# Visual Programming of Plan Dynamics Using Constraints and Landmarks

**Julie Porteous, Jonathan Teutenberg, David Pizzi** and **Marc Cavazza**
School of Computing,
Teesside University,
Middlesbrough TS1 3BA,
United Kingdom
{j.porteous,j.teutenberg,d.pizzi,m.o.cavazza}@tees.ac.uk

## Abstract

In recent years, there has been considerable interest in the use of planning techniques in the area of new media. Many traditional planning notions no longer apply in the context of these applications. In particular, it can be difficult to answer the important question of what constitutes a good plan for the domain, but there is an emerging consensus that plan dynamics plays an important role. As a consequence, it is important to support representation of such aspects. Our solution is to introduce a meta-level of representation that is an abstraction of the domain with respect to both time and causality, and to develop a visual representation of this in the form of a *narrative arc*. This visual representation can then be used in a visual programming approach to the exploration and specification of plan dynamics. In the paper we outline this approach to meta-level representation using *constraints* along with the visual programming interface we have developed. We illustrate the approach with examples of visual programming in the development of an interactive entertainment system based on Shakespeare's play "The Merchant of Venice".

## Introduction

With the development of new media, such as Interactive Storytelling (IS), a major new application area for AI planning is emerging. In this area, planning technology is used to generate narratives for entertainment systems that feature 3D interactive presentation of the narrative using animations. This approach has its roots in the adoption of planning as a technology for virtual agents which was later transferred to reasoning about virtual actors (Geib 1994). It was first proposed for IS in (Young 2000a) and since then it has emerged as the core technology for IS prototype systems (Cavazza et al. 2007; Riedl and Young 2010). These new media domains differ from the sorts of benchmark problems that have featured in planning research, as seen in the sorts of domains that feature in the bi-annual ICAPS International Planning Competitions. In particular a key planning property sets new media domains apart, namely: the criteria used to assess plan quality.



Historically, optimality in terms of plan length was *the* criterion for assessing the quality of plans generated by automated planning systems (Chapman 1987). This has been relaxed more recently with the advent of forward state-space planners that trade off plan quality for performance gains (Bonet and Geffner 1999; Hoffmann and Nebel 2001). In addition, there have been moves to specify criteria for plan quality using metrics (Hoffmann 2003) and adequacy criteria such as preferences and constraints (Chen, Wah, and Hsu 2006; Baier, Bacchus, and McIlraith 2007).

Quality criteria for new media applications differ since planning goals need not equate to the end of the narrative (i.e. serve as the narrative driver) since they can evolve over time, re-planning is frequently required in response to user interaction, interesting narrative events may be missed if the emphasis is on generating optimal solutions, and there is frequently a need to generate sub-optimal trajectories in order to display narrative concepts such as suspense. Indeed, there is growing consensus that for these types of domains quality criteria should be more concerned with the dynamics of the plan, the shape of its *trajectory*, i.e. the intermediate states that will be traversed when it is executed. Different application-specific interpretations will apply to the trajectories: for example in new media they relate to narrative features such as suspense, pace and so on.

Based on these observations, we have addressed the problem of how to specify quality criteria for domains, such as new media, where quality relates to features such as plan dynamics and trajectory. Since controlling trajectory by direct manipulation of the domain model is a considerable challenge and because trajectory suits a visual representation our solution has been to use a visual programming approach. However, visual programming for complex representations such as connected sets of planning operators brings its own challenges. This is why we propose to use an intermediate level of abstraction between the visual interface and the planning operators. From a theoretical perspective, this intermediate representation should be compatible with some form of semantic mapping as well as informing the selection of plan actions. Recent advances in declarative control representations, such as landmarks (Hoffmann, Porteous, and Sebastia 2004), served as an inspiration for our approach.

In the paper we present this solution and illustrate it throughout with examples taken from an IS application we

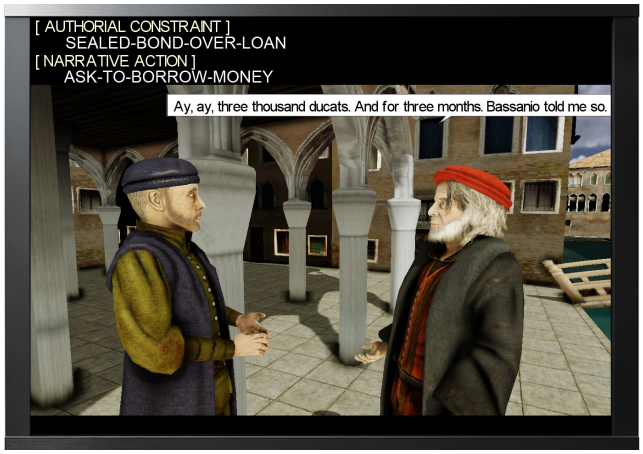


Figure 1: A screenshot showing 3D visualisation of a narrative action in our IS system which features virtual characters and scenarios inspired by Shakespeare's *Merchant of Venice*.

have developed based on Shakespeare's *Merchant of Venice*. In the system, generated narratives are staged as 3D animations featuring virtual characters (as shown in the screenshot in figure 1. The organisation of the paper is as follows. We start with background on new media applications and issues involved in creating and modelling them. This is followed with an overview of our planning approach and the meta-level representation it supports. Then we describe the key features of our visual programming approach. We include an evaluation that demonstrates the applicability of the approach for identification of plan quality criteria. We finish with consideration of closely related work and conclusions.

## Background

In certain areas of new media such as IS, planning has been enthusiastically adopted for the task of narrative generation due to its ability to propagate causality within an output narrative and the utility of using planning has been clearly demonstrated by the IS research community (Cavazza et al. 2007; Riedl and Young 2010). Different views are taken on the relationship between plans and narratives: some equate the narrative to the plan (Young 2000b) whereas we adopt the weaker assumption and view the plan as a representation of the narrative (Porteous, Cavazza, and Charles 2010).

Planning is well suited to narrative generation since it enables narratives to be naturally modelled as sequences of actions. One difficulty that remains is how to specify what constitutes a good plan although there is growing consensus that a fundamental aspect is plan dynamics. An important aspect of this is narrative tension, which can be represented via the temporal distribution of key narrative actions that form part of all narrative ontologies. This should be underpinned by the semantic tagging of operators corresponding to these actions, reflecting their inherent contribution to narrative tension. This represents key actions (beats) rather than an indirect measure of suspense-as-progression. The inclusion of narrative tension enables the overall shape of narratives to be

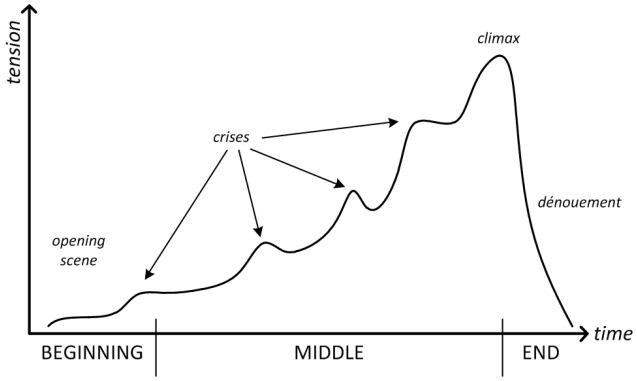


Figure 2: An example narrative arc displaying the classic Aristotelian shape of tension rising to a final climax and denouement. In our system tension levels are controlled by the user (see section "Narrative Planning Approach"). They are used by the planner to control output trajectory shape. As shown, peaks of tension are often local. These important intermediate trajectory points are used by our implemented planner to drive plan generation.

represented visually using a *Narrative Arc*. Figure 2 shows an arc that corresponds to the classic Aristotelian structure (and also the original structure of our example narrative, *The Merchant of Venice*). It features rising tension through a series of minor crises towards a final climax which is resolved with the denouement at the end.

## Narrative Planning Approach

Approaches to the problem of supplying information about plan dynamics include the introduction of *author goals* to *complexify* the planning problem (Riedl 2009) and HTN approaches where this information is embedded in method decompositions (Hoang, Lee-Urban, and Munoz-Avila 2005; Kelly, Botea, and Koenig 2007). Our solution to supplying this information is to exploit a meta-level which is an abstraction with respect to both time and causality (Porteous, Cavazza, and Charles 2010). This meta-level is represented using *constraints* which are key narrative situations for a domain[1]. The idea is that the constraints can be used by a planner as intermediate goals to guide generation of plans that display these features.

Constraints provide a high-level mechanism to control the selection of operators and hence to ensure that the corresponding trajectory exhibits the desired properties for that particular application (for example, in narrative these are associated with pace, suspense and so on). As an example, in the *Merchant of Venice*, important narrative situations include: the signing of a bond between the characters Antonio

[1]We use the nomenclature of (Porteous, Cavazza, and Charles 2010) but observe the similarity to author goals (Riedl 2009). We also note the similarity to the notion of landmarks (Hoffmann, Porteous, and Sebastia 2004). Unlike landmarks, constraints don't always have to be made true in order to solve the goal, only to conform to desired plan dynamics.

and Shylock, by which the latter agrees to lend 3000 ducats to the former without interest; and the receipt, by Antonio and Shylock, of the verdict of the court when Antonio has subsequently defaulted on the loan. These situations can be represented as predicates as follows:

*(sealed-bond-over-loan shylock antonio)*
*(received-verdict-of-court antonio shylock).*

A constraint is a predicate that is required to occur in a generated narrative. We have found the PDDL3.0 modal operators *sometime* and *sometime-before* suitable for their representation (Porteous, Cavazza, and Charles 2010). The *sometime* operator can be used to specify constraints that must occur in output narratives but are unordered with respect to other constraints and the *sometime-before* operator can be used if the ordering of two constraints is important. For example, it makes sense in the context of the *Merchant of Venice* pound-of-flesh sub-plot (Hinely 1980) for the bond to have been sealed between Shylock and Antonio *before* they have received the verdict of the court. Both these predicates are hence constraints and the order can be represented as follows:

*(sometime-before (received-verdict-of-court antonio shylock)*
*(sealed-bond-over-loan shylock antonio))*

In order to enable visual representation of plan dynamics numeric tension levels are associated to each of the constraints. They can be set and adjusted by the user during interaction with the system otherwise they are allocated default values.

Our planner is a state-space forward search heuristic planner which supports the use of landmarks (Hoffmann, Porteous, and Sebastia 2004) and accepts planning problems expressed using the subset of PDDL3.0 described above. As such, it represents a more modern planning approach than those that have traditionally been used in IS (such as partial order planning (Riedl and Young 2010)). It uses a partially ordered set of constraints to decompose the process of narrative generation into a sequence of sub-problems, where each sub-problem has a constraint as its goal and the planner generates a narrative for each decomposed sub-problem in turn. If the planner can't generate a narrative for a sub-problem then it simply continues with the next constraint, thus ensuring planner continuation. Once all sub-problems have been tackled, a final narrative can be assembled by composition of the narratives for each sub-problem. However, since our planner is integrated within an IS system a complete plan need not be output in the traditional sense. Instead operators are sent one at a time for 3D visualisation to a user.

We have implemented a control mechanism that integrates this narrative planner within an IS system. The control mechanism handles constraint and problem instance selection at run-time. Constraints for a particular narrative world are represented using PDDL3.0 and form a partially ordered set of predicates. For a particular planning instance a subset of the constraints are selected (e.g. based on user preferences and interaction history) and are topologically sorted to linearise them. The planner then uses this to drive narrative generation. Initial problem instances are created for each session on the basis of factors such as user preference and enforced variation between runs.

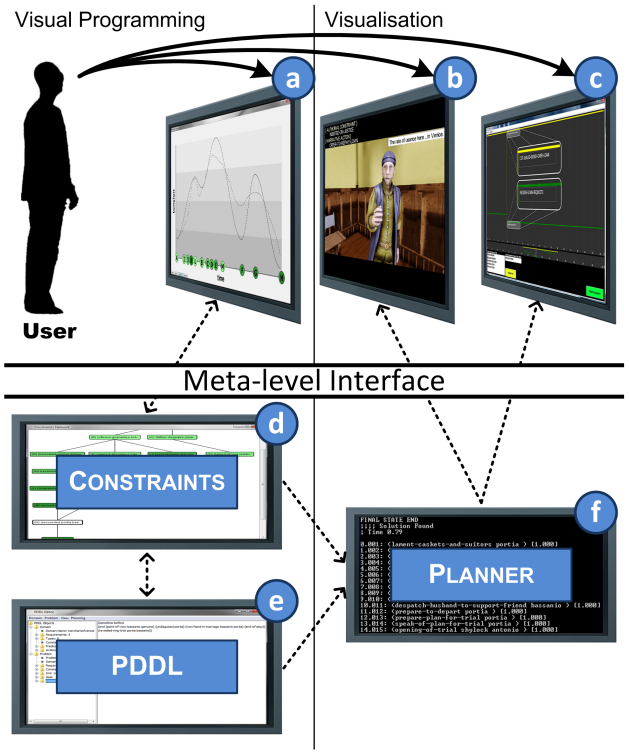


Figure 3: Visual programming of plan dynamics: user interacts via narrative arc window, drawing desired arcs, adjusting tension levels (a); global properties of generated narratives assessed via visualisation windows (b) and (c); lower level components accessed via interface (d) and (e); user invocation of planner using specified plan dynamics (f).

## Visual Programming: System Overview

Our constraint-based approach to narrative generation introduces a meta-level of representation which is an abstraction with respect to both time and causality since a selected sequence of constraints gives us just the bare outline of a narrative. We have developed a visual representation of this meta-level that uses the tension levels assigned to the constraints to plot them on a narrative arc. Building on this we have developed a system for visual programming of plan dynamics: users can manipulate the narrative arc to specify different dynamics and assess their global properties.

An overview of the system architecture is shown in figure 3. The user interacts at the meta-level via a *Narrative Arc Window* (a) and can also explore generated narratives via two visualisation windows: an *Animation Window* (b); and a *Timeline Window* (c). The system also features some hierarchically organised lower level components. They include the constraints (d) and other PDDL constituents of the domain model (e) (user interaction with these lower level components is outside the scope of this paper since our focus is meta-level aspects). Invocation of the planner (f) is driven by the user in order to explore the narrative possibilities of different sets of plan dynamics.

The Narrative Arc Window and its role in the system is

shown in figure 3 (a) and it's also shown enlarged in the screen shots in figure 4. This window provides a visual programming interface which enables users to draw and manipulate narrative arcs as a means to specify plan dynamics. When the user draws or manipulates an arc in this window there is an information flow from the meta-level to lower levels so the system can relate the user arc to lower level information (e.g. the constraints) in order to produce a system arc that represents the "best fit" to it given the current domain model. The information that is communicated is the tension values that have been determined for the constraints and the trajectory that has been drawn by the user. The system uses this information to: produce a best fit to the users arc in terms of an ordering of constraints and spacing between them; then represent the constraints as a smoothly changing contour in the Narrative Arc Window.

Through the interface the user can also choose to explore narratives generated using the arrangement of constraints represented in the system arc. This involves communication from the meta-level to the lower level system components and invocation of the planner. The generated narrative is communicated back to the meta-level for presentation to the user via the visualisation windows (figure 3 (b) and (c)).

## Visual Programming: Enabling Technology

The principle underlying our visual programming approach is a mapping between the shape of the dramatic intensity curve and the constraints which serve as a meta-layer in our plan-based representation. The bi-directional nature of this mapping supports an interactive implementation and makes the curve an interface with which to modify the constraint-based plan representation.

**Finding "Best Fit" to user drawn arc:** The constraints in the domain model form a partially ordered set. The first step to finding their best fit to a user drawn trajectory is to linearise them via a topological sort.

We use information communicated from the meta-level – the tension levels assigned to the constraints – to inform this topological sorting. The tension values for each valid topological sort can be arranged, evenly spaced, on a line and their distance from the value of the user arc calculated. In IS planning tasks the meta-level constraints roughly correspond to beats in a narrative, of which there are typically in the order of 10 to 15 (Mateas and Stern 2002). In addition, the constraint graphs tend to have a number of articulation points – vertices that separate constraints into distinct sets – that reduce the scope of possible orderings. In general between 5 and 30 thousand orderings need to be considered, which on modern consumer hardware can be processed well within limits for real-time response to users.

In addition to constraint ordering, the temporal spacing between constraints is important – non-uniform spacing makes it possible it to find a closer fit to the users desired trajectory. Our solution to finding the best fit with non-uniform spacing, is to give each constraint a time value. The resolution of these values is set to the greater of 50 or 3 times the number of constraints. The problem then becomes one of assigning each constraint to one of the 50+ time points. This corresponds to the well-known NP-hard assignment problem. However, since the constraints are partially ordered we can exploit algorithms that are efficient in practice.

Hence, rather than directly solve the assignment problem, our solution is to decompose it into a series of sub-problems applied to each valid topological sort of the constraints[2]. The resulting sub-problem for each topological sort is then to find the assignment of constraints to time values that maintains a given ordering, that is, an instance of the Order-Preserving Assignment Problem (Scott and Nowak 2006). A bipartite graph is constructed in which one vertex set contains a vertex for each constraint, and the other set contains one vertex for each possible point in time. An edge exists from each vertex in the first set to each vertex in the second, with weight equal to the squared difference between the constraint's tension value and the value of the desired tension arc at that time point. The problem is then one of assigning each constraint vertex to one time point, such that overall weight is minimised and that the ordering of the selected time points is the same as the ordering of the constraints (i.e. the order is still the same as the topological sort we are working from for the current sub-problem).

In addition to the standard order-preserving requirements, there is also a minimal distance that must be maintained between constraints' time points. This is based on the minimal number of actions in any plan that must occur between the given constraint facts. As with generating the topological sorts, these minimal distances need only be calculated when changes are made to the constraint graph.

The best match to constraints is found with a dynamic programming solution. A 2D space of possible constraint times is explored while finding the matching, with each cell representing a matching of one constraint to one time point. The cost of each cell with constraint $i$ and time $j$ is given by:

$$C_{i,j} = d_{i,j} + minCost(i-1, j - M_{i,i-1})$$

where $d_{i,j}$ is the squared difference between the desired contour and the constraint's value, $M_{i,i-1}$ is the minimum distance permitted between the $i$th and $(i-1)$th constraints, and $minCost$ is the lowest cost value for the previous constraint:

$$minCost(i,j) = \begin{cases} C_{i,0} & j = 0 \\ min(C_{i,j}, minCost(i, j-1)) & j > 0 \end{cases}$$

The values for the $minCost$ of the previous constraint and the cost for each time point can be computed together in a single pass – with a constant number of operations per cell. So the overall time complexity for filling the 2D space is $O(nm)$ for $n$ constraints and $m$ time points.

[2]This requires us to generate the set of all topological sorts but as changes to the constraint graph occur less frequently than changes to user narrative arcs, this set can be cached and the computation required by this stage of the process can be amortised across the application's run time.

**Representation of System Arc:** Once the ordering and distances between constraints that best reflect a user's selected trajectory has been determined, it must then be presented in a format compatible with the expected view of a narrative arc (i.e. a smoothly rounded contour as shown in figure 2). We abstract the tension values that have been assigned to the constrained predicates to a contour sharing the following properties with typical narrative arcs: only the high-level trajectory is shown, without fine detail; important events (peaks and troughs) are visible; and the curve is continuously differentiable. To achieve this we first remove all non-peak tension values (i.e. those values for which the prior tension is higher and the following tension lower, or vice-versa). The contour between peaks is then generated using cosine interpolation, enabling smoothly changing values to be drawn in the output contour.

## Visual Programming: User Interaction

User interaction with the visual programming system is via the Narrative Arc Window which enables a user to draw and manipulate differently shaped narrative arcs and subsequently use these arcs to explore narrative possibilities.

A series of screen shots of this window are shown in figure 4 and illustrate the constituent parts of the window that are presented to the user. The axes are the duration of the narrative and level of narrative tension and the labelled circles along the x-axis represent narrative constraints (the constraint name is displayed when the mouse is run over it). These screen shots were generated whilst interacting with our *Merchant of Venice* system and constraints (E) and (M) correspond to the constraints *(sealed-bond-over-loan shylock antonio)* and *(received-verdict-of-court antonio shylock)* as discussed earlier. The dotted arc in the window is a desired arc created by a user and the solid arc represents the best fit to the constraints constructed by the system – the system arc. The height of the system arc at any point reflects the tension level of the corresponding constraint. At any time users can set, or re-set, the tension level of constrained predicates by way of a drop down menu.

When an arc is drawn in this window it triggers the system to generate the best fit of constraints to it using the solution detailed in the previous section. Then the system re-displays this best fit arrangement of constraints and system arc. It is this re-ordered set of constraints which will be used to control the planner, later, when narrative generation is required. Screen shot (1) shows a desired user arc and the arrangement of constraints and system drawn arc that best fit it.

The user can also manipulate the arc in order to specify different plan dynamics – figure 4 shows this process of interaction. The initial user arc has been drawn in the window (screen shot (1)) and then the user modifies the arc by dragging different segments of it (screen shot (2)). In response to the user modifications the system recalculates the best fit and redisplays the constraints and system arc (screen shot (3)). For the constraints along the x-axis, observe how both the relative spacing between them and the ordering of the constraints has changed. In particular, constraint *I* has moved from its position between constraints *D* and *B*, to between constraints *M* and *N*.

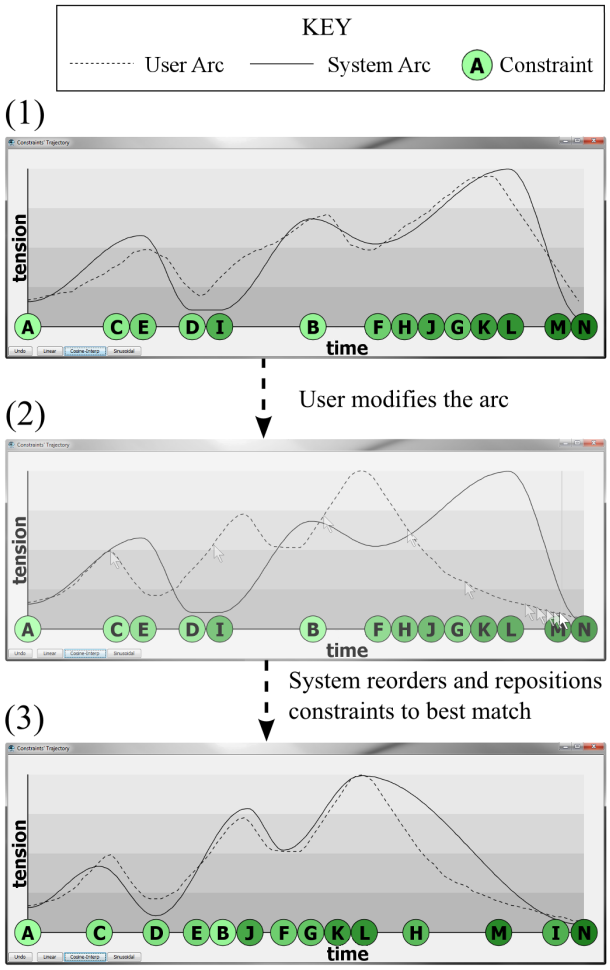


Figure 4: Visual programming of plan dynamics using the narrative arc window: (1) an initial arc drawn by the user with system best fit constraints and corresponding system arc; in (2) the user is modifying their desired arc by dragging different sections of the arc; and (3) shows the positioning of constraints and system arc after computing the best fit.

Should they wish to, the user can also specify plan dynamics via manual manipulation of the ordering of constraints along the x-axis of the narrative arc window. This can be achieved by dragging a constraint in either direction along the x-axis, within the limits defined by the partial temporal orders between constraints. When the user manipulates the constraints in this way it triggers the system to regenerate its arc to reflect the re-arranged constraints.

The Narrative Arc Window provides a means for a user to specify different plan dynamics and then to explore possible narratives that display that shape. When a user has produced an arc with their desired shape they can choose to generate a narrative for the current best fit constraint configuration and assess its global properties using the narrative visualisation windows. When the user invokes the planner, the current constraint configuration, along with the remainder

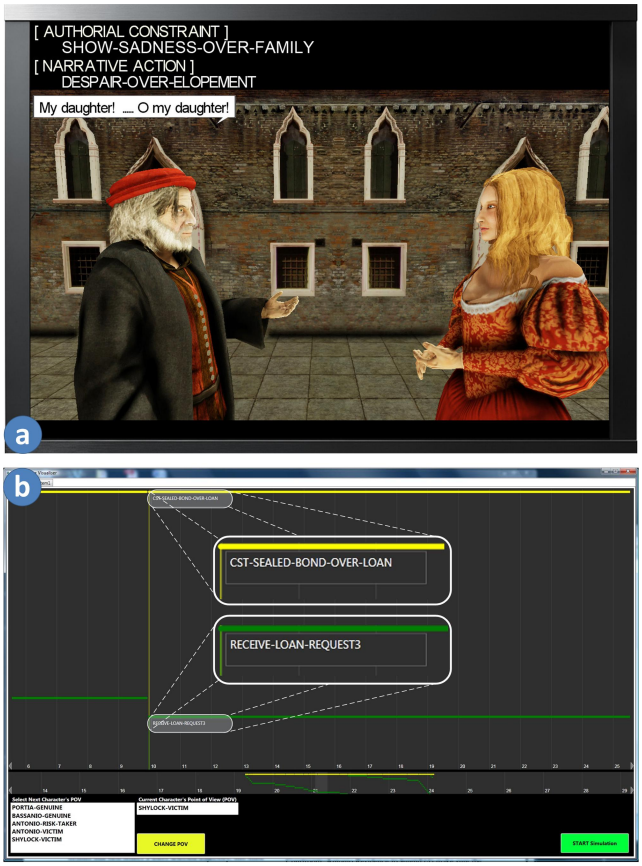


Figure 5: Visualisation Windows: (a) 3D visualisation of narrative actions with information bar at top (the constraint goal of the current sub-problem and the action being visualised) and character dialogue presented using speech bubbles; (b) a timeline representation of the output narrative which is updated incrementally as the planner proceeds.

of the domain model is passed to the planner. Our planner uses a decomposition approach and as it proceeds through the sequence of sub-problems actions are passed to the visualisation windows for presentation to the user.

Examples of the narrative visualisation windows are shown in figure 5. In the Animation Window, screen shot (a), individual actions are staged as a sequence of 3D animations featuring virtual characters. Figure 5 (a) shows a scene from our *Merchant of Venice* system, featuring a conversation between the virtual characters Shylock and his daughter Jessica. The other visualisation window features a timeline representation of the narrative as shown in (figure 5 (b)). It shows the segment of the narrative that contatins the action that is currently being visualised. It is updated in real-time: as the planner proceeds through the sequence of decomposed sub-problems actions are passed for visualisation and the timeline is updated (figure 5 (b)). Together, these windows enable the user to observe the 3D animation whilst also relating this to forthcoming actions.

## Evaluation

**Quantitative:** Our objective was to demonstrate acceptable performance of the system when recalculating the best fit to a user-specified narrative arc. Tests were run on a Java 1.6 virtual machine, using a single 2.26 GHz CPU. The number of constraints ranged from between 12 and 30 constraints, with our typical IS domains using 15 or fewer. The number of topological sorts is usually between 5 and 30 thousand. With 30 thousand topological sorts, it took less than $1/3$ of a second to respond to a user's input when there were 15 constraints. We feel this provides a satisfactory user experience. With 25 constraints, responses took $3/4$ of a second and at 30 constraints slightly over 1 second.

Our results showed that the time taken to determine the best arrangement of constraints scales close to linearly with the number of constraints. From 17 constraints upward, both the number of possible time values and the number of constraints increase, giving a quadratic upper bound on the complexity. It is therefore a very positive result that in practice the theoretical worst-case is not reached.

**Qualitative:** Our objective was to provide answers to key questions concerning whether our approach supports the visual programming of plan dynamics, enables user exploration of the narrative space and supports the identification of plan quality criteria that relate to features such as trajectory shape. We considered these questions with reference to two example plans generated by our system (shown in figure 6). The system generates variants with an average of 40 operators required to span the main plot. The consistency of these narrative variants has been demonstrated in previous work (Porteous, Cavazza, and Charles 2010) although there was hitherto no guarantee that the generated variants were interesting ones. Hence this approach is a first attempt to filter out narrative variants via a high-level representation that encompasses some measures of narrative quality or interest.

The two parts of the figure show very different user narrative arcs. The shape of narrative arc (a) follows the classic Aristotelian contour, with minor climaxes of increasing tension levels before the final climax of the play and subsequent denouement. This climax is the end of the "pound-of-flesh" sub-plot (Hinely 1980), where it appears all hope is lost for the titular merchant of the play, Antonio, having defaulted on a loan from Shylock, who is unwilling to show mercy and demands justice – represented in our domain with constraint (L) *(insisted-on-justice shylock antonio)*. Segments of the plan generated from the best-fit constraints for this arc, along with shots from its 3D visualisation, are shown running down the figure. The constraints labelled (B), (I) and (H) play an interesting role in the narrative because they show Shylock being subjected to bad treatment at the hands of Antonio and suffering a series of personal losses which help to motivate his merciless behaviour with respect to Antonio and his demands for his "pound-of-flesh". One consequence of the inclusion of these actions in the early phase of the narrative in part (a) of figure 6 is that Shylock is portrayed in a sympathetic light to the audience.

In contrast, narrative arc (b) has a very different shape with the major climax much earlier, followed by reduction of

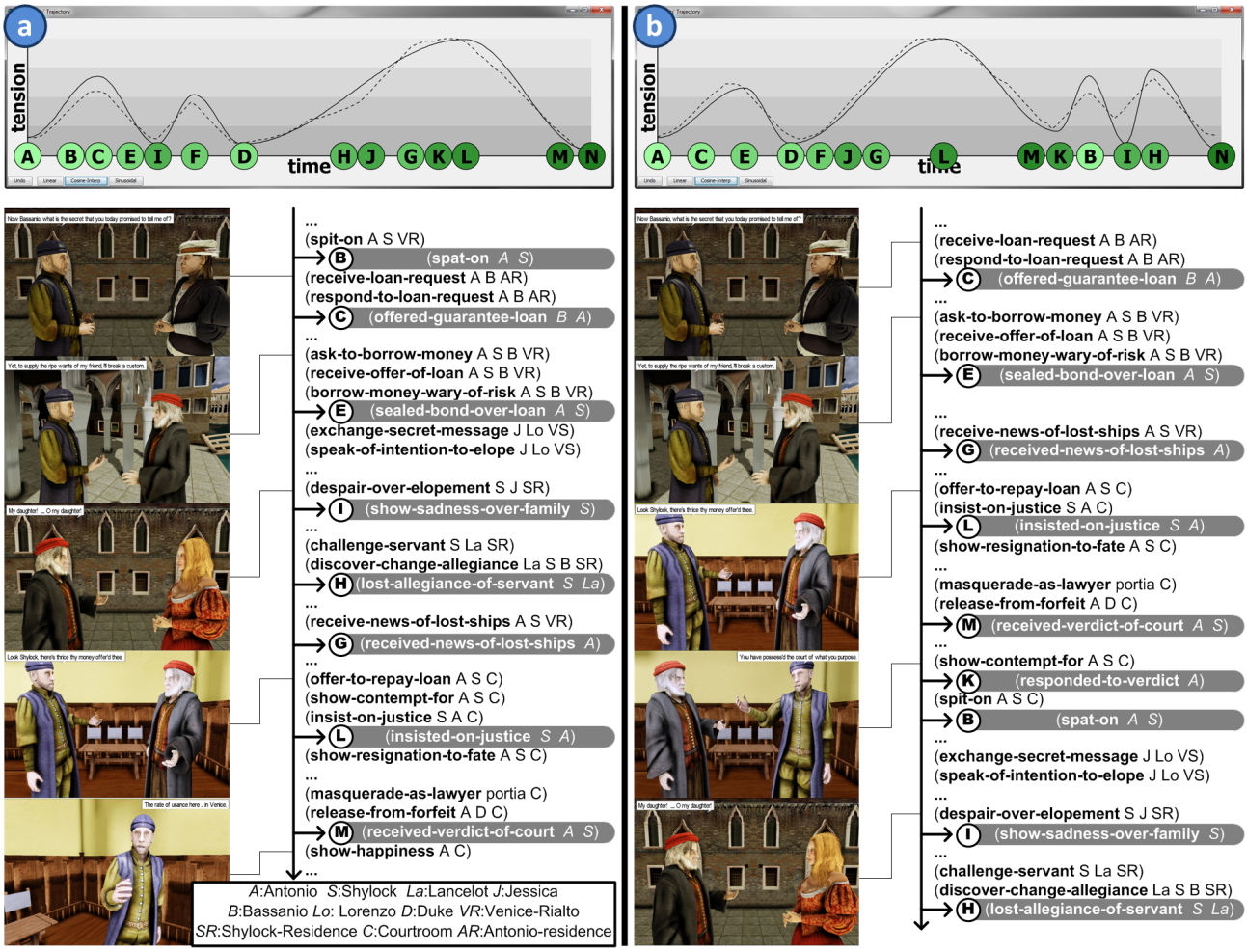


Figure 6: Parts (a) and (b) both show a Narrative Arc Window with user drawn arc and best fit system arc. The user has invoked the planner and the system has passed the configuration of constraints shown in the arc window to the planner to drive plan generation. Selected actions and constraints from the output plan are shown running downwards. The constraints are highlighted in grey and are annotated with their label from the Narrative Arc Window. The series of screen shots running alongside the plan are taken from the 3D animation of the actions. For comparison of the arcs and output narrative see text.

tension over an extended period, with minor crises towards the end of the narrative. This user drawn arc has resulted in a different arrangement of constraints and best fit system arc to that of part (a). This has interesting consequences for the semantics of the generated narrative. In particular, the constraints relating to Shylock's bad treatment at the hands of Antonio and his unfortunate personal circumstances, constraints (B), (I) and (H), now appear towards the end of the narrative. This example is interesting because it illustrates how different narrative arcs and the plan dynamics that they specify can reveal different semantics. For narrative arc (b), the semantics are very different to those of (a): there is no justification for Shylock's merciless treatment of Antonio during the trial and any later persecution or personal suffering on the part of Shylock can be perceived as retribution.

These examples help us answer our earlier questions. The first of these was whether our approach supported the visual programming of plan dynamics. The examples in figure 6 show that this is the case and that a user is free to manipulate visual elements to specify plan dynamics. In this example, the different visual arcs specified plan dynamics which enabled the generation of narratives with different semantics.

The second question of interest to us related to whether the approach enabled user exploration of the narrative space. This has also been supported by the examples illustrated in figure 6. They show that when the planner was invoked, the different narrative arcs drawn by the user led to the generation of narratives with very different semantics. The visual programming interface enables the generation of different narratives displaying the specified dynamics and also the potential for discovering new semantic possibilities.

Finally, we were interested in showing whether our approach supports the identification of plan quality criteria that relate to features such as trajectory shape. Evidence to sup-

port this follows from our earlier observations: we illustrated how the interface can enable the abstract specification of trajectory shape along with the exploration of generated narratives. Hence users are provided with the means to assess global properties of trajectories and output narratives.

## Conclusions

The need for supporting the creation of domain models for real-world planning applications has been recognised, for example with systems such as ITSIMPLE2.0 (Vaquero et al. 2007) and GIPO (Simpson, Kitchin, and McCluskey 2007). Our work shares the same spirit as these but addresses a rather different issue with the support for non-optimal planning domains. In addition to these general approaches, other work such as SCRIBE (Medler and Magerko 2006), PRISM (Cheong et al. 2008) have targeted new media applications. Our work represents a novel contribution to this endeavour focussing on visual support for users.

The central contribution of this paper is the development of a visual programming approach which enables users to specify plan dynamics and explore global properties of output narratives. Without this sort of support users face a considerable challenge if they are required to specify plan trajectories by direct manipulation of the domain model.

In the paper we illustrated our approach with reference to an IS application we have developed. However, this approach is of general applicability to other domains where quality criteria relates to the plan trajectory shape and hence are well suited to the use of a visual representation. Investigation of this is an area for future work.

## Acknowledgements

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

Thanks also to Terry Borst for emphasising to us the role of trajectory representation in the authoring of narratives.

## References

Baier, J. A.; Bacchus, F.; and McIlraith, S. A. 2007. A heuristic search approach to planning with temporally extended preferences. In *Proceedings of the 20th Int. Joint Conf. on Artificial Intelligence (IJCAI 2007)*, 1808–1815.

Bonet, B., and Geffner, H. 1999. Planning as heuristic search: New results. In *Proc. of the European Conference on Planning (ECP 1999)*, 360–372. Springer.

Cavazza, M.; Lugrin, J.; Pizzi, D.; and Charles, F. 2007. *Madame Bovary* on the Holodeck: Immersive Interactive Storytelling. In *Proc. of the 15th Int. Conf. on Multimedia (ACMMM 2007)*, 651–660.

Chapman, D. 1987. Planning for conjunctive goals. *Artificial Intelligence* 32:333–377.

Chen, Y.; Wah, B.; and Hsu, C. 2006. Temporal Planning using Subgoal Partitioning and Resolution in SGPlan. *Journal of Artificial Intelligence Research (JAIR)* 26:323–369.

Cheong, Y.; Kim, Y.; Min, W.; Shim, E.; and Jim, J. 2008. PRISM: A Framework for Authoring Interactive Narratives. In *Proc. of 1st Int. Conf. on Interactive Digital Storytelling*.

Geib, C. W. 1994. The Intentional Planning System: ItPlanS. In *Proceedings of the 2nd Int. Conf. on Artificial Intelligence Planning Systems (AIPS-94)*.

Hinely, J. L. 1980. Bond Priorities in The Merchant of Venice. *Studies in English Literature, 1500-1900* 20(2):217–239.

Hoang, H.; Lee-Urban, S.; and Munoz-Avila, H. 2005. Hierarchical Plan Representations for Encoding Strategic Game AI. In *Proceedings of Artificial Intelligence and Interactive Digital Entertainment Conference (AIIDE-05)*. AAAI Press.

Hoffmann, J., and Nebel, B. 2001. The FF Planning System: Fast Plan Generation through Heuristic Search. *Journal of Artificial Intelligence Research (JAIR)* 14:253–302.

Hoffmann, J.; Porteous, J.; and Sebastia, L. 2004. Ordered Landmarks in Planning. *Journal of Artificial Intelligence Research (JAIR)* 22:215–278.

Hoffmann, J. 2003. The Metric-FF Planning System: Translating "Ignoring Delete Lists" to Numeric State Variables. *Journal of Artificial Intelligence Research* 20:291–341.

Kelly, J. P.; Botea, A.; and Koenig, S. 2007. Planning with Hierarchical Task Networks in Video Games. In *Proceedings of the ICAPS-07 Workshop on Planning in Games*.

Mateas, M., and Stern, A. 2002. A Behavior Language for Story-Based Believable Agents. *IEEE Intelligent Systems* 17:39–47.

Medler, B., and Magerko, B. 2006. SCRIBE: A Tool for Authoring Event Driven Interactive Drama. In *Technologies for Interactive Digital Storytelling and Entertainment (TIDSE)*.

Porteous, J.; Cavazza, M.; and Charles, F. 2010. Applying Planning to Interactive Storytelling: Narrative Control using State Constraints. *ACM Transactions on Intelligent Systems and Technology (ACM TIST)* 1(2):1–21.

Riedl, M. O., and Young, R. M. 2010. Narrative Planning: Balancing Plot and Character. *Journal of Artificial Intelligence Research* 39:217–267.

Riedl, M. 2009. Incorporating Authorial Intent into Generative Narrative Systems. In *Proc. of AAAI Spring Symposium on Intelligent Narrative Technologies*.

Scott, C., and Nowak, R. 2006. Robust Contour Matching Via the Order-Preserving Assignment Problem. *IEEE Transactions on Image Processing* 15(7):1831–1838.

Simpson, R.; Kitchin, D.; and McCluskey, L. 2007. Planning domain definition using GIPO. *Knowledge Engineering Review* 22(2).

Vaquero, T.; Romero, V.; Tonidandel, F.; and Silva, J. 2007. itSIMPLE 2.0: An Integrated Tool for Designing Planning Domains. In *Proc. of the Int. Conf. on Automated Planning and Scheduling (ICAPS 2007)*.

Young, R. M. 2000a. Creating Interactive Narrative Structures: The Potential for AI Approaches. In *AAAI Spring Symposium in Artificial Intelligence and Entertainment*. AAAI Press.

Young, R. M. 2000b. Notes on the use of Plan Structures in the Creation of Interactive Plot. In *Working Notes of the AAAI Fall Symposium on Narrative Intelligence*.