In Computational Intelligence Journal, Volume 26 Issue 2, pp. 183-211

Drama Management and Player Modeling for Interactive Fiction Games

Manu Sharma, Santiago Ontañón, Manish Mehta, and Ashwin Ram

Cognitive Computing Lab (CCL)
College of Computing, Georgia Institute of Technology
Atlanta, Georgia, USA
manu@gatech.edu, {santi, mehtama1, ashwin}@cc.gatech.edu

Abstract. A growing research community is working towards employing drama management components in story-based games. These components gently guide the story towards a narrative arc that improves the player's gaming experience. In this paper we evaluate a novel drama management approach deployed in an interactive fiction game called Anchorhead. This approach uses player's feedback as the basis for guiding the personalization of the interaction. The results indicate that adding our Case-based Drama manaGer (C-DraGer) to the game guides the players through the interaction and provides a better overall player experience. Unlike previous approaches to drama management, this paper focuses on exhibiting the success of our approach by evaluating results using human players in a real game implementation. Based on this work, we report several insights on drama management which were possible only due to an evaluation with real players.

Keywords: Drama Management, Interactive Fiction, Player Modeling, Case-Based Reasoning

1 Introduction

The game industry has continuously presented advancements in various disciplines like graphics, animation, effects and audio to keep up with the increasing popularity of computer games. Comparatively, Artificial Intelligence in games (henceforth referred as Game AI) has received lesser attention. Lack of complex Game AI is generally attributed to its high computational cost. The limitations of a primitive or scripted Game AI can result in a break in player experience. For example, expert gamers quickly get bored of playing against the built-in Game AI of a real-time strategy game. They can easily isolate and exploit flaws in the limited set of Game AI's strategies. Furthermore, unrealistic responses of game characters in role-playing or adventure games break the illusion of an ongoing narrative. Games with advanced AI approaches that address these issues can offer promising new levels of entertainment. As a step towards this direction, the novel AI approach presented in this paper provides an improved player experience in interactive fiction games.

Game AI has not escaped the attention of the current Artificial Intelligence research community as a challenging domain [1]. A related field, *Interactive Drama*, has gained

popularity as a growing research area over the past couple of decades [2, 3]. In an Interactive Drama, the author wishes to communicate an interesting narrative to the participant (i.e. the player of the game). The drama increasingly adapts to the participant's interaction as the narrative unfolds. Narratives in games are distinguished from narratives in other genres because they delegate to the participant a limited degree of freedom and in doing so enable him or her to have an influence on the trajectory of the plot. Situated in an Interactive Drama domain, our work addresses the problem of creating story-based interactive fiction games that adapt to improve individual participant's experience.

Interactive Drama is characterized by the player's participation in an ongoing narrative. Creation of an interactive drama presents the intriguing problem of maintaining an interesting balance between the drama's original author conceived aesthetics and personalization of player experience. There are two inter-related sub-problems that have to be addressed.

- Building an artificial intelligence component in charge of guiding the complete dramatic experience towards a particular narrative arc.
- Creating a model of the player to recognize interesting personalized narrative arcs.

Both the sub-problems are non-trivial due to several reasons. First, the decision space, i.e. the set of possible ways in which a story can be influenced by the AI, is enormous. Second, individual actions can be small, for example 'pick up the silver key', but it's hard to understand what the player is trying to do at an abstract level, and the extent to which these actions should affect the environment. Most recent interactive games, e.g. Mass Effect (BioWare, 2007), Fallout 3 (Bethesta Game Studios, 2008) and Spore (Electronic Arts, 2008), attempt to have interesting non-linear plots, various interplaying and branching subplots, and multiple endings. These problems hold promising interest to the game research community as well as the game industry.

In this paper we present C-DraGer (Case-based Drama manaGer), a drama management approach based on both search and case-based reasoning [4–6]. C-DraGer has been implemented in an interactive drama game called Anchorhead [7]. We present an evaluation of our approach based on a C-DraGer enabled version of Anchorhead played by human players. With the goal of providing the player a compelling dramatic experience, we use player feedback as a guide for our evaluation.

In particular, to address the sub-problem of guiding the player towards interesting narratives, our drama manager employs a set of actions provided at appropriate points in the ongoing game. These actions attempt to influence the game in such a manner that the player is maneuvered towards certain aspects of the narrative. To address the latter subproblem of personalizing player's experience, our approach uses a *case-based player modeling* technique to predict the interestingness of game events for the player. We first introduced player modeling as a key component for the success of drama management based approaches in [8, 9].

Anchorhead, created by Michael S. Gentry, is an interactive game with a complicated story, several plots and subplots. These features make it amenable for drama management studies. Anchorhead has been previously used as a test bed for testing different drama management approaches with simulated players [10]. In order to test our approach, we reimplemented a subset of the Anchorhead game where the player interacts

through a text-based interface (similar to one of the first interactive game, Zork). We created an intervention where the players were asked to play the game twice, once with C-DraGer included as part of the game and another time without C-DraGer. To gather further drama management design insights, we observed the player's interaction and discussed their subjective opinion at the end of each evaluation (as detailed in Section 5). Evaluation with human players aided us in obtaining valuable information that we could not have noticed otherwise.

The rest of the paper is organized as follows. Section 2 summarizes the related work in the fields of drama management and player modeling. It compares and contrasts the similarities and the differences between the existing approaches in these fields and C-DraGer. In Section 3, we present a brief introduction to the game used as our testbed, Anchorhead. This is followed by our technical approach to drama management in Section 4. We begin the discussion by presenting the definitions of the most common technical terms used in this paper. Section 5 presents the results of our evaluation. It emphasizes our qualitative analysis of the player's experience while playing the game followed by an evaluation of the performance of the player modeling in our system. Finally, in Section 6, we conclude the paper with final thoughts and future directions.

2 Related Work

The field of drama management involves developing computational theories for cognitive/emotional agents, presentation style, and drama [3] as well as developing architectures to incorporate relevant developments in AI into modern game engines [11]. Interactive drama implements the idea of the *multiform plot* presented by Murray [12]. In a multiform plot, the *interactor*, (Murray's reference to the player) is given some agency to influence the story. One of the biggest challenges is to balance the amount of agency given to the player. Too much agency can result in losing control of the plot, but limiting the agency of the player can make the player aware of the medium and therefore reduce the engagement.

Interactive dramas need some architecture to create an appealing dramatic experience for their players. One approach to solve the problem of guiding dramatic experience towards specific narrative arcs has been referred in the existing literature as *Drama Manager* (DM) [13] or *Director* [14].

We have organized the remainder of this section into two main categories: interactive drama and player modeling in games.

2.1 Interactive Drama

An important topic in interactive drama is the development of plot representation formalisms. For example, Mateas et al.[15] presented a formalism called *beats*. Beats are storyline threads i.e. they include a chain of narrative goals as well as possible variations and reactions depending on the player's interaction. Nelson et al. [16] as well as Case-based Drama manaGer (C-DraGer) use the *plot point* representation. It is based on representing a plot as a graph where nodes represent important events in the plot and the edges represent any dependencies between these events. The *recursive graph model*

[17], aids in capturing notions like views, thoughts, motives, plans and emotional states of characters in the narratives. The plot point representation does not capture these notions. At the same time, the plot point representation is expressive enough for the purposes of C-DraGer. Magerko [18] presented a comparison of some of these representations in the literature.

As mentioned earlier, interactive drama and fiction games have been a popular commercial success as well as been a field of considerable interest to the research community. While discussing 'drama' as a key element for generating *virtual reality*, Bates [3] strikes an interesting analogy between the drama management approach and the two player turn-based game Chess. The Director (the module that enforces drama management) plays its turn by observing the moves of the player interacting with the game (i.e. the opponent). The only difference between the planning approaches is that in case of drama management the Director's aim is not to defeat this player but to ensure that the interaction is presented as an interesting experience to the player. This work addresses interesting planning issues that the Director needs to incorporate; for instance, the granularity of the Director's actions, and the extent to which the actions should affect the virtual reality environment. The approach to drama management presented by Bates exhibits a lot of similarities to the ideas presented in C-DraGer. Both works address the problem by requesting the Drama Manager to plan more interesting narratives for the player. Instead of aiming to achieve virtual reality's theme of "go anywhere, do anything", C-DraGer attempts to combine the player interest and the author-defined aesthetics for the game to present an appealing experience to the player.

Another significant set of contributions to the field of drama management are related to the *Oz System Architecture* [13]. While the Oz architecture focused on delivering the three key elements 'characters', 'presentation', and 'drama', it delineated the major components and their necessary relationships for the architecture to support drama management. The architecture incorporates a Drama Manager (or Director), an interactor (a human participant experiencing the drama), a theory of presentation, an explicit model of the body and a model of the non-playing character's mind. The Drama Manager can directly influence the non-playing character's actions, the physical environment and the presentation theory. This work presents various "live interactive drama experiments" performed as an attempt to understand the underlying requirements to create an architecture for an appealing dramatic experience. These experiments involved actual drama settings, an interesting plot, an interactor, an audience, plot's characters enacted by artists, and a director. The director is a human that observed the entire drama while being outside the view of the characters. The director verbally communicated with the artists on stages via headphones. This was a strict one-way communication where the director issued commands to individual artists. Although this work is not based on any automated game, a striking similarity between the Oz architecture and C-DraGer is that both strive to provide drama management for human participants.

Another approach to drama management is that of *Façade* [15]. It employs a beat-based management system suited towards tighter story structures where all the activities contribute towards the story. The *Mimesis architecture* [11] proposes a story planning based approach for real-time virtual worlds. In this approach, the story plans are tagged with causal structure and the system handles player actions that might threaten the

causal links. This is achieved by either replanning of the story or disallowing the player the opportunity to carry out the threatening action. However, in such an approach to drama management, only the author specifies the concrete goals that the planner should achieve. This approach does not incorporate a player interest model to guide the player experience during the game. C-DraGer plans based on combined feedback from the player interest model and the author specified goals.

Magerko [19] presented the *Interactive Drama Architecture* (IDA), that incorporates a *Director* module that makes use of player models in order to manage the drama. IDA utilizes the player models to predict the behavior of the player and anticipate problematic situations before they occur. Although a lot of work was put into making sure that IDA was tested against synthetic players represented by variety of player types (defined as "player archetypes"), there was no indication on how IDA would perform against human players. Moreover, IDA uses hand-coded player models as compared to C-DraGer that automatically learns player interest models at run-time.

By extending his previous work, Magerko addressed *the boundary problem* [20]. In an interactive drama, the boundary problem occurs when the player actions bring a dramatic experience outside the boundaries of the authored content. Magerko's work has been implemented using a 3D world created in Unreal Tournament engine and Haunt 2 (the game environment used by IDA). The director follows the story progression and attempts to predict player's future actions. Based on this prediction, the director preemptively steers the player away from causing the boundary problem.

Bates first proposed the idea of treating drama management as an optimization problem. The approach termed *Search Based Drama Management* (SBDM) was based on the fact that the drama manager chooses its best available action with expectation-calculating nodes and the player is typically assumed to act according to some probabilistic model. In his dissertation, Peter Weyhrauch [21] further developed the idea of a SBDM with a tree-based search (similar to the algorithms used for playing Chess) that used an author-specified evaluation function to measure the interestingness value for a particular story. However, the DM employed was not connected to a concrete story world and the techniques were tested using simulated players. Furthermore, the approach ignored capturing the player's interest for a particular story. Continued work on SBDM is presented in [22, 23].

In another approach, Nelson et al. defined a *Declarative Optimization based approach to Drama Management* (DODM) [16]. The central premise of their technique is to provide the author with the ability to specify what constitutes a good story. Given a set of plot points and author specified evaluation functions, it uses a reinforcement learning approach to optimize drama manager actions in response to player actions. Plot points are significant intermediate story events (see definition in Section 4.1). This approach also uses a simulated player model to predict the next player action. Furthermore, the approach ignores constructing a player interest model.

Finally, it is interesting to mention that there are a few non-academic products related to interactive drama and storytelling. For instance, *DreamPath* [24], is an authoring system for interactive "gamebooks" similar to the style of the create-your-own-adventure collections. Another example is *SWAT* (StoryWorld Authoring Tool), by *Sto-*

ryTron [25]. SWAT lets the artists author story worlds that are narrated by a story telling tool. However, none of these tools incorporate drama management or player modeling.

2.2 Player Modeling in Computer Games

Player modeling can be employed as a critical element in improving player experience [26]. Also, player modeling is generally accepted as a prerequisite towards achieving adaptiveness in games [27, 28]. Different approaches towards player modeling can be classified in two groups, namely:

- *Direct-measurement* approaches, that employ physiological measures to directly monitor player's emotional state during the game (such as the work presented by Prendinger et al. [29]).
- *Indirect-measurement* approaches, that try to infer (in opposition to directly monitor) information about the current player (e.g. the skill level). This is generally done by computing a set of features (e.g. actions per minute, number of aggressive actions) during the ongoing interaction in the game (such as the work presented by Togelius et al. [30]).

An example of direct measurement approach is that of Predinger et al. [29], where sensed data such as heart rate is used to modify the behavior of an empathic virtual agent situated in a job environment. In our approach towards player modeling, indirect measurements were better suited as we were interested in modeling the player from the data that can be derived from the player actions taken in the game. Previous work on indirect-measurement techniques for player modeling focuses on modeling the player's skill for automatic adjustment of game level. Cowley et al. [31] presented a decision theoretic framework to model the choices that players make in the well known game *Pacman*. They observe the player's deviation from the optimal path and use that to model the player's skill level. One of the observations from interviews carried out as part of the evaluation of C-DraGer suggested that the player skill level is an important measure for determining various drama management strategies to improve player experience (see Section 5.1).

Thue et al. [32] have presented work signifying the importance of player modeling in interactive storytelling. This work introduced *Player-Specific Stories via Automatically Generated Events* (PaSSAGE), an interactive storytelling system that learns a player model to dynamically select the content of an interactive story. While the game is being played, the player's actions are used for constructing the player's playing-style model. The player model is a vector of weights assigned to five playing styles: Fighters, Power Gamers, Tacticians, Storytellers, and Method Actors. The authors treat interactive storytelling as a general decision-making problem comprised of three levels: *selection*, *specification*, and *refinement*. The selection level aids in making high level decisions to questions like 'What plots should be presented to the player?', or 'What will prompt the player to initiate certain interactions?'. The specification level determines the time, location and participants of the plot. Given the player's interests, the refinement level answers the question 'How should the actors behave?'. Thus, it determines the behavior of each character in the game. This work performed extensive evaluation

(with multiple human participants) for their proposed hypothesis that the adaptive versions of the game were more entertaining and provided a higher agency than their fixed story counterparts.

Albrecht et al. [33] presented an interesting approach using user modeling for an adventure game. This work applied an approach to keyhole plan recognition for predicting the players' goals in an adventure game. This approach concentrates on gathering observations from the player actions in a Multi-User Dungeon (MUD), and then training a dynamic belief network to predict the players' goals. In contrast with Albrecht et al., this paper does not focus on modeling player's goals, but restricts itself to model the player's interests.

Yannakakis and Maragoudakis [26] presented interesting results on the usefulness of deploying a player model. In particular, the authors defined a set of rules by which the game *Pacman* can be considered interesting, and an evolutionary algorithm for the behavior of the enemies in the game. In that framework, they exhibit the usefulness of a player model to help the evolutionary algorithm achieve more interesting games. Instead of defining a set of rules, we focus on obtaining the answer to the question 'How interesting is a story arc?' from the player feedback.

There has been work on applying a player modeling technique to a racing game [30]. The player model captures the player's behavior for a given track. Instead of generating hand-made tracks, Togelius et al. use the learned models to automatically generate new tracks that exhibit similar characteristics (speed achieved, difficulty, etc.) when the learned player models are used to drive in the tracks. In this work, they build *player-action* models (i.e., modeling the behavior of the players) whereas we focus on modeling the interests of the player to provide a better playing experience.

On the other hand, Laurel [34] critiques that 'user models' are useless. Based on Laurel's proposed argument, if the computer models the user, the user usually models the computer. Now the user will model the model that the computer has of the user, and that will create an infinite recursion. Our system's Player Modeling Module does not fall into this trap because we are not modeling the player, but the player's interests (see Section 4.3).

3 Anchorhead

Interactive fiction games are a genre of games where the player usually explores some locations (e.g. a haunted house, a lost city) while trying to achieve some objective. The player typically tackles different puzzles, comes across clues, revisits places with additional information, objects or even supernatural powers to continue the pursuit for the objective. Classic examples of commercial interactive fiction games are Zork (Infocom, 1979), Planetfall (Infocom, 1983), Amnesia (Electronic Arts, 1987), The Secret of Monkey Island (LucasArts, 1990), and Curses (Infocom, 1993). As part of the Oz project, the interactive game implementation, Façade [15], has been used for active research.

In our work we have reimplemented a subset of the game Anchorhead, a game created by Michael S. Gentry [7]. The story takes place in the town of Anchorhead. The main character, i.e. the player, explores the town for clues about a mansion that the

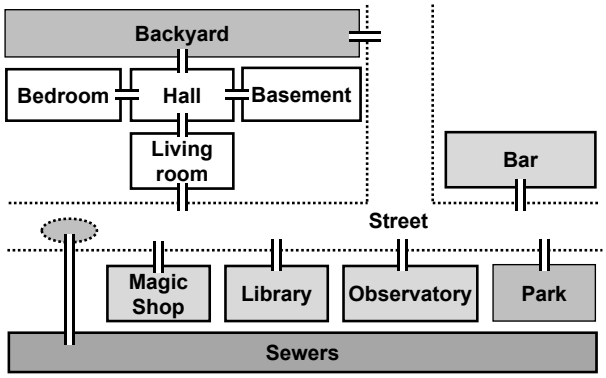

Fig. 1. Map of the various locations in our implementation of the interactive fiction game Anchorhead. The player gradually discovers the secrets of the Verlac family by participating in various subplots in different locations. The player is free to move around anywhere in this map and can choose to perform a variety of player actions.

player inherits at the beginning of the game. The player has to investigate the odd history of the previous occupants, the Verlac family. The mansion belonged to the player's distant cousin Edward Verlac. The lawyer that was supposed to help out the player has gone missing. The player starts with the information that Edward killed his entire family and then committed suicide at the mental asylum. The original story is divided in five days of action. For this work, we have focused on a subpart of the story, identified by Nelson et al. [10] as interesting for evaluating drama management approaches. The subpart is based on the second day of the original game. The resulting story has multiple subplots that are related but can lead to two different endings.

Figure 1 shows the map of all the locations considered in our implementation of Anchorhead. While playing the game, the player is free to explore each location, interact with several natives (e.g. the bum, the owner of the magic shop, or the bartender), and objects (e.g. use the telescope at the observatory, open the coffin in the graveyard, or pick up flask at the bar). Our implementation has a text-based interface for the player's interaction (very similar to the classic game Zork). Figure 2 shows an example of an interaction with our implementation of Anchorhead. As shown in the figure, at any given instant, the player is presented with the current location in the map, the inventory contents, and a list of valid player actions. At a given situation, the player selects one of the many listed player action descriptions. Effects of a player action, effects of the Drama Manager, and conversations with Non-Playing Characters (NPCs) are all expressed as textual messages from the game engine.

4 C-DraGer: Case-based Drama manaGer

The Drama Manager (DM) observes the player's interaction during the complete course of the game and is responsible for producing interesting story arcs (or narratives). Our approach captures the complete game of any given player as an *experience*. This in-

```
==\\Current Location = [magic_shop]
==//Knapsack contents = [Crypt Key]
0. Quit Game.
1. Window shop in the crazy shop.
2. Get out of here. Go into the streets.
3. Buy the spooky magical ball (shows future with 99% guarentee)
4. Buy the shiny protective amulet.
>> 3
>>Hi, I am the magic ball. Welcome to the world of Future. Ironically, my future lies in
your hands ;)

==\\Current Location = [magic_shop]
==//Knapsack contents = [Crypt Key, Magic Ball]
0. Quit Game.
1. Get out of here. Go into the streets.
2. Buy the shiny protective amulet.
3. Window shop in the crazy shop.
>> 1
>>

==\\Current Location = [street]
==//Knapsack contents = [Crypt Key, Magic Ball]
0. Quit Game.
1. Go back to the Mansion.
2. Go to the big and deserted backyard.
3. Go to the big library (La Biblotheque).
4. Go to the mysterious Magic Shop.
5. Go to the town observatory.
6. Go to the pleasent park.
7. What the hell. Lets get drunk. Lets go to the bar!
8. Go to the stinking sewer.
>>
```

Fig. 2. A snippet from a player interaction with our implementation of Anchorhead. In the above snippet, the player is initially located at the Magic Shop along with a Crypt key in the inventory (Knapsack). The player decides to purchase the magical ball. The ball is added to the player's inventory and the game provides a textual notification. The player then moves out to the street that connects to most locations in the town.

cludes, apart from many other parameters, the complete time-ordered history of the events in the game, the interventions from the DM, and feedback given by the player at the end of the game. In the feedback, the player specifies the parts of the game that were liked or disliked. Based on the captured past experiences, the Drama Manager attempts to maximize interestingness of the story arcs for the current player.

Case-Based Reasoning (CBR) provides an approach for tackling unseen but related problems based on past experiences [4–6]. The Drama Manager needs to lazily decide the best possible course of action to enrich the player's experience. In our approach, we use CBR to learn from past experiences and determine the course of action that is better suited to a specific player. This names our approach Case-based Drama manaGer (C-DraGer). Our architecture for drama management consists of three modules (see Figure 3), as introduced in Sharma et al. [35]: a *game engine*, responsible for actually running the game and interacting with the player; a *player modeling module*, responsible for analyzing the actions of the current player and developing a player interest model; and

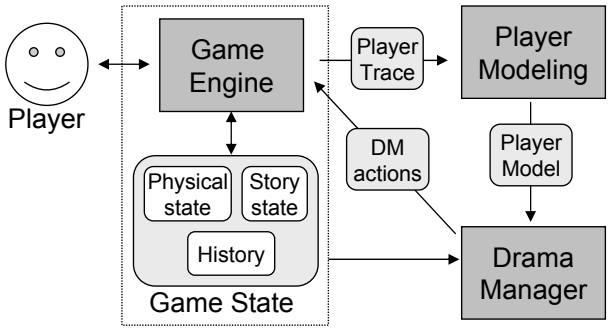

Fig. 3. Architecture of the Case-based Drama manaGer (C-DraGer). C-DraGer oversees the player's interaction with the game and attempts to make the story arcs interesting by building the player model and maximizing the player's interests.

a *drama management module*, influencing the story progression of the game and making it more appealing to the player.

As shown in Figure 3, during a typical run of the game, the player interacts with the game engine that displays a textual description of the current game state and the possible set of player actions. When the player selects an action, it is the game engine's responsibility to execute the player action and update the game state. While the game engine processes the player action, two steps are taken: first, this player action is sent to the player modeling module, which maintains a *player model*; and second, the updated player model is sent to the drama management module. It uses this player model to decide whether it is going to influence the game or not. The drama management module influences the game by executing a *drama manager actions* (explained in the following sections). If a drama manager action is indeed selected, it is again the game engine's responsibility to execute the action and thereby influence the game state. The following sections explain the various terms used across the paper followed by the details of all the modules presented in Figure 3.

4.1 Terms and Definitions

Let us start the explanation of C-DraGer with a collection of definitions for the most common terms used in the remainder of this paper. Note that these definitions are simply meant to serve as clarifications for the usage of these terms in this paper:

- A *player action* is any action that the player can execute to interact with the game. E.g. "open Willam's coffin", or "browse through the objects at the Magic shop". See Table 1 for the types of player actions available in Anchorhead.
- A *drama manager action* is an action that the drama manager can execute to change the course of the game. E.g., "hint the crypt's location", or "temporarily deny entrance to the observatory". Table 2 lists all the twenty-eight drama manager actions employed by C-DraGer.

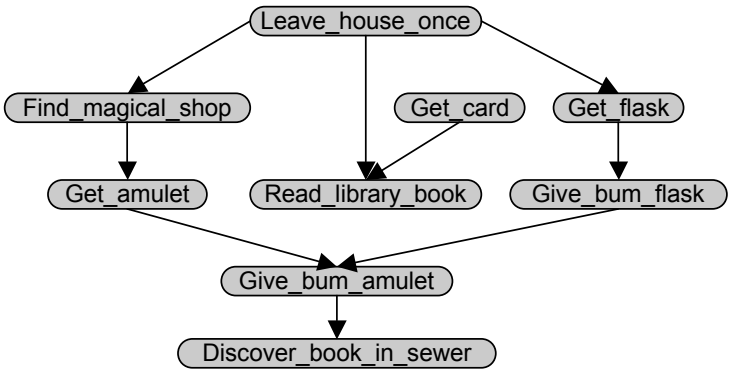

Fig. 4. A subset of the story used by our implementation of Anchorhead. The basic entities in the representation are the intermediate game events called "plot points". Plot points are organized as the nodes of a directed graph. The directed edges represent the dependencies between the corresponding plot points.

- A *plot point* is an event that is relevant to the game story. E.g., "the player has discovered the existence of a hidden room", and "the librarian met with a car accident". Our game's story is composed of a collection of plot points. In the remained of this paper, we will use the words *game event* and plot point to refer to the same concept. In Figure 4, the nodes of the graph represent plot points.
- The *plot* of a game in our framework is the collection of plot points that constitute the story. The complete plot of the game might have several subplots, each one composed from of a subset of all the plot points in the story. E.g., in Anchorhead, one of the subplots of the whole story revolves around discovering the existence of a character named William. Such a subplot consists of about ten plot points.
- A *Story arc* (or *narrative*) denotes the particular order in which a collection of plot points are visited. Thus, the phrase "the drama manager plans interesting story arcs for the player" implies that the drama manager plans a specific sequence of plot points that is believed to be interesting for the player.
- The term *interestingness* to denotes how much a player is interested in a particular plot point. While "interest" is the feeling experienced by a person when he is interested in something, "interestingness" is the condition or quality of being interesting. For example, players are "interested" in stories, and stories have some degree of "interestingness" to the players.

4.2 Game Engine

The game engine performs three functions: a) maintaining the current game state, b) presenting the player with the game state and valid player actions, and c) handling player input. The game engine holds the following components corresponding to the game state:

– The *physical state*, represents the status of the physical objects and characters in the world, i.e. it contains information such as "the character X is at (x1, y1, z1) coordinates", or "the crypt in the backyard of the mansion is currently open". In particular, for our implementation of Anchorhead, the physical state corresponds to the player location in the game (e.g. "the magic shop", "the street", etc.).
– The *story state* is represented as a set of *plot points* [21]. As explained earlier, a plot point is an event that is relevant to the game story. Plot points are structured as a directed graph, where each plot point acts as a node in the graph and the arcs represent dependencies. A dependency states that a particular plot point cannot happen unless another set of plot points have already occurred. Also, each plot point has other associated conditions that need to be satisfied for the plot point to occur in the game. For example, the player needs to be at a specific location (the bar) for a specific plot point (visited bar) to occur in the game. Thus, the collection of plot points that have already occurred in the game denote the portion of the story already visited by the player. To be able to experience the game's entire story, the player needs to satisfy all the dependencies thereby visiting all the plot points. Our implementation of Anchorhead encodes these dependencies between plot points in the form of logical expressions. These are simple expressions comprising the logical *AND*, *OR* and *NOT* operators. Figure 4 shows a particular example of a plot point dependency graph. The player cannot give the bum an amulet unless the amulet was bought at the Magic shop and the bum was given a flask at the park. If a plot point does not have any parent in the graph, it simply implies that there are no dependencies and the plot point would occur as soon as the associated condition for the plot point is satisfied. For instance, the plot point "Get_card" in Figure 4, will occur as soon as the player finds and takes the card located in a drawer at living room of the mansion. The story state is simply a list that indicates the set of plot points in the story graph that have already occurred. Initially, the story state is empty; indicating that no plot points in the story graph have happened.
– The *history* contains information on the evolution of the game until the present state, i.e. a trace of player and drama manager actions from the beginning of the game as well as other meaningful events during the game execution. In the remainder of the paper, the terms "history" and "game trace" will have identical meaning.

It is important to remark that in order to use drama management techniques similar to C-DraGer with any game, the only requirements that the game has to satisfy are: a) it must record an explicit history of the game events, b) it must maintain an explicit representation of the story state, c) it has to allow the drama manager to read the current physical and story state, d) it has to allow the drama manager to influence the story in some way. This could be achieved either by the using *drama manager actions* (as we explain in Section 4.4) or by direct manipulation of the physical and story states.

4.3 Player Modeling Module

The player modeling module (PMM) constantly builds and maintains a player model for the current player. The player model encodes the game playing characteristics of the

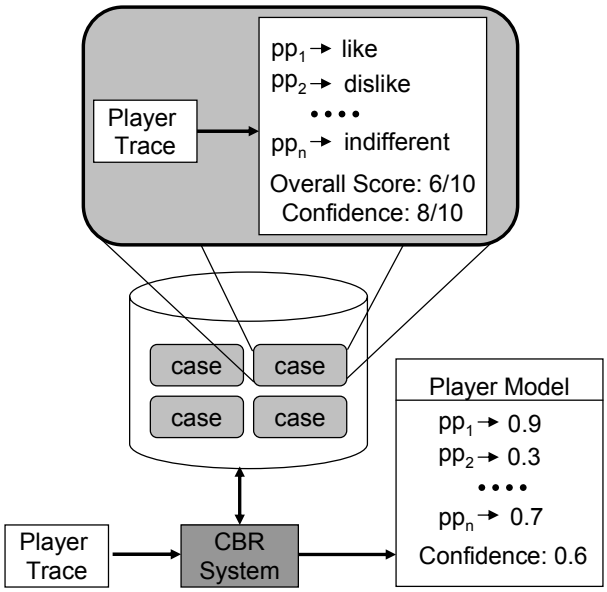

Fig. 5. The Player Modeling Module (PMM) uses a Case Based approach to build the player model. The player model is a mapping from the player's game playing characteristics to the interests in the plot points.

current player and maps it to plot points and an associated numeric interestingness predicted for that plot point. Thus, the player model becomes a prediction of the player's interests in the upcoming plot points of the story. The numeric interestingness is based on the feedback provided by past players at the end of each game. This feedback contains the player's opinions on the game, including the plot points they found interesting and those that they did not. The intent is to capture the interestingness of the plot points encountered by the player during the game. At the end of each game, the PMM stores this player feedback along with the corresponding history of the game.

As mentioned earlier, our proposed approach uses Case-Based Reasoning (CBR) [4] for constructing the player model. As shown in Figure 5, given a player trace, the PMM contains a CBR module that constructs a player model. Given the player's game playing characteristics, the player model is basically a mapping between plot points and interestingness values. In Figure 5, given the player action trace, each plot point pp_i is mapped to a number that represents its predicted interestingness for the player. In order to construct this player model, the CBR module has access to a *case base*. The case base is a collection of cases, where each case contains the feedback provided by a past player. The PMM works on the assumption that players with similar game playing characteristics have similar interests.

The intuition behind this assumption is that it is likely that people with similar interests tend towards performing similar actions in the context of a game. This happens because the players tend to select actions with similar goals in mind. For example, as

Player Action Type	Effect In Game	Example
Movement	Changes the player's current location to one of the twelve possible locations	Walk from street to home
Conversation	Speak with a Non-Playing Character in the game	Talk to bum lying on the ground
Picking Objects	Adds an item from the game to the player's inventory	Accept library card
Using Objects	Interact with an object in the game or the player's inventory	Use magic lens on telescope
No Effect	Does not contribute towards any story arc	Drink French wine

Table 1. Player action classes available in Anchorhead. Anchorhead provided a total of fifty-two possible player actions.

soon as the game starts, the fact that two players decide to visit the "bar" as opposed to exploring the "mansion" suggests some commonality between the players. This is a common assumption in player modeling and. For instance, Thue et al. [32] rely on this assumption with very good results. Moreover, as Section 5.2 further discusses, finding the correct set of features to extract from the game playing characteristics in order to find some useful correlation with player interests is a hard research problem.

After the player completes a game, the game's user interface sequences the plot points in the order that the player visited them over the course of the entire game. From the list, the player is asked to select his interest in the plot points based on a 5 point Likert scale classification: *strongly like*, *like*, *indifferent*, *dislike* and *strongly dislike*. In addition to this, the player is asked to enter a whole number on a 5 point scale representing the overall interestingness of the interaction. The player also provides a confidence value (on the above rating) on a 5 point scale. Again, notice that in our drama management system, the player model is a player interest model, i.e. we are only modeling the interest of a particular player for each plot point in the story. The system builds a case as soon as the player completes the form. The case is added to the case base in the following manner:

- Interest for each plot point pp_j is converted to a number $\delta(pp_j)$ using the mapping: strongly dislike = -1, dislike = -0.5, indifferent = 0, like = 0.5 and strongly like = 1.
- The overall interest of the player is converted to a number $s \in [-1, 1]$ by linearly mapping 0 to -1 and 4 to 1.
- The confidence (on the above interest) provided by the player is converted to a number $c \in [0, 1]$ using a linear mapping where 0 is converted 0 and 4 to 1.
- The plot point interestingness ppi of each plot point pp_j is computed as $ppi(pp_j) = \frac{\delta(pp_j)+s}{2}$, i.e. the average between the overall interestingness value and the particular interestingness annotation for that plot point. We assume that players do not provide random feedback leading to cases where s is high and all the pp_j values are low, or vice versa.
- A new case consists of the player trace, the interestingness values for each plot point $ppi(pp_j)$, and the confidence c.

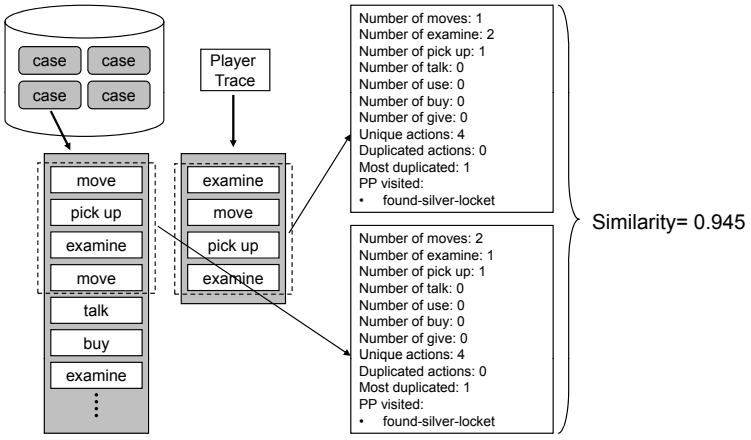

Fig. 6. Illustration of the similarity comparison between player traces. Each player trace is represented by a feature set. Similarity between the player traces is the Euclidean distance between the feature sets.

While a player is playing the game, his game trace is compared to the traces within the different cases stored in the case base. For our text-based interactive game, these characteristics are computed by analyzing the player actions, their choice and ordering. Figure 6 illustrates this process. As mentioned before, the assumption behind player modeling is that players with similar game playing characteristics will have similar interests. To facilitate calculating the similarity between these player action traces, we have categorized the player actions in a collection of classes: move, talk, examine, etc. Table 1 lists the various player action types available in Anchorhead. To illustrate the representation of player actions in our implementation of Anchorhead, consider the following example:

$$pa_8 = \begin{bmatrix} op = request\_for\_help\_with\_puzzle \\ type = Conversation \\ msg1 = \textit{“Request Magic shop owner to help with the puzzle.”} \\ msg2 = \textit{“He solves it to find a magic lens within the puzzle box!”} \\ p_l = magic\_shop \\ p_{pp} = find\_magical\_shop \wedge open\_safe \wedge \neg open\_puzzle\_box \\ e_l = \emptyset \\ e_{pp} = open\_puzzle\_box \end{bmatrix}$$

The player action pa_8 is a Conversation type action. $msg1$ represents the textual description of the player action. $msg2$ represents the textual feedback to the player provided by the game engine as soon as it executes pa_8. The player action's representation incorporates prerequisites (p_l, p_{pp}) and effects (e_l, e_{pp}) for locations as well as plot points. pa_8 is a valid player action only at the magic shop (p_l). Its plot point dependencies (p_{pp}) are defined as a logical expression that require the player to have found the magic shop, opened the safe in the mansion, and not have already opened the puzzle

box. Empty e_l indicates that pa_8 has no effect on the player's location. But this player action unlocks a plot point $open\_puzzle\_box$ for the player (e_{pp}). Thus, using pa_8, the player can request the Magic shop owner's help to open the puzzle box and discover a magic lens.

Figure 6 shows how the similarity between a short player trace and a trace stored in a case is computed. First, each action is classified into its class. Then, a collection of features is computed: number of actions of each class, number of duplicated actions, etc. The feature values are then normalized to be in the interval [0,1] by dividing them by the total number of actions in the trace. Similarity between the player traces is then computed by using a simple Euclidean distance among the features. When we compare two traces of different lengths, only the first n actions of each trace is considered for similarity, where n is the number of actions in the shorter trace.

The PMM retrieves a certain fixed number of similar cases. The interestingness value for each plot point is then computed as a weighted average of the plot point interestingness values in the similar cases. The weight for individual case is its similarity metric value. The output of the PMM is a player model that consists of the predicted interestingness of each plot point for the current player and also a confidence c_{PM} on this player model (as shown in Figure 5). The confidence c_{PM} is computed as the mean of the confidences introduced by the players in the retrieved cases weighted by the similarity with these cases.

4.4 Drama Management Module

Given the player interest model, the current game state, and a set of *author specified story guidelines*, the Drama Management Module (DMM) plans story arcs that maximize player interest (in accordance with the player model) and narrative coherence (in accordance with the author specified story guidelines). Specifically, at every game cycle the DMM uses this information to select, if necessary, a particular action to influence the story. One of the novel aspects of our approach is that it combines a learned player model along with a set of author specified guidelines in order to generate interesting narratives.

In our implementation, we have used the following author specified guidelines in order to maintain story coherence: *thought flow*, *activity flow*, and *manipulation*. Thought flow measures how much the plot changes from one topic to another, favoring plots that do not alternate too much between different topics. Activity flow measures how much the different events in the story are concentrated in the same locations without the player having to move around too much. Finally, manipulation measures how much the DM intervenes to manipulate the story, favoring plots with less intervention from the DM. These author specified guidelines are implemented as a collection of functions that return a number between 0 and 1. This number represents the story's score on the given guideline. See Weyhrauch's work [21] for a detailed explanation on implementing these guidelines. Notice however, that these guidelines do not constrain the way that the story is represented. These are functions that rate a particular story line according to different factors.

The game's author specifies both the sets of actions i.e. the player actions as well as the *drama manager actions* (DM actions). These actions represent the ways in which

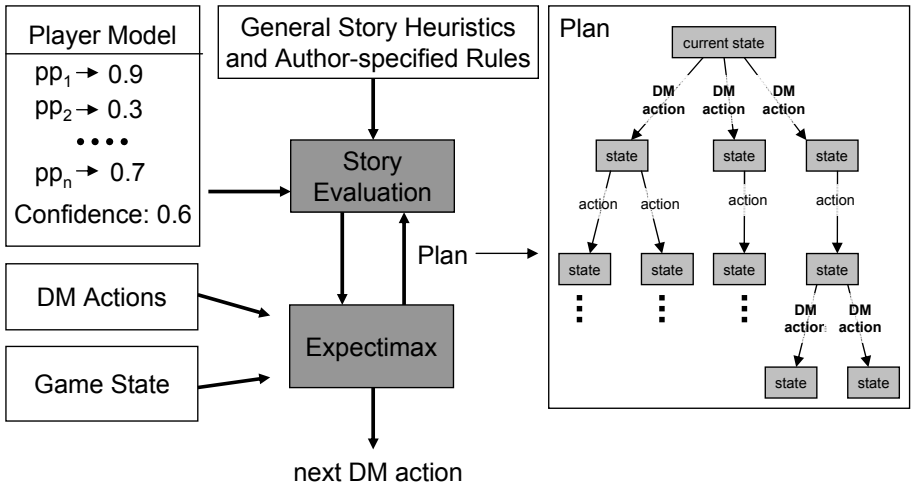

Fig. 7. The Drama Manager Module consists of a planner and a story evaluation module. Given the history of the current game and the player's interest model, this module is responsible to decide a Drama Manager action that is likely to lead the player to interesting story arcs.

the DM can influence the game, e.g. "prevent the player from entering the library by locking the door" or "make the bar tender start a conversation with the player about the suspicious person". The DM has access to the complete current state of the game. It uses search-based techniques to look ahead for possible combinations of player actions as well as predict the effects that different DM actions might produce. Based on the search result, the DM will select some DM action to execute. This DM action is sent to the game engine, that in turn executes the action, thereby affecting the current state of the game. For example, if the DM predicts that by executing a particular player action pa the player might reach a situation that the player will not find interesting, the DM will try to block action pa by executing a DM action (if such a DM action exists) that prevents action pa.

The DM actions can be classified in these groups:

- *Causers* are designed to lead the player towards a particular arc in the narrative, i.e. they *cause* some plot point to happen. Causers can be 'hints' or 'direct causers'. E.g., a hint to the player to go inside the stinking sewer, or a causer to directly make something happen in the game.
- *Deniers* are designed to prevent the player to move towards a particular arc in the narrative. Deniers could be in form of 'hints' or 'direct deniers'. In our design, we have only used direct deniers.
- *Temporary Deniers* are designed to block the player from satisfying some sub-plot for a given period of time. E.g. temporarily hide the keys from the player's view. As expected, for each temporary denier, there exists an accompanying *Re-enabler*. E.g. make the keys visible to the player.

Table 2 lists the DM actions deployed by C-DraGer in Anchorhead. It is important to recognize 'no operation' as a potential DM action. To illustrate the representation of DM actions available in Anchorhead, consider the following example:

$$dma_3 = \begin{bmatrix} op = bum\_hints\_crypt\_key\_is\_in\_basement \\ msg = \textit{"Bum: Did you happen to find the key at the basement?"} \\ p_l = park \\ p_{pp} = true \\ e_l = \emptyset \\ e_{pp} = \emptyset \\ h_a = \{obtain\_crypt\_key\} \end{bmatrix}$$

This DM action dma_3 causes one of the characters in the game, the bum, to tell the player that there is a crypt key hidden in the mansion's basement. It is important for the player to find this key to advance in one of the subplots. Specifically, this action is a *hint*, and h_a represents a set of player actions at which this DM action hints; i.e. after this DM action has been executed during the game, the player is more likely to choose an action from this set. In particular dma_3 hints the action $obtain\_crypt\_key$, which is the internal name that the game engine uses to designate the action that the player executes to obtain the crypt key. msg represents the hint message that the DM action requests the game engine to output as part of the game's interaction. Similar to the player action, the DM action's representation incorporates prerequisites (p_l, p_{pp}) and effects (e_l, e_{pp}) for locations as well as plot points. A DM action can affect player location or cause a particular plot point to be triggered. For example, a denier for a particular action a is implemented by creating a plot point that prevents action a to occur. Then this plot point is added to the DM action's effects (e_{pp}). In the above example, dma_3 does not have any effects. The only prerequisite for this DM action is that the player needs to be at the park $(p_l = park)$. As the game progresses, C-DraGer will choose to execute this DM action if it realizes that providing the key to the player potentially causes the player to reach plot points that would be interesting. In our implementation, we have defined twenty-eight different DM actions shown in Table 2.

In order to decide the next DM action to execute, the DMM uses an *expectimax* algorithm [36]. The expectimax algorithm is very similar to the minimax algorithm used in Chess-like games. Both algorithms construct a search tree and use an evaluation function in the leaf nodes thereby providing a score for each leaf. These scores are propagated up the tree. The difference is that the minimax algorithm assumes that the opponent has opposite goals, whereas the expectimax algorithm does not. Whereas the minimax algorithm picks the minimum score out of all the scores associated with the child nodes, expectimax computes the average score.

As shown in Figure 7, the starting node of the search tree is the current game state. In the odd plys of the tree, each branch consists of a DM action (including a branch for performing no action). In the even plys of the tree, each branch consists of a player action. In our evaluation, we have kept a fixed depth of 5 plys so that the time required by the DMM to search is not appreciable by the player. Each leaf in the tree corresponds to the game state (including the physical state, story state and history) resulting from applying all the actions in its branch to the current game state. For each leaf node (l_j),

DM Action	DM Action Type	Effect In Game
no op	no Effect	No effect
bum amulet	Hint	The bum offers to trade the amulet for a secret
bum crypt	Causer	The bum brigs up the crypt in a conversation
bum Anna	Causer	The bum brings up Anna in a conversation
bum himself	Causer	The bum reveals information about himself
bum crypt	Hint	The bum hints the crypt's location
bum William	Causer	The bum brings up William in a conversation
lock library	Temp Deny	Temporarily deny borrowing books form the library
unlock library	Re-enable	Re-enable borrowing books from the library
deny safe	Denier	Makes discovering the safe impossible
lock safe combo	Temp Deny	The safe combo temporarily disappears from the bed
unlock safe combo	Re-enable	The safe combo reappears in the bed
lock observatory	Temp Deny	Temporarily locks the observatory door
unlock observatory	Re-enable	Unlocks the observatory door
deny evil god	Denier	Damages the telescope (blocking one of the endings)
lock amulet	Temp Deny	Amulet sold out at the magic shop
unlock amulet	Re-enable	Amulet is available again at the magic shop
lock bum amulet	Temp Deny	The bum would not accept the amulet
unlock bum amulet	Re-enable	The bum becomes friendly to you again
deny book	Denier	Cannot locate book (blocking one of the endings)
deny crypt key	Denier	Hides the crypt key
silver locket	Hint	Hints towards finding the silver locket
card	Hint	Hints towards finding the library card
safe combo	Hint	Hints checking out the bed
amulet	Hint	Hints buying the amulet
flask	Hint	Hints buying the flask to bribe the bum
pages	Hint	Hints reading files in the bedroom
scare bum	Hint	Hints scaring the bum

Table 2. List of all the twenty-eight DM actions used by C-DraGer in Anchorhead.

the DMM computes two interestingness values: $p(l_j)$ and $a(l_j)$. $p(l_j)$ is the average interestingness of the visited plot points in l_j (computed using the player model) and $a(l_j)$ is the interestingness obtained using the author defined rules.

Once both $p(l_j)$ and $a(l_j)$ have been computed, the interestingness value of a node is computed as:

$$node_i(l_j) = c_{PM} \times p(l_j) + (1 - c_{PM}) \times a(l_j)$$

where c_{PM} is the confidence value returned by the player modeling module via the constructed player model. The intuition behind the calculation is that if the confidence in the player model is high, weigh the player model higher. If the confidence is low, the system should rely in the author specified guidelines.

The interestingness values are propagated up in the tree by selecting the maximum interestingness in the plys with DM actions and the average interestingness in the plys with player actions. Moreover, if a hint DM action has been executed, the hinted actions

by that DM action will have double the probability of being executed in the subtree below the hint DM action for averaging purposes. Notice that multiplying by two is just an approximation, a more accurate way would involve an analysis of how obvious each hint is, and increase probabilities accordingly. In the end, each DM action in the first ply has an associated interestingness value (obtained by propagating up the interestingness), and the DMM executes the DM action with maximum interestingness. If the maximum interestingness is given to the branch with no DM action, then no DM action will be executed. The result of this process is that the DMM selects the action that leads the story in a direction that is expected to be more interesting for the player.

Finally, the fact that the drama manager is a separate module from the game engine does not limit the way in which it can influence the game. The drama manager's influence is defined by the set of DM actions. As long as the game engine supports a rich collection of DM actions, the drama manager will not be limited in influencing the game. The limitation of separating the drama manager from the game engine is that an explicit game state representation is needed to pass the state information between them. The interaction between these two components can be very frequent in complex real-time games.

5 Evaluation and Analysis

We recruited twenty two participants (referred as P1 ... P22), 4 females and 18 males. The participants ranged across multiple races, education levels, and ages (from 22 to 37 with an average age of 25.5). Nine of these participants had absolutely no or low gaming experience (we call them *non-gamers*), and none of twenty-two participants had played Anchorhead before. Each player was provided with an explanation on Anchorhead. This included a small introduction to the game plot, and instructions on how to interact with the game interface. Each player was asked to play twice. The DM was either absent or present along with the game. However, the players were not disclosed which of the two games enabled the DM (until after they had finished playing both games). Players were asked to sign a consent form before starting the game. The players filled a background questionnaire to help us gather information such as their previous gaming experience and types of games they liked to play. There was no reward offered to the participants of this study.

During each game, one of the authors observed the player's interaction with the game and logged information. This included any unusual reactions such as 'surprised by a game event', or 'amused by a witty hint'. On an average, a complete player interaction (two games) lasted for about 45 minutes. After completing the two games, the participants were interviewed about their experience. The questions are listed in Table 3. The evaluation using player's responses to these questions is elaborated in Section 5.1.

To leverage our case-based approach towards player modeling, the intent of our evaluation methodology is to collect relevant cases at the end of each game interaction. The evaluation was constructed with the following two aspects in mind:

- Account for the order in which players played with and without C-DraGer. Player interest changes from the first to the second time they play the game. Thus, we

designed experiments that compensate for this issue of ordering the two games. Half the players first played with the DM and then without the DM, and the other half of the players played in the reverse order.
- Make participants play with different sizes of case library i.e. vary C-DraGer's knowledge about the player interests.

Based on these two criteria, the evaluation was conducted in the following six phases:

- Phase I: Players P1 to P6 played the game once without the DM. By collecting player feedback at the end of each game, we obtained six cases (C1 . . . C6).
- Phase II: Players P7 to P11 first played without the DM and then with the DM (that used cases C1 . . . C6). Since five players played Anchorhead twice, we obtained ten cases (C7 . . . C16).
- Phase III: Players P12 to P16 first played with the DM (that used cases C1 . . . C6) and then without the DM. The different orders in phases II and III helped in accounting for any possible discrepancy in results due to the order in which the participants play with or without the DM. Like Phase II, we collected ten more cases (C17 . . . C26).
- Phase IV: Players P1 to P6 played the game again with DM that used cases C1 . . . C26.
- Phase V: Players P17 to P22 played the game once with the DM (using cases C1 . . . C26).
- Phase VI: Players P17 to P22 played the game without the DM. Phases V and VI were conducted to account for any possible discrepancy in results due to the order in which players P1 and P6 played with and without DM in Phases I and IV.

The above classification of players might seem complicated, but it essentially divides the players in two groups: the first group (Phases II and III) will play the game with 6 cases (C1 . . . C6) in the case base, and the second group (Phases I, IV, V and VI) will play the game with 26 cases (C1 . . . C26) in the case base. Inside each of these two groups of players, half the players played the game first with DM and then without DM, and the other half played in the reverse order. Players were assigned to Phase I, II, III, IV, V or VI at random. This division allows us to evaluate the effect of the DM and the effect of larger case bases with just twenty-two human participants.

We transcribed the player responses from the interviews and observed players' actions during the games. We analyzed the data obtained, focusing on a qualitative analysis of the results. In order to perform a qualitative analysis we used the well-known qualitative analysis method, *Grounded Theory* [37]. Additionally, we performed an analysis of the player modeling component.

Using grounded theory principles, we made notes for each player action during the game play and their responses from the interview. We processed the transcriptions, first to take open-ended notes and second to highlight more salient ideas expressed by players. We generated codes, such as 'Player is not able to locate the library' and 'Player did not enjoy the ending', and listed multiple player quotes for each idea in an Excel spreadsheet. Within the spreadsheet, we then conducted data analysis where related

codes were grouped together. For example, all codes that had something to do with DM strategies were initially grouped together into related concepts. We went through an iterative process of describing each concept and conceptually linking them to each other. We formed a single hierarchy of the concepts, went back to the spreadsheet and organized the same concept with a paragraph to describe each concept along with the top two to three quotes for each concept. This hierarchical organization of concepts served as an outline for our qualitative findings. A subset of the codes are shown in Table 5.

5.1 Qualitative Results

One of the authors observed every player play the two games. These observations involved recording the player's pacing of the game (e.g. rash player), specific actions the player found interesting (e.g. enjoyed bribing the bum), how well the player utilized the hints from the DM (e.g. player recovered from being completely lost using the DM hint about cracking safe combination), any particular comment on the game (e.g. "I do not like the Magic ball"), and anything specific to that particular player's interaction. At the end of the second game, the author presented the participant with a set of questions (see Table 3). The questions were designed to gather as much data as we could from the player's subjective opinion of the interaction (i.e. both the games combined). The qualitative analysis incorporates the player responses to these questions as well as the author's observations.

Before interviewing the players, the authors disclosed the order in which the players played the games. To apply Grounded Theory, we gathered all the player responses against the questions in a tabular representation. Table 4 shows responses from seven players for three questions (Q4, Q5, Q6 in Table 3) and the prominent author's observations. All the participants answered that the system was easy to use (response for Q1). Most participants answered that they would like natural language elements in the game (response for Q2). The player responses for the remaining questions are mentioned in the relevant upcoming sub-sections. From the qualitative analysis, we draw the following four conclusions.

DM improves overall player experience, especially for inexperienced players. Participants with little or no gaming experience explicitly commented that the "hints from the DM were particularly helpful" (P7). Non-gamers felt that the hints from the DM helped them in collecting objects and finding suitable use for these objects in the game (see comment by P4 in Table 5 under 'DM helped Player'). Note that the author has already disclosed the game where DM was actually present. This author explained the specific game events influenced by the C-DraGer. Since the authors knew the effects of all the DM actions, they made specific notes whenever they observed the player encounter the prominent effects of a DM action. Analysis of observations made during the game play further showed that non-gamers found themselves lost in the game on multiple occasions (see P11 and P12 in Table 4). Sometimes they struggled finding or using existing clues in the game.

It is easy to observe that non-gamers admit that they never followed any specific strategy to play the game (see column Q6 in Table 4 and player comment under 'Lack

Number	Question
Q1	Was it easy or difficult to use the system?
Q2	Would you prefer to use natural language with the characters instead of selecting numbered choices?
Q3	Did you enjoy your interaction? Share a specific narrative that you particularly liked and/or disliked.
Q4	Would you like to play this game (or similar games) again? Why or why not?
Q5	Did you feel that you were able to influence the outcome?
Q6	Did you follow any specific strategies?
Q7	Did you feel as if the Non-Playing Characters reacted to you appropriately?
Q8	Did you ever feel that your input did not effect the Non-Playing Character's behavior? If so, did you change your strategy at that moment?
Q9	How would you like the game to be improved?
Q10	Would you like to comment on any other aspect of the evaluation?

Table 3. Questionnaire presenting the interview questions that the players answered after completing the two games.

of Strategy' in Table 5). A small number of non-gamers felt frustrated during the game play without the DM. Without hints, they had trouble advancing in the game. For instance, one player (P15) asked "Can I quit the game?" while playing without the DM. For a long time, he wandered around the town without successfully advancing in the game. He did manage to finish the game after moving back and forth between multiple sub-plots.

Some players realized the importance of the DM when they played the game without the DM. For instance, player P18 played first with DM and then without DM. During his first game, he did not realize that some of the hints that he was receiving were actually generated by the DM. He assumed that the hints were part of the game itself. Notice that this is a positive player feedback, since this points out that the DM was not very intrusive. Later, while playing without the DM, P18 realized that the DM interventions were important in the previous runs as it provided necessary hints and guidance whenever he was lost. P11 and P7 made similar comments, and were completely lost without the DM. P11, P15, and P17 were non-gamers. This reinforces our thesis that the DM is very helpful for non-gamers, since it can help them when they are lost.

As noted earlier, one intuitive observation from the player interactions was that they were much more comfortable playing the game for the second time. P10 felt that "playing the first game helped him while playing the second time" and that he had a better sense of the context and the goals of the game. This is generally true for most interactive games. But the availability of a DM can aid players with a variety of playing styles based on the their player models even if it is the first time they play the game. This is an important observation for an interactive game, as incorporating a DM can help attract a wider audience that includes non-gamers. Most non-gamers responded positively to playing games in the genre of interactive fiction in future (see column Q4 in Table 4).

Player	Gamer	Q4	Q5	Q6	Observations
P3	Yes	Yes. I can discover the story at my own pace.	Yes (Second Game)	Explore everything	Rash player. Prefers small action set.
P4	No	Yes. I enjoyed the bar. Game presents mystery.	Yes (Second Game)	Loiter	Initially, scared to select spooky actions.
P6	No	Yes. Inventory contents unlock fun actions.	No (Short Game)	No	Avoided conversing with the bum.
P9	No	Yes. I enjoy discovering multiple game endings.	First game aided second	No	Slow paced. Enjoyed hints.
P11	No	No. I do not enjoy puzzles in a game.	Yes	No	Lost without the DM. Hints helped finish game.
P12	No	Yes. I enjoyed puzzles and keeping track of sub-stories in game.	Yes	No	Lost briefly. Quickly recovered using the DM hints.
P14	Yes	Yes. I enjoyed the text-adventure.	Yes	Play all actions	Enjoyed spooky areas. Extremely exploratory.
P19	No	Yes. Game makes you think about decisions.	No (Short game)	Explore everything	Played faster than other non-gamers.

Table 4. Responses from a subset of the twenty-two participants to some of the questions in Table 3: Q4 (Would you like to play such a game again?), Q5 (Did you feel that you influenced the outcome?)), and Q6 (Did you follow any specific strategies?).

DM strategies should depend on player's gaming expertise. Our current DM strategies do not take into account the player's previous game playing expertise and the number of times the player has played Anchorhead since these features were observable by the system. Experienced gamers' comments on this aspect indicated that the hints provided by DM were sometimes "too obvious" (P3) and should be "more subtle" (P5). However, as indicated in the previous subsection, non-gamers felt that hints were extremely useful. These results suggest that the text corresponding to a particular hint should be designed differently depending upon the player's general game-playing expertise. While providing comments about the game (Q10 in Table 3), P3 suggested that the hints could be modeled as additional player actions as opposed to text. These player actions are not available unless the drama manager explicitly enables them as an effect of executing a hint. This gives the players an option to choose whether they wish to acquire some information or not. This feedback can be extended to perhaps embody the drama manager in the form of a Non-Playable Character (e.g. a companion). The player could choose to ignore a conversation with this companion.

The easiest way to collect this information is to ask the player. As soon as the game starts, we can ask the player for previous gaming experience (over some scale) and the number of times the player has played Anchorhead. This information should be included in the player model that the DM uses to plan story arcs. In our future work, this information will be incorporated in the $node_i$ expression used by the Drama Management Module (presented earlier in Section 4.4).

Category	Player Comments
DM helped Player	*"I would have never thought (unless told by DM) that picking up the skull would ever be useful."*
With DM then Without DM	*"I understood the help that I was getting from DM only after I played the game without it."*
Player liked a plot point	*"I enjoyed solving the puzzle to find the magic lens."*
Liked the game	*"Game allows you to discover game elements at ones own pace."*
Game Endings	*"I felt that I could influence the particular outcome in the game."*
Game Duration	*"I would have preferred a longer game."*
Gameplay	*"My actions varied with every new inventory content."*
Game's story	*"The story presented mystery, puzzles and a lot of spooky situations."*
Lack of Strategy	*"I did not follow any specific strategy to play the game."*
Player Action Preference	*"I enjoyed conversing with the Non-Playing Characters."*
DM Actions	*"I did not like deniers."*

Table 5. Applying Grounded Theory to categorize player responses to the post-game questionnaire.

Player model should be richer. An important observation from relating the player's interview responses and their feedback on plot points is that the player model should be enriched to account for the player's liking for hints as well. Certain players liked a particular plot point, but not the hints presented to them at those plot points. Consider this scenario from the evaluation study: Player P_x encountered a puzzle, had an interesting experience solving it, and provided a positive feedback for the plot points associated with the puzzle. Later, P_y (another player with similar playing characteristics) reached the puzzle. At this point P_y received a hint from the DM that eased the puzzle solving process. The DM chose this hint to ensure that P_y completed the puzzle solving story event. At the end of the game, P_y did not mark the plot points associated with the puzzle as interesting.

If the feedback at the end of the game included an opinion on the DM actions (hint, in this example), the player model can help avoid situations like the one mentioned above. P_y would mark the hint as not interesting. This means that the next player that has similar playing characteristics as P_x and P_y would indeed be directed towards the puzzle story arc but would not be presented with the particular hint.

Let us consider another example from or evaluation. In player P21's game, the DM decided to use a denier to prevent P21 to perform an action that lead to a game ending (with the belief that P21 might dislike the ending). Note that this was P21's second game and he knew the existence of the denied player action that lead to the game ending he reached in his first game. The denier was so obvious that P21 felt manipulated. He commented that he would have preferred a hint towards the other ending, instead of a denier. When he was asked if he would play similar games again, he responded "Yes, if there are no deniers". If the player model could predict that P21 didn't like deniers, it would have provided a hint instead of the denier. This scenario conforms with Laurel's comments on introducing constrains in an interactive drama. Laurel [34] explains that the constraints should be applied without shrinking the participant's perceived range of freedom of action. Constraints should not limit what the participant can do, but what

the participant is likely to think of doing next. In the particular game for P21, C-DraGer constrained the player's actions thereby leading to reduced interestingness. Although, as argued by Harrel and Zhu, such shrinking in player's agency could be used deliberatively as an expressive tool [38].

DM strategies should depend on player's strategy. We observed that non-gamers do not follow any particular strategy. They are good followers of the hints from the DM, and thus remain unsure if they actually influenced the ending of the game. They find interactions with non-playing characters relatively more interesting, e.g., bribing the bum, talking with the owner of magic shop. Experienced gamers intelligently pick items that they feel would be useful later in the game.

Experienced gamers typically follow a "breadth-first" search strategy to play a game, i.e. when they reach a room, they try to execute every possible action they can in that room before moving to the next room (see responses from P3 and P14 to the question of being able to influence outcome of the game in Table 4). These gamers admit that a drama manager can be very useful in much larger games where a strict breadth-first search in the game cannot reach the goal of the game without wasting an impractical amount of time. This general feedback can be observed by noting the responses from P6 and P19 when they were questioned about being able to influence the outcome of the game (see Table 4). A typical strategy used by our participants (gamers or non-gamers) was to explore a lot by going to different locations in the map. As a design insight for future, we want to incorporate this observation in DM design. When the DM observes that the player actions are not performing any action that advances towards reaching a subgoal for some finite amount of time, it can provide a hint with the assumption that the player is confused. If the player was indeed trying to simply explore and did not like this hint, he would provide a negative feedback for the particular hint. Later, if another player has similar playing characteristics, he would not be presented with the same hint during those game events.

5.2 Player Modeling Analysis

As the next step of our analysis, we want to validate our assumption that players with similar game playing characteristics (or playing patterns) have similar interests in the context of the game. We performed two independent experiments. The fist one analyzed the correlation between game playing characteristics and player interests. The second one applied automatic clustering techniques on the game playing characteristics.

Correlating Player Interest and Game Playing Characteristics. In our first analysis, we correlate the player interest for individual plot points and the overall experience with different player features used in player modeling. We divided the feature set into four parts namely: Average number of unique actions performed when player visits a location in the game for the first time (F1), average player action time (F2), DM action trace (F3), and other player action trace features (F4). An action trace simply refers to a list of player actions sequenced in the order of selection. We wanted a subset from this feature set that would help us detect similar players in terms of interest for the

Player Model Feature	Correlation with Feedback
F1	-0.021495
F2	0.074829
F3	0.258544
F4	-0.077339

Table 6. The correlation of the four different features of the player trace with player feedback.

game. We observed that typically, players with low gaming experience spent a lot of time moving in different locations without performing a lot of possible actions at any given location. F1 helped incorporate player actions taken across the twelve different locations in Anchorhead. As a typical game for a gamer is of shorter duration than a non-gamer, we used feature F2 to capture this game playing characteristic. The DM action trace (F3) is tested for similarity (defined only when the DM is present) on 7 different features that include length of the DM action trace, number of hints, non-hint causers, permanent deniers, temporary deniers and associated re-enabling actions, and the DM actions without any effect. F4 incorporates 9 features that express the player action trace. These include general trace features like length of the player action trace, the number of unique actions, duplicate actions and the cardinality of the most occurring duplicate action as well as the number of actions from the five possible player action classes (movement, conversation, picking objects, using objects and no effect).

As we observe from Table 6, most of the features (except F3) did not provide appreciable correlation values. One of the possible reasons for a low correlation value can be understood from the following example in the evaluation. One of the players (P7) found a challenge presented to him during the game to be interesting and accordingly provided a high interest for the plot point associated with that challenge. When another player (P9) interacts with the game and has very similar playing characteristics, DM picks a sequence of actions that leads P9 to the story arc that makes him confront the challenge. In order to guide P9 towards the story arc that contains a challenge, DM provides the player hints to lead him towards the challenge plot point. P9 indicated during his interview that as a result of being provided with hints, he didn't find the challenge that interesting. As a result, even though P9 had similar playing characteristics, he provided a lower interest for the plot points associated with the challenge. This suggests that the player model should include whether players like particular DM actions or not, and also that the DM might need hints of different degrees of subtlety for different types of players.

The best correlation was achieved between player feedback and DM action trace features (F3). The high correlation indicates that two players that are presented with similar DM actions during a game typically have similar overall interest in the game and the plot points. For future DM design, this result indicates that the DM action set employed for a particular player can provide an important measure of similarity between different players. This is an interesting finding, and can be explained in the following way. In order to predict player interests, the system needs features that reveal player preference that is easy to distinguish between any two game events. For instance, Thue et al. [32] accomplished this by presenting the player with a choice that elicited

Model	2 Classes		3 Classes		4 Classes	
Feature	INTRA	INTER	INTRA	INTER	INTRA	INTER
F1	0.0784	0.0962	0.0337	0.1153	0.0410	0.1057
F2	0.0640	0.2194	0.0642	0.1682	0.0867	0.1520
F3	0.0899	0.9489	0.0952	0.7921	Not Possible	
F4	0.2448	0.2949	0.2616	0.2879	0.2678	0.2869

Table 7. Spectral clustering on the player models over the four features and an increasing number of classes. INTRA and INTER respectively represent the average intraclass distance for all classes and the average distance between the classes.

the kinds of interactions players prefer. In our game, the different DM actions executed by our system for a given player are the best way to elicit such information.

Features such as the average time a player takes to execute an action (F2), although interesting for drama management purposes (as next sub-section shows), do not exhibit any correlation with predicting interests. Due to low correlation values and different DM strategies employed, our results could not conclusively find a good subset of features that would be useful for detecting the interestingness value of a given plot point for the current player. These results indicate that further research needs to be carried out to find the correct subset of features to use as a player model. As a future step, we want to incorporate these findings to modify the DM design.

Player Model Evaluation. As an offline experiment to find the distinguishing features in the existing player traces, we used spectral clustering [39]. This helped us cluster different players according to the different combinations of features available in their game playing characteristics. Spectral clustering is one of the most common and successful clustering techniques; the main idea is to use the eigenvectors of the similarity matrix of the data points to perform dimensionality reduction and then perform clustering with a reduced set of dimensions.

This involved measuring the similarity in the player traces stored at the end of each game collected from our participants using the selected four sets of features (F1-F4). During clustering, we experimented by increasing the number of classes to find the set of player traces that always form a class irrespective of the increase in the number of classes.

These results lead to interesting observations. While clustering the player traces with the average number of unique actions taken when the player visits a particular location for the first time (F1), we observed that most non-gamer participants always formed a single class when we used 3 and 4 classes. This result is supported by our previous observation that non-gamers typically roamed across various locations without performing a lot of actions at any particular location. They typically revisited the places after being hinted by the DM.

In Table 7, the spectral clustering for the feature set comprising the average time taken by the player to perform an action in the game (F2) provided a strong classification of the player traces. Upon close observation of the player traces in each class, they suggested certain interesting facts about the influence of factors like gaming experience

and the number of times the participant has played the game. Since these classes are based on different amounts of time taken by the users in the game, we would expect one class to be consisting of the players who took a long time to complete the game. This included non-gamers who were lost in the game as well as some gamers who were exploring a lot. The player traces that took, on an average, the least amount of time to select a player action were fairly unaffected by the change in the number of classes being increased. For the majority of the players, their second game belonged in this class. This clearly shows that the average time taken to select a player action is fairly similar (for both gamers and non-gamers) when they play the game for the second time. Thus, this feature can be useful in distinguishing players that play the game for the first time.

The feature set of the DM action trace, F3, provides a good clustering. This is expected as the spectral clustering is able to identify the cases that contain DM action trace well. Since, there are just two classes (DM action trace is either present or absent in the case), spectral clustering is not able to classify the player models in four or more classes. The set consisting of the general player action trace features does not provide a very good clustering. This may partly be due to the fact that it is a collection of nine features and we need to perform a better analysis to find a subset that represents these general features sufficiently and performs good player model classification.

Discussion. We can draw several conclusions from our player model evaluation. First, features such as the average time to take an action (F2) can be useful to distinguish non-gamers from gamers. This is useful, since our qualitative analysis shows that gamers and non-gamers have difference preferences to hints. At the same time, F2 did not strongly correlate with player interests. Other features such as the DM action trace (F3) predict better player interest. This suggests that the player modeling module should use different feature subsets to predict different aspects of the player (gaming experience, interests, etc.). The fact that some features exhibit a positive correlation with respect to player interests constitutes evidence for our assumption that players with similar game playing characteristics have similar interests in the context of a game. Since some of our feature set presented low correlation with player interest, our analysis exhibits a need of careful selection of these features in order to exhibit superior player interest modeling.

Our qualitative evaluation has provided us with evidence that player modeling helps the DM in taking the right decisions. For example, P9 first played with C-DraGer disabled and reached one of the game endings that he disliked. Next, he played with C-DraGer enabled. C-DraGer prevented P9 from reaching the first ending. Subsequently, P9 found the other game ending, which he found more interesting. This shows that even with a low (although positive) correlation between player's game playing characteristics and interests, our player modeling module is able to capture an average interestingness of the plot points that is sufficient to guide the players towards interesting narratives.

6 Conclusions and Future Work

In this paper, we have presented a drama management approach and its evaluation with human players interacting with a text-based interactive fiction game. Our results indi-

cate that incorporating Case-based Drama manaGer (C-DraGer) in Anchorhead provided a better playing experience. This was particularly visible for players with no or low gaming experience. We have shown that player modeling is a key factor in drama management. Our system leverages an automatically learned player model and a set of author specified story guidelines to generate personalized experience for individual players.

Moreover, the evaluation with human players provided us with several extremely useful insights that will guide our future research. Our findings can be summarized as follows:

- The use of a Drama Manager in interactive drama provides a better overall playing experience. This is especially true for inexperienced players as the Drama Manager helps them progress smoothly during the game.
- Player modeling is a key factor for the success of any drama management approach aimed towards significantly personalizing player experience.
- Drama Manager should take into account the player's previous gaming experience. For example, providing more guidance to an experienced game player makes the game less challenging and may become an obstacle for such player's interests.
- Drama Manager strategies should depend on player's strategies. Hints are more useful when a player is lost, than when a player is still advancing in the plot.
- The player model should incorporate player feedback on plot points as well as the strategies used by the Drama Manager during the interaction. For example, the players can provide feedback in the form of interest in causers and deniers.
- Player modeling is hard. Some aspects of the player model are hard to learn and they are easier to obtain by asking the player before starting the game. For example, previous gaming experience.

As future work, we plan to incorporate various design insights into C-DraGer. We want to extend the player modeling capabilities to incorporate how different DM strategies affect interestingness of the game. We also plan to apply our techniques to a more complex game with a longer story. In order to achieve that, we plan to improve C-DraGer's search techniques by applying non-exhaustive and hierarchical search techniques. In order to further improve the search and predictive capabilities of C-DraGer, we want to incorporate a player action modeling module. It will be responsible to learn the actions that a player is likely to perform in different game circumstances. This will aid in predicting player actions, as well as pruning branches of the search tree with low probability of occurrence. Finally, to provide an immersive experience, we are currently working on an advanced game interface that supports real-time games with a graphical presentation.

7 Acknowledgments

The authors would like to thank all the twenty-two participants of the Anchorhead drama management evaluation and the anonymous reviewers for their detailed suggestions that contributed to increase the quality of this paper.

References

1. Ram, A., Ontañón, S., Mehta, M.: Artificial intelligence for adaptive computer games. In: Twentieth International Florida Artificial Intelligence Research Society Conference (FLAIRS), AAAI Press (2007)
2. Laurel, B.: Toward the Design of a Computer-Based Interactive Fantasy System. PhD thesis, Drama Department, Ohio State University (1986)
3. Bates, J.: Virtual reality, art, and entertainment. PRESENCE: The Journal of Teleoperators and Virtual Environments, MIT Press **1**(1) (1992) 133–138
4. Aamodt, A., Plaza, E.: Case-based reasoning: Foundational issues, methodological variations, and system approaches. Artificial Intelligence Communications **7**(1) (1994) 39–59
5. Kolodner, J.: Case-Based Reasoning. Morgan Kaufmann, San Mateo, CA (1993)
6. Leake, D.: Case-Based Reasoning: Experiences, Lessons, and Future Directions. AAAI Press/MIT Press, Menlo Park, CA (1996)
7. Gentry, M.S.: Anchorhead. available online at http://www.wurb.com/if/game/17.html (1998)
8. Sharma, M., Mehta, M., Ontañón, S., Ram, A.: Player modeling evaluation for interactive fiction. In: Third Artificial Intelligence for Interactive Digital Entertainment Conference (AIIDE), Workshop on Optimizing Player Satisfaction, AAAI Press (2007)
9. Sharma, M., Ontañón, S., Mehta, M., Ram, A.: Drama management evaluation for interactive fiction games. In: AAAI Fall Symposium on Intelligent Narrative Technologies, AAAI Press (2007)
10. Nelson, M., Mateas, M., Roberts, D., Isbell, C.: Declarative optimization-based drama management in interactive fiction. IEEE Computer Graphics and Applications **26**(3) (2006) 33–41
11. Young, R., Riedl, M., Branly, M., Jhala, A., Martin, R., Sagretto, C.: An architecture for integrating plan-based behavior generation with interactive game environments. Journal of Game Development **1**(1) (2004)
12. Murray, J.H.: Hamlet on the Holodeck: The Future of Narrative in Cyberspace. MIT Press, Cambridge, MA, USA (1998)
13. Kelso, M.T., Weyhrauch, P., Bates, J.: Dramatic presence. PRESENCE: The Journal of Teleoperators and Virtual Environments, MIT Press **2**(1) (1992)
14. Magerko, B., Laird, J., Assanie, M., Kerfoot, A., Stokes, D.: Ai characters and directors for interactive computer games. In: Proceedings of the 2004 Innovative Applications of Artificial Intelligence Conference. (2004)
15. Mateas, M., Stern, A.: Integrating plot, character, and natural language processing in the interactive drama facade. In: Proceedings of the 1st International Conference on Technologies for Interactive Digital Storytelling and Entertainment. (2003)
16. Nelson, M., Roberts, D., Isbell, C., Mateas, M.: Reinforcement learning for declarative optimization-based drama management. In: Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems. (2006)
17. Ryan, M.L.: Possible Worlds, Artificial Intelligence, and Narrative Theory. Indiana University Press, Bloomington, IN, USA (1991)
18. Magerko, B.: A comparative analysis of story representations for interactive narrative systems. In: Third Artificial Intelligence for Interactive Digital Entertainment Conference (AIIDE), AAAI Press (2007)
19. Magerko, B.: Player Modeling in the Interactive Drama Architecture. PhD thesis, University of Michigan (2006)
20. Magerko, B.: Evaluating preemptive story direction in the interactive drama architecture. Journal of Game Development **2**(3) (2006)
21. Weyhrauch, P.: Guiding Interactive Drama. PhD thesis, Carnagie Mellon University (1997)

22. Nelson, M.J., Mateas, M.: Search-based drama management in the interactive fiction anchorhead. In: First Artificial Intelligence for Interactive Digital Entertainment Conference (AIIDE), AAAI Press (2005)
23. Lamstein, A., Mateas, M.: A search-based drama manager. In: AAAI 2004 Workshop on Challenges in Game AI
24. Flaishon, I.: Dreampath. available online at http://drakevision.com/projects/dreampath/ (2009)
25. Crawford, C., Mixon, L.J.: Storytron. available online at http://www.storytron.com/ (2008)
26. Yannakakis, G.N., Maragoudakis, M.: Player modeling impact on players entertainment in computer games. In: User Modeling 2005. Number 3538 in Lecture Notes in Computer Science. Springer-Verlag (2005) 74–78
27. Houlette, R.: Player modeling for adaptive games. In: AI Game Programming Wisdom II. Charles River Media (2004) 557–566
28. Charles, D., Black, M.: Dynamic player modeling: A framework for player-centered digital games. In: Proceedings of the International Conference on Computer Games: Artificial Intelligence, Design and Education. (2004) 29–35
29. Prendinger, H., Mori, J., Ishizuka, M.: Recognizing, modeling, and responding to users affective states. In: User Modeling 2005. Number 3538 in Lecture Notes in Computer Science. Springer-Verlag (2005) 60–69
30. Togelius, J., Nardi, R.D., Lucas, S.M.: Making racing fun through player modeling and track evolution. In: SAB'06 Workshop on Adaptive Approaches for Optimizing Player Satisfaction in Computer and Physical Games. (2006) 61–70
31. Cowley, B., Charles, D., Black, M., Hickey, R.: Using decision theory for player analysis in pacman. In: SAB'06 Workshop on Adaptive Approaches for Optimizing Player Satisfaction in Computer and Physical Games. (2006) 41–50
32. Thue, D., Bulitko, V., Spetch, M., Wasylishen, E.: Interactive storytelling: A player modelling approach. In: Third Artificial Intelligence for Interactive Digital Entertainment Conference (AIIDE), AAAI Press (2007)
33. Albrecht, D.W., Zukerman, I., Nicholson, A.E.: Bayesian models for keyhole plan recognition in an adventure game. user modeling and user-adapted interaction. (1998) 5–47
34. Laurel, B.: Computers as Theatre. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA (1993)
35. Sharma, M., Ontañón, S., Strong, C., Mehta, M., Ram, A.: Towards player preference modeling for drama management in interactive stories. In: Twentieth International Florida Artificial Intelligence Research Society Conference (FLAIRS), AAAI Press (2007) 571–576
36. Michie, J.: Game-playing and game-learning automata. In Fox, L., ed.: Advances in Programming and Non-Numerical Computation. (1966) 183–200
37. Strauss, A., Corbin, J.: Basics of Qualitative Research: Grounded Theory Procedures and Techniques. Sage (1990)
38. Harrell, D.F., Zhu, J.: Agency play: Dimensions of agency for interactive narrative design. In: AAAI Symposium on Intelligent Narrative Technologies II, AAAI Press (2008)
39. Ng, A.Y., Jordan, M.I., Weiss, Y.: On spectral clustering: Analysis and an algorithm. In: Advances in Neural Information Processing Systems 14, MIT Press (2001) 849–856