihood that a_1 will succeed relative to a_2, assuming that only one of them will succeed:

$$0 \leq \text{i\_balance}(c, a_1) = \frac{\pi(T_1)}{\pi(T_1) + \pi(T_2)} \leq 1$$

It should never occur that $\pi(T_1) + \pi(T_2) = 0$ unless actors form subplans with a 0% chance of success. In this case, $\text{i\_balance}(c, a_1) = 0$.

Egri asserts that a conflict is more interesting when the opposing agents are evenly matched [13]. Based on this observation, the formula for overall balance is structured to be high when both participants are evenly matched and low when one is clearly more likely to succeed:

$$\text{o\_balance}(c) = 1 - |\text{i\_balance}(c, a_1) - \text{i\_balance}(c, a_2)|$$

After he is captured and tortured by Humperdinck, Westley is "mostly dead" and ill-equipped to rescue Buttercup. The balance of the conflict between Westley and Humperdinck is skewed in the prince's favor, meaning that balance is low for Westley, high for Humperdinck, and low overall. Once Westley is revived, he is a more formidable adversary. Normally, he would overpower Humperdinck easily (and the balance would be skewed in Westley's favor), but his near death leaves him physically weak with only his wits to defend himself. The climax of the story occurs when a strong but foolish Humperdink faces off against a weak but cunning Westley. Finally, the balance is not skewed in anyone's favor, making the overall balance high.

3.3.7 Resolution

> *The beginning of an action always presents us with a situation in which there is some element of instability, some conflict; in the middle of that action there is a period of readjustment of forces in the process of seeking a new kind of stability; in the end of an action, some point of stability is reached, the forces that have been brought into play have been resolved.* [8, p. 78]

Resolution measures the outcome, favorable or not, of a conflict for some actor. It is that actor's change in utility between the start and end of the conflict. As such, its range is -1 to 1.

Consider the resolution of conflict c from a_1's point of view. Let E be the set of executed steps occurring before $\max(\text{index}(m_1), \text{index}(m_2))$. In other words, E is the story up until the conflict begins.

Let T'_1 be all the executed steps from T_1 and let T'_2 be all the executed steps from T_2. Let C be all the steps which are causal ancestors of any step in $T'_1 + T'_2$. Finally, let $V = E + T'_1 + T'_2 + C$. In other words, V is the story up until the conflict begins plus only those steps which must be executed to reach the state when the conflict has ended.

Note that V may not be equivalent to the story up until the conflict ends. Steps not related to the conflict are excluded from V because they might influence the utility of the actors. We only want to measure the change in utility which was brought about by the subplans in question, not the overall change in utility. For example, say Bob and Joe both love the same woman and both plan to marry her. While the resulting conflict plays out, Bob wins the lottery. This greatly influences his utility. However, if Joe marries the woman, Bob's resolution value *for this conflict* needs to be negative, even if his *overall* change in utility is positive due to unrelated events.

Having established these sets, resolution can be measured like so:

$$-1 \leq \text{i\_resolution}(c, a_1) = \text{utility}(a_1, V) - \text{utility}(a_1, E) \leq 1$$

Again, overall resolution is the average of individual values:

$$\text{o\_resolution}(c) = \frac{\text{i\_resolution}(c, a_1) + \text{i\_resolution}(c, a_2)}{2}$$

In addition to the continuous values for resolution, several discrete classes of resolution can be identified [45]. At the highest level, these are win/win, win/lose, and lose/lose resolutions. However, conflict in fictional narrative differs from real world conflict in that a win/win situation is often not desirable. Identifying additional classes of resolution (such as trickery) will be an important direction for future work.

4 Generating Stories with Conflict

A model of narrative conflict can be helpful on its own as a tool for story analysis, but its power to inform the story generation process is also quite valuable. Here I discuss the problem of generating stories with interesting narrative conflict. I begin with a survey of previous work and then present my own preliminary solution along with a discussion of its strenghts and limitations.

4.1 Previous Work on Generating Conflict in Computer Narrative

4.1.1 Pre-Scripted Stories

As far back as Talespin [29] and as recently as PaSSAGE [46], narrative generation systems have relied on their creators to supply the conflict that drives the story. This method is common in the video game industry: most story-based games have their plots written at design time rather than generated at run time.

This method has been somewhat generalized by systems like Universe [24] and Mexica [31], which combine pre-scripted plot fragments (or plot grammars) to produce whole stories. However, the general problem of building well-structured plot fragments from scratch remains unsolved:

> The goal state is simply assumed to be an interesting one with no further justification other than our own experience with melodramatic stories. This avoids the need for detailed analysis of what makes a plot fragment interesting. [24, p. 496]

Systems which rely on pre-scripted plots or plot fragments model conflict implicitly. By making conflict explicit in the model, we gain a greater ability to reason about this essential phenomenon and adapt interactive stories.

4.1.2 Adversarial Planning and Game-Playing

One tempting solution for producing conflict would be to generate stories via adversarial planning or some kind of zero sum game-playing algorithm as suggested by Smith and Witten:

> To facilitate the interaction of the antagonist and the protagonist, the story planner allows the characters to take turns at selecting primitive actions for the story events. Both characters use the same heuristic functions for evaluating next states, but the antagonist does so with the purpose of directing the story to low-valued situations. For this reason, when the protagonist is searching the story space for an optimum plan of action from the current situation, he must bear in mind that the next event will be determined by his opponent and, thus, will most probably not be in his best interest. [42, p. 12]

This is an oversimplification of the antagonist's role; it is not simply a malevolent force trying to make trouble for the protagonist at every available opportunity. The antagonist is a force with its own goals, and thwarts the plans of the protagonist only when those goals require it to do so. In other words, the antagonist needs a reason for its actions. An ideal model of conflict should reflect this.

Zambetta, Nash, and Smith modeled conflict in stories as a system of differential equations that simulate an arms race scenario [55]. While this may be helpful as high-level control for the pace of a story, it cannot explain the individual motivations of the participants in a conflict.

4.1.3 Dilemma-Based Stories

Barber and Kudenko [4] create dramatic tension in their GADIN system with *dilemmas*, which they define as decisions the user must make which will negatively affect at least one character. If the user has any concern for these characters, the decisions will be difficult and serve as a source of engagement. In the process of generating a story, GADIN detects when these dilemmas are applicable and uses them to engage the user. Similarly, Szilas measures conflict in his IDtension system based on actions that a character can take to achieve a goal but which violate that character's moral values [44].

These methods represent progress toward encoding an understanding of conflict into the story generation process. However, these dilemmas are a small subset of all the possible kinds of conflict available to story writers. Also, because a dilemma arises and is resolved immediately, it is difficult to model an ongoing or thematic conflict.

Conflict is often used to structure a narrative. The adversarial and dilemma-based models above reason about when conflict *can* occur, but are not equipped to reason about when it *should* occur in order to provide rising action, pacing, and other macro-structural features of stories.

4.2 Evolution of CPOCL

I now present CPOCL, an algorithm for generating stories based on my definition of conflict as threatened causal links. This solution is preliminary because it only solves the foundational problem of *allowing* conflict to arise in plan-based stories. Ideally, it would produce *only* stories which contain conflict. I will revisit this issue and why it is difficult at the end of the section.

Recall from section 3.1 that partial order causal link (POCL) plans resemble story fabulae [52] and can be mapped onto psychological models of narrative [11]. The search space of a POCL refinement algorithm resembles search through a universe of stories, so these algorithms have been a popular foundation for narrative planners. More modern Graph-based, SAT-based, and HSP-based planning algorithms ([5], [23], and [7] respectively) usually perform faster, but since all classical planning algorithms are at least PSPACE complete in complexity [18], the representational richness of POCL refinement algorithms makes them a good foundation for this work.

I describe three algorithms, each one building on the last:

1. The basic POCL refinement planning algorithm
2. An intentional planning algorithm, named IPOCL*
3. A conflict planning algorithm, named CPOCL

4.2.1 The POCL Algorithm

A classical planning problem begins with an initial state, a set of operators which describe the actions that can be taken in the world, and a goal state. The task of the planner is to find a sequence of steps that begins at the initial state and arrives at the goal.

POCL planning is a kind of refinement search [22]. The search space can be described as a directed tree in which each node is either a solution or an incomplete solution to the problem. Incomplete solutions, called *partial plans*, are annotated with flaws that describe why the plan is not yet a solution. At each step of the search, a leaf partial plan node is chosen and one of its flaws is repaired. For every way that the flaw can be repaired, a new partial plan is generated. These partial plan nodes are made the children of the parent. Eventually, a solution (a *complete plan* which has no flaws) is found, or the algorithm fails [50].

Repairing a flaw may cause new flaws to arise. Some flaws cannot be repaired at all, which prompts the search to backtrack. POCL algorithms and their descendants revolve around defining a set of flaws and a set of procedures to repair those flaws.

As defined in section 3.2.1, a POCL plan is a 4-tuple, $\langle S, B, O, L\rangle$, where S is a set of steps, B a set of variable bindings, O a set of ordering constraints over the steps in S, and L a set of causal links. Steps have preconditions which must be satisfied and effects which change the world state. In a complete plan, every precondition of every step must be the head of some causal link which describes how that precondition was established. Further more, no causal link may be threatened. These requirements give rise to the two kinds of flaws:

Definition 18. An **open precondition flaw** indicates that some precondition of a step has not yet been satisfied by a causal link. Formally, an open precondition flaw in a plan $P = \langle S, B, O, L, I\rangle$ 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.

Definition 19. A **threatened causal link flaw** indicates that the condition established by some casual link might be undone before it is needed. Formally, a threatened causal link flaw in a plan $P = \langle S, B, O, L\rangle$ is

Algorithm 1 The POCL (Partial Order Causal Link) Planning Algorithm is a general characterization of similar partial order planning (POP) algorithms defined by others [50].

POCL ($P = \langle S, B, O, L \rangle$, F, Λ)

Where P is a partial plan, F a set of flaws, and Λ a set of operators.
Initially, P is the null plan, and F contains an open precondition flaw for each precondition of the end step.

1. **Termination:** If B or O is inconsistent, fail. If $F = \emptyset$, return P.
2. **Plan Refinement:** Non-deterministically choose a flaw f from F. Let $F' = F - \{f\}$.
   (a) **Goal Planning:** If f is an open precondition flaw $f = \langle s_{need}, p \rangle$, then let s_{add} be some step with an effect p. Non-deterministically choose how to create s_{add}:
      i. **Reuse:** Choose s_{add} from the steps already in S.
      ii. **New Step:** Create a new step s_{add} from Λ which has an effect e that unifies with p. Let $S' = S + \{s_{add}\}$. For each precondition c of s_{add}, add a new open precondition flaw $\langle s, c \rangle$ to F'.

      Create a new causal link $l = s_{add} \xrightarrow{p} s_{need}$. Let $L' = L + \{l\}$. Let $B' = B \cup \mathrm{MGU}(e, p)$. Let $O' = O + \{s_{add} < s_{need}\}$.
   (b) **Threat Resolution:** If f is a threatened causal link flaw $f = \left\langle s \xrightarrow{p} u, t \right\rangle$, then non-deterministically choose a way to prevent the threat:
      i. **Promotion:** Let $O' = O' + \{t < s\}$.
      ii. **Demotion:** Let $O' = O' + \{u < t\}$.
      iii. **Restriction:** Add bindings to B' which cause the threatening effect of t not to unify with p.
3. **Threat Detection:** If any casual link $l \in L'$ has become threatened by any step $s_{threat} \in S'$, add a new threatened causal link flaw $\langle l, s_{threat} \rangle$ to F'.
4. **Recursive Invocation:** Call **POCL** ($P' = \langle S', B', O', L' \rangle$, F', Λ).

---

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 some step in S with effect $\neg p$, and $s < t < u$ is a consistent ordering given the constraints in O.

POCL (Algorithm 1) begins by creating a null plan which contains only a placeholder start step and end step. This becomes the root of the search space tree. POCL then iteratively refines partial plans until a solution is found or no solution is possible.

Definition 20. The **start step**, s, has no preconditions, and it has effects equivalent to the initial state of the planning problem. An **end step**, e, has no effects, and it has a precondition for each fact which must be true in the goal state. The **null plan** is the partial plan $P = \langle \{s, e\}, \emptyset, \emptyset, \emptyset \rangle$.

Now that the flaws have been defined, we can describe an algorithm to repair those flaws.

An open precondition flaw is resolved by adding a new causal link to L. The tail step of that causal link can be an existing step or a newly created step. Creating this link often requires imposing additional variable binding constraints on B. This is done via a most general logical unification function $\mathrm{MGU}(s_1, s_2)$, where s_1 and s_2 are function-free ground predicate literals. Details can be found in Weld [50].

Threatened casual link flaws can be resolved by imposing ordering constraints on O which prevent the threatening step from being ordered between the tail and head steps of the causal link, or they can be resolved by imposing additional binding constraints on B which prevent the threatening effect from unifying with the negation of the causal link's label.

POCL has been proven sound and complete [50]. A **sound** planner only returns plans which are guaranteed to achieve the goal state. A **complete** planner is capable of finding all possible solutions—in other words, it will always return a solution if one exists.

The goals of the planning problem can be thought of as author goals: those things which that author desires be true at the end of the story. However, characters have goals too, and POCL is ill-equipped to represent these.

4.2.2 The IPOCL* Algorithm

Building on POCL, Riedl and Young [34] described an intentional planning algorithm called IPOCL. It explains not only how the author goals of a story are achieved but also what individual goals each character is pursuing when taking action.

Section 3.2.2 described the new structures used to represent intentionality in plans. Each operator and step is annotated with A, a set of actors who must consent to carry out that action. Intentional plans contain **intention frames** which describe why a character adopts a goal and what steps it takes to fulfill that goal. An intentional plan is a 5-tuple, $P = \langle S, B, O, L, I \rangle$, where S, B, O, and L are defined as they are for POCL plans, and I is a set of intention frames. An intention frame is a 5-tuple $\langle a, g, m, \sigma, T \rangle$, where a is an actor, g is a goal literal, m is the motivating step which has an effect (`intends` a g), σ is the satisfying step which has an effect g, and T is a set of steps taken in pursuit of g.

IPOCL: Working Backwards

IPOCL extends POCL by managing a set of intention frames during the planning process. New intention frames get created when steps which could be satisfying steps get added to the plan. Specifically, when a new step is added to an intentional plan, IPOCL may choose to treat each effect of that step as a character goal, creating a new intention frame for it. It must later choose a motivating step to explain why the actor of that frame adopted the goal.

This process is dubbed "working backwards," because the planner first chooses a satisfying step and later chooses a motivating step. One result of this "working backward" method is that some (`intends` a g) effects in the steps of a complete plan might not have a corresponding intention frame. Only those `intends` effects which *must* be used to motivate a frame will be used.

This property of IPOCL is limiting because some characters may not adopt goals when it is appropriate to do so. A more restrictive commitment to intentionality would require that every `intends` effect in the plan *must* have a corresponding intention frame in I[4]. I present an intentional planning algorithm called IPOCL*, which is similar to IPOCL, but achieves this more restrictive commitment to intentionality via a "working forward" method.

IPOCL*: Working Forwards

Every time a step which contains an (`intends` a g) effect is added to an intentional plan, IPOCL* adds a new intention frame to I. IPOCL* must later choose a satisfying step for the frame to explain how the goal was achieved. This is the key difference: whereas IPOCL began with the satisfying step and later chose the motivating step, IPOCL* begins with the motivating step and later chooses the satisfying step. This method ensures that every intention has an intention frame to explain how it was achieved.

The "working forward" method makes the search spaces of IPOCL* and IPOCL different in shape. It is not clear if the differences are beneficial; likely they are helpful for some problems and harmful for others. However, it does address one issue that arises in IPOCL when adding steps that require consent from multiple actors. When adding a step that requires consent from two or more actors, IPOCL may choose to create an intention frame for each actor in each group of the power set of actors. This gives the IPOCL search tree a high branching factor at those decision points. IPOCL* does not experience this because it is able to delay those commitments until later.

The need to satisfy an intention frame with a satisfying step can be formulated as a flaw:

Definition 21. An **unsatisfied intention frame flaw** indicates that a step has not yet been chosen to satisfy the goal of an intention frame. It can be described as a 1-tuple $\langle f \rangle$, where f is some intention frame.

[4] This requirement that every intention be explained by an intention frame is more in keeping with the popular Beliefs Desires Intentions (BDI) model of intelligent agents [**?**]. Bratman claims that the difference between a desire and an intention is commitment; a goal is only an intention if an agent makes plans to achieve it. It is possible that IPOCL did not impose this restriction because intentional planning often causes conflict to arise, and IPOCL was not designed to handle conflict.

Algorithm 2 The IPOCL* (Intentional Partial Order Causal Link) Planning Algorithm

IPOCL* $(P = \langle S, B, O, L, I \rangle, F, \Lambda)$

Where P is a partial plan, F a set of flaws, and Λ a set of operators.
Initially, P is the null plan, and F contains an open precondition flaw for each precondition of the end step. If the start step contains any effects like (`intends` a g), then I contains intention frames for those effects and F contains unsatisfied intention frame flaws for those frames.

1. **Termination:** If B or O is inconsistent, fail. If $F = \emptyset$ and P contains no orphans, return P. Else, if $F = \emptyset$, fail.

2. **Plan Refinement:** Non-deterministically choose a flaw f from F. Let $F' = F - \{f\}$.

   (a) **Goal Planning:** If f is an open precondition flaw $f = \langle s_{need}, p \rangle$, then let s_{add} be some step with an effect p. Non-deterministically choose how to create s_{add}:

      i. **Reuse:** Choose s_{add} from the steps already in S.

      ii. **New Step:** Create a new step s_{add} from Λ which has an effect p. Let $S' = S + \{s_{add}\}$. For each precondition c of s_{add}, add a new open precondition flaw $\langle s, c \rangle$ to F'.
      For each effect of s_{add} like `(intends a g)`, add a new intention frame $r = \langle a, g, s_{add}, \emptyset, \emptyset \rangle$ and add a new unsatisfied intention frame flaw $\langle r \rangle$ to F'.

   Create a new causal link $l = s_{add} \xrightarrow{p} s_{need}$. Let $L' = L + \{l\}$. Let $B' = B \cup \mathrm{MGU}(e, p)$. Let $O' = O + \{s_{add} < s_{need}\}$.
   For every intention frame $r = \langle a, g, \sigma, m, T \rangle \in I$, if $s_{add} \notin T$ and $s_{need} \in T$ and $a \in A$ for s_{add}, then add a new intent flaw $\langle s_{add}, r \rangle$ to F'.

   (b) **Threat Resolution:** If f is a threatened causal link flaw $f = \left\langle s \xrightarrow{p} u, t \right\rangle$, then non-deterministically choose a way to prevent the threat:

      i. **Promotion:** Let $O' = O' + \{t < s\}$.

      ii. **Demotion:** Let $O' = O' + \{u < t\}$.

      iii. **Restriction:** Add bindings to B' which cause the threatening effect of t not to unify with p.

   (c) **Satisfaction Planning:** If f is an unsatisfied intention frame flaw $f = \langle r = \langle a, g, m, \emptyset, T \rangle \rangle$, then let s_{sat} be some step with an effect g. Non-deterministically choose s_{sat} the same way that s_{add} is chosen. Let $T' = T + \{s_{sat}\}$, and let $r' = \langle a, g, m, s_{sat}, T' \rangle$. Let $I' = I - \{r\} + \{r'\}$.

   (d) **Intent Planning:** If f is an intent flaw $f = \langle s_{orphan}, r = \langle a, g, m, \sigma, T \rangle \rangle$, then non-deterministically choose how to handle s_{orphan}:

      i. **Inclusion:** Let $T' = T + \{s_{orphan}\}$, let $r' = \langle a, g, m, \sigma, T' \rangle$, let $I' = I - \{r\} + \{r'\}$, and let $O' = O + \{m < s_{orphan}\}$.
      For each causal link $s \xrightarrow{p} s_{orphan}$ in L, if $a \in A$ for s, add a new intent flaw $\langle s, r' \rangle$ to F'.

      ii. **Exclusion:** Do nothing.

3. **Threat Detection:** If any casual link $l \in L'$ has become threatened by step $s_{threat} \in S'$, add a new threatened causal link flaw $\langle l, s_{threat} \rangle$ to F'.

4. **Recursive Invocation:** Call **IPOCL*** $(P' = \langle S', B', O', L', I' \rangle, F', \Lambda)$.

IPOCL and IPOCL*: Populating Frames

After an intention frame $f = \langle a, g, m, \sigma, T \rangle$ is created in an intentional plan $P = \langle S, B, O, L, I \rangle$, it must then be populated by including steps from S in T. In a complete intentional plan, T contains all the steps which the actor takes in order to achieve the frame's goal. While IPOCL and IPOCL* create frames differently, both planners populate frames in the same way.

When a new causal link is created, it is possible that it will link a step outside an intention frame (a step not in T) to a step inside an intention frame (a step 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.

Definition 22. An **intent flaw** occurs when a causal links $s \xrightarrow{p} u$ is created such that, for some intention frame $r = \langle a, g, m, \sigma, T \rangle$, $s \notin T$, $u \in T$, and a is one of the actors which must consent to s. It can be described as a 2-tuple $\langle s, f \rangle$, where s is the step which may need to be included in frame f.

Intent flaws are solved by either including the outside step in the frame or ignoring the flaw. Ignoring the flaw, however, creates an *orphan*.

Definition 23. When a step requires consent from some actor, but that step is not yet a member of any intention frames for that actor, then the actor is said to be an **orphan**.

The presence of an orphan in a plan indicates that some step is not yet fully intentional. In other words, it is not clear why the orphan actor is choosing to take that step. A plan with one or more orphans is not a complete intentional plan.

However, the presence of an orphan is not a flaw because it cannot be directly repaired. In other words, IPOCL* has no means of directly eliminating an orphan from a plan. Nevertheless, a complete plan must be free of orphans. This might seem to imply that any time an orphan is created the algorithm must backtrack, but that is not the case. Orphans can be removed later while repairing other intent flaws. When IPOCL* chooses to ignore an intent flaw and create an orphan, the hope is that another intent flaw will arise later for that same step which can be solved by including the step in another frame of intention.

It should be noted that IPOCL* does not reason about contradictory intentions. IPOCL defines a flaw called an *intentional threat*, which occurs when a character simultaneously holds contradictory intentions, and it resolves these by ordering the frames so that they do not occur simultaneously. This same mechanism could be used in IPOCL*. It was excluded because it rules out potentially interesting stories which contain internal conflict, the phenomenon of a character being in conflict with itself.

IPOCL* (Algorithm 2) represents progress toward a narrative planner, however it will fail unless every character's intentions can be satisfied. In most stories, at least one subplan will fail, and IPOCL* cannot support this.

4.2.3 The CPOCL Algorithm

CPOCL, a conflict planning algorithm which builds on IPOCL*, does not require the addition of any new flaw types (though one must be redefined). Conflicts revolve around threatened causal links, and these are already reasoned about in POCL.

The important addition to CPOCL is that each step in a CPOCL plan is marked as executed or non-executed. Non-executed steps are ones which some actor intended to carry out but could not due to interference from other steps in the plan. Because non-executed steps do not occur, they cannot causally contribute to executed steps. Though they do not affect the world, they are important to the story because they embody information about the plans of intentional agents.

When a step is added to a plan, it is initially marked as non-executed. If ever a causal link forms from a non-executed step to an executed step, the tail step and all its causal ancestors must then be marked as executed. Note that by using this method, the complete plan will have only those steps marked as executed that must occur to achieve the goal. This is in keeping with the *least commitment* paradigm of POCL and its descendants. Some non-executed steps in the plan may also be executable, and any system using CPOCL plans is free to test for this and mark them as such.

Recall that the presence of non-executed steps also affects the definition of a *threatened causal link*. Specifically, a certain subset of threatened causal links are narrative conflicts, and thus no longer considered flaws.

A **conflict** in a CPOCL plan $P = \langle S, B, O, L, I \rangle$ occurs just when:

- There exists as causal link $s \xrightarrow{p} u \in L$.
- There exists a step $t \in S$ with effect $\neg p$.
- $s < t < u$ is a valid ordering given the constraints in O.
- u and t belong to different intention frames.
- Either u or t (or both) is non-executed.

Indeed, a conflict is a threat to a subplan by definition, however it is a threat that does not affect the causal soundness of the story as a whole.

CPOCL (Algorithm 3) is very similar to IPOCL*. A complete CPOCL domain, problem, and plan can be seen in Appendix A, along with a trace of the decisions made at each branch of the search.

4.2.4 Comparing POCL, IPOCL*, and CPOCL

Appendix A demonstrates one story that CPOCL can produce, but it is perhaps more important to consider what can be said about *all* the plans which CPOCL produces.

For this discussion, we will consider only the executed steps of a story—in other words, only the steps which would actually get narrated when telling the story to an audience. By definition, only CPOCL can produce stories with non-executed steps. These provide helpful information about character plans and alternate paths that the narrative might take, but since all steps in POCL and IPOCL* plans are executed steps, the comparison will be more fair if we define a story to be only the executed steps.

POCL can be used to produce a wide variety of stories, but many of them would contain steps which are not clearly motivated by character goals (e.g. the hero and villain might work together for no apparent reason). The plans IPOCL* produces are a subset of those produced by POCL—an important subset in which characters are seen to act on their own goals. IPOCL* narrows the solution space of POCL to stories which more closely meet audience expectations.

However, IPOCL* can only produce a solution when every character goal can be satisfied. This is rarely the case in fictional narratives. The space of CPOCL solutions is also a subset of POCL, but it is a *superset* of IPOCL*. Specifically, it is the set of stories in which characters act on their own goals but not every character's subplan has to succeed. Each character has goals, makes plans for their goals, and works to achieve their goals, but some of them may fail in order bring the story to its desired end state. This is why CPOCL is said to be a narrative planner which supports conflict.

Algorithm 3 The CPOCL (Conflict Partial Order Causal Link) Planning Algorithm

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

Where P is a partial plan, F a set of flaws, and Λ a set of operators.
Initially, P is the null plan (with both the start and end steps marked as executed), F contains an open precondition flaw for each precondition of the end step. If the start step contains any effects like `(intends a g)`, then I contains intention frames for those effects and F contains unsatisfied intention frame flaws for those frames.

1. **Termination:** If B or O is inconsistent, fail. If $F = \emptyset$ and P contains no orphans, return P. Else, if $F = \emptyset$, fail.

2. **Plan Refinement:** Non-deterministically choose a flaw f from F. Let $F' = F - \{f\}$.

   (a) **Goal Planning:** If f is an open precondition flaw $f = \langle s_{need}, p \rangle$, then let s_{add} be some step with an effect p. Non-deterministically choose how to create s_{add}:

      i. **Reuse:** Choose s_{add} from the steps already in S.

      ii. **New Step:** Create a new non-executed step s_{add} from Λ which has an effect p. Let $S' = S + \{s_{add}\}$. For each precondition c of s_{add}, add a new open precondition flaw $\langle s, c \rangle$ to F'. For each effect of s_{add} like `(intends a g)`, add a new intention frame $r = \langle a, g, s_{add}, \emptyset, \emptyset \rangle$ and add a new unsatisfied intention frame flaw $\langle r \rangle$ to F'.

   Create a new causal link $l = s_{add} \xrightarrow{p} s_{need}$. Let $L' = L + \{l\}$. Let $B' = B \cup \text{MGU}(e, p)$. Let $O' = O + \{s_{add} < s_{need}\}$.
   For every intention frame $r = \langle a, g, \sigma, m, T \rangle \in I$, if $s_{add} \notin T$ and $s_{need} \in T$ and $a \in A$ for s_{add}, then add a new intent flaw $\langle s_{add}, r \rangle$ to F'.
   If s_{need} is executed, make s_{add} and all its causal ancestors executed.

   (b) **Threat Resolution:** If f is a threatened causal link flaw $f = \left\langle s \xrightarrow{p} u, t \right\rangle$, then non-deterministically choose a way to prevent the threat:

      i. **Promotion:** Let $O' = O' + \{t < s\}$.

      ii. **Demotion:** Let $O' = O' + \{u < t\}$.

      iii. **Restriction:** Add bindings to B' which cause the threatening effect of t not to unify with p.

   (c) **Satisfaction Planning:** If f is an unsatisfied intention frame flaw $f = \langle r = \langle a, g, m, \emptyset, T \rangle \rangle$, then let s_{sat} be some step with an effect g. Non-deterministically choose s_{sat} the same way that s_{add} is chosen. Let $T' = T + \{s_{sat}\}$, and let $r' = \langle a, g, m, s_{sat}, T' \rangle$. Let $I' = I - \{r\} + \{r'\}$.

   (d) **Intent Planning:** If f is an intent flaw $f = \langle s_{orphan}, r = \langle a, g, m, \sigma, T \rangle \rangle$, then non-deterministically choose how to handle s_{orphan}:

      i. **Inclusion:** Let $T' = T + \{s_{orphan}\}$, let $r' = \langle a, g, m, \sigma, T' \rangle$, let $I' = I - \{r\} + \{r'\}$, and let $O' = O + \{m < s_{orphan}\}$.
      For each causal link $s \xrightarrow{p} s_{orphan}$ in L, if $a \in A$ for s, add a new intent flaw $\langle s, r' \rangle$ to F'.

      ii. **Exclusion:** Do nothing.

3. **Threat Detection:** If any casual link $l \in L'$ has become threatened by some step $s_{threat} \in S'$, add a new threatened causal link flaw $\langle l, s_{threat} \rangle$ to F'.

4. **Recursive Invocation:** Call **CPOCL** ($P' = \langle S', B', O', L', I' \rangle$, F', Λ).

5 Conclusions

5.1 Limitations of CPOCL

The primary limitation of CPOCL is that it only *allows* conflict rather than explicitly creating it. If the characters in the story can achieve their goals without coming into conflict, CPOCL may return a plan devoid of conflict. If conflict is considered an essential element of stories, this deficiency limits the power of CPOCL as a narrative planner.

This limitation exists because a refinement planner is a means-ends analysis tool. Elements are only added to the plan to satisfy some need. Introducing conflict when there is no need for it would require a means of reasoning about which new goals a character would need to adopt in order to form a subplan that will cause conflict with some existing subplan. This solution might look something like Smith and Witten's adversarial planning approach, in which the antagonist works against the protagonist without a clear motivation [42]. Also, adding a step for no reason other than creating conflict can lead to *dead ends* in the plan, or steps whose effects do not contribute causally to any of the plan goals. Dead ends are undesirable in stories [25].

Narrative characters are expected to act believably [16], so forcing characters to go out of their way to create conflict when none need exist may violate the expectations of the audience. One alternative solution to ensure that conflict arises might be to modify the planning problem. Many narrative systems are realized in virtual worlds over which the author has a great deal of control. Since the initial state and goal state are just another part of the story, narrative systems can modify them to be more conducive to ideal plot structures [49]. Research is already underway on initial and goal state revision algorithms [36, 49].

Another weakness of CPOCL is that its search is not guided by the seven dimensions identified in section 3.3. Ideally, CPOCL would take as input not only the planning problem but also a list of desired conflicts with constraints on the values of their seven dimensions. Users could specify constraints like "there must exist a conflict with intensity higher than 0.75." Guiding the search based on dimension values is particularly difficult in a least-commitment planner because the world state before and after a step is not known until the plan is complete, and even then it will differ based on which total ordering is chosen.

This limitation can be addressed by building on a different planner such as HSP, in which the current state and all previous states are always known, or GraphPlan which provides much more information about current and future states. Because HSP and GraphPlan are significantly faster than POCL algorithms, this will also make this research more usable in a real world system.

Lastly, CPOCL does not reason about when an actor should attempt to replan. This is not an issue in a traditional planner because either no plan exists or any plan returned is guaranteed to succeed. In CPOCL, a character's subplan might fail. Sometimes this will cause that character to adopt a new subplan with the same goal, and sometimes not. Characters cannot simply replan every time they fail or the story may never end. Also, when replanning happens, characters should not simply form the same subplan again, since it is likely to fail in exactly the same way. However, the planner does not understand this, so it is likely to generate the same subplan over and over. CPOCL should be extended to support replanning in some believable way that still brings the story to and end in a timely manner. Research into this problem is also underway [14].

5.2 Future Work

Conflict is an essential element of interesting stories, but little work has been done to model it formally. Until such a model exists, mechanical systems cannot directly reason about conflict in narrative, and this is a significant deficiency for a story generating system that wishes to leverage the engagement and structure that conflict provides.

In this document, I compiled such a model from several narratological sources. Conflict occurs when a goal-seeking agent forms a plan which can be thwarted by another force (usually another goal seeking agent, but possibly the agent itself or the environment). Each conflict also has at least seven dimensions which help to classify it: participants, subject, duration, directness, intensity, balance, and resolution. I provided some very simply heuristics for estimating these dimensions.

Since this definition of conflict is based on planning, and since planning is a good basis for modeling stories, I described a story planner called CPOCL which allows conflict to arise. Traditional planners seek to remove all conflicts because of the causal inconsistencies they create. However, when the planner is extended to reason about character intentions and failed plans, I demonstrated that certain conflicts can be considered narrative conflicts. These can be preserved without damaging the plan.

This work is preliminary in many ways. I believe it is valuable because it establishes, for the first time, a robust formal model of conflict and a method for generating stories based on that model. Like any excursion into a new research problem, there are numerous features that need to be improved:

- The model may need further refinement. The current list of seven dimensions may need to be expanded or contracted. If not, some evidence should be provided that it is complete and that each dimension measures a distinct phenomenon. If synergy exists between dimensions, this should be identified.
- The heuristics for measuring the dimensions of directness, intensity, balance, and resolution can all be improved by research from economics, agent modeling, psychology, etc.
- This model is the first to measure conflict as anything more than a single value, but additional granularity is always helpful. For example, a more detailed analysis of the different classes of resolution and how each one affects the narrative would be helpful.
- Some method needs to be devised which ensures that a generated story contains conflict if any is possible. This might be solved directly by algorithms which explicitly introduce conflict, or it might be solved indirectly by engineering the initial and goal state of the story such that conflict must arise.
- Story generation should reason about the dimensions of conflict as the story is being constructed. Related to this is a need to describe, in formal terms, the structure that conflict should take in certain situations. Should a story be dominated by one high-duration conflict that is punctuated by many shorter conflicts? Should intensity increase or decrease as the story progresses? It is likely that no perfect answer exists to these questions, but insights from narratology and screen writing can provide guidance in specific situations, perhaps on a per-genre basis.
- Story planners need some way of detecting when an agent who fails needs to replan. The newly formed plan should not simply be a repeat of the previous plan.
- It may be helpful to group conflicts based on similar characteristics or goals. For example, I represent Prince Humperdinck's two plans to start a war as two distinct conflicts. Perhaps these should be considered one conflict, or perhaps a hierarchy should be established to convey the relationship between them.

Despite these open questions, I hope that my model and algorithm will provide a useful foundation for conflict-based story generation. The ability to reason about and manipulate this important feature of narrative will increase the ability of story generating systems to meet the expectations of human audiences.

A Example CPOCL Plan: Bob's Broken Heart

To illustrate how CPOCL works, I present a planning problem and trace through the algorithm as it reaches a solution. We will assume that each non-deterministic choice is made correctly to avoid describing a massive search space.

A.1 Sample Domain: Love World

The sample domain and problem are described in PDDL, the Planning Domain Definition Language which has been established as a defacto knowledge representation in the field of planning. Support for modal predicates and intentional planning are not standard in PDDL, so I have attempted to extend the language in a reasonable manner.

This domain defines three actions:

- `(fall-in-love ?lover ?target)` causes `?lover` to desire marriage to `?target`. Note that it is a *happening*, an action which does not require consent from any actors.
- `(woo ?wooer ?target)` represents how `?wooer` will make `?target` fall in love with him. It is initiated by the `?wooer` and thus requires only his consent.
- `(marry ?groom ?bride)` represents the union of two lovers, and requires the consent of both.

```
(define (domain love-world)
  (:requirements :strips :intentionality)
  (:predicates (likes ?liker ?target)
               (loves ?lover ?target)
               (married ?husband ?wife))
  (:modalities (intends ?someone ?something))
  (:action fall-in-love
    :parameters   (?lover ?target)
    :actors       ()
    :precondition ()
    :effect       (and (loves ?lover ?target)
                       (intends ?lover (married ?lover ?target))))
  (:action woo
    :parameters   (?wooer ?target)
    :actors       (?wooer)
    :precondition (and (loves ?wooer ?target)
                       (likes ?target ?wooer))
    :effect       (and (loves ?target ?wooer)
                       (intends ?target (married ?wooer ?target))))
  (:action marry
    :parameters   (?groom ?bride)
    :actors       (?groom ?bride)
    :precondition (and (loves ?groom ?bride)
                       (loves ?bride ?groom)
                       (single ?groom)
                       (single ?bride))
    :effect       (and (married ?groom ?bride)
                       (not (single ?groom))
                       (not (single ?bride)))))
```

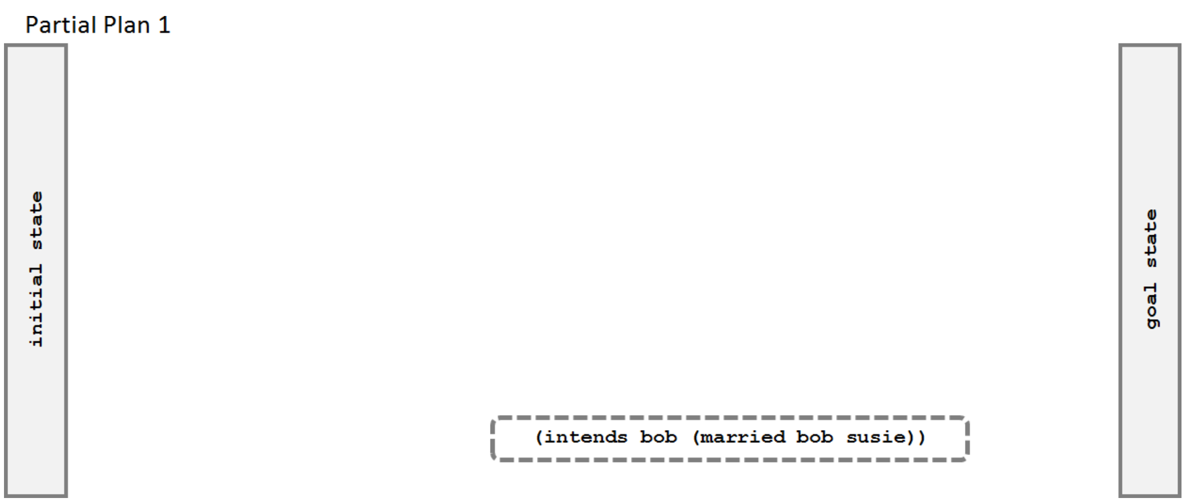
A.2 Sample Problem: Bob's Broken Heart

The problem introduces three characters: Bob, Joe, and Susie. All three consider the others to be friends, but Bob is in love with Susie and wishes to marry her. The author goals, however, require that Joe be wed to Susie. This situation is ripe for conflict!

```
(define (problem broken-heart)
  (:domain love-world)
  (:init (likes bob joe)
         (likes bob susie)
         (likes joe bob)
         (likes joe susie)
         (likes susie bob)
         (likes susie joe)
         (single bob)
         (single joe)
         (single susie)
         (rich joe)
         (loves bob susie)
         (intends bob (married bob susie)))
  (:goal (married joe susie)))
```

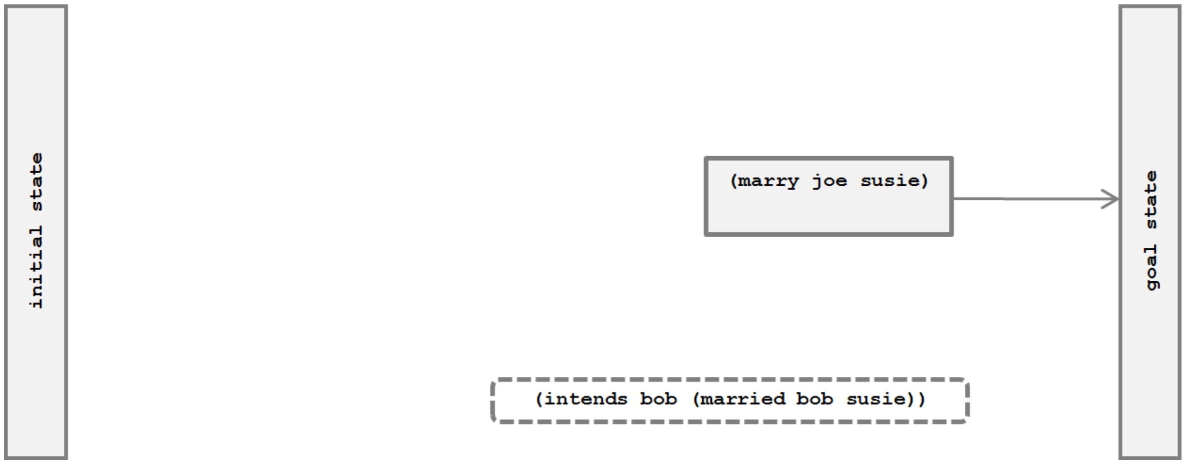
A.3 CPOCL Trace

1. CPOCL begins by constructing the null plan for this problem: a start step with effects equivalent to the `initial` state of the problem, and a goal step with preconditions equivalent to the `goal` of the problem. Each literal of the goal state has a corresponding open precondition flaw; in this case there is only one: `(married joe susie)`. The plan begins with one intention frame motivated by `initial` which must describe how `bob` will achieve `(married bob susie)`. There is also one unsatisfied intention frame flaw for that intention frame, so 2 flaws in total in the null plan.



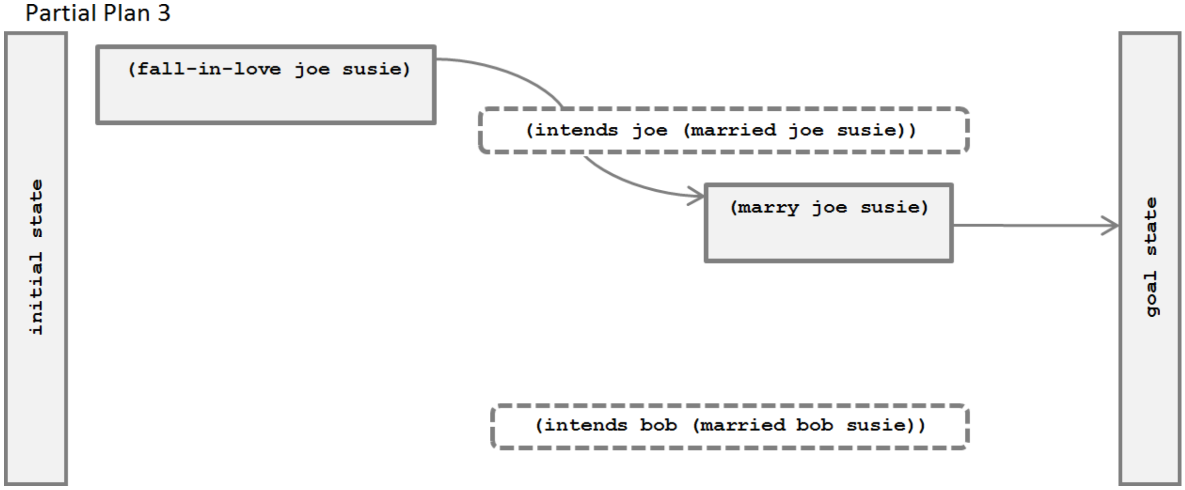
2. CPOCL chooses to repair the open precondition `(married joe susie)` in `goal`. It creates a new non-executed step from the `marry` operator: `(marry joe susie)` to be s_{add}. Four new open precondition flaws are generated for `(marry joe susie)`: `(loves joe susie)`, `(loves susie joe)`, `(single joe)`, and `(single susie)`. A new causal link is added from `(marry joe susie)` to `goal` with the label `(married joe susie)`. Because `goal` is an executed step, `(marry joe susie)` must also be marked as executed. No intention frames need to be created and no threats have occurred, but note that all the actors who must consent to `marry` (actors `joe` and `susie`) are currently orphans. CPOCL is invoked recursively.

Partial Plan 2



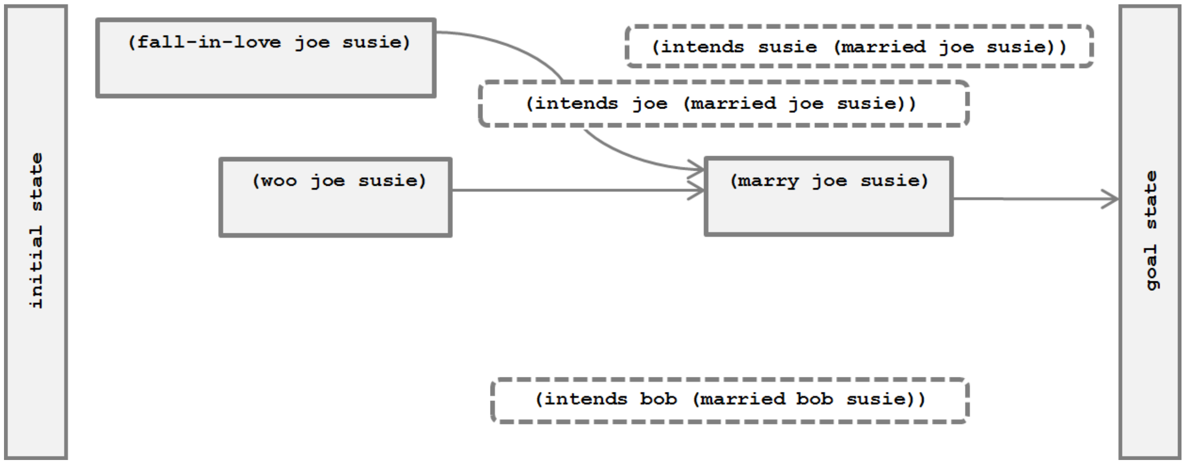
3. CPOCL chooses to repair the open precondition `(loves joe susie)` in `(marry joe susie)` by creating a new step from the `fall-in-love` operator: `(fall-in-love joe susie)`. CPOCL creates a causal link from `(fall-in-love joe susie)` to `(marry joe susie)` with label `(loves joe susie)`. This step has no preconditions, so no new open precondition flaws are generated. `(fall-in-love joe susie)` is marked as executed. Because `(fall-in-love joe susie)` has the effect `(intends joe (married joe susie))`, a new intention frame is created for joe which must explain how he will achieve `(married joe susie)`. A new unsatisfied intention frame flaw for this new frame is added, for a total of 3 remaining flaws.

Partial Plan 3



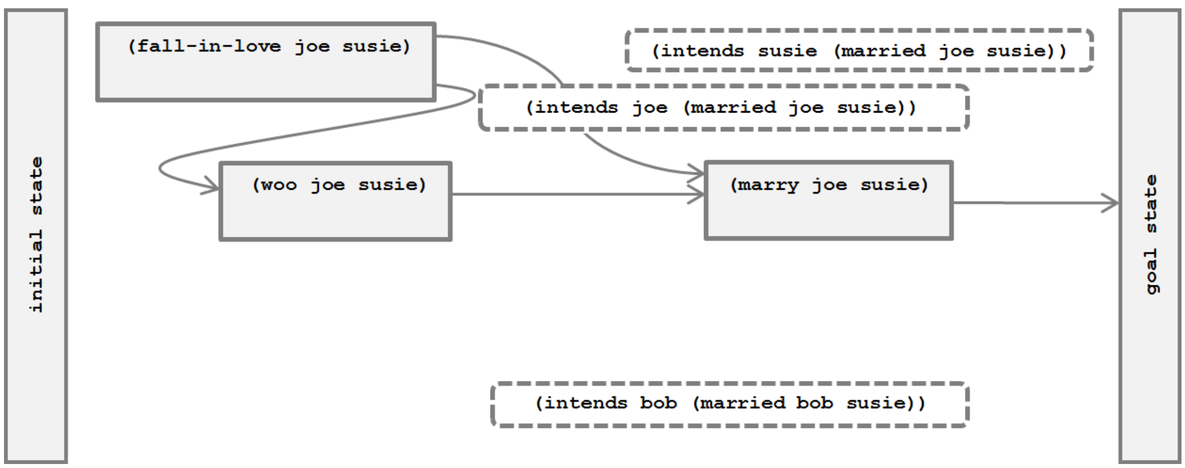
4. The open precondition `(loves susie joe)` is satisfied by a causal link from a new step: `(woo joe susie)`. It gets marked as executed and creates a new intention frame to describe how `susie` will achieve `(married joe susie)`. It also adds an orphan to the plan. Appropriate open precondition and unsatisfied intention frame flaws are added.

Partial Plan 4



5. The open precondition `(loves joe susie)` in the step `(woo joe susie)` is satisfied with a causal link from the existing step `(fall-in-love joe susie)`.

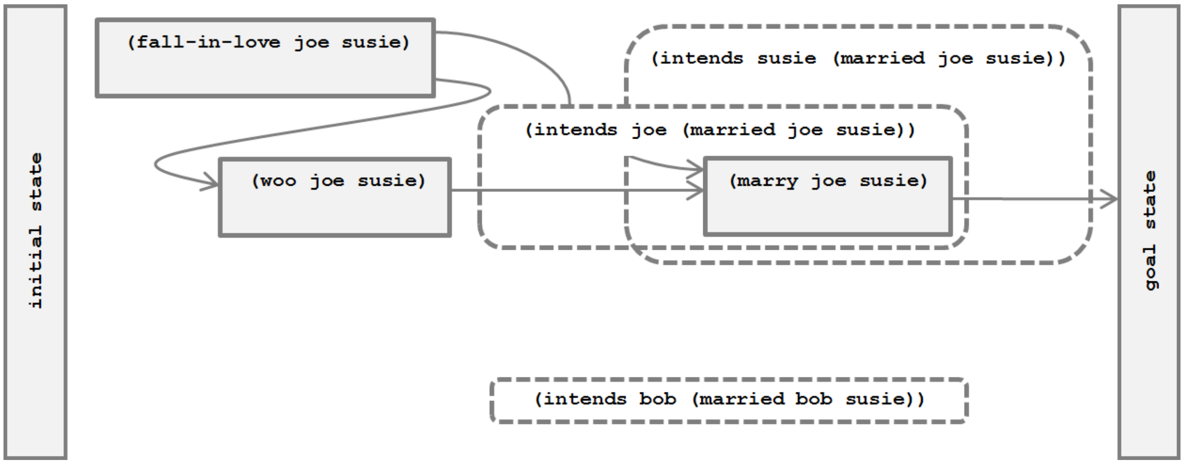
Partial Plan 5



6. The frame that explains how `joe` will achieve `(married joe susie)` is satisfied by the existing step `(marry joe susie)`. This removes an orphan, because now it is clear why `joe` consented to carry out the `marry` action. However, this also causes an intent flaw to arise: since `(woo joe susie)` has a causal link leading to `(marry joe susie)`, we must consider adding `(woo joe susie)` to `joe`'s intention frame as well.

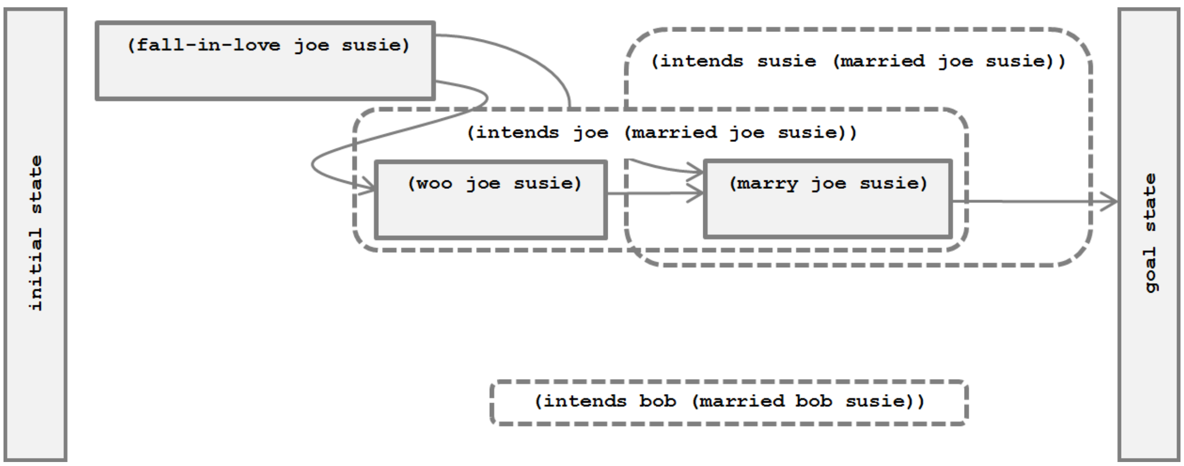
7. The step `(marry joe susie)` also satisfies the frame that explains how susie will achieve `(married joe susie)` and removes another orphan.

Partial Plan 7



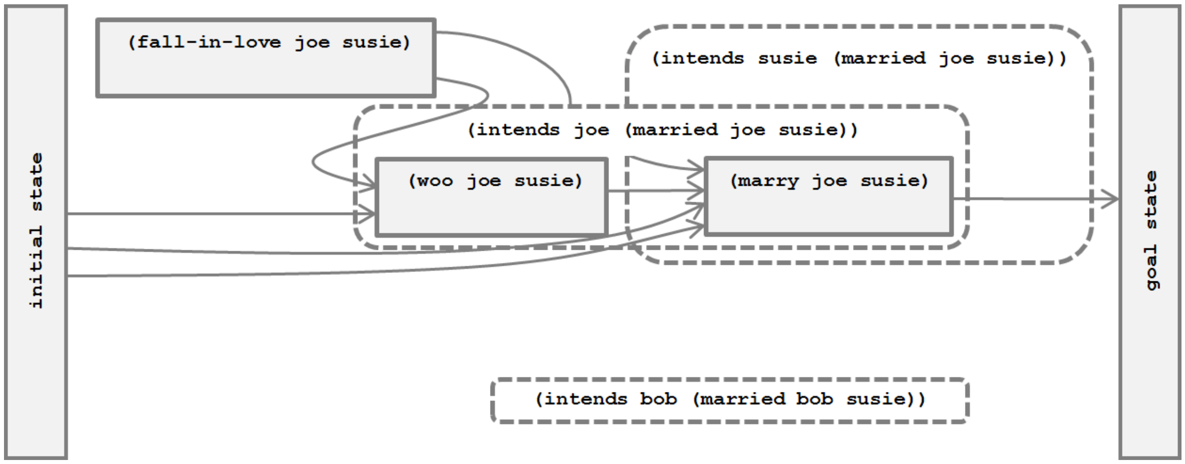
8. CPOCL chooses to add `(woo joe susie)` to `joe`'s intention frame, resolving the intent flaw and making `joe` in `(woo joe susie)` no longer an orphan. At this point, every step is fully intentional—that is, the plan has no orphans.

Partial Plan 8



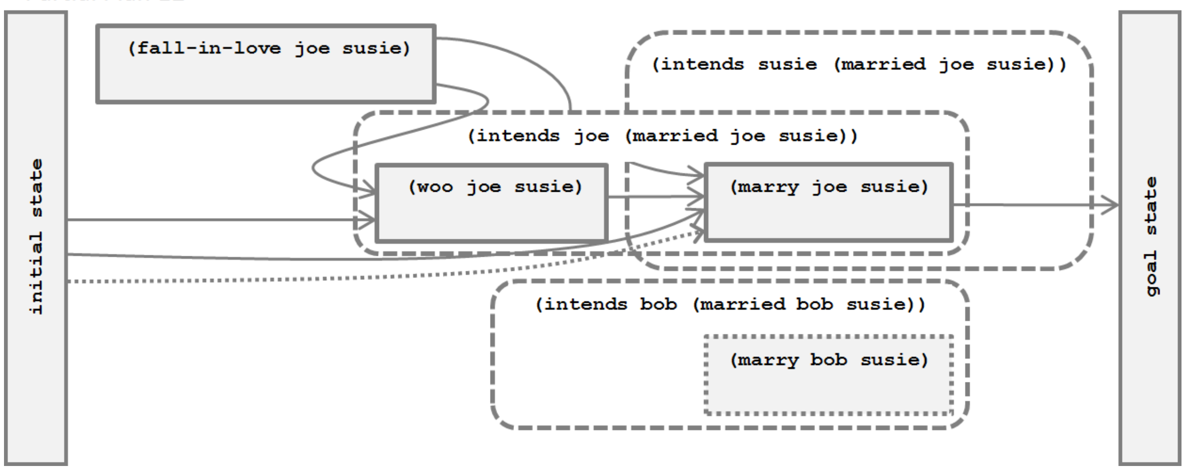
9. A causal link from `initial` to `(woo joe susie)` satisfies the open precondition `(likes susie joe)`.
10. A causal link from `initial` to `(marry joe susie)` satisfies the open precondition `(single joe)`.
11. A causal link from `initial` to `(marry joe susie)` satisfies the open precondition `(single susie)`.

Partial Plan 11



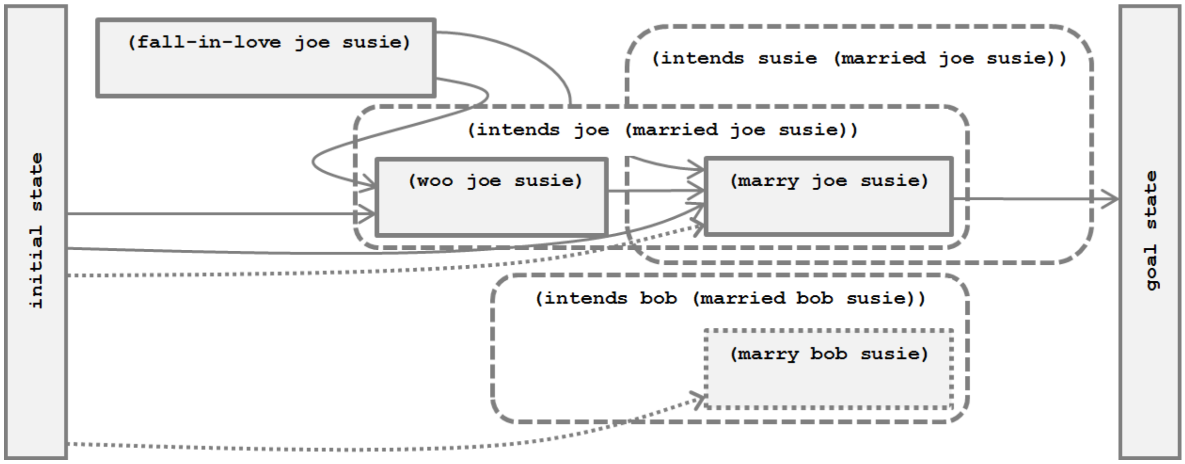
12. One flaw remains: the unsatisfied intention frame that explains how `bob` will achieve `(married bob susie)`. It gets satisfied by a new step `(marry bob susie)`. Relevant open precondition flaws are added. Because no casual links lead from this step to an executed step, it remains non-executed. The addition of this step also causes a threat to arise. The effect `(not (single susie))` threatens the causal link going from `initial` to `(marry joe susie)` with label `(single susie)`. Because `(marry bob susie)` is a non-executed step, this threat is a conflict and does not result in an threatened causal link flaw.

Partial Plan 12



13. CPOCL chooses to repair the open precondition flaw for `(single susie)` in `(marry bob susie)` with a new causal link from the `initial` step. This new link is threatened by the step `(marry joe susie)` which has the effect `(not (single susie))`. Again, this threat is a conflict because `(marry bob susie)` is non-executed.

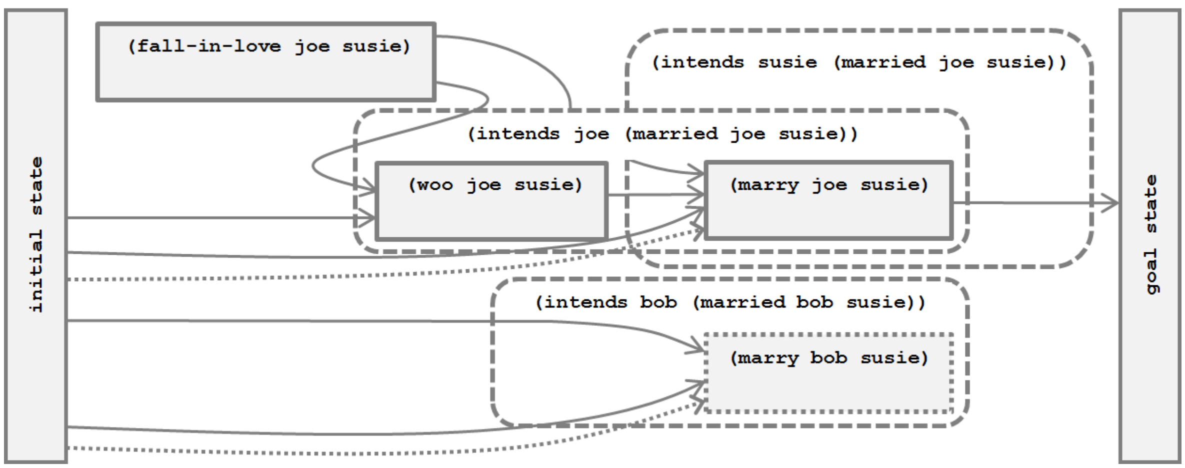
Partial Plan 13



14. A causal link from `initial` to `(marry bob susie)` satisfies the open precondition `(loves bob susie)`.

15. A causal link from `initial` to `(marry bob susie)` satisfies the open precondition `(single bob)`.

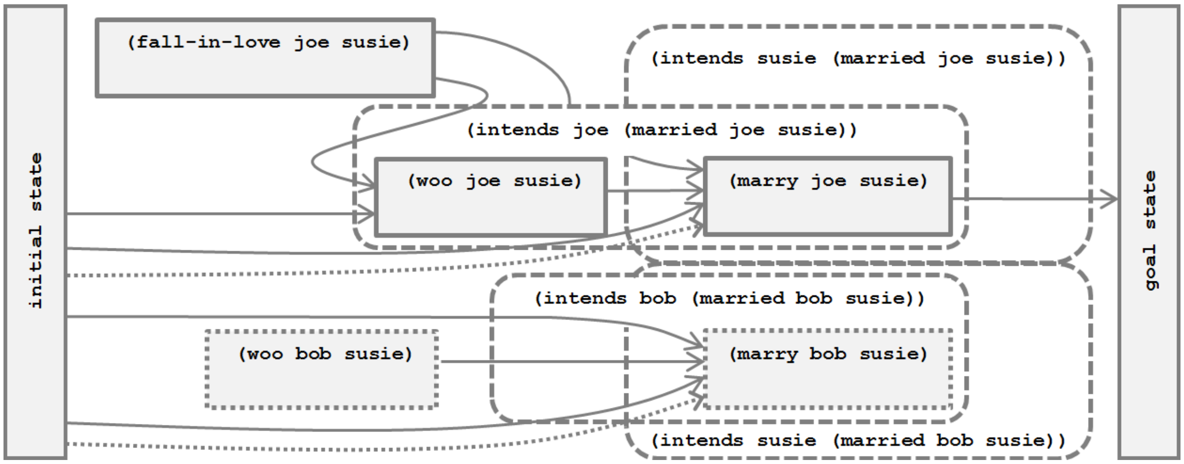
Partial Plan 15



16. The open precondition `(loves susie bob)` is satisfied with a causal link from a new step `(woo bob susie)`. This new causal link creates an intent flaw for the step `(woo bob susie)` and the frame which describes how bob will achieve `(married bob susie)`. Because the new step only has causal links leading to non-executed steps, it remains non-executed. Adding this step creates a new intention frame to explain how `susie` will achieve `(married bob susie)`.

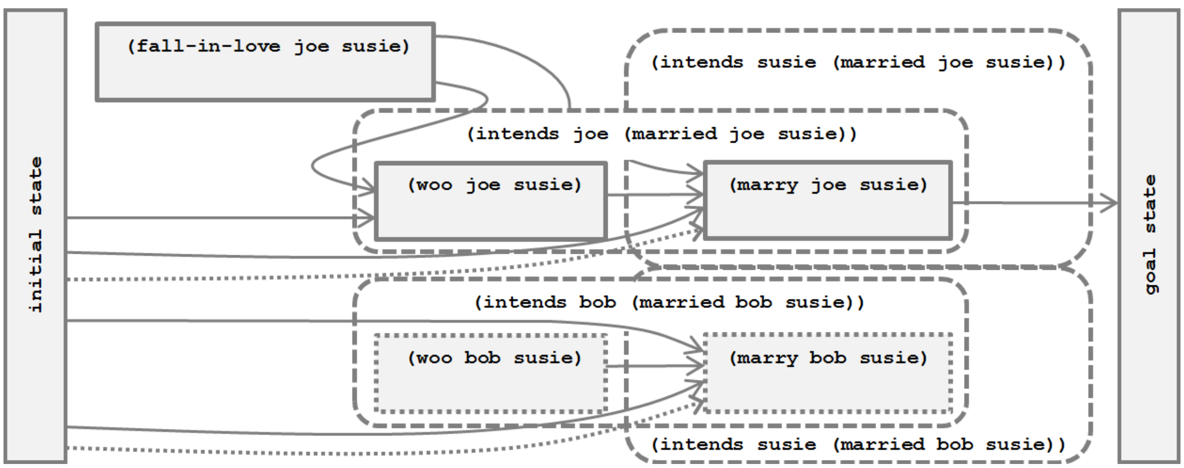
17. This new frame is satisfied by the existing step `(marry bob susie)`.

Partial Plan 17



18. CPOCL chooses to include (`woo bob susie`) in `bob`'s intention frame to achieve (`married bob susie`), which removes an orphan.

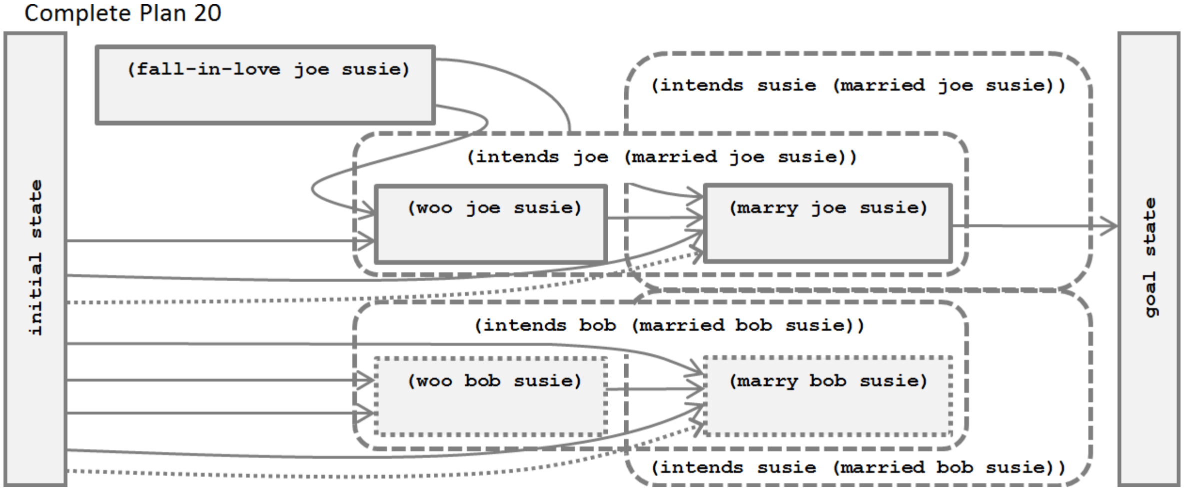
Partial Plan 18



19. A causal link from `initial` to (`woo bob susie`) satisfies the open precondition (`loves bob susie`).

20. A causal link from `initial` to (`woo bob susie`) satisfies the open precondition (`likes susie bob`).

Complete Plan 20



21. The plan has no flaws remaining and no orphans, so it is returned as a solution.

A.4 Solution

The solution plan is included below as output by CPOCL, followed by a description of each part.

```
(define (plan broken-heart-solution)
  (:problem broken-heart)
  (:steps (0 (init) t)
          (6 (woo bob susie) nil)
          (3 (fall-in-love joe susie) t)
          (4 (woo joe susie) t)
          (5 (marry bob susie) nil)
          (2 (marry joe susie) t)
          (1 (goal) t))
  (:orderings (0 1) (2 1) (3 1) (2 3)
              (4 1) (4 2) (3 4) (5 1)
              (6 1) (6 5))
  (:clinks (2 (married joe susie) 1)
           (3 (loves joe susie)  2)
           (4 (loves susie joe) 2)
           (3 (loves joe susie) 4)
           (0 (single joe) 2)
           (0 (single susie) 2)
           (0 (likes susie joe) 4)
           (0 (loves bob susie) 5)
           (0 (single bob) 5)
           (0 (single susie) 5)
           (6 (loves susie bob) 5)
           (0 (loves bob susie) 6)
           (0 (likes susie bob) 6))
  (:iframes ((intends bob (married bob susie)) 0 (6 5))
            ((intends joe (married joe susie)) 3 (4 2))
            ((intends susie (married joe susie)) 4 (2))
            ((intends susie (married bob susie)) 6 (5)))
  (:conflicts ((0 (single susie) 2) 5))
              ((0 (single susie) 5) 2))
```

The above plan is a sample of the output from my system. Each part is described below:

- The `:steps` section lists the steps of the plan in an executable total order. Each step begins with an ID number, followed by its name and parameter values, followed by `t` if it is an executed step or `nil` if it is non-executed.
- The `:orderings` section is a set of tuples that each name two steps. The first step must be executed before the second.
- The `:clinks` section lists the causal links in the plan. Each link begins with the ID of its tail step, followed by its label, followed by the ID of its head step.
- The `:iframes` section lists the intention frames in the plan. Each frame begins with an `intends` predicate that names the actor and goal of the frame, followed by the ID of the motivating step, followed by a list of steps which belong to the frame. The last step in the list is always the satisfying step that achieves the goal.
- The `:conflicts` section lists the conflicts in the plan. Each conflict is composed of a causal link followed by the step which threatens the link.

The conflicts section lists only two items because there are two threats. However, in both cases the threatening step has two consenting actors. This means that there are actually 8 conflicts: Bob vs. Joe, Joe vs. Bob, Bob vs. Susie, Susie vs. Bob, Joe vs. Susie, Susie vs. Joe, Susie vs. Susie, and Susie vs. Susie. Note how the conflicts arise in complementary pairs: Bob's plan to marry Susie threatens Joe's plan to marry Susie, and vice versa. Susie's plans threaten each other, which is a nice example of internal conflict.

When CPOCL returns a plan, only steps which *must* be executed are marked as such. However, many systems will find it helpful to execute *every* step that can be executed. This makes characters appear more like goal-oriented agents. Step 6, in which Bob woos Susie, is marked as non-executed because it is not needed to achieve the story goals, but all of its preconditions are met, so it can be carried out.

Assuming that step 6 is marked as executed, the plan might be translated into natural language like so:

> *Once upon a time there lived three friends: Bob, Joe, and Susie. Joe was rich. Bob loved Susie and planned to marry her. Bob wooed Susie with flowers and love songs. However, Joe fell in love with Susie and also attempted to woo her. In the end, Susie married Joe.*

This simple story is important because it demonstrates the abilities of CPOCL as a narrative planner. This example problem can only be solved by POCL if character intentionality is ignored. It cannot be solved by IPOCL* because there is no way to achieve both `(married bob susie)` and `(married joe susie)`. Only CPOCL is able to maintain causal soundness and character believability while still producing a solution, thanks to its ability to handle conflict.

A.5 Analysis

If we establish some very simple functions for calculating the necessary values within this domain, we can analyze two of the above conflicts based their seven dimensions.

Let us define the closeness of two actors based on emotional distance.

```
function closeness(actor a, actor a)
    closeness = 0;
    if (likes a b) closeness += 0.5;
    if (loves a b) closeness += 0.5;
```

Let us define *utility* based on wealth, social status, and relationship status (considering unrequited love).

```
function utility(actor a)
    utility = 0.4;
    if (rich a) utility += 0.2;
    if (exists actor b such that (likes b a)) utility += 0.2;
    if (exists actor b such that (loves a b) and (married a b)) utility += 0.2;
    if (exists actor b such that (loves a b) and (not (married a b))) utility -= 0.4;
```

Let us define *likelihood of success* such that wealth makes most actions easier.

```
function liklihood-of-success(step s)
    if (step like (fall-in-love a)) liklihood = 1;
    if (step like (woo a b)) {
        if (rich a) liklihood = 0.8;
        else liklihood = 0.5;
    }
    if (step like (marry a b)) {
        if (rich a) or (rich b) liklihood = 1;
        else liklihood = 0.5;
    }
```

Based on these functions and the formulas presented in section 3.3, we get the following values for each dimension:

Dimension	**Parameters**	**Value**	**Parameters**	**Value**
participants		`bob` vs. `joe`		`bob` vs. `susie`
subject		`(single susie)`		`(single susie)`
duration		2		1
i_directness	bob to joe	0.5	bob to susie	1
	joe to bob	0.5	susie to bob	1
o_directness		0.5	overall	1
i_intensity	for bob	0.6	for bob	0.6
	for joe	0.6	for susie	0.2
o_intensity		0.6	overall	0.4
i_balance	for bob	0.238	for bob	0.2
	for joe	0.762	for susie	0.8
o_balance		0.476	overall	0.4
i_resolution	for bob	+0	for bob	+0
	for joe	+0.2	for susie	+0.2
o_resolution		+0.1	overall	+0.1

Several observations can be made about the story based on these dimensions:

- The conflict between Bob and Joe has a longer **duration**.
- The conflict between Bob and Susie is more **direct** because they are lovers rather than just friends.
- The conflict between Bob and Joe is more **intense** because both men have a lot to gain and a lot to lose. The conflict between Bob and Susie has a lower intensity because, no matter what, Susie is going to end up with a low utility.
- In both cases, the odds are stacked against Bob because he is not rich, but the **balance** is slightly closer to 1 in the conflict with Joe because even Joe's wealth cannot guarantee that he will win Susie's affections.
- The **resolution** of both conflicts has the same value because Bob's utility never changes and both Joe and Susie experience a slight gain in utility as a result of their plans.

References

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

[2] B.C. Bae and R.M. 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, 2008.

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

[4] H. Barber and D. Kudenko. Dynamic generation of dilemma-based interactive narratives. In *Proceedings of Artificial Intelligence and Interactive Digital Entertainment (AIIDE)*, 2007.

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

[6] J. Blythe. An overview of planning under uncertainty. *Artificial Intelligence Today*, pages 85–110.

[7] B. Bonet and H. Geffner. Planning as heuristic search: New results. *Lecture Notes in Computer Science*, pages 360–372, 2000.

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

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

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

[11] D.B. Christian and R.M. Young. Comparing cognitive and computational models of narrative structure. In *Proceedings of the National Conference of the American Association for Artificial Intelligence*, pages 385–390, 2004.

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

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

[14] M.W. Fendt. Dynamic social planning and intention revision in generative story planning. In *Proceedings of Foundations of Digital Games*, pages 254–255, 2010.

[15] 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.

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

[17] A.C. Graesser, K.L. Lang, and R.M. Roberts. Question answering in the context of stories. *Journal of Experimental Psychology: General*, 120(3):254–277, 1991.

[18] M. Helmert. New complexity results for classical planning benchmarks. In *Proceedings of the International Conference on Automated Planning and Scheduling*, pages 52–61, 2006.

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

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

[21] 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.

[22] 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.

[23] H. Kautz and B. Selman. Planning as satisfiability. In *Proceedings of the European Conference on Artificial Intelligence*, pages 359–363, 1992.

[24] M. Lebowitz. Story-telling as planning and learning. *Poetics*, 14(6), 1985.

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

[26] M. Mateas and A. Stern. Façade: An experiment in building a fully-realized interactive drama. In *Proceedings of the Game Developers Conference*, page 82, 2003.

[27] D. McAllester and D. Rosenblitt. Systematic nonlinear planning. In *Proceedings of the National Conference of the American Association for Artificial Intelligence*, pages 634–639. Massachusetts Institute of Technology Press, 1991.

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

[29] J.R. Meehan. Tale-spin, an interactive program that writes stories. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 1977.

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

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

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

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

[34] M.O. Ridel and R.M. Young. Narrative planning: balancing plot and character, 2010.

[35] M. Riedl and A. Stern. Believable agents and intelligent story adaptation for interactive storytelling. In *Proceedings of Technologies for Interactive Digital Storytelling and Entertainment*, pages 1–12, 2006.

[36] M.O. Riedl and R.M. Young. Open-world planning for story generation. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 2005.

[37] S. Rimmon-Kenan. *Narrative fiction: Contemporary poetics*. Routledge, 2002.

[38] S. Russell and P. Norvig. *Artificial intelligence: a modern approach*. Pearson Education, 2003.

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

[40] E.D. Sacerdoti. The nonlinear nature of plans. 1975.

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

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

[43] 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.

[44] N. Szilas. IDtension: a narrative engine for interactive drama. In *Proceedings of the International Conference on Technologies for Interactive Digital Storytelling and Entertainment*, 2003.

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

[46] D. Thue, V. Bulitko, and M. Spetch. Making stories player-specific: Delayed authoring in interactive storytelling. In *Proceedings of the International Conference on Interactive Digital Storytelling (ICIDS)*, pages 230–241, 2008.

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

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

[49] S.G. Ware and R.M. Young. Rethinking traditional planning assumptions to facilitate narrative generation. In *Proceedings of the Association for the Advancement of Artificial Intelligence Fall Symposium on Computational Models of Narrative*, 2010.

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

[51] D.S. Weld. Recent advances in AI planning. *AI magazine*, 20(2):93, 1999.

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

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

[54] 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.

[55] 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, 2007.