Controlling Narrative Generation with Planning Trajectories: the Role of Constraints

Julie Porteous and Marc Cavazza

School of Computing
University of Teesside
{j.porteous}{m.o.cavazza}@tees.ac.uk

Abstract. AI planning has featured in a number of Interactive Storytelling prototypes: since narratives can be naturally modelled as a sequence of actions it has been possible to exploit state of the art planners in the task of narrative generation. However the characteristics of a "good" plan, such as optimality, aren't necessarily the same as those of a "good" narrative, where errors and convoluted sequences may offer more reader interest, so some narrative structuring is required. In our work we have looked at injecting narrative control into plan generation through the use of PDDL3.0 state trajectory constraints which enable us to express narrative control information within the planning representation. As part of this we have developed an approach to planning with such trajectory constraints. The approach decomposes the problem into a set of smaller subproblems using the temporal orderings described by the constraints and then solves these subproblems incrementally. In this paper we outline our method and present results that illustrate the potential of the approach.

1 Introduction

A key component of an Interactive Storytelling (IS) system is a narrative generator. Since narratives can be naturally modelled as a sequence of actions, AI planning has emerged as the technology of choice in this field and a range of planning approaches have been successfully applied to the task of narrative generation. For example, partial order planning in the tradition of UCPOP [1] has been used with adaptations to intentions [2] and emotion [3]; state based planning in the style of HSP [4] was used to generate plans in an authoring tool [5]; and HTN planning [6] was adapted to handle anytime user intervention [7].

As these systems demonstrate, it is clearly possible to generate narratives using planning technology but they are not without their limitations. For instance, both partial order (plan space) planning and state space planning approaches suffer from an IS perspective, because they were developed with the objective of building a planner that would find the shortest plan to a goal (or a close approximation to it). This is appropriate in the sorts of domains for which planners were first developed, such as robot stacking and logistics, but has little meaning in

the context of IS, where errors and convoluted sequences may be desirable characteristics. On the other hand, HTN planning does provide a means to include information about desirable ways for the narrative to be developed, in the form of decompositions of methods in the task network. However, because the control information is embedded in the task network it is not possible to manipulate the information independently and since this is a feature that would be useful in an IS environment, for example to respond to user interaction, it suggests some other approach to narrative structuring is required.

A review of the IS literature reveals a number of previous attempts at narrative structuring. The approach taken by some has been to use HTN planning which, as previously mentioned, permits the encoding of authors intentions of possible ways to develop the narrative, in the form of the decomposition methods themselves [7, 8]. Others have proposed the use of search based drama management: Weyhrauch [9] suggested the encoding of aesthetics in an evaluation function to be used by the drama manager; Magerko et al [10] exploited a meta-level controller in the form of a drama manager; Mateas and Stern's Façade drama [11] featured a beat based drama manager; and Riedl and Stern [12] used an automated story director to blend user agency with narrative plot control. Other researchers have looked to adapt the plan generation algorithm itself to structure the narrative along aesthetic lines. For example Riedl and Young [2], used simulated intention reasoning to guide the planner to promote story coherence and temporal consistency whilst Cheong and Young [8] demonstrated how the narrative structure can be post-processed to help promote suspense.

In our work we have looked at an alternate way to structure the narrative: by forcing the planner to make certain key events occur in the course of the narrative and in a given order. Riedl [13] referred to this as "complexifying" the planning process and extended his intent driven planner [2] to plan with the inclusion of "author goals". However, we have taken a rather different approach based on the observation that the state trajectory constraints provided in the planning domain representation language PDDL3.0 appear to provide a means to express the key events that are required in a narrative and also any ordering between them. Our hypothesis is that these constraints can be used to express narrative structuring control knowledge and that any narratives generated that satisfy these constraints will display the desired narrative structure. We were encouraged by the results of our initial experiments with MIPS-xxl [14], a planner that can reason about PDDL3.0 constraints, which supported our hypothesis. However that planner appeared too slow, given IS requirements, and this led us to develop a novel method for planning with these constraints that could perform within the time limits imposed by an IS environment. In the paper, we present an overview of this work and some preliminary results.

The paper is organised as follows. We begin in section 2 with background on the use of constraints to inject narrative control into story generation. Then in section 3 we discuss the novel algorithm that we have developed for planning with trajectory constraints. In section 4 we discuss our results and at the end of the paper in section 5 we conclude and suggest directions for future work.

2 Narrative Structuring using Constraints

To illustrate the use of constraints we'll use an example from an IS domain based on the novel Goldfinger by Ian Fleming (with a history both in narrative formalism [15] and IS [16]). The novel features James Bond, a British secret service agent who is sent to investigate gold smuggling by the eponymous Auric Goldfinger. Suppose that we want to generate a narrative with a goal of Bond defusing a bomb set by Goldfinger. We could do this using a planner but a problem with the narrative, from an IS perspective, is that it will only contain actions that are directly relevant in pursuit of the planning goal. Yet the narrative may be more interesting if, for example, James Bond were placed in a perilous situation (such as facing death by a laser beam), but such events would only feature in the narrative if they served some purpose with respect to achieving the final goal (such as Bond gaining further information). What is required is a means to specify important events that are to occur in the narrative, as well as the final narrative goals. Our hypothesis is that the state trajectory constraints in the planning representation language PDDL3.0 offer a way to do this.

State trajectory constraints are one of the novel language features introduced in PDDL3.0 [17] to describe benchmark problems for the 5th International Planning Competition. They are expressed independently to the rest of the PDDL problem definition and assert conditions that must be met by the entire sequence of states visited during the execution of a solution. The language provides a number of modal operators for expressing these constraints. In our experiments we have used the following:

operator	meaning
(sometime-before a b)	b must be made true for the first time before a
(sometime a)	predicate a must be true at some stage of the narrative
(at-end a)	predicate a must be true at the end of the narrative

The *sometime* and *sometime-before* operators allow us to specify events that must occur in the narrative and this gives a temporal order over these events. For the Goldfinger example these constraints could include:

(sometime-before (seduced bond jill) (won-cards bond goldfinger card-game))
(sometime-before (seduced bond jill) (got-mission bond))
(sometime (won-golf bond goldfinger golf-game))

which gives a partial temporal order. It is partial because not all the events are ordered with respect to each other: the event *(won-golf bond goldfinger)* must appear at some point in the narrative, but the exact timing is left unspecified. The *at-end* constraints are equivalent to planning goals, and they are ordered last in the temporal order (and the initial state is implicitly ordered to be earliest). An example *at-end* constraint for Goldfinger could be:

(at-end (defused-bomb bond fort-knox))

The temporal order defined by these constraints is as shown in figure 1 below.

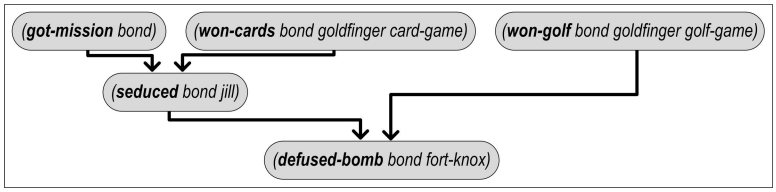

Fig. 1. Partial Temporal Order between constrained predicates.

The subset of PDDL3.0 constraints operators that we have discussed fit our purposes since they enable us to specify both: narrative events that must occur in the course of the narrative; and any orders between them. They also result in a number of benefits over alternate approaches. For example, from a computational perspective, because the constraints are specified independently from the rest of the PDDL model they can be independently manipulated, perhaps to respond to user interaction, without affecting the rest of the model. This is in contrast to approaches where control information is embedded in the operator pre-conditions or in the decompositions of an HTN and where making changes is much harder. Also, from the perspective of authoring, the independent specification of the constraints may make them easier to express and may help facilitate the testing of different story variants since the constraints can be independently changed without changing the rest of the representation.

3 Decomposition Planning with Trajectory Constraints

We saw in the previous section that the constraints define a temporal order over key narrative facts. This temporal order gives strong hints about events that will feature earlier in any narrative that satisfies the constraints, and suggested to us that a narrative could be built incrementally, starting with those events which are likely to feature earlier on. Based on this observation our planning approach is: use the orders in the constraints to decompose the problem into a number of temporally ordered subproblems; solve those subproblems in turn; and "grow" the narrative forwards from the initial state to the goal state by appending the solution for the current subproblem to the narrative developed so far.

This approach is based on the method for planning with *landmarks* of Hoffmann et al [18]. They exploit *landmarks* - facts that must necessarily be made true en route to solving a final problem goal - to decompose the planning problem and then use the resulting subproblems to guide a search control algorithm that is wrapped around a base planner (any planner that accepts basic STRIPS [19] input). Instead of landmarks, we use the events that have been supplied as constraints to decompose the problem. Although these facts aren't landmarks as defined by Hoffmann et al (they don't *have* to be made true to achieve the final goals) they nevertheless have to be achieved in order to satisfy the constraints.

The first step of the planning process is to construct a *constraints tree* using the constraints that are included in the input domain model. The input is a

Input: F, I, G, C, O **Output**: P

---

1 build constraints tree CT
2 repeat until $CT := \{\}$
3 D = leaf nodes of constraints tree
4 call planner with: A, S, and disjunctive goal $D \rightarrow P'$
5 if planner didn't find a solution P' then fail
6 else if planner did find a solution P':
7 $P := P \circ P'$
$S :=$ result of executing P' in S
8 remove from CT all $L \in D$ with $L \in add(o)$ for some $o \in P'$
9 call planner with: A, S and conjunctive goal $G \rightarrow P'$
10 if planner didn't find solution P' then fail
11 else if planner did find a solution P':
12 $P := P \circ P'$, output P

Fig. 2. Outline Algorithm: Decomposition Search Control

subset of a PDDL3.0 planning problem consisting of: F, a set of facts that can be used to describe the world; G, a goal condition such that $G \subseteq F$; and C, a (possibly empty) set of constraints. The output is a constraints tree CT with nodes N and edges E. The nodes N in the tree CT are any facts $f \in F$ that: appear in any of the *sometime* constraints; or any facts that appear as problem goals or *at-end* constraints which can be treated as equivalent. For the example constraints given in section 3 the set of nodes N is:

{*(defused-bomb bond fort-knox), (won-cards bond goldfinger card-game), (seduced bond jill), (got-mission bond), (won-golf bond goldfinger)*}

Edges *(x,y)* in the tree are directed from x to y and indicate that x must be made true in the plan before y. Continuing with the example constraints given in section 3 the set of edges E is:

{*((won-cards bond goldfinger card-game), (seduced bond jill)),*
((got-mission bond), (seduced bond jill)),
((seduced bond jill), (defused-bomb bond fort-knox)),
((won-golf bond goldfinger), (defused-bomb bond fort-knox))}

An outline of the decomposition planning algorithm is shown in figure 2. The input is a PDDL3.0 planning problem consisting of $< F, I, G, C, O >$ where: F is a set of facts that can be used to describe the world; I and G are an initial state of the world and goal condition such that $I \subseteq F$ and $G \subseteq F$; C, a (possibly empty) set of constraints; and O, a ground set of operators each with an Add, Delete, and Precondition list. Upon successful termination of the algorithm, the output is a plan P that satisfies the constraints and achieves all final goals.

The first step in the algorithm is the construction of the constraints tree, CT, which we described above. Then the algorithm loops until the constraints tree is empty (lines 2-8). Within each loop, a new subproblem is formulated and passed to the base planner (line 3-4). The goal of this subproblem, D, is *disjunctive* and is formed from the leaves of the constraints tree: these goals are the earliest in the temporal order, which we want the base planner to work on first. We know nothing about whether these facts can be made true at the same time or their order with respect to each other, so leave it up to the planner to decide which goal to work on next. We achieve this by expressing them as *disjunctive goals* using the method proposed by Gazen and Knoblock [20] which allows them to be expressed in standard STRIPS format. Then a separate base planner is invoked with this disjunctive subproblem (line 4). If the base planner returned a solution, P', to this disjunctive subproblem then a new plan is formed by appending P' onto the current plan so far and a new initial state is formed by executing the returned plan P' in the current subproblem initial state (line 7). Also, the constraints tree CT is updated so that any facts in leaf nodes of the tree that are made true by the execution of plan P' are removed (line 8). Once the constraints tree is empty (or if no constraints are supplied) the search control proceeds with the original top level conjunctive goal G (line 9). When the plan for this conjunctive goal subproblem is returned this is appended to the end of the plan so far to produce the final plan which is then output (line 12).

4 Results

The central hypothesis of our work is that PDDL3.0 constraints can be used for narrative control. In this section we present the results of a qualitative evaluation that support this hypothesis, via analysis of a selection of sample narratives.

For the evaluation we developed a decomposition planner, referred to as *DPC*, which is an implementation of the algorithm outlined in figure 2. *DPC* uses *FF-v2.3* [21] as a base planner, although any propositional planner that accepts STRIPS input could have been used. We used *DPC* to generate sample narratives for a number of input narrative variants that featured constraints (all narrative plans generated by *DPC* were validated using the PDDL3.0 validator VAL [22] and shown to satisfy the constraints). In these experiments performance of *DPC* was found to be acceptable for IS purposes, with an average subproblem solution time within 500ms.

For the purposes of the evaluation we developed a PDDL3.0 representation of the classic spy novel Goldfinger from a description of the novel inspired by the actions of the main characters. Characters' attributes, including such things as their location, activities and allegiance became the predicates of the planning domain. The main actions, those that modify characters' attributes, were represented as planning operators. The predicates were then used to specify goals and constraints for individual characters in different story variants and the planner, *DPC*, used the operators to generate narratives that satisfied the individual characters' goals and constraints (i.e. the perspective of the planner is at the

narrative level not the character level). The Goldfinger model that we have developed is a rich one where the use of constraints enables *DPC* to generate a wide range of story variants such as a trajectory consisting of 42 narrative actions that capture the main outline of the novel (this is the sequence of operators shown down the centre of figure 4).

Example Narrative Generation

In this example we discuss how our planner *DPC* generates the narrative shown in figure 3. This narrative features a goal where Bond triumphs over Goldfinger by defusing a bomb that Goldfinger was intending to use to irradiate the US gold reserve, represented with the predicate *(defused-bomb bond fort-knox)*.

If we were to pose this scenario as a planning problem *without* constraints, then *DPC* would simply invoke the base planner with the final goal (in terms of the algorithm in figure 2 the constraints tree would be empty so control would pass directly to line 9). A narrative plan would be generated and the output narrative would be directed towards the goal resulting in a sequence of operators that contribute directly to achieving the goal. This sequence is shown down the left hand side of figure 3 culminating in Bond defusing the bomb, represented by the planning operator *(defuse-bomb bond fort-knox)*.

However, the structure of the narrative can be dramatically changed with the introduction of constraints which force certain key events to occur. For example, the sequence of actions down the centre of figure 3 shows the different narrative that results when the following constraints are included: at some stage Bond will beat Goldfinger at golf, represented in the model with the predicate *(won-golf bond goldfinger golf-game)*; Bond will seduce Goldfingers assistant Jill Masterson, *(seduced bond jill)*; and before the seduction Bond will beat Goldfinger at cards, *(won-cards bond goldfinger card-game)* and also receive his mission, *(got-mission bond)*. These constraints were discussed in section 2 and the corresponding temporal order is shown in figure 1.

Following the algorithm in figure 2, *DPC* tackles the constraints in order starting with the earliest. In this example there is a choice of earliest predicates: *(got-mission bond), (won-cards bond goldfinger card-game)* and *(won-golf bond goldfinger golf-game)*, and so *DPC* formulates these as a disjunctive subproblem goal using the method of [20]. To do this we introduce an artificial predicate as the goal of the current subproblem and then for each predicate in the disjunction we add an artificial operator with: that predicate as the precondition; the artificial predicate as a single add effect; and no delete effects.

Thus for this example, the goal of the first subproblem is represented using the predicate *(artificial)* and three operators are introduced that add it: one with *(got-mission bond)* as a precondition; one with *(won-cards bond goldfinger card-game)*; and one with *(won-golf bond goldfinger golf-game)*. Then the base planner determines which operator to use to achieve the artificial goal. For this example, the base planner chooses the artificial operator with the precondition *(got-mission bond)* and this is achieved with narrative sequence that takes Bond to London to receive his mission. At the end of this first iteration of the algorithm the constrained predicate has been achieved and the other two constrained pred-

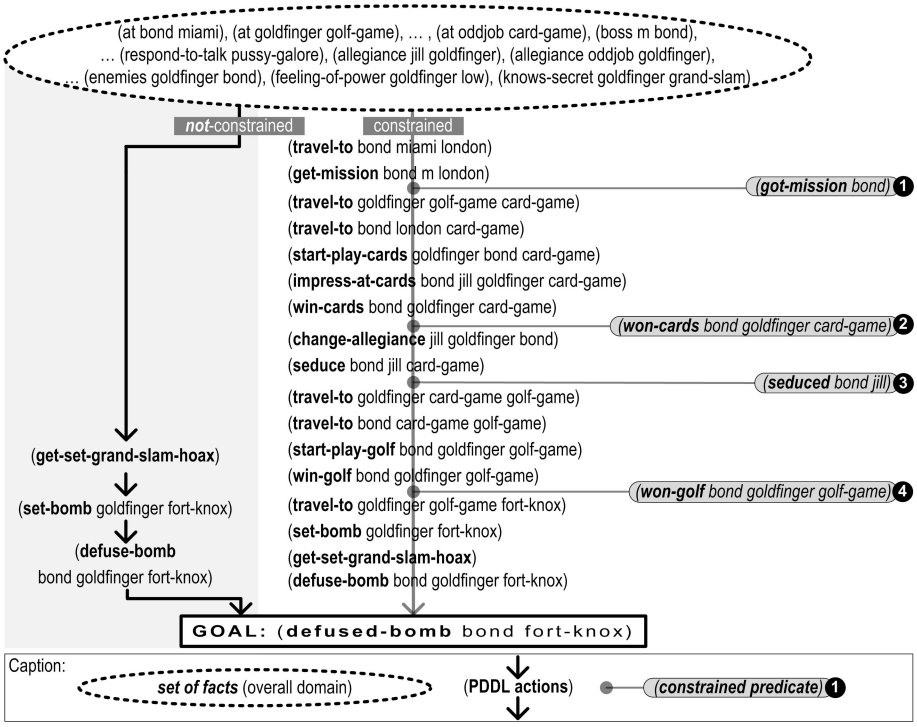

Fig. 3. Using constraints to shape a narrative (see text for worked example).

icates remain in the temporal order to be tackled on subsequent iterations. The sequence of operators to achieve *(got-mission bond)* forms the initial narrative and *DPC* then grows the narrative forwards from this point.

Attention now turns to the current earliest constrained predicates in the temporal order. These are *(won-cards bond goldfinger card-game)* and *(won-golf bond goldfinger golf-game)*. Once again *DPC* poses this as a disjunctive problem and the state that has been reached at the end of the narrative so far (ie Bond is in London having received his mission) is used as the starting situation for the next subproblem. For this subproblem the base planner chooses to next solve *(won-cards bond goldfinger card-game)*, shown as constrained predicate (2) in figure 3, with a sequence of operators involving the characters moving to the venue of the card game and participating in the game. These operators are then appended to the developing output narrative as it is grown forwards.

Now the earliest predicates in the temporal order are: *(seduced bond jill)* and *(won-golf bond goldfinger golf-game)* and once again these are posed as a disjunctive problem and the base planner is invoked. As shown in figure 3 the base planner chooses to solve *(seduced bond jill)* next with a narrative sequence that involves Jill changing her allegiance and being seduced by Bond. At this point there is a single earliest predicate in the temporal order, *(won-golf bond*

goldfinger golf-game), and the base planner outputs a narrative sequence for this which is appended to the output narrative.

Once all the constrained predicates have been achieved in order, the base planner is then invoked with the final conjunctive goal, which in this example consists of the single predicate *(defused-bomb bond fort-knox)*. Finally the narrative for this final goal is appended to the output narrative generated so far.

This example demonstrates the way in which the constraints can be used to support a high level description of a narrative so that it follows the required dramatic arc. In fact, so long as the set of constraints for a planning instance accurately reflects the high level control that is required for the narrative then any plan that is generated that satisfies the constraints (i.e. a valid plan), can be said to display the required narrative control. This example also shows how the use of constraints can allow for variation in the narratives that are generated when we have a partial order over predicates in the domain, since a number of different variants will satisfy the same set of constraints.

Narrative Segment: introducing conflict

Figure 4 shows a Goldfinger narrative that captures the main outline of the novel. One way to look at this sequence of events is from the perspective of the novel's main theme, the conflict between the protagonist James Bond and his adversary Auric Goldfinger. As observed by Cheong and Young [8], this type of conflict, where one characters' individual goals are the negation of anothers', is an important prerequisite for creating suspense in a narrative. Since suspense is a key feature of Goldfinger, we can use a dimension that places these characters and the pursuit of their goals in opposition and to view story progression through the sequence of operators and resulting character situations. Interestingly, the "trajectories" along that dimension represent story evolution in a similar way to "dramatic arcs" [23, 24].

We can use constraints to introduce conflict into the narrative by imposing constraints that draw the narrative in directions that set the goal of one of the central characters against the others. As an example consider the constrained predicates 8–13 in the lower segment of figure 4. Some of these constraints place Goldfinger in a position of superiority over Bond (for example, when he has captured Bond and is attempting to kill him with an industrial laser (10) and when he has Bond handcuffed to a nuclear bomb (12)), whereas others place Bond in the superior role and serve to negate Goldfingers goals (for example, Bond spying on Goldfinger in Switzerland (9) and Bond seducing Pussy Galore and securing her allegiance (11)).

Narrative Segment: introducing causality

Another way in which the constraints can be used is to promote causality in the trajectory that is output, for example, to ensure that character behaviour is justified. As an illustration, consider the constrained predicates numbered 3–5 in figure 4. Here the constraint *(won-cards bond goldfinger)* is followed by the constraint *(seduced bond jill)* and yields a narrative sequence that sets up Jills betrayal of Goldfinger – she is impressed by Bonds card playing and this causes

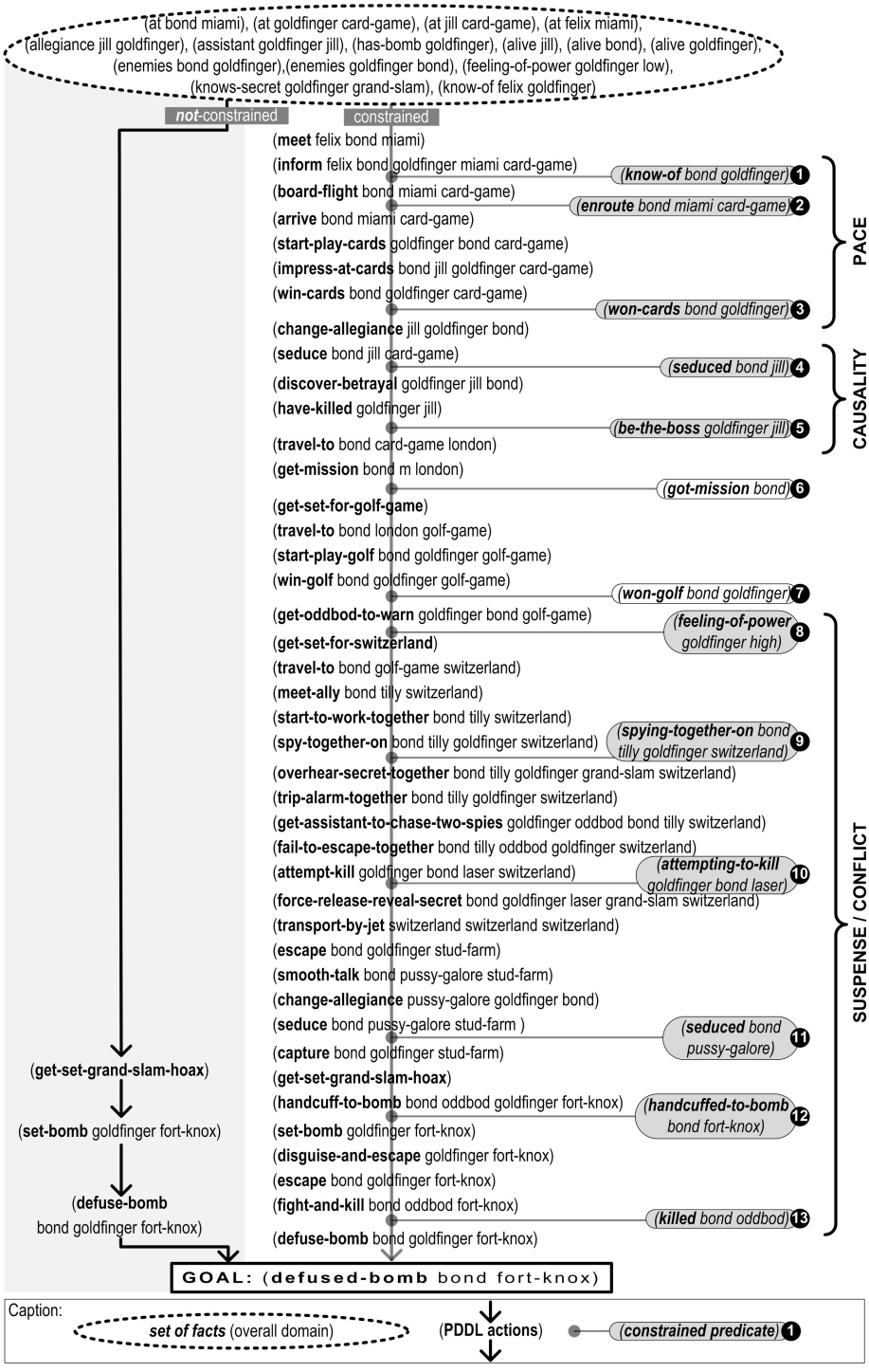

Fig. 4. Using constraints to shape a narrative which reflects the central theme of the novel and introduces conflict between Bond and Goldfinger (see text for more details).

her to change allegiance – which justifies the subsequent sequence of actions where Goldfinger has her killed once he becomes aware of her betrayal in order to assert his position of power.

Narrative Segment: varying pace
We can use the constraints to vary the pace of parts of the narrative. For example, consider the constrained predicates numbered 1–3 in figure 4. Bond is initially in Miami and must travel in order to take part in a card game with Goldfinger. In the absence of constraints a planner would generate a narrative which has Bond travelling directly to the venue, with the key action *(travel-to bond miami card-game)*. However the consequence of introducing constraints that order *(know-of bond goldfinger)* to occur in the plan before *(enroute bond miami card-game)* which in turn must occur before *(won-cards bond goldfinger card-game)* is the addition of the key actions which have Bond and Felix Leiter meeting in Miami, Felix informing him about Goldfinger and his nefarious activities, and then introducing detail about Bonds journey from Miami. The inclusion of the constraints results in the addition of a number of relevant operators which change the pace by increasing the detail about this characters activities during this segment of the narrative.

5 Conclusions and Future Work

In the paper we have introduced a novel method for injecting control into automatically generated narratives for IS. The approach uses features of the planning representation language PDDL3.0 to encode control knowledge in the form of constraints and we have developed a novel method for planning with these constraints. Our results are encouraging and support the hypothesis that these constraints can be used for narrative control.

Using constraints means that narrative structuring information is specified independently to the rest of the representation and yields important benefits. For example, one benefit is that the structuring information can be independently manipulated and changed without affecting the rest of the representation (this is much harder if the control information is embedded within pre-conditions of operators or HTN decompositions). Also for authoring, it appears that the declarative nature of the constraints makes them easier to express and facilitates author testing of alternate story variants.

We are currently extending our approach to handle interactivity and in particular the re-specification of constraints "on-the-fly" in response to user interaction. We have identified mechanisms which should support the re-specification of constraints during an interactive session, suggesting that interactivity could affect not just the planning domain but also the constraints themselves thus opening the way for interpreting user intervention at multiple levels.

Acknowledgements This work has been funded (in part) by the European Commission under grant agreement IRIS (FP7-ICT-231824).

References

1. Weld, D.: An introduction to least commitment planning. AI magazine (1994)
2. Riedl, M., Young, M.: An Intent-Driven Planner for Multi-Agent Story Generation. In: Proceedings of AAMAS. (2004)
3. Aylett, R., Dias, J., Paiva, A.: An affectively-driven planner for synthetic characters. In: Proceedings of ICAPS. (2006)
4. Bonet, B., Geffner, H.: Planning as heuristic search: New results. In: Proceedings of the European Conference on Planning (ECP). (1999)
5. Pizzi, D., Cavazza, M., Whittaker, A., Lugrin, J.: Automatic Generation of Game Level Solutions as Storyboards. In: Proceedings of AIIDE. (2008)
6. Erol, K., Hendler, J., Nau, D.: UMCP: A sound and complete procedure for hierarchical task-network planning. In: Proceedings of AIPS. (1994)
7. Cavazza, M., Charles, F., Mead, S.: Character-based Interactive Storytelling. IEEE Intelligent Systems, special issue on AI in Interactive Entertainment (2002)
8. Cheong, Y., Young, R.: A Computational Model of Narrative Generation for Suspense. In: AAAI 2006 Computational Aesthetic Workshop. (2006)
9. Weyhrauch, P.: Guiding Interactive Drama. PhD thesis, School of Computer Science, Carnegie Mellon University (1997)
10. Magerko, B., Laird, J., Assanie, M., Kerfoot, A., Stokes, D.: AI Characters and Directors for Interactive Computer Games. In: Proceedings of AAAI. (2004)
11. Mateas, M., Stern, A.: Structuring Content in the Façade Interactive Drama Architecture. In: Proceedings of AIIDE. (2005)
12. Riedl, M., Stern, A.: Believable Agents and Intelligent Story Adaptation for Interactive Storytelling. In: Proceedings of TIDSE. (2006)
13. Riedl, M.: Incorporating Authorial Intent into Generative Narrative Systems. In: Proceedings AAAI Spring Symposium on Intelligent Narrative Technologies. (2009)
14. Edelkamp, S., Jabbar, S., Nazih, M.: Large-Scale Optimal PDDL3 Planning with MIPS-XXL. In: Proceedings of Planning Competition, ICAPS. (2006)
15. Barthes, R.: Image Music Text. Fontana Press (1993)
16. Cavazza, M., Charles, F., Mead, S.: Multi-modal Acting in Mixed Reality Interactive Storytelling. IEEE Multimedia (2004)
17. Gerevini, A., Long, D.: Plan Constraints and Preferences in PDDL3. Technical report, Department of Electronics for Automation, University of Brescia, Italy (2005) http://www.cs.yale.edu/homes/dvm/papers/pddl-ipc5.pdf.
18. Hoffmann, J., Porteous, J., Sebastia, L.: Ordered Landmarks in Planning. JAIR **22** (2004)
19. Fikes, R., Nilsson, N.: STRIPS: A new approach to the application of theorem proving to problem solving. Artificial Intelligence (1971)
20. Gazen, B.C., Knoblock, C.: Combining the Expressiveness of UCPOP with the Efficiency of Graphplan. In: Proceedings of European Conference on Planning (ECP). (1997)
21. Hoffmann, J., Nebel, B.: The FF Planning System: Fast Plan Generation through Heuristic Search. JAIR (2001)
22. The Strathclyde Planning Group: VAL: The Automatic Validation Tool for PDDL including PDDL3 and PDDL+ http://planning.cis.strath.ac.uk/VAL/.
23. Mateas, M.: A Neo-Aristotelian Theory of Interactive Drama. In: AAAI Spring Symposium on AI and Interactive Entertainment. (2000)
24. Zagalo, N., Barker, A., Branco, V.: Story Reaction Structures to Emotion Detection. In: Proceedings of ACM Multimedia 2004, Workshop on Story Representation, Mechanism, Context. (2004)