Intelligent Narrative Technologies and Social Believability in Games:
Papers from the AIIDE 2015 Joint Workshop

Symbolic Plan Recognition in Interactive Narrative Environments

Rogelio E. Cardona-Rivera and **R. Michael Young**
Liquid Narrative Group
Department of Computer Science
North Carolina State University
recardon@ncsu.edu, young@csc.ncsu.edu

Abstract

Interactive narratives suffer from the narrative paradox: the tension that exists between providing a coherent narrative experience and allowing a player free reign over what she can manipulate in the environment. Knowing what actions a player in such an environment intends to carry out would help in managing the narrative paradox, since it would allow us to anticipate potential threats to the intended narrative experience and potentially mediate or eliminate them. The process of observing player actions and attempting to come up with an explanation for those actions (i.e. the plan that the player is trying to carry out) is the problem of *plan recognition*. We adopt the framing of *narratives as plans* and leverage recent advances that cast *plan recognition as planning* to develop a symbolic plan recognition system as a proof-of-concept model of a player's reasoning in an interactive narrative environment. In this paper we outline the system architecture, report on performance metrics that demonstrate adequate performance for non-trivial domains, and discuss the implications of treating *players as plan recognizers*.

Introduction

Interactive narratives are systems that mediate a player's interaction within a virtual environment through a narrative framing. These systems afford their players the opportunity to step into a dramatic role to influence the development of a storyline through their actions [Riedl and Bulitko, 2013]. This type of virtual environment has become increasingly commonplace in educational, training, and entertainment contexts, but remains a challenging task to develop. One reason for this challenge is due to what Aylett [2000] calls the *narrative paradox*: for an experience to count as a story it must have some kind of satisfying structure, which participants can directly affect and make incoherent.

One way to help ameliorate the narrative paradox is to attempt to model the player's reasoning process to predict what goal the player intends to accomplish in the game, as well as how the player aims to achieve that goal. With that information, a *drama manager* [Nelson et al., 2006] could potentially adapt the interactive narrative dynamically, making sure to avoid narratively undesirable states, while guiding the player toward narratively desirable ones. In this paper we present a proof-of-concept symbolic plan recognition system within an interactive narrative. While the plan recognition system attempts to find the player's plan given their actions, we use the system as a *proxy for the player's reasoning process in an interactive narrative*. This is because we posit that as a player engages with an interactive narrative, she is trying to perform plan recognition herself by attempting to identify what author-intended narrative plan she fits into. Our approach builds upon prior work that represents *narratives as plans* [Young, 1999] from an automated planning context. We additionally present the reasons why we represent *players as plan recognizers*, as well as an evaluation of the plan recognition system's performance for a non-trivial domain.

Players in Interactive Narratives

The work outlined here is meant as a proof-of-concept of a model of one aspect of a player's reasoning in an interactive narrative environment. Specifically, we care to characterize the player's story reasoning during her interactive narrative experience: we would like to model what the player intends to accomplish given her understanding of the story, her perceived narrative role, and her perceived affordances to act within the story context [Young and Cardona-Rivera, 2011]. In this section, we outline some of the implications of considering *players as plan recognizers* within interactive narrative contexts. Our focus here is on interactive narratives from a player-centric perspective. We therefore look to what some cognitive psychologists and narratologists tell us about the way in which humans engage with narrative artifacts.

...as Problem Solvers

The concept of "gameplay" is difficult to precisely define [Salen and Zimmerman, 2003]. While we do not care to weigh in on what gameplay *is*, we do care to point out that a great deal of research work in interactive narrative play tacitly assumes that the game player acts as a *problem-solver* during gameplay [Roberts and Isbell, 2007; Riedl and Bulitko, 2013]; that is, the player is assumed to be reasoning forward through potential courses of action toward a specific goal. Indeed, prior work [Young et al., 2013] has cast a player's experience of playing through an interactive narrative as a deliberative problem solving process, computationally represented in a classical planning paradigm, which can be used to model key structural aspects of narratives, including the causal relationship between narrative events and the relative ordering between them [Young, 1999]. In this view, a player monitors the current world state, attempts to identify which planning operators have their preconditions satisfied, and choses actions that lead to a desired world state. This view is consistent with an empirical cognitive psychology account of a person's story understanding process: Gerrig and Bernardo [1994] frame *readers as problem solvers* whereby humans transport themselves into a story they are reading, and attempt to solve the plot-related problems on behalf of the protagonist(s). Given the mapping between narrative structures and planning data structures and the psychological experimental support, automated planning seems a natural fit to characterize a player's interactive narrative experience.

...as Plan Recognizers

However, according to the narratologist Herman [2013], a key component of consuming a narrative is the ascription of intentions on behalf of the audience to the narrative's author. This ascription is akin to Dennett's [1989] stance: in essence, when explaining and predicting the behavior of an object, we can choose to view it at varying levels of abstraction. Herman reviewed converging evidence in cognitive psychology and narratology that support the idea that when consuming a narrative, the default mode of ascription is in the *intentional plane*, where we are concerned with beliefs, desires, and intentions (BDI) of an object to explain its behavior. Herman [2013] claims that this level of reasoning goes beyond the attribution of BDI to characters, extending intentional attribution to the author herself. The importance of this reasoning process to interactive narrative play has been noted by Murray [1998] who stated that interactive narrative designers must constrain a player's reasoning process to identify (in the context of their play experience) the set of *dramatically appropriate* actions, those that an interactive narrative expects the player to execute to successfully advance the narrative experience. Concordantly, players are expected to make their in-game actions meaningful in the context of the designed experience; a designer affords a coherent experience if the player behaves in coherent ways [Adams, 2013]. The reasoning process by which players correctly identify the sequences of action that successfully complete their narrative experience (i.e. those intended by the game designer) is therefore key during interactive narrative play. Ascription of intent in the classical AI sense [Cohen and Levesque, 1990] involves identifying the goal the agent (in our case, the author) is attempting to pursue, and the corresponding plan that will be pursued in service of the goal. In other words, players must perform plan recognition *a priori* to being able to realize a particular plan in an interactive environment. Put simply, a player in an interactive narrative must identify how the game wants her to proceed in order to bring about the plan that achieves game progress. Plan recognition is a paradigm well suited to computationally modeling this reasoning process.

Plan Recognition

In the plan recognition problem, we seek two things: a) the goals that explain the observed actions (often referred to as the *goal recognition problem* [Sukthankar et al., 2014]), and b) the actions that will occur next in pursuit of the expected goal [Carberry, 2001]. Plan recognition appears in three different forms: a) plan recognition when the agent is aware and actively cooperating in the recognition (called *intended plan recognition*), b) plan recognition when the agent is unaware of the recognition (called *keyhole plan recognition*), and c) plan recognition when the agent is aware of and actively obstructs the recognition process (called *obstructed plan recognition*). Importantly, plan recognition assumes *a priori* that there exist a set of possible goals G_i that an agent cares to achieve. We denote the set of these disjunctive goals as $\mathcal{G} = \{G_0, \ldots, G_n\}$. Plan recognition has historically operated over a *plan library*, a set of *recipes* that record likely actions toward assumed goal states. In this light, the plan recognition problem is to identify which goal-recipe pair is the observed agent's intended course of action, given the actions that the agent has undertaken in an environment. Recognizing that plan recognition is *planning in reverse*, Ramírez and Geffner [2009] proposed a novel way of casting the plan recognition problem that avoids the need for a plan library: by assuming that an observed agent in a plan recognition task will prefer optimal plans (i.e. those plans with minimal cost), Ramírez and Geffner proposed to compile the agent's observed actions as additional goals (to the set of assumed goals) that an agent must plan for. As part of this compilation process, new planning operators are added to the domain that uniquely satisfy the observed action goals, effectively placing a lower bound on the optimal plans in the compiled domain; i.e. the agent must pursue plans in service of the assumed possible goals that *satisfy* the observations that were identified in the plan recognition task.

Definition 1. *An action sequence* $\pi = a_0, \ldots, a_n$ satisfies *the observation sequence* $OBS = obs_0, \ldots, obs_m$ *if there is a monotonic function* f *mapping the observation indices* $j = 0, \ldots, m$ *into action indices* $i = 0, \ldots, n$ *such that* $a_{f(j)} = obs_j$.

This formulation of plan recognition depends on a model of automated planning. We adopt a typical STRIPS-like representation [Fikes and Nilsson, 1971], which has been used to model key aspects of stories and the discourse about them [Young et al., 2013].

Definition 2 (Planning Domain). *A* Planning Domain *is a triplet* $\mathcal{P} = \langle F, I, A \rangle$*, with atoms* F*, initial state* $I \subseteq F$*, and action operators* A*. A state is a conjunction of function-free atoms that describe a state of the world. An action operator is a template for an action that can occur in the world, defined by* preconditions *and* effects*: preconditions describe the conditions of the world that must be true in order for the operator to execute, and effects describe the conditions made true in the world via the execution of the operator. Operators may contain variable terms to convey ideas such as* $move(x, y)$ *to convey "move from* x *to* y*".*

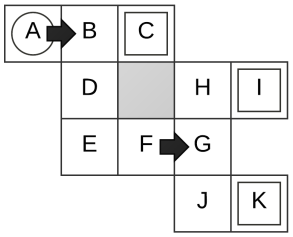

Figure 1: Plan recognition in a robot navigation domain with initial position A, possible goals $\mathcal{G} = \{C, I, K\}$, and observed unit-cost actions $move(A, B)$ and $move(F, G)$.

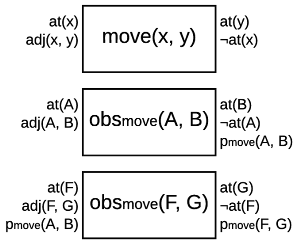

Figure 2: Observation compilation for the domain example illustrated in Figure 1 adds two actions $obs_{move(A,B)}$ and $obs_{move(F,G)}$ based on the operator $move(x, y)$. Operators are shown with their preconditions (left) and effects (right).

Example

In lieu of the full description of how to achieve plan recognition with planning, we present an example to help explain the intuition behind the procedure. For this, we leverage the example used by Ramírez and Geffner: plan recognition in a robot navigation domain, illustrated in Figure 1. In the original paper by Ramírez and Geffner, there was an implicit assumption that we make explicit here: recognized plans are represented by the goals they achieve, i.e. there is only ever *one optimal plan* that achieves a given assumed goal $G_i \in \mathcal{G}$ (if there exists more than one plan, then the selection of which plan satisfies the observations is done randomly). In Figure 1, the initial position of the robot agent is given by the circle (the agent is in room A at the start), and the possible goals for the agent are to be in any one of the squared rooms C, I, or K. In addition, we have observed the agent's unit-cost actions of moving from room A to room B, as well as moving from room F to room G. Ramírez and Geffner [2009] frame a plan recognition task as a *plan recognition over a domain theory*:

Definition 3 (Plan Recognition Theory). *A* plan recognition theory *is a triplet* $\mathcal{T} = \langle \mathcal{P}, \mathcal{G}, OBS \rangle$*, where* $\mathcal{P} = \langle F, I, A \rangle$ *is a* planning domain*,* $\mathcal{G}$ *is the set of sets of possible goals* $G_i \subseteq F$*, and* $OBS = obs_0, \ldots, obs_m$ *is an observation sequence with each* $obs_i \in A$.

The plan recognition theory T for the example domain is:

$\mathcal{P} =$
$F = \{$
$at(x)$, denoting that the agent is at location x
$adj(x, y)$, denoting that x is adjacent to y
$A \ldots K$, denoting rooms A through K
$\}$
$I = \{at(A)\}$, the agent is at room A
$A = \{move(x, y)\}$, the agent can move from x to y (with preconditions and effects as in Figure 2)

$\mathcal{G} = \{G_0 = \{at(C)\}, G_1 = \{at(I)\}, G_2 = \{at(K)\}\}$

$OBS = \{obs_0 = move(A, B), obs_1 = move(F, G)\}$

Intuitively, the plan recognition compilation proposed by Ramírez and Geffner creates two new actions $obs_{move(A,B)}$ and $obs_{move(F,G)}$ from the input observations of the plan recognition theory, as shown in Figure 2. The action $obs_{move(A,B)}$ that is created from the first observation $obs_0 = move(A, B)$ has all the preconditions and effects from the $move(x, y)$ operator in A. However, the operator is fully grounded with the preconditions $at(A), adj(A, B)$, and effects $at(B)$ and $\neg at(A)$. In addition, the operator's effects have been augmented with a fluent that denotes that the operator has executed; in essence, the effect $p_{move(A,B)}$ reifies that the operator has happened. The action $obs_{move(F,G)}$ that is created from the second observation $obs_1 = move(F, G)$ is defined in a similar manner, with the added precondition $p_{move(A,B)}$, because the observation of the action $move(A, B)$ comes *before* the observation of the action $move(F, G)$. In addition to the expansion of the planning domain operators, the compiled domain also expands the existing set of assumed goals; each assumed goal becomes a set of assumed goals, and is expanded by the atoms representing the observed actions. Thus, the transformed plan recognition theory T' is:

$\mathcal{P}' = \langle F', I', A' \rangle$
$F' = \{$
$at(x)$, $adj(x, y)$, $A \ldots K$,
$p_{move(A,B)}$, denoting that $move(A, B)$ happened
$p_{move(F,G)}$, denoting that $move(F, G)$ happened
$\}$
$I' = \{at(A)\}$
$A' = \{move(x, y), obs_{move(A,B)}, obs_{move(F,G)}\}$,

$\mathcal{G}' = \{$
$G'_0 = \{at(C), p_{move(A,B)}, p_{move(F,G)}\}$,
$G'_1 = \{at(I), p_{move(A,B)}, p_{move(F,G)}\}$,
$G'_2 = \{at(K), p_{move(A,B)}, p_{move(F,G)}\}$
$\}$
$OBS' = \{\varnothing\}$

The solution to the plan recognition theory is then found by identifying the plans of minimal cost that achieve the respective assumed goals. Each pair $(\mathcal{P}, G_i)$ with $G_i \in \mathcal{G}$ forms a *planning problem* that a planner attempts to solve. The cost of any plan π_i is denoted by $c(\pi_i)$ and is defined as the sum of the cost of the actions that make up the plan: $c(\pi_i) = \sum_{a_i \in \pi_i} c(a_i)$. In this implementation, only one plan π_i serves as the solution to a given planning problem, such that goals uniquely identify planning problems. We therefore use the notation $c(G_i)$ to denote the cost of the plan that achieves the G_i. Assuming unit costs for every action, the cost of a plan will be the plan's length. For the example above, the cost for the plan to pursue each of the goals is as follows: $c(G'_0) = 10$, $c(G'_1) = 7$, and $c(G'_2) = 7$. Thus the recognized plans are the plans that achieve G'_1 and G'_2.

Existing Approaches in Interactive Narratives

Existing approaches to activity recognition have either focused on statistical approaches [Albrecht, Zukerman, and Nicholson, 1998; Synnaeve and Bessière, 2011] or on the related but distinct problem of *goal recognition* [Mott, Lee, and Lester, 2006; Gold, 2010; Baikadi et al., 2013]. In order to represent narratives as plans and perform plan recognition given this knowledge representation, we developed a plan recognition system that follows a more symbolic approach. Our implementation of plan recognition leverages Ramírez and Geffner's [2009] intuition in a plan-based interactive narrative context [Young et al., 2013].

The Plan Recognition Optimality Assumption

Our implementation of plan recognition assumes that the observed agent will act *optimally*, which in our case means that the agent (i.e. the player) will pursue plans of the smallest possible cost to any goal from the state the agent is observed to be in. To continue the previous example, in Figure 1 the last action we have observed is the agent transitioning from room F to room G. Since it is the last action the recognizer knows of, the most conservative assumption to make is that the agent is at room G. We additionally know that from room G it is more costly to return to room C than to continue to either room I or K. Thus, the recognizer identifies plans to get to either room I or K as optimal. In our current implementation, we assume that the player is playing optimally with respect to executing the shortest possible sequence of actions to achieve his or her goal. This notion of optimality is idealized and we admit that it is possible (and even likely) that players will play in sub-optimal ways; exploratory behavior, for instance, would be poorly modeled by our basic plan recognizer, since it may involve several sub-optimal (*vis-à-vis* plan cost/length) moves and repetition. While our assumption may not be tenable in elaborate environments, we will not account for this discrepancy in this work. Future work will address how to relax this optimality assumption.

Implementation

We used an existing interactive narrative system called the *General Mediation Engine* (GME) [Robertson and Young, 2014], a unified game engine and experience manager that creates and manages gameplay experiences based on a Planning Domain Description Language (PDDL) [McDermott, 2000] domain and problem file, and a linear narrative generator/planner. GME was embedded into a Unity[1] game environment for our prototype, where it linked game assets to internally represented plan objects [Robertson and Young, 2015]. The GME works with a range of planners, including FAST DOWNWARD [Helmert, 2006] a classical planning system based on heuristic search. Within the GME, game actions are discretized, but not limited: the player can take game actions, but only those that conform to the automated planning knowledge representation that represents the domain will be interpreted by the GME as having occurred. We used the Unity GME to create a game environment where the player could accomplish one of many pre-established goals.

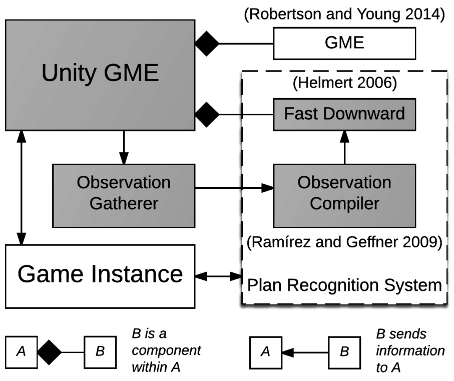

Figure 3: Architecture for our implementation. The plan recognition system has access to the Unity GME-driven game. Components in the *plan recognition loop* are in gray.

The Plan Recognition Architecture

Our implementation is illustrated in Figure 3. The plan recognition component was implemented by taking the code base that Ramírez and Geffner [2009][2] created and adapting it to work with FAST DOWNWARD [Helmert, 2006]. As the player plays in the environment, the *plan recognition loop* (illustrated in Figure 3) triggers: For every action the player takes that GME recognizes as part of the domain: the action is logged by the OBSERVATION GATHERER, which sends it to the OBSERVATION COMPILER developed by Ramírez and Geffner [2009]. Then, the compiler produces a new planning domain as outlined prior and passes it to the FAST DOWNWARD planner, which produces the output plan that is recognized by using A* with a context-enhanced additive heuristic [Helmert and Geffner, 2008]. This system produces one such plan, settling ties randomly.

The Planning Domain

Our planning domain was designed to reflect a fantasy-genre adventure game in the style of *The Legend of Zelda* [Nintendo R&D4, 1986]. The domain contains seven action operators representing the *types* of actions that collectively support completing key-lock quests [Ashmore and Nitsche, 2007]. It contains 23 predicates that describe relationships between objects in the domain, and 20 objects the predicates could describe. This domain was encoded in PDDL and input to the Unity GME, which created a virtual environment similar to that illustrated in Figure 4. In this example, the player can complete the game by getting one item from each non-`Center` location.

[1]http://unity3d.com

[2]This code can be found at http://goo.gl/38vLZ2

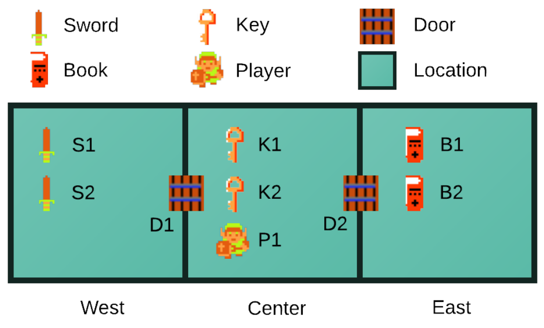

Figure 4: An illustration of a procedurally generated game instance generated by the GME from a plan recognition theory. The player can beat the game by obtaining one sword from the `West` location and one book from the `East` location. There are four such winning conditions, which are specified in disjunctive normal form to the plan recognizer as discussed in the **Plan Recognition** Section.

Evaluation

Blaylock and Allen [2003] defined four metrics to evaluate statistical goal recognizers, which we present in the context of our own work:

1. *Speed of computation* – since gameplay happens in real-time, the recognizer needs to be able to execute fast enough to avoid affecting the player's overall play experience in a way that would interrupt the game.
2. *Early/Partial prediction accuracy* – the recognizer should identify plans as economically/accurately as possible.
3. *Convergence* – ideally, as more information is provided to the recognizer, the more specific the recognized plan should be. This criterion may not be tenable in a realistic game context, since players always have the option of abandoning their current plan in service of another and are not required to communicate when or what they switch to ahead of time.
4. *Generalizability* – the recognizer should use as little domain-specific heuristics as possible in its calculation of the recognized plan.

Our evaluation focused on *speed of computation*, although we indirectly addressed the *generalizability* criterion since we did not use domain-specific information in our system. Future work will address the remaining criteria in an experimental game context, for which a reasonably fast speed of computation must be guaranteed to avoid adversely affecting the player's experience (hence our focus on the first criterion in this work). However, speed is evaluated *in context*. Prior work by Ramírez and Geffner [2009] evaluated the performance of the plan recognizer across several task domains. In this work, we were concerned with plan recognition within an interactive narrative environment that is managed by the Unity GME. We therefore configured our domain to run performance tests in several scenarios. We then characterized runtime trends of the plan recognition loop in these scenarios to explore the feasibility of this proof-of-concept for recognizing player actions.

Materials

We developed a game with the Unity (v5.1.0) game engine, using the Unity GME and planning domain discussed in the **Implementation** Section. The game ran on a Origin-brand Personal Computer with a 3.50 GHz Intel Core i7 Processor and an available 8.0 GB of RAM. The operating system was the 64-bit version of Windows 7 Home Premium (SP1).

Procedure

The first author generated all of the data for this experiment manually by playing all of the diverse game configurations, never playing sub-optimally *vis-à-vis* plan cost/length. The data was produced by executing the plan recognition loop after every GME-recognized player step. The plan recognizer ran as a separate thread within the Unity environment. To evaluate the plan recognition system within the Unity GME, we ran several configurations of the planning domain. Overall we collected data on 36 different configurations and each configuration was run 10 times. The different configurations were along four dimensions:

1. *Number of goals* $|\mathcal{G}|$. Because the recognizer must work to find potential plans of players toward potential goals, we varied the number of potential goals $G_i \in \mathcal{G}$ a player could have. We arbitrarily chose two values for the number of goals: 4 and 8. Each goal G_i was constructed as a conjunction of two literals such that the actions needed to accomplish both overlap in a way that makes it difficult for the recognizer to identify one unique plan as recognized. In effect, the number of goals in the plan recognition theory dictates the amount of planning problems the recognizer must consider and the number of overlapping actions represents how difficult it is for the recognizer to perform its task.
2. *Length of the optimal plan* $length(\pi^*)$. This is one estimator of the amount of processing time a domain-independent planner consumes during the search process. Since the plan recognition theory defines a set of planning problems, the evaluation domain was configured to support optimal plans of equal length for all planning problems from the initial state of the domain. These optimal plans used three of the seven available operators in the domain. We selected two values for the varying lengths: 9 and 15.
3. *Number of actions in the domain* $|A|$. This is another estimator of the amount of processing time a domain-independent planner consumes during the search process. To inflate the number of actions in the domain, we created differently-named copies of each of the seven domain actions, leading us to three values for the total number of actions: 7, 14, and 21. For each set of actions, the compiled actions during plan recognition proportionately increase, so the overall branching factor impact is captured in this dimension.
4. *Percentage of optimal plan actions observed* (O%). As gameplay continues, the observations that are compiled and input to the domain compiler increases. We observed runtime performance of the plan recognition loop across three monotonically increasing percentages of optimal plan actions observed: 30%, 50%, and 70%.

Table 1: Linear regression results for Equation 1. Of the parameter estimates, only β_4 is practically significant.

Parameter	Estimate (in ms)	SE (in ms)	p-value
Intercept	β_0 =1178.9653	67.399632	< 0.0001
$\|\mathcal{G}\|$	β_1 =-52.8709	5.9510353	1
$length(\pi^*)$	β_2 =73.4028	2.7986799	< 0.0001
$\|A\|$	β_3 =30.8058	2.0786601	< 0.0001
O%	β_4 =417.3162	41.319289	< 0.0001

$F_{stat} = 270.7372$ ($p < 0.0001$), $R^2_{adjusted} = 0.4065$

Results and Analysis

We conducted an exploratory analysis to assess variations in the response variable (runtime) as a function of the linear combination of the predictor variables $|\mathcal{G}|$, $length(\pi^*)$, $|A|$ and O%. Because these parameters are continuous, our model was a linear regression of the following form:

$$\text{time} = \beta_0 + |\mathcal{G}|\beta_1 + length(\pi^*)\beta_2 + |A|\beta_3 + O\%\beta_4 + \epsilon \quad (1)$$

The alternate hypothesis used in the linear regression was $H_A > 0$ (under the null $H_0 = 0$) because we expected increases in any of the predictor variables to lead to an increase in the response variable. The results of the regression are illustrated in Table 1. We fail to reject the null hypothesis for $|\mathcal{G}|$, the number of goals predictor, but reject the null in all other cases. One likely reason for having failed to reject the null hypothesis for $|\mathcal{G}|$ is due to significant ($p < 0.0001$) interaction effects detected between $|\mathcal{G}|$ and both $length(\pi^*)$ ($\beta = 10.6272$) and $|A|$ ($\beta = -9.9678$). Thus, care should be taken in how to interpret results with regards to $|\mathcal{G}|$. Intuitively, as mentioned in the **Procedure** Section, the number of goals input to the plan recognizer dictates how many unique planning problems the solver must consider in its identification of a recognized plan. We therefore expect more nuanced experimentation to reflect the intuition that a higher number of goals will result in a more time-consuming computational process.

The other three predictors were statistically significant, but only one of these (the O% predictor) was practically significant in terms of magnitude increases to runtime: for a unit increase in the number of observed actions over the optimal plan in the domain, the runtime increases on average by 0.417 seconds. This indicates that a promising area of future work is the intelligent selection of observed actions to input to the plan recognition loop. Finally, this work suggests that this technique for plan recognition inside an interactive narrative will scale well, since other predictors were not detected as practically significant. Of course, future work shall address this last point in a more exhaustive manner since this work was more exploratory.

Limitations Unfortunately, no benchmark tasks exist for plan recognition systems. Thus, our evaluation here was done according to our intuition of what plan recognition theories would be difficult for the plan recognizer to perform its task. Namely, we designed the goals of the plan recognition theory to require overlaps in the actions for the optimal plan from the initial state. Our implementation of plan recognition is planning-based, and automated planning is a PSPACE-complete problem in general [Bylander, 1994]. Our system solves several planning problems in order to come up with a solution to the plan recognition theory, so even though our evaluation was not normative, it is informative. An interesting avenue for future work is to quantify the difficulty of a plan recognition theory, but that is beyond the scope of this work. A second limitation is that we assume the player will play optimally. This means that, in the context of gameplay, it will be easy for the player to "fool" the plan recognizer (e.g. through backtracking). Future work shall address how we could potentially relax this limitation by intelligently selecting the information the recognizer pays attention to during gameplay (which may have positive effects in the overall performance, as suggested in the previous section). A third limitation is that the architecture currently executes the plan recognition loop after every player step, which is a naïve strategy – players do not necessarily change their mind with regards to what they wish to accomplish in the environment after every step, thus eliminating the need to recognize her plan after every action. Thus, future work shall find principled ways to schedule the plan recognition process to better reflect the player's thought process as she engages the interactive narrative.

Conclusion

We have discussed and evaluated a symbolic plan-recognition system inside an interactive narrative virtual environment. This system relies on automated planning data structures and computation, which provide advantages for representing player actions over statistical-based approaches including a readable knowledge representation and clear semantics for arbitrary domains. While future work is needed for this approach to gain traction as a procedure to model a player's expectation of upcoming action, we are optimistic about this line of work given the favorable performance metrics that we have presented.

Several narrative scholars have highlighted the importance of understanding intention attribution when studying how story consumers engage with narrative artifacts. We have taken a step in service of this directive by modeling intention attribution as a plan recognition process. Our overall goal is to use plan recognition as a proxy for a player's interactive narrative comprehension processes. The results presented here form the baseline reasoning that we posit a player is engaging in within interactive narrative contexts and future refinements will help more closely approximate a player's actual cognitive engagement. We thus advocated for considering *players as plan recognizers* and have proposed a computational representation and procedure to describe their reasoning process.

Acknowledgments

We thank David R. Winer for his help in an earlier version of the work presented here and Justus Robertson for his help with the Unity GME. This work was supported in part by the U.S. Dept. of Energy under grant number DE-FG02-97ER25308. Opinions, findings, and conclusions expressed

here are solely those of the authors.

References

Adams, E. W. 2013. *Resolutions to Some Problems in Interactive Storytelling*. Ph.D. Dissertation, Teesside University.

Albrecht, D. W.; Zukerman, I.; and Nicholson, A. E. 1998. Bayesian Models for Keyhole Plan Recognition in an Adventure Game. *User Modeling and User-Adapted Interaction* 8(1–2):5–47.

Ashmore, C., and Nitsche, M. 2007. The Quest in a Generated World. In *Proceedings of the 2007 Digital Games Research Association Conference: Situated Play*, 503–509.

Aylett, R. 2000. Emergent Narrative, Social Immersion and 'Storyfication'. In *Proceedings of the 1st International Workshop on Narrative and Interactive Learning Environments*, 35–44.

Baikadi, A.; Rowe, J. P.; Mott, B. W.; and Lester, J. C. 2013. Improving Goal Recognition in Interactive Narratives with Models of Narrative Discovery Events. In *Proceedings of the 6th Workshop on Intelligent Narrative Technologies at the 9th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 2–8.

Blaylock, N., and Allen, J. 2003. Corpus-based, Statistical Goal Recognition. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, 1303–1308.

Bylander, T. 1994. The Computational Complexity of Propositional STRIPS Planning. *Artificial Intelligence* 69(1):165–204.

Carberry, S. 2001. Techniques for Plan Recognition. *User Modeling and User-Adapted Interaction* 11(1-2):31–48.

Cohen, P. R., and Levesque, H. J. 1990. Intention is Choice with Commitment. *Artificial Intelligence* 42(2):213–261.

Dennett, D. C. 1989. *The Intentional Stance*. MIT Press.

Fikes, R. E., and Nilsson, N. J. 1971. STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving. *Artificial Intelligence* 5:189–208.

Gerrig, R. J., and Bernardo, A. B. I. 1994. Readers as problem-solvers in the experience of suspense. *Poetics* 22(6):459–472.

Gold, K. 2010. Training Goal Recognition Online from Low-Level Inputs in an Action-Adventure Game. In *Proceedings of the 6th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 21–26.

Helmert, M., and Geffner, H. 2008. Unifying the Causal Graph and Additive Heuristics. In *Proceedings of the 18th International Conference on Automated Planning and Scheduling*, 140–147.

Helmert, M. 2006. The Fast Downward Planning System. *J. of AI Research* 26:191–246.

Herman, D. 2013. *Storytelling and the Sciences of Mind*. MIT Press.

McDermott, D. M. 2000. The 1998 AI planning systems competition. *AI Magazine* 21(2):35.

Mott, B.; Lee, S.; and Lester, J. 2006. Probabilistic Goal Recognition in Interactive Narrative Environments. In *Proceedings of the 21st National Conference on Artificial Intelligence*, 187–192.

Murray, J. H. 1998. *Hamlet on the Holodeck: The Future of Narrative in Cyberspace*. The MIT Press.

Nelson, M. J.; Mateas, M.; Roberts, D. L.; and Isbell, C. L. 2006. Declarative optimization-based drama management in interactive fiction. *IEEE Computer Graphics and Applications* 26(3):32–41.

Nintendo R&D4. 1986. *The Legend of Zelda*. Nintendo.

Ramírez, M., and Geffner, H. 2009. Plan Recognition as Planning. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence*, 1778–1783.

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

Roberts, D. L., and Isbell, C. L. 2007. Desiderata for Managers of Interactive Experiences: A Survey of Recent Advances in Drama Management. In *Proceedings of the 1st Workshop on Agent-Based Systems for Human Learning and Entertainment at the 2007 International Conference on Autonomous Agents and Multiagent Systems*.

Robertson, J., and Young, R. M. 2014. The General Mediation Engine. In *Proceedings of the Experimental AI in Games Workshop at the 10th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 42–48.

Robertson, J., and Young, R. M. 2015. Automated Gameplay Generation from Declarative World Representations. In *Proceedings of the 11th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, (to appear).

Salen, K., and Zimmerman, E. 2003. *Rules of Play: Game Design Fundamentals*. MIT Press.

Sukthankar, G.; Geib, C.; Bui, H. H.; Pynadath, D.; and Goldman, R. P., eds. 2014. *Plan, Activity, and Intent Recognition: Theory and Practice*. Elsevier.

Synnaeve, G., and Bessière, P. 2011. A Bayesian Model for Plan Recognition in RTS Games Applied to StarCraft. In *Proceedings of the 7th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 79–84.

Young, R. M., and Cardona-Rivera, R. E. 2011. Approaching a Player Model of Game Story Comprehension Through Affordance in Interactive Narrative. In *Proceedings of the 4th Workshop on Intelligent Narrative Technologies at the 9th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 123–130.

Young, R. M.; Ware, S.; Cassell, B.; 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. *Sprache und Datenverarbeitung (SDV): International Journal of Language Processing, Special Issue on Formal and Computational Models of Narrative* 37(1–2):67–77.

Young, R. M. 1999. Notes on the use of Plan Structures in the Creation of Interactive Plot. In *Proceedings of the AAAI Fall Symposium on Narrative Intelligence*, 164–167.