Suggesting New Plot Elements for an Interactive Story

Spyridon Giannatos, Mark J. Nelson, Yun-Gyung Cheong, and Georgios N. Yannakakis
Center for Computer Games Research
IT University of Copenhagen
Rued Langgaards Vej 7
Copenhagen, Denmark
{spgi,mjas,yugc,yannakakis}@itu.dk

Abstract

We present a system that uses evolutionary optimization to suggest new story-world events that, if added to an existing interactive story, would most improve the average interactive experience, according to author-supplied criteria. In doing so, we aim to apply some of the ideas from drama-managed storytelling, such as authorial aesthetic control, in an unguided setting more akin to emergent storytelling: rather than guiding or directing a player towards an experience in line with an author's aesthetic goals, the storyworld is augmented with new content in a way that will tend to align with an author's goals, even if the player is not guided. In this paper, we present an offline system, and demonstrate its robustness to a number of variations in authorial criteria and player-model assumptions. This is intended to lay the groundwork for a future system that would generate new content online, allowing for interactive stories larger than those explicitly written by the author.

Introduction

The simplest kind of interactive narrative is an explicitly branching narrative: a story is written to have certain bifurcation points, where the player's action can influence which branch is taken. Past a certain level of branching, however, this explicit branching tends to get unwieldy to write and manage. As a result, interactive-story systems commonly define the branching space implicitly, by specifying a story space, rather than explicitly specifying every possible traversal of that story space.

When a story space is specified in this implicit manner, there are many stories (traversals of the story space) that have not been explicitly written by the author. Thus it is not necessarily the case that the stories will all be interesting or coherent. There are a few possible responses. One approach, taken by advocates of emergent narrative, is to argue that we should embrace the lack of control over which particular traversals emerge, and focus our authoring efforts more on the story world: if we produce interesting story worlds, then players will have interesting narrative experiences when they traverse them (Aylett et al. 2006).

A second approach is to factor the narrative logic into a separate component from the logic defining the story world itself; this separate drama-manager or story-director will then monitor and guide traversals of the story world (Laurel 1986; Bates 1992). This lets the author first construct a broad narrative possibility space, without worrying that all possible traversals must be good ones, and then separately prune the undesired possibilities with a component that intervenes in real time (e.g. by enabling and disabling possible events and interactions). The pruning can be done according to various criteria, e.g. by requiring that all traversals meet certain narrative goals (Charles et al. 2003; Young et al. 2004), or by defining an experience-quality metric and promoting more highly-rated traversals (Weyhrauch 1997).

We investigate a third approach: expanding the story-possibility space to improve the average quality of traversals. As with emergent-narrative approaches, we simply let story emerge as the user traverses a story space. But as with drama-management approaches, we have criteria for the kinds of narrative experiences we prefer. Instead of using the criteria to intervene in the story world, however, we adapt the story world according to the criteria. More specifically, we *expand* the story world in ways that improve it. While other kinds of adaptation are possible, such as removing parts of the story-world or changing constraints within it, one of our larger goals is to work towards a more generative narrative space, where players can can go off infinitely in any direction, so we prefer to focus on improvements that add to the story-world.

In this paper, we start tackling the problem by looking at offline adaptation, conceived as an authoring tool that suggests new story elements to an author. We adopt the plot representation used by Weyhrauch (1997), which defines a narrative possibility space as a set of plot points—important events that can take place in the story—each of which has a series of semantic annotations indicating its role in the story, and a set of ordering constraints that the story world imposes.[1] Our system suggests the single plot point that would, if added, most increase the average quality of traversals of the story space, according to the author's supplied

[1] A secondary advantage of using a story representation also used by drama managers is that we can later add a drama-manager to our system as well. There is interesting future research into how a drama manager might need to change when the story-possibility space is not fixed.

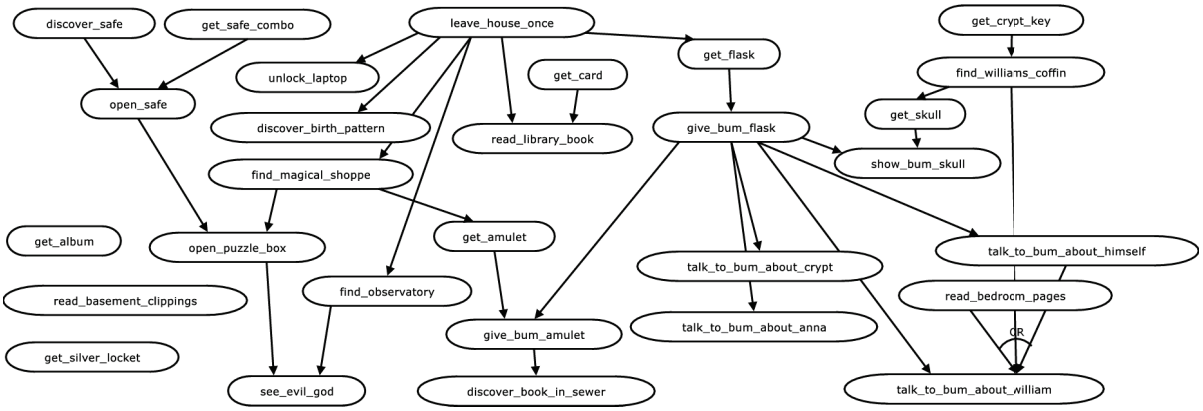


Figure 1: Precedence-constraint graph of *Anchorhead*'s plot points. Note that this is not a branching story graph, but rather an encoding of ordering constraints. The equivalent branching-story graph would be much larger, and would include all order-respecting traversals of this constraint graph.

experience-quality rating function.

Problem formulation

Following Weyhrauch (1997), our system works on an abstraction of a story's possibility space, defined by *plot points*. A plot point is a possible story-world event that has narrative importance, such as the player taking some narratively-relevant action, or learning a piece of information. Since plot point names are by themselves simply opaque symbols, each one is annotated with some information about its role in the story. Simple kinds of annotations include which subplot a plot point could be part of; its location; whether it foreshadows other plot points; and so on.

These plot points are then stored in a graph representation that summarizes the ordering constraints imposed by the story world (for example, plot points that take place in a locked room might not be possible until a *get-key* plot point takes place). A possible story is any traversal of the plot-point graph that respects the ordering constraints. The resulting stories (in their abstract versions, as sequences of annotated plot points) can then be rated by an evaluation function that scores them according to various desired features.

Figure 1 shows the plot points, along with their ordering constraints, for a portion of the interactive fiction *Anchorhead*, as adapted by Nelson & Mateas (2005). This plot-point graph defines the space of possible specific stories that can arise when the interactive story is played. Following Weyhrauch (1997), we can rate the quality of any single story by adding together several rating functions, each specifying a desired property of the story. For example, a story might be rated on its spatial coherence (not having events jump around too often); or on whether it leaves any foreshadowed events as unresolved loose ends.

From these ratings of specific stories, we can infer a rating for the overall quality of an interactive story, by defining it as the average quality of stories that can arise, given the ordering constraints.[2] This requires a player model that determines how likely each story is to actually occur. In the absence of empirical data for such a player model, we can approximate one using simple player models, such as a player that acts completely randomly, or that takes actions in proportion to their distance. Testing several such player models can also help us understand how sensitive the rating is to assumptions about player behavior.

Our goal here is to suggest the single plot point that would, if added, most improve the average quality of the stories. To keep things understandable, we start by looking at adding points to maximize single desired features: for example, what is the plot point that, if added, would most increase the average spatial locality of stories?

Fitness functions

The story evaluation (fitness) functions rate sampled stories on several criteria, such as spatial locality. These are converted to fitness functions for a candidate plot point through a simulation as depicted in Figure 2. The fitness of a candidate plot point equals the average story evaluation function difference between the augmented story and the existing story.

For the experiments presented in this paper, we tested three fitness functions proposed by Weyhrauch (1997) which are based on features of the plot point (Location, Thought, Motivation).

Location flow The location flow fitness function (f_L) calculates the spatial locality of plot points. Specifically, it measures the proportion of plot points that occur in the same location as the preceding plot point. Therefore, this fitness function scores stories highly if clusters of events happen in the same location, before moving on to more actions elsewhere, and penalizes stories where sequences of events rarely happen in the same location.

[2] Other definitions are possible; for example, we might conservatively want to just avoid very poor stories, and thus want to maximize the minimum story quality, rather than maximizing the average. For this work, however, we stick to average story quality.

Thought flow The thought fitness function (f_T) is calculated in the same way as the location fitness function, but using the thought feature instead; i.e. it measures the proportion of plot points that are on the same thought topic as the plot point before them. This function promotes the evolved stories that contain short snippets of coherent sub plots. For example, `give_bum_amulet` and `ask_bum_about_Anna` are both annotated with the thought feature `bum`, so the thought flow function will prefer plots in which the player gives the amulet to bum and then asks him about Anna (or vice versa), rather than giving the amulet to him, getting distracted by something else and then asking about Anna.

Motivation Some plot points are annotated with the information that they would motivate another plot point. This motivation function (f_M) measures whether the plot points occur out of nowhere or occur after other plot points have been motivation for them in player's mind. For example, first finding the observatory (`find_observatory`) and then noticing that the telescope is missing a lens would make opening the puzzle box and finding a lens inside (`open_puzzle_box`) motivated, while opening the puzzle box without having found the observatory would make the discovery of the lens unmotivated. (This is distinct from the ordering constraints in the story graph, because those specify what *must* happen for a plot point to even be possible, not what we would *like* to have first happened for a plot point to be motivated.)

Player models

We use two dissimilar player models to investigate the impact of player model idiosyncrasies to the performance of the optimization algorithm and the quality of the generated plot point graph. Player models could be derived from empirical player data; however, we use ad-hoc designed players for this initial study.

Random player We assume that this player has no knowledge of the story and the consequences of his actions, so he acts in a random manner to explore the story. The random player uses a uniform random distribution for choosing his next action, as all transitions must be equally likely to happen.

Location-biased player This player also uses a uniform distribution for the next action which is, however, biased by its previous location. First, we check whether a random number exceeds the *locationBias* threshold (30% in this paper) of the player model and if this is the case, the next transition will be in the same location as it was before. If there are many possible transitions within the same location, the model chooses one of them randomly. If the *locationBias* threshold is not exceeded, the player model chooses a random transition which is independent of the previous location.

The main advantage of this model is that it has some characteristics of a real player. Imagine that a real player is located at the `pub` and her action is `give_flash_to_bum`. Most probably, she will choose her next action at the same location, for example `talk_to_bum_about_William` and not something that she could do at the `house` far away from the `pub` and then return back to chat with the bum.

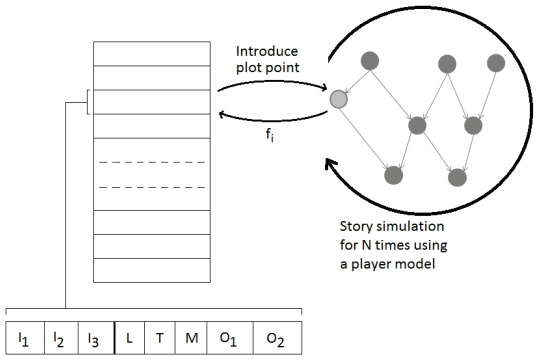


Figure 2: Assessing the fitness of a new plot point. The GA chromosome in this paper includes three incoming edges (I_1, I_2 and I_3), two outgoing edges (O_1, O_2), as well as the location (L), the thought (T) and the motivation (M) feature of the plot point. Each chromosome (plot point) is introduced to the existing plot-point graph and the story is simulated for N times via a player model. The simulation tests the fitness (f_i) of the new plot-point.

Genetic search

This section presents our genetic search approach to plot point generation. The section includes the presentation of the different story evaluation (fitness) functions tested and the two player models designed.

We generate new plot points using a standard genetic algorithm (GA) (Mitchell 1998). [3] According to our implementation, each member of the GA population represents a possible new plot point. To compute its fitness, we construct a story graph that includes the plot point candidate, and sample N possible playthroughs (N is 100 in this paper) from the augmented story, using two types of player model (location-biased and random players, as explained below). The fitness is computed as the average of the fitness values obtained from evaluating the 100 playthroughs. Figure 2 shows the basic steps for assessing the fitness value of each new plot point generated. The set of fitness functions tested is presented in detail in the following subsection.

Each plot point contains a set of features—i.e., location (L), thought (T), motivation (M) in this paper—and a set of incoming (I) and outgoing (O) edges (the ordering constraints); these values are represented as integers. The interval of the features lies between 0 and the total number of each feature, while the interval of the edges lies between 0 and the total number of plot points — when a gene has the value 0, the specific feature or edge is not used. Therefore, the genotype is an array of integers, while the phenotype is

[3] A general introduction to the use of evolutionary computation for procedural game content generation (such as levels, characters, stories, and camera views) can be found in (Togelius et al. 2011; Yannakakis and Togelius 2011).

Table 1: Best fitness values obtained for all fitness functions and player models investigated in GA runs of 600 generations. Numbers in parentheses denote the fitness of the original story.

Fitness Function	Random Player	Location-biased
f_L	0.6845 (0.4093)	0.7112 (0.5173)
f_T	0.1807 (0.1020)	0.2143 (0.1162)
f_M	0.6537 (0.0967)	0.6315 (0.0926)
f_W	0.4360 (0.2357)	0.5176 (0.2928)

a plot point composed by its features and its edges. In a GA algorithm, this is a direct encoding, because each part of the genotype maps to a specific part of the phenotype.

Each generation of the GA implementation presented in this paper contains the following algorithmic steps. Each population member (new plot point) is introduced to the existing story and it is evaluated according to the procedure depicted in Fig. 2. Then, a number of parents is selected using elitism: 60% of the best fit chromosomes are chosen as candidate parents (we use a population size of 100 in this paper). The selected parents are recombined and breed a number of offspring that amounts to 40% of the population size. Uniform crossover occurs with a probability of 65%. Next, mutation is applied on all generated offspring with a rate of 2%; mutation randomly picks a new integer via a uniform distribution to replace the current value of the gene to be mutated. The generated offspring—after crossover and mutation—replace the 40% least fit members of the current population. To minimize the non-deterministic effects of genetic search we executed 10 independent runs for all GA experiments presented in this paper.

Results

This section presents the main set of experiments performed in this paper. The focus of our experimentation is two-fold: first, we examine whether genetic algorithms are able to select a highly fit new plot point across all story-evaluation functions designed, and, second, we test the impact of the player model to the performance of the GA. Then, we examine the new story graphs generated and discuss the benefits of plot points added from an authorial perspective.

Table 1 summarizes the main results obtained for each of the three fitness functions — location (f_L), thought (f_T), and motivation (f_M) — independently and their weighted sum (f_W) fitness function; in these results equal weights are used giving the evolutionary process an equal amount of selective pressure from each fitness functions. Results from both player models are included. Given the findings of Table 1, it appears that genetic search is efficient as the average story quality manages to improve in all scenarios tested. Moreover, it is clear that the algorithm is robust across different fitness functions and player models. Figure 3 illustrates the fitness evolution graphs of all fitness functions examined and demonstrates that it takes the GA approximately 400 to 500 generations to converge to highly fit new plot points.

Table 2: Experiments with different combinations of weights (first three columns) for the weighted sum fitness function f_W evaluated on the Location-biased player model. The f_W value depicted is the best fitness obtained in 600 generations. The f_W value in parentheses is the original story's fitness.

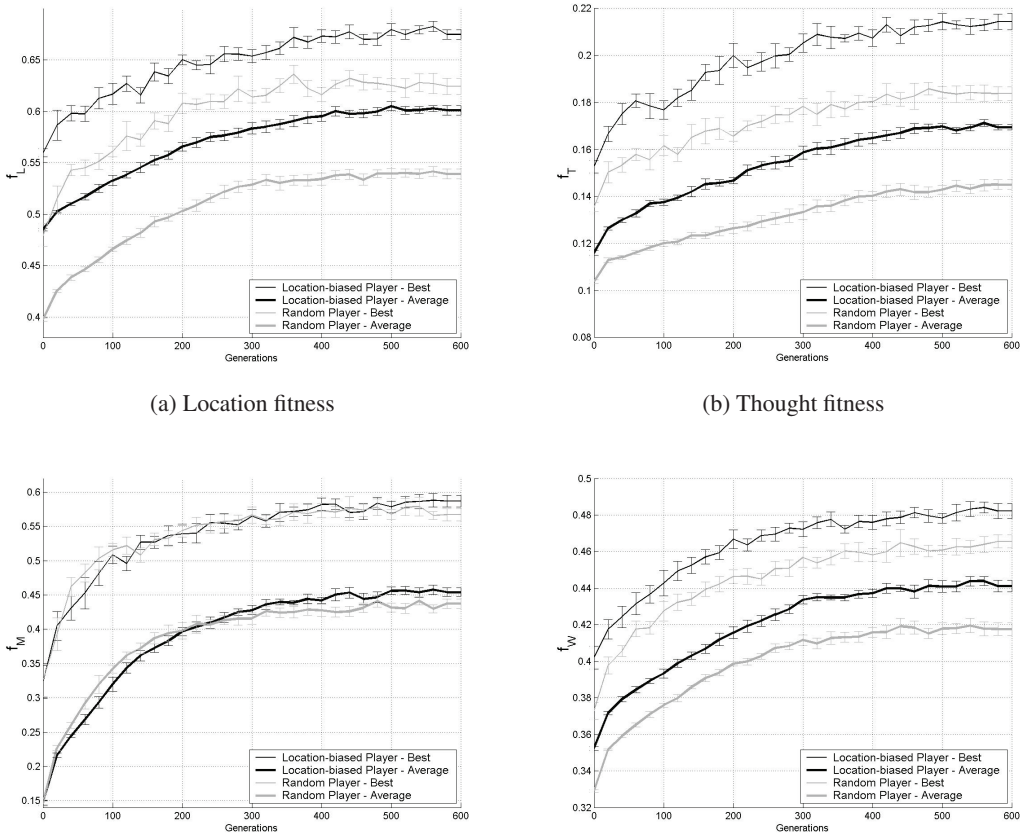
w_L	w_T	w_M	f_W
1	2	2	0.4812 (0.3309)
2	1	3	0.5176 (0.2928)
2	4	2	0.4318 (0.2603)
4	2	4	0.4979 (0.2981)
3	2	1	0.4732 (0.3554)
1	4	3	0.3914 (0.2037)
2	1	3	0.4832 (0.2875)
3	1	2	0.4864 (0.3409)

After testing the ability of the GA for efficiency and robustness we attempt to examine the impact of different weight vectors of the f_W to the performance of the algorithm. Table 2 depicts the performance of genetic search across different vector weights for the f_W function. Observing the best fitness obtained it is obvious that the GA appears to be robust in generating plot points that yield high fitness values across all different weight vectors. One may view these weights as knobs that an author may twist to influence the generation of new plot points. For example, a high f_L weight will most likely result in a plot point that is spatially close to the location of its connected plot points.

In the following we speculate a highly fit new plot point and — after linking it to the existing plot point graph — we observe its potential impact to the existing story world.

Augmenting plot-point example Figure 4 illustrates the best story graph produced by the Location-biased player model using the f_W fitness function, with weights shown in the last row of Table 2. In this story, the new highest-fit plot point, which we name `look_under_the_safe`, has the location `house`, and thought `safe`. This plot point appears reasonably good as its constraints link it to plot points of the same location and thought features; in addition, the safe is located in the house, so that line of thought is likely to coincide with its location.

Notice that this plot point becomes enabled only after `leave_house_once`, `get_silver_locket` and `discover_safe` have been visited. All of these plot points are located in the house, so a possible valid gameplay scenario is as follows: the player searches the house, gets the silver locket, discovers the safe and opens it. Most of the important actions that the player can perform in the house are done, so she leaves the house. However, this new plot point's outgoing edges make it a precondition for both of the story endings: (`see_evil_god` and `discover_book_in_sewer`. Thus, the new plot point represents some entity in the house that causes the player to



(a) Location fitness

(b) Thought fitness

(c) Motivation fitness

(d) Weighted Sum fitness

Figure 3: Evolution of the fitness functions tested across the two player models (Location-biased and Random). Figures depict average values of 10 runs and corresponding 95% confidence interval bars every 20 generations.

think about the safe, which must be done before any endings can be reached.

The chosen name for this plot point (`look_under_the_safe`), is based on the above-mentioned context, in which the player finds William's manuscript. Notice that in the original *Anchorhead* story the `get_silver_locket` plot point has no edges at all, which means that the player can finish the game without ever getting the locket. However, in the new story the locket must be taken and the player might find information about William's manuscript by examining and opening the locket, more closely connecting that event to the necessary causal structure of the story.

Another interesting aspect is that this added plot point has significantly reconfigured all possible playthroughs, by inserting itself as a precondition for both story endings. In future work it would be worth investigating an option to constraint suggested plot points to less far-reaching plot changes, in the case where the author merely desires to elaborate on a mostly finished story, rather than receive suggestions that involve major plot changes.

Conclusions and future work

We have demonstrated an approach to augmenting an interactive story by inserting new plot points into its story graph, in a way that improves the average quality of stories players encounter when they traverse the story space, according to an author-supplied evaluation (fitness) function. For now, the main purpose is to suggest to authors new plot points they might add to their interactive story, if they wish to improve it according to a set of predefined criteria.

Our longer-term goal is to generate plot points online, while the story is being played via indicators of player experience such as affective recognizers (Yannakakis and Togelius 2011; Aylett et al. 2006). That would permit an "infinite story" type of experience; in addition, it would let us tune the new plot points being generated to both the particular player currently playing the game, and to the location of the story they have reached in their play thus far. However, the representation we currently use is an abstraction of the game world: plot points and their annotations capture narratively-important elements of the game world, but are not themselves the game world; they omit many details and concrete elements, from minor intermediate events

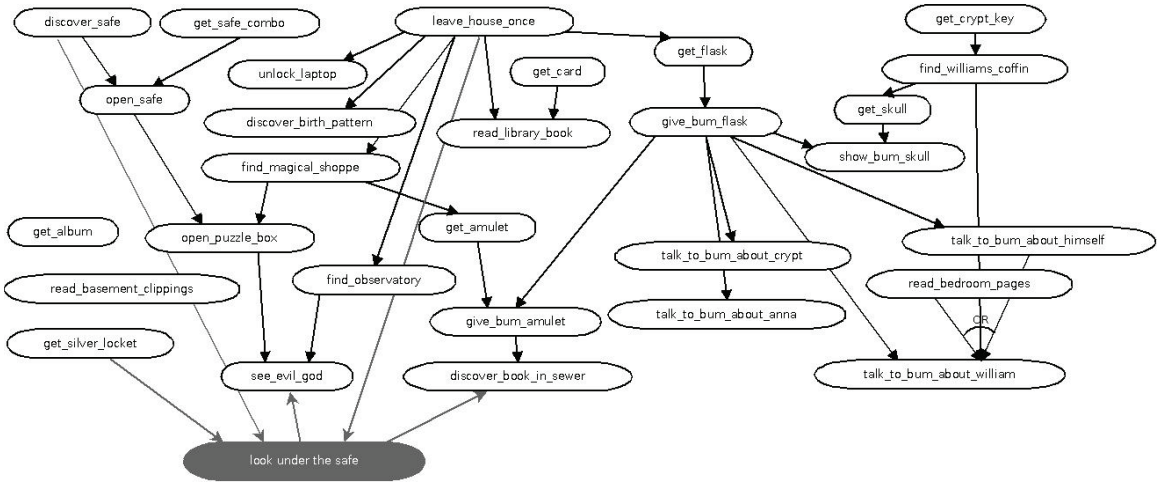


Figure 4: Best story graph obtained through the f_W fitness function and the Location-biased player model. The new point point `look_under_the_safe` appears as the grey ellipse.

to dialog trees and character behavior. That could be overcome in several possible ways. One way would be to have a second phase of generation, where the new high-level plot point is turned into a concrete new portion of the story. This could be done by, for example, adapting a library of existing concrete game elements, in the manner of case-based reasoning; or by handing off specific kinds of plot points to a specialized generator, such as a quest generator (Sullivan, Mateas, and Wardrip-Fruin 2009). The other way would be to use a story representation that is both semantically meaningful enough to generate new elements, and yet directly executable (Magnusson and Doherty 2008; McCoy et al. 2010).

Acknowledgements

This work has been supported in part by the EU FP7 ICT project SIREN (project no: 258453).

References

Aylett, R.; Louchart, S.; Dias, J.; Paiva, A.; Vala, M.; Woods, S.; and Hall, L. 2006. Unscripted narrative for affectively driven characters. *IEEE Computer Graphics and Applications* 26(3):42–52.

Bates, J. 1992. Virtual reality, art, and entertainment. *Presence: The Journal of Teleoperators and Virtual Environments* 2(1):133–138.

Charles, F.; Lozano, M.; Mead, S. J.; Bisquerra, A. F.; and Cavazza, M. 2003. Planning formalisms and authoring in interactive storytelling. In *Proceedings of the First International Conference on Technologies for Interactive Digital Storytelling and Entertainment (TIDSE)*.

Laurel, B. 1986. *Toward the Design of a Computer-Based Interactive Fantasy System*. Ph.D. Dissertation, Drama department, Ohio State University.

Magnusson, M., and Doherty, P. 2008. Logical agents for language and action. In *Proceedings of the Fourth Artificial Intelligence and Interactive Digital Entertainment Conference (AIIDE)*, 72–77.

McCoy, J.; Treanor, M.; Samuel, B.; Tearse, B.; Mateas, M.; and Wardrip-Fruin, N. 2010. Comme il Faut 2: A fully realized model for socially-oriented gameplay. In *Proceedings of the Third Workshop on Intelligent Narrative Technologies*.

Mitchell, M. 1998. *An Introduction To Genetic Algorithms*. MIT Press.

Nelson, M. J., and Mateas, M. 2005. Search-based drama management in the interactive fiction Anchorhead. In *Proceedings of the First Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE)*, 99–104.

Sullivan, A.; Mateas, M.; and Wardrip-Fruin, N. 2009. QuestBrowser: Making quests playable with computer-assisted design. In *Proceedings of the 8th Digital Arts and Culture Conference*.

Togelius, J.; Yannakakis, G. N.; Stanley, K. O.; and Browne, C. 2011. Search-based Procedural Content Generation: A Taxonomy and Survey. *IEEE Transactions on Computational Intelligence and AI in Games*. (in print).

Weyhrauch, P. 1997. *Guiding Interactive Drama*. Ph.D. Dissertation, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA. Technical Report CMU-CS-97-109.

Yannakakis, G. N., and Togelius, J. 2011. Experience-driven Procedural Content Generation. *IEEE Transactions on Affective Computing*. (in print).

Young, R. M.; Riedl, M. O.; Branly, M.; Jhala, A.; Martin, R. J.; and Saretto, C. J. 2004. An architecture for integrating plan-based behavior generation with interactive game environments. *Journal of Game Development* 1(1).