Directing Intentional Superposition Manipulation

Justus Robertson[1] and **Adam Amos-Binks**[1] and **R. Michael Young**[2]

Liquid Narrative Group

[1] Department of Computer Science
NC State University
Raleigh, NC, USA, 27695
{jjrobert|aaamosbi}@ncsu.edu

[2] School of Computing and
The Entertainment Arts and Engineering Program
University of Utah
Salt Lake City, UT, USA, 84112
young@eae.utah.edu

Abstract

Strong story interactive narratives (IN) are stories that branch based on participant actions where all branches conform to a set of predefined constraints. However, participants in these systems may create branches where the constraints no longer hold. Strong story *experience management*, the process of generating IN trees, can be viewed as a game where the experience management agent wins if the story constraints hold during gameplay and loses if they are broken. In domains where the player has incomplete information of the story world, the experience manager can take action by shifting the player between alternate states that are consistent with the player's observations in order to maximize the probability that constraints will hold. This process is called *superposition manipulation*. In this paper we present a method of estimating the number of goal states reachable from different states in order to make informed decisions during superposition manipulation.

Introduction

Interactive narratives are branching stories whose events change based on actions a player takes during gameplay. Experience managers are agents that automatically generate and control interactive narratives. Strong story experience managers (Riedl and Bulitko 2013) are agents that generate interactive narratives using a central data structure that controls non-player character (NPC) behaviors. Strong story experience managers are beneficial when domain authors want to preserve an amount of control over what events happen during the interactive story. However, in order to retain authorial control while generating interactive narratives, strong story experience managers must solve the *boundary problem* (Magerko 2007) or *narrative paradox* (Louchart and Aylett 2003). This problem arises when a player deviates from the author's constraints and creates a branch where authorial control no longer holds.

One way to retain authorial control when the player breaks constraints is to perform *perceptual experience management* (Robertson and Young 2014a; 2015). Perceptual experience management uses a model of player knowledge to construct a perceptually realistic interactive narrative. A perceptually realistic interactive narrative is one that appears consistent to the player, but may be dynamically modified and arranged by the experience manager. A perceptual experience manager can modify past story events and story world update mechanics, as long as these modifications don't contradict what the player has observed in the story world. However, one drawback to these methods is they passively wait until the player has nullified an authorial constraint to modify the story world. This passive approach may miss opportunities to steer the player into world states that avoid dead end branches where author constraints no longer hold.

Superposition manipulation (Robertson and Young 2016) addresses this problem by explicitly modeling all consistent world states during perceptual experience management as a superposition of possible worlds from the player's perspective. Whenever the player makes an observation that differentiates between two or more states in their superposition, the model splits into two or more superpositions based on whether they observe the statement to be true or false. At that point, the experience manager must decide which of the split superpositions to reveal to the player as existing in the world. The experience manager should choose the superposition that maximizes the probability that the author's constraints will continue to hold in the story world.

In this paper we use a data structure called an intention dependency graph (Amos-Binks, Potts, and Young 2017) to estimate the story plans consistent with author constraints from a given superposition state in intentional planning domains (Riedl and Young 2010). We use this information to give utility estimates of different states when a superposition is collapsed and use the estimates to choose which collapsed superposition will be shown to the player in the story world.

Related Work

This paper presents a heuristic for choosing between superpositions while performing perceptual experience management. The experience management framework is grounded in plan-based story generation (Young et al. 2013; Porteous, Cavazza, and Charles 2010). Specifically, it operates on plan-based stories driven by characters who act intentionally toward their goals. The original system to perform intentional planning was a least commitment (Weld 1994) system called IPOCL (Riedl and Young 2010). Since IPOCL was first described, intentional planning has been achieved

in an off-the-shelf state space planner (Haslum 2012), a multiple agent framework (Teutenberg and Porteous 2013), and a specialized narrative state space planner that supports conflict (Ware and Young 2014). This paper performs superposition manipulation on intentional planning domains.

Automated planning's generative nature leads to a disconnect between the formulation of a domain-problem pair and the resulting structure of the solutions. Obtaining satisfying performance and results from a planner often requires analyzing the properties of the planning problem in advance of the actual planning process and is known as problem formulation. First efforts into problem formulation centered on capturing application context knowledge in a common representation (McCluskey and Porteous 1997), later developed into the Planning Domain Description Language (PDDL) (McDermott et al. 1998). Problem formulation has also led to the development of the planning graph (Blum and Furst 1997) and subsequent extensions (e.g. (Hoffmann and Nebel 2001)) that increased search speed.

The intention dependency graph (IDG) (Amos-Binks, Potts, and Young 2017) uses a modified planning graph (Ware and Young 2014) to leverage a fundamental property of intentional plans: that every action in a solution plan is part of one or more character intentions. While a planning graph is being constructed, the IDG explicitly keeps track of character intentions that depend on one another. These dependencies are combined into exemplar trajectories that consist of conjunctions of intentions, each exemplar representing a class of solution plans. The exemplar trajectories of an IDG enables us to estimate the number of solution classes to a planning problem. As we describe below, representing a superposition as a planning problem would enable IDG construction, and in turn the IDG's exemplar trajectories can be used to estimate the classes of wins.

The system presented in this paper is a strong story experience manager that uses planning (Riedl and Bulitko 2013). Plan-based experience managers use a process called mediation to control interactive experiences. Mediation is a plan-based process of creating branching story structures from an exemplar narrative. The original mediation algorithm was created for the Mimesis system (Riedl, Saretto, and Young 2003) and used a least commitment planner. A later algorithm called ASD (Riedl et al. 2008) focused on a tiered replanning strategy for generating branches. The PAST system (Ramirez and Bulitko 2014) was built on ASD and introduced player modeling to the mediation process to tailor interactive stories for participants. The GME system (Robertson and Young 2014b; 2014c) introduced a state space model for mediation and a procedural content generation pipeline that automatically configured an interface for the player from the underlying state transition model.

This paper presents a heuristic for choosing between world models consistent with player observations in a perceptual simulation. Perceptual simulations were first used to reduce the processor load for large scale crowd simulations in sandbox games (Sunshine-Hill and Badler 2010) by retroactively creating alibis that explain away randomized NPC behavior. In narrative domains, perceptual simulations are similar to the process of initial state revision (Riedl and Young 2005; Ware and Young 2010), a process of dynamically modifying the initial state of a narrative planning problem to better facilitate author goals, and late commitment (Swartjes, Kruizinga, and Theune 2008), a method of improvising new world states during emergent narrative gameplay. A crowd-sourced narrative model called plot graphs (Li et al. 2013) has also been used to generate alibis for NPCs (Li et al. 2014). Methods called event revision (Robertson and Young 2013; 2014a) retroactively replans past story events, and domain revision (Robertson and Young 2015), which retroactively shifts between story world mechanics, have been used to generate alternate story events and world mechanics in interactive domains.

Superposition Manipulation

Superposition manipulation (Robertson and Young 2016) combines adversarial gameplay, experience management, and perceptual simulation in a framework that shifts users between possible worlds in order to maintain authorial control over an interactive story world. The approach differs from previous plan-based perceptual interactive narrative simulations by proactively shifting players between alternate possible worlds. Previous mediation-based perceptual simulations wait until the player takes an exceptional action (Riedl, Saretto, and Young 2003; Harris and Young 2009), or one that deviates from the current plot, before searching through alternate possible worlds to maintain author constraints. The strength of superposition manipulation is in its proactive maintenance of a model of all possible worlds. Superposition manipulation transitions the player between those world models whenever he or she makes an observation that collapses the superposition model. This paper characterizes the utility of possible world state superpositions with intention dependency graphs in order to choose between them when a player makes a differentiating observation. Here we introduce the superposition manipulation method and examine how a predictive model of state utility can be used to shift the player between possible worlds whenever a superposition is collapsed.

Adversarial Experience Management

Superposition manipulation views experience management as an adversarial game played by the interactive storyteller and a player. The Oz Project (Weyhrauch 1997; Mateas 1999) first viewed experience management through this lens and is useful because it allows the system to quantify story outcomes as good or bad. The experience manager "wins" the game if a series of events plays out in the story world that corresponds to a set of constraints given by an external author. The manager "loses" if the constraints are broken.

A main assumption made when using adversarial experience management is that important experience features like story states (e.g., a happy ending) or narrative structures (e.g., conflict) can be modeled as a set of constraints on trajectories through the story world. For example, all solutions in this paper's framework contain intentionally-driven character actions. Any trajectory where characters act in opposition to their desires is considered a "loss" by our system.

These states and features are called *author constraints*, and adversarial experience management assumes that if the system maintains these constraints over the course of an interactive story it results in a desired player experience. Conversely, it is assumed that if the system cannot maintain the constraints, it results in an undesirable player experience.

State Utility

In an adversarial experience management framework, the utility of any state is the probability that author constraints will hold from that state until the end of the experience. In our framework, this probability can be quantified given two components: the set of possible trajectories from the current state and the actions a participant will take in each trajectory. The set of trajectories can be calculated by fully expanding the game tree under the state. The actions a participant will take may not be known ahead of time, but may be approximated with a model of choice preference (Yu and Riedl 2013), goal recognition (Cardona-Rivera and Young 2015), and/or role assignment (Domınguez et al. 2016). This paper assumes that players act according to a uniformly random distribution, but this assumption is modular. A more robust predictive model of player action can be inserted into this framework and it will update its utility predictions.

Knowledge Model

Superposition manipulation operates on a model of player knowledge. Like the predictive model of player action, the player knowledge model is a black box that can be removed, expanded, and re-inserted and the rest of the framework will operate effectively. The current knowledge model is simply that story characters observe every action and object they are co-located with. According to our model, players believe that the effects of every action they observe obtain in the state resulting from the action's execution and players believe every property that holds of each object they are co-located with as soon as they co-locate.

Expanding and Collapsing State Superpositions

Instead of players existing in a single state, superposition manipulation allows players to exist in a collection of states consistent with their knowledge model. This collection of possible current states is called the player's *state superposition*. The superposition is expanded whenever a story world character takes an unobserved action. If the player exists in a superposition of a single state and an unobserved story character has a choice between two actions, the player's state superposition will grow from one to two states. The two new states will represent the two possible successor states created for each of the story character's two unobserved actions.

The state superposition collapses whenever a literal that is true in some superposition states but false in other superposition states is observed by the player. At this point, the system must decide whether it will show the player that the literal is true or false. This observation, once made, will collapse the superposition into the set of worlds where the literal is true or the set of worlds where the literal is false. These two mechanics, expansion and collapse, allow superposition manipulation to track the set of possible worlds that are consistent with the player's knowledge model at any point in an interactive story. The problem of superposition manipulation is choosing how superpositions should collapse to maximize the probability that author constraints are maintained.

Superposition Utility

The final component of superposition manipulation is determining the superposition to transition the player to when a new observation collapses their superposition. This choice should be made based on the superposition that offers the highest utility of preserving author constraints. This utility function can be calculated with four components: number of winning goal states, number of losing dead end states, a predictive model of player action, and a process that determines the utility of superpositions from its individual states.

This utility calculation depends on the composition of the space below each state in the superposition, the space's number of goal states vs. dead end states, and the actions a player takes through each space during gameplay. A baseline approach for these calculations would be to fully expand every node under each state in each superposition until either a goal or a dead end was reached in each branch. A predictive model of player action would then assign a probability distribution on each set of choice options the player receives under each state. One open problem is how to measure the utility of a collection of superposed states without fully expanding their underlying game trees. The next section introduces a data structure that enables us to estimate the number of reachable goal states below a single state.

Intention Dependency Graphs

In most domains, representing multiple states in opposing superpositions requires an intractable amount of time and space to fully expand game trees beneath states. To address this limitation, we investigate the effectiveness of using a heuristic function that can approximate the full utility function while taking less time and space. This paper uses a data structure called an *intention dependency graph* to approximate superposition manipulation's full utility function. This section describes intention dependency graphs and how they can be used to calculate the first (winning goal states) of the four components of a superposition utility function.

An intention dependency graph (IDG) computes an estimate of the number of different solutions to a story planning problem. As we describe in more detail below, it accomplishes this by explicitly keeping track of dependencies between character goals. It is constructed separately from, but during, the construction of another problem formulation methodology, the planning graph used by Glaive (Ware and Young 2014). Classes of solutions are characterized by exemplar trajectories consisting of character goal conjunctions.

A solution to a story planning problem differs from a classical planning solution on one key property. Every action in a story planning problem must be causally linked to achieving at least one goal of some character agent. Thus, every solution to a story planning problem will have a derived set of character goals that the solution's actions support. The

IDG leverages this property in two ways. The first is to enable efficient construction. Secondly, exemplar trajectories capture conjunctions of character goals that characterize a solution enabling a planner to focus its search. We discuss each in turn in the following two sections.

IDG Construction

When a player character makes an observation, we must decide how to collapse the superposition that is consistent with the players knowledge model so we are best positioned for a win. To determine this, each state within a superposition is captured in its own story planning problem, consisting of an initial state, goal state, domain, objects, and character agents. Differences in superposition states are represented as different initial states. We can then construct an IDG for each story planning problem.

We construct the IDG in an efficient manner by extending the modified planning graph developed for the Glaive story planner (Ware and Young 2014). Modifying Glaive's planning graph enables the IDG to be constructed at the same time in a separate graph structure. Planning graphs provide an efficient structure to estimate the remaining cost to a solution. They alternate layers of propositions and actions to represent a relaxed version of a planning problem by ignoring steps' delete lists. A first layer represents the propositions true in the initial state and then layers of applicable actions and their effects are added until the goal conditions are reached. The Glaive story planner extends the planning graph to intentional planning by ensuring all actions added to a plan graph are potentially motivated. That is, an action is only added after it is confirmed to exist on an action sequence towards a character goal. This is confirmed through a structure called a goal graph that captures action sequences of a character acting towards achieving their goal. A character goal is in the final layer of a goal graph and is preceded by layers of actions, where each action is causally connected to at least one action in the next layer. The result is a graph structure with a set of possible action sequences to accomplish a goal. Before the plan graph is constructed, a goal graph is constructed for each character goal in the planning problem. When an action is added to the planning graph, the IDG keeps track of the character goal that provides the motivation. As new actions are added to the planning graph, the IDG determines if other character's goals were required to establish the actions preconditions. Using the planning graph to determine the underlying dependencies between character goals enables the IDG to obtain information about the solution space without incurring the cost of generating multiple plans.

To represent the IDG, we use a directed layered graph and provide a formal definition:

Definition 1 (Intention dependency graph (IDG))**.** The IDG of a planning problem Φ, $IDG(\Phi)$, is a directed layered graph represented by a tuple of three elements $\langle V, E, f \rangle$ where V is a set of n vertices $v_0, v_1 \ldots, v_{n-2}, v_g$ each with a label, $L(v_i)$, consisting of a set of character goals and the graph layer it is positioned in. E is a set of directed edges between vertices, and f a vertex labeling function that takes sets of character goals as input and outputs the union set.

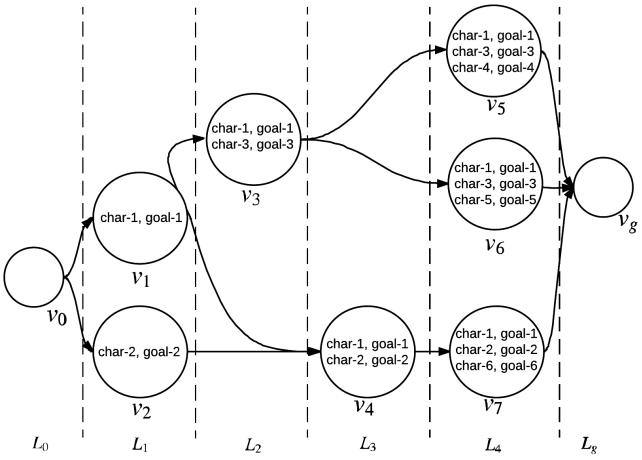

Figure 1: IDG with three exemplar trajectories, each containing three character goals, connected to v_g

Intuitively, a vertex is added to the IDG under two conditions. The first is when an action has been added to the planning graph that supports a new character goal being pursued (e.g. v_1,v_2 in Figure 1). A second is when we learn through the story planning graph that an existing character goal depends on another character's actions (v_4). A new vertex is added with the two character goals as labels. Edges are added between vertices when a new dependencies is identified. The vertex that results is connected to the necessary vertices in previous layers.

Exemplar Trajectories

The final layer of an IDG contains a single vertex (v_g) that is connected to neighbor vertices in previous layers (Figure 1). These vertices are referred to as exemplar trajectories and are a second way that the IDG leverages intentional plans.

An IDG maintains explicit dependencies between character goals, represented by a vertex label consisting of a list of character-goal pairs. When a vertex is connected to v_g, the character-goal pairs list represents a conjunction of character-goal pairs that could solve the story planning problem. This property allows the IDG to partition the search space into exemplar trajectories representative of plans with the same character-goal pair. Exemplar trajectories can then be used to direct a story planner to find solutions that differ by character goals and still respect authorial constraints.

While obtaining an estimate of the solution space rather than solving it explicitly has speed advantages, we lose some accuracy. For the purpose of superposition manipulation, the biggest limitation is that the planning graph does not use *mutex* edges to capture precedence and interference relationships between actions. This results in unreachable plans being considered in the planning graph and consequently overestimating the number of exemplar trajectories in the IDG.

We use the exemplar trajectories to inform an experience manager of opportunities to proactively change the player's state and choose the appropriate superposition.

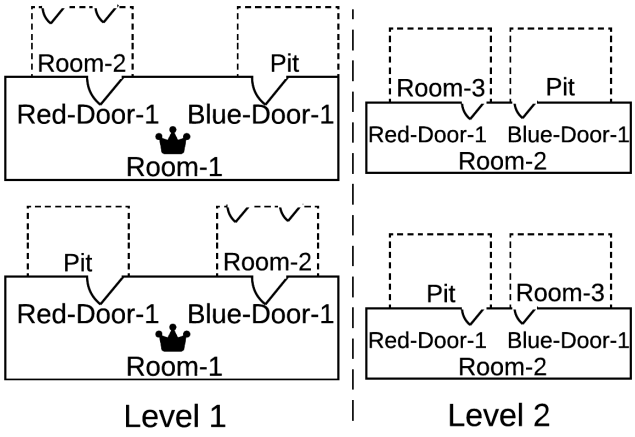

Figure 2: Room configurations after each of Merlin's actions

Superposition utility using IDGs

Our utility function to calculate the probability we reach a goal state from superposition is based on the IDG exemplar trajectories. Recall that exemplar trajectories captured by the IDG of a planning problem each represent a class of solutions with the same character-goals. Exemplar trajectories can then used to direct an intentional planner to find solutions in a specific part of the search space.

We calculate our utility function as the number of exemplar trajectories for each state within a superposition and the chosen superposition as the $argmax$ in Equation 1,

$$\upsilon(SP) = \underset{sp \in SP}{\operatorname{argmax}} \sum_{s \in sp} |\eta(s)|, \tag{1}$$

where SP is the set of superpositions being evaluated, η is the exemplar trajectories of a superposition state s.

One constraint of this approach is that we do not estimate the number of plans each exemplar represents.

Example

In this section we present a toy domain to illustrate the mechanics of IDG as a heuristic for choosing between competing superpositions during a collapse after a new player observation. The toy problem represents a procedurally configured dungeon where a dungeon master places pre-created rooms together as a player explores the space. In the example problem there are three pre-created rooms: Room1, Room2, and Room3. The player, named Arthur, begins in Room1. The player's intention and the author's goal is for the player's character to arrive at Room3. There are two doors in Room1 and Room2, a red door and a blue door. All four doors are initially connected to a void, represented in the planning problem as an open pit. The player has a key for each room that opens one of its doors, red or blue, but once the key has been used once it must be discarded. Once the player has opened a door with a key they observe the room the door leads to. To maintain the illusion that the dungeon is static, once a door has been opened no room can be connected the other side. It is the job of the dungeon wizard character, Merlin, to attach rooms to doors in order to lead the player to Room3 while avoiding the situation in which the player opens a door to a pit. Merlin has an intention to connect all rooms to all possible doors, so he is free to act as he pleases in order to achieve the author's goal.

Merlin acts first. The four possible actions Merlin can take are: {*(place Merlin Room1 Room2 Blue-Door1 Pit 1),(place Merlin Room1 Room2 Red-Door1 Pit 1),(place Merlin Room2 Room3 Blue-Door2 Pit 2),(place Merlin Room2 Room3 Red-Door2 Pit 2)*}. The resuting configurations are represented in Figure 2. Since Merlin is not located in the room with the player's character, all of his actions are unobserved and a single superposition is created of four individual states that correspond to the successors of the four enabled, unobserved Merlin actions. There is a state where Room2 is connected to Blue-Door1, a state where it is connected to Red-Door1, a state where Room3 is connected to Blue-Door2, and a state where Room3 is connected to Red-Door2. At this point, the player faces a decision of opening Blue-Door1 or Red-Door1. For the purpose of our example, the player chooses to open Red-Door1.

When the player chooses to open Red-Door1 they will make an observation of what room the door leads to. This will split the superposition into two new superpositions. One superposition (sp_1) contains a single state representing Merlin executing a *place* action of Room2 behind Red-Door-1 so the player may observe Room2. Another superposition (sp_2) is a collection of three states where Merlin took *place* actions resulting in Room2 being placed elsewhere. At this point, the IDG is constructed to assess the utility of each possible superposition at level 1.

Recall from Equation 1 that our utility function is the total number of exemplar trajectories in a superposition. In level 1 we need to determine the utility between of superposition sp_1 and sp_2. Since sp_1 contains a single state, only a single IDG is constructed to assess its utility. The IDG is constructed from a planning problem that uses the resulting state after Merlin places Room2 behind Red-Door-1 the initial state. The domain, goal state and objects are those outlined in Figure 3. The IDG results in a single exemplar trajectory where Merlin's goal is to connect Room1-3 and Arthur's goal is to get to Room3, captured as a single win in row 2, column 4 in Table 1.

The utility of sp_2 is calculated from three states that result from three Merlin actions that lead to Arthur not observing Room2. Each state serves as the Initial State to three planning problems, and an IDG is created for each planning problem. No exemplar trajectories were found in the IDGs, resulting in a utility of 0. For level 1, we apply our utility function (Equation 1) over our set of superpositions (SP). We find that while sp_1 only has one state associated with it, it provides the possibility of a win (the IDG may overestimate the wins) in the form of a single exemplar trajectory, whereas despite having three states, sp_2 has zero potential wins, thus we proceed with sp_1 (column 5, Table 1).

In level two of our example, we have two superpositions (sp_3 and sp_4) each with a single state. Once the states have been translated to planning problems and the IDGs constructed, we see that sp_3 and sp_4 have one and zero exem-

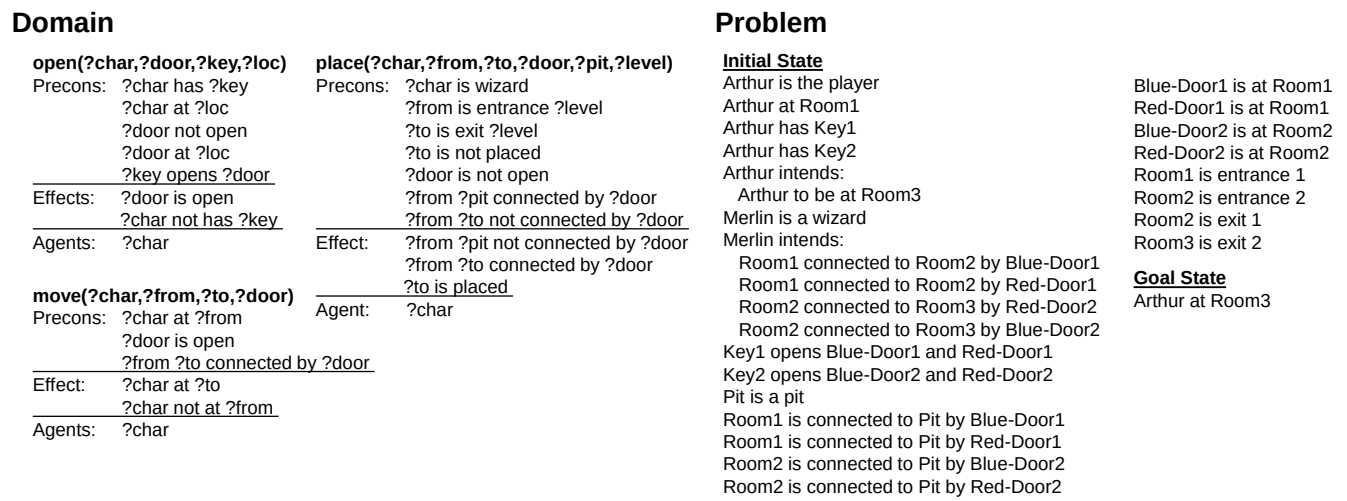

Figure 3: A simplified representation of PDDL for the intentional planning domain and problem used for an example.

Level	Superposition	States	$\lvert\eta(sp)\rvert$	$v(SP)$
1	sp_1	1	1	sp_1
1	sp_2	3	0	
2	sp_3	1	1	sp_3
2	sp_4	1	0	

Table 1: IDG results for each superposition

plar trajectories, respectively (rows 4, 5 in Table 1). Since sp_3 has more possibilities of reaching the goal state, Equation 1 will choose it as the most desirable superposition to be in.

Discussion and Future Work

This document describes how an IDG can be used to estimate the first of the four components that make up the full superposition manipulation utility function. While using wins alone for individual states is a good start and performs better than a random baseline as a utility estimate, it lacks information necessary to identify the optimal decision in all situations. First, leaving losses out of the utility function can lead to the wrong decisions. For example, two states could have a small difference in the number of winning states beneath them but a large difference in number of losing states. If the first state, s_1, has w number of wins and a second state, s_2, has $w + 1$ wins then according to the utility function in this paper $s2$ is the better option. However, if s_1 has l losses and s_2 has $l+1000$ losses, then the ratio of wins to losses for s_1 is much more beneficial than the ratio of losses for s_2. In this situation, this paper's utility function would choose s_2 when s_1 is the optimal choice. A more robust utility function would calculate losses in addition to wins.

Second, omitting player decisions from the utility function can lead to the wrong decisions. For example, if sp_1 has three outgoing actions the player can choose from where one choice leads to a goal state but two choices lead to dead ends and sp_2 also has three outgoing actions the player can choose from where two choices lead to a goal state and only a single choice leads to a dead end, but this paper's utility estimation and the win + loss utility function described in the last paragraph will choose state sp_2. However, both utility functions assume the player is equally likely to choose any of their possible actions but if the player reliably chooses the action that leads to a goal state from sp_1 and the action that leads to a dead end from sp_2 both utility functions will choose wrong. If the utility function uses a probability distribution over the player's possible actions it can make a more informed decision. For example, if a predictive model of player behavior assigned a 70% probability of the player taking the winning edge in sp_1 and a 70% probability of the player taking the losing edge in sp_2 will lead to a utility function that incorporates player choice model making a correct decision when the previous functions did not.

Third, even a utility function that incorporates a predictive player behavior model only gives the utility for single states, not sets of states that represent superpositions. Once utility calculations have been made for each individual state within each superposition, a second round of decision making must take place to determine which aggregate set of states presents the best probability of reaching a goal state. A full utility function should be able to synthesize utility information from individual states and make a full calculation over sets of superposed states.

A last element of future work is that while the IDG provides a heuristic for collapsing superpositions in intentional domains, the current superposition manipulation framework does not support expanding intentional domains. Intentions allow domain authors to specify the goals that characters will pursue and thus the actions that can be added to a plan. Observed actions in a superposition manipulation framework are driven by an intentional planner, but unobserved actions are used to expand superpositions. In order to pre-

serve the intentional planning constraint of only intentionally motivated actions being performed by NPCs, superposition manipulation must differentiate between intentional and non-intential character actions when adding states to a superposition and only adding intentional actions.

Acknowledgements

This material is based upon work supported in whole or in part with funding from the Laboratory for Analytic Sciences (LAS). Any opinions, findings, conclusions, or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the LAS and/or any agency or entity of the United States Government.

References

Amos-Binks, A. A.; Potts, C.; and Young, R. M. 2017. Planning Graphs for Efficient Generation of Desirable Narrative Trajectories. In *Intelligent Narrative Technologies Workshop*.

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

Cardona-Rivera, R. E., and Young, R. M. 2015. Symbolic Plan Recognition in Interactive Narrative Environments. In *Intelligent Narrative Technologies Workshop*.

Domınguez, I. X.; Cardona-Rivera, R. E.; Vance, J. K.; and Roberts, D. L. 2016. The Mimesis Effect: The Effect of Roles on Player Choice in Interactive Narrative Role-Playing Games. In *Proceedings of the ACM Conference on Computer-Human Interaction (CHI)*.

Harris, J., and Young, R. M. 2009. Proactive Mediation in Plan-Based Narrative Environments. *IEEE Transactions on Computational Intelligence and AI in Games* 1(3):233–244.

Haslum, P. 2012. Narrative Planning: Compilations to Classical Planning. *The Journal of Artificial Intelligence Research* 44:383–395.

Hoffmann, J., and Nebel, B. 2001. The FF planning system: Fast plan generation through heuristic search. *The Journal of Artificial Intelligence Research* 14:253–302.

Li, B.; Lee-Urban, S.; Johnston, G.; and Riedl, M. 2013. Story Generation with Crowdsourced Plot Graphs. In *Proceedings of the Annual Conference of the Association for the Advancement of Artificial Intelligence*.

Li, B.; Thakkar, M.; Wang, Y.; and Riedl, M. O. 2014. Data-Driven Alibi Story Telling for Social Believability. In *Social Believability in Games Workshop*.

Louchart, S., and Aylett, R. 2003. Solving the Narrative Paradox in VEs – Lessons from RPGs. In *Proceedings of the Conference on Intelligent Virtual Agents*, 244–248.

Magerko, B. 2007. Evaluating Preemptive Story Direction in the Interactive Drama Architecture. *Journal of Game Development* 2(3):25–52.

Mateas, M. 1999. An Oz-centric review of interactive drama and believable agents. In *Artificial intelligence Today*. Springer. 297–328.

McCluskey, L., and Porteous, J. 1997. Engineering and compiling planning domain models to promote validity and efficiency. *Artificial Intelligence* 95(97):1–65.

McDermott, D.; Ghallab, M.; Howe, A.; Knoblock, C.; Ram, A.; Veloso, M.; Weld, D.; and Wilkins, D. 1998. PDDL-the planning domain definition language. In *International Conference on Artificial Intelligence and Planning Systems*.

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* 1(2):10.

Ramirez, A., and Bulitko, V. 2014. Automated Planning and Player Modelling for Interactive Storytelling. *IEEE Transactions on Computational Intelligence and AI in Games* 375–386.

Riedl, M., and Bulitko, V. 2013. Interactive Narrative: An Intelligent Systems Approach. *AI Magazine* 34(1):67–77.

Riedl, M. O., and Young, R. M. 2005. Open-World Planning for Story Generation. In *International Joint Conference on Artificial Intelligence*, 1719–1720.

Riedl, M. O., and Young, R. M. 2010. Narrative Planning: Balancing Plot and Character. *The Journal of Artificial Intelligence Research* 39(1):217–268.

Riedl, M. O.; Stern, A.; Dini, D. M.; and Alderman, J. M. 2008. Dynamic Experience Management in Virtual Worlds for Entertainment, Education, and Training. *International Transactions on Systems Science and Applications* 4(2):23–42.

Riedl, M.; Saretto, C. J.; and Young, R. M. 2003. Managing Interaction Between Users and Agents in a Multi-Agent Storytelling Environment. In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems*, 741–748.

Robertson, J., and Young, R. M. 2013. Modelling Character Knowledge in Plan-Based Interactive Narrative to Extend Accomodative Mediation. In *Intelligent Narrative Technologies Workshop*, 93–96.

Robertson, J., and Young, R. M. 2014a. Finding Schrödinger's Gun. In *Proceedings of the Conference on AI and Interactive Digital Entertainment*, 153–159.

Robertson, J., and Young, R. M. 2014b. Gameplay as On-Line Mediation Search. In *Experimental AI in Games Workshop*.

Robertson, J., and Young, R. M. 2014c. The General Mediation Engine. In *Experimental AI in Games Workshop*, 65.

Robertson, J., and Young, R. M. 2015. Interactive Narrative Intervention Alibis through Domain Revision. In *Intelligent Narrative Technologies Workshop*.

Robertson, J., and Young, R. M. 2016. A Model of Superposed States. In *Experimental AI in Games Workshop*, 65–71.

Sunshine-Hill, B., and Badler, N. I. 2010. Perceptually Realistic Behavior through Alibi Generation. In *Proceedings of the Conference on AI and Interactive Digital Entertainment*, 83–88.

Swartjes, I.; Kruizinga, E.; and Theune, M. 2008. Lets Pretend I Had a Sword. *Interactive Storytelling* 264–267.

Teutenberg, J., and Porteous, J. 2013. Efficient Intent-Based Narrative Generation using Multiple Planning Agents. In *Proceedings of the 2013 International Conference on Autonomous Agents and Multi-Agent Systems*, 603–610.

Ware, S. G., and Young, R. M. 2010. Rethinking Traditional Planning Assumptions to Facilitate Narrative Generation. In *AAAI Fall Symposium: Computational Models of Narrative*.

Ware, S. G., and Young, R. M. 2014. Glaive: A State-Space Narrative Planner Supporting Intentionality and Conflict. In *Tenth Conference on Artificial Intelligence and Interactive Digital Entertainment*.

Weld, D. S. 1994. An Introduction to Least Commitment Planning. *AI Magazine* 15(4):27.

Weyhrauch, P. 1997. *Guiding Interactive Drama*. Ph.D. Dissertation, Carnegie Mellon University Pittsburgh, PA. CMU-CS-97-109.

Young, R. M.; Ware, S. G.; Cassell, B. A.; and Robertson, J. 2013. Plans and Planning in Narrative Generation: A Review of Plan-Based Approaches to the Generation of Story, Discourse and Interactivity in Narratives. *SDV. Sprache und Datenverarbeitung*.

Yu, H., and Riedl, M. O. 2013. Data-Driven Personalized Drama Management. In *Ninth Conference on Artificial Intelligence for Interactive Digital Entertainment*, 191–197.