A Stochastic Approach to Character Growth in Automated Narrative Generation

Josh Wade,* Josiah Wong,* Max Waldor,* Lucas Pasqualin,* Klaus Jantke, Rainer Knauf,*** Avelino Gonzalez***

*Department of Computer Science, Univ. of Central Florida – Orlando, Florida
[joshwade@knights.ucf.edu, josiah.w.ucf@knights.ucf.edu, maxwaldor@gmail.com, lucas.pasqualin@knights.ucf.edu, gonzalez@eecs.ucf.edu]

**ADISY Consulting – Weimar, Germany [Klaus.P.Jantke@adisy.de]

***Technical University – Ilmenau, Germany [Rainer.Knauf@tu-ilmenau.de]

Abstract

This paper describes the AESOP program, a modular storytelling framework that composes storyboards for stories that emphasize character development. Through a combination of modules designed to represent story elements, such as Characters, Actions, and Conditions, AESOP combines the novelty of stochastic events with the grounding of logic progression. These modules form a unique approach that synthesizes previous storytelling systems. A prototype system was built and tested in cooperation with the Fraunhofer Inst. for Digital Media Technology. Our tests experimented with the inclusion and removal of certain modules from the story origination process, and our results showed that the introduction of our modules improved the perceived quality of the generated stories over random story origination. Though the system has a number of shortcomings in terms of its details, readability, and flexibility, it makes great strides in automated story origination.

Introduction

Narrative is a pervasive and essential element of the human experience. It is present in areas such as art, psychology, written literature, and theatrical performing arts. All these areas manifest the various ways that humans represent their experiences in the form of narrative, a phenomenon called "narrative intelligence" (Mateas & Sengers, 1999). Thus, an intriguing challenge for AI researchers today is imbuing intelligent systems with this narrative intelligence so that computer systems can tell stories of their own.

Storytelling is a tradition that predates history. It is interesting that for being one of our most familiar and long lasting traditions, researchers are still examining the minute details of how it works and why we use it. Nevertheless, there are many questions about stories that scholars and psychologists answered long ago. Stories are how we pass on our values, encode our culture, and craft meaning for our time in this world; if nothing else, they can be a very effective distraction. But what makes a story good? Is the quality that makes a story powerful not only observable, but also measurable? Can we find exact rules that govern all great stories? If so, could a machine follow these rules and create such stories without human guidance?

These are the questions and curiosities that drove our research described here - a process of program design with the goal of creating a smart artificial story generator and storyteller. We studied story structure and story creation, endeavoring to include every beneficial element. Our methods were heavily motivated by the work of Hollister (2016), who wrote his doctoral dissertation on the power of oral story origination and telling through digital means. Additionally, Knauf and Jantke, the creators of the storyboard model (Jantke & Knauf, 2005), were both inspiration and guidance for us in this project.

Inspiration from Hollister's work (i.e., the means to express our stories both formally and graphically through the storyboard model), and research from a number of previous story origination all played critical roles in the development of our *Automatic Eclectic Story Origination Program* research (AESOP). In many ways, our program climbs above the shoulders of giants on which it started through a number of innovations. These new directions include a system that considers character attributes that define a character's skills and traits, contextual graphs that define hard rules for different sections of the story, relationships between characters that define how they feel about each other, and an algorithm

that allows story actors to plan paths through the story to achieve certain objectives. These innovations seek to take digital story origination and, if only by a small amount, push forward the state of the art.

Background

Our program is not the first story origination program, and it certainly will not be the last. To develop our program more fully, we drew inspiration from a number of programs.

One of the earliest attempts at narrative origination was MINSTREL (Turner, 1992), which used a combination of author-level and character-level goals to fulfill story plans, accounting for character motivations such as finding love. This allowed characters to pursue goals within a premade framework, but author-level goals had to be specified in code, severely limiting scalability.

MEXICA (Pérez y Pérez, 2007) was a narrative generation system that introduced the Engagement-Reflection (ER) cycle. During the Engagement stage, the program created story content with emotional links and inter-character tensions. In the Reflection stage, this content was revaluated to maintain consistency between events and character relationships. MEXICA excelled in creating coherent and novel stories, but was limited in its reliance on a case library of previous stories and its lack of a guaranteed story resolution.

Riedl and Young (2010) created the IPOCL (Intent-based Partial Order Causal Link) planning system which aimed to ensure event causality and the illusion of characters acting with intent, using high-level plans to guarantee story completion and frames of commitment to ensure that all actions were in pursuit of character goals. IPOCL succeeded in creating complex stories where motivations were clearly conveyed and resolved within a coherent story but was unable to run in real-time or allow stories where characters failed.

Riu (Ontañón & Zhu, 2010) was a narrative generation system that created new stories by using force dynamics and case-based reasoning. In Riu, story events involved an Agonist affected by forces from an Antagonist, their various interactions marked by "phases" that were completed using elements from a previous story that was ranked as the most similar to the story at hand. Riu successfully created stories through analogy to past stories, but was limited by its need for these stories to exist a priori.

More recently, CAMPFIRE STS (Hollister, 2016) is an interactive narrative generation system that uses a contextual approach to construct stories that can be manipulated by a user in real-time without the user embodying a character in the story. CAMPFIRE STS used a variety of solution contexts to fill in a story plan that created an array of stories where all elements added to a story were consistent with the various themes selected for the story (such as character traits and story setting). CAMPFIRE STS was also able to accommodate changes requested by the user and regenerate the story on the spot, which was shown to improve story acceptance. Its main limitation was the voluminous prewritten content necessary for the system.

Other work on narrative generation also provided inspiration for our system. Prevoyant (Bae & Young, 2014) took preexisting narratives and reordered its events to incorporate flashbacks and foreshadowing. Kruizinga's work (2007) analyzed several story planners that can be incorporated into character agents and a modified partial-order planner.

This is a brief review of the literature, as the page limitations do not permit a full discussion of the literature. The interested reader is referred to (Hollister, 2016) for a more in-depth discussion of the state of the art.

Our work sought to exploit the best parts of recent narrative generation systems while avoiding their pitfalls. AESOP, like IPOCL, can use context graphs to make a story plan, guaranteeing an interesting event sequence while also allowing characters to pursue their own goals. Character actions are informed by their internal traits as in CAMPFIRE and their relationships as in MEXICA. However, AESOP avoids the computational cost of IPOCL and its inability to let characters fail. It also guarantees story completion, unlike MEXICA, and does not rely upon large amounts of preauthored material as does CAMPFIRE. Our driving innovation stems from the hypothesis that well-crafted narratives are driven by characters who change and transform probabilistically in relation to other characters and an interesting overarching plot. In the following section, we describe how AESOP implements character growth in interesting stories.

Approach

The roots of our system rely on four fundamental structures: Characters, Conditions, Actions, and Objects. These modules were created and chosen to give us the greatest ability to pursue unpredictable stories with foreseeable character growth and believable event progression. Through these modules, we gain a great deal of flexibility without causing issues of randomness or inconsistency. There are other important modules, such as Location, which dictates where in the story world a module is located, or Destiny, which defines certain Actions as having or lacking actors and controlling how often those Actions happen. However, for the sake of brevity and focus, only the modules most essential to our program's functionality will be described here.

Characters, the most complex module (see Figure 1), represent agents in the story space; these are the individuals that move the story forward through their deeds. The most significant components of this module are Attributes, Relationships, and Goals. Attributes describe variables and details about the character, such as the character's strength, intelligence, or charisma. Each Attribute has a name to show what it represents and a value from 0 to 100. For example, an Attribute with the name "Strength" and a value of 80

would show that a hypothetical agent labeled Character A is very physically strong. Relationships describe the character in relation to other characters. Each Relationship, like each Attribute, has a value and a name; however, Relationships also have valences to represent negative feelings, as well as a direction to show which character "receives" these feelings. A Relationship with a name "Like," a value of 50, a negative valence, and a direction of "Character B" would show that Character A dislikes, but does not hate, Character B. Goals are a list of Conditions that a character "wants" to happen. Actions that help these Goals succeed have their probability weighted slightly. Further explanation of Conditions follows in the paragraph below. What makes our system unique is that these individual details can change throughout the story based on what Actions are chosen – more details on how this works are explained later on in the Approach section. With these details, we are able to model a Character as a pseudo intelligent actor that changes and grows throughout the story.

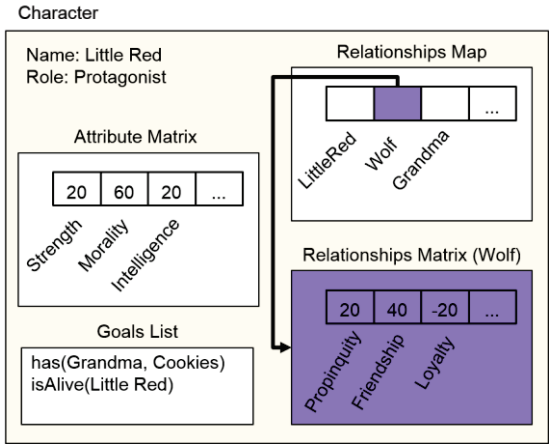

Figure 1. *The Character module.*

Conditions are statements that describe a state in the story space, whether those states are true or not. In common planning terminology, this has an equivalent with logical clauses. A Condition consists of a Name, Parameters, a Status (true or false), and a String. The Name of a Condition describes what it represents – for example, if a Character is holding an Object, the Name of the Condition would likely be "isHolding" or something similar. The Parameters are the arguments that specify who or what is involved in that Condition. To reuse our previous example, Character A and Object B could be Parameters to "isHolding". A Status is, in simplest terms, a Boolean value to represent whether or not a Condition is currently valid in the story world. The Condition would have a Status of "true" as long as Character A holds Object B, but this would be set to false if an Action causes Character A to drop Object B, if Character C steals Object B, or some similar Action were to take place. If a Condition is absent from a given Condition list, it is assumed to have a negative Status. Further explanation of Actions follows in the paragraph below. Finally, a String is a representation of that Condition written in plain English, such as "Character A holds Object B." Through Conditions, we can keep track of what our story world currently is.

Actions are changes in the story world, usually incorporating Characters – they are equivalent to planning actions in terms of common digital storytelling vocabulary. Actions contain Parameters, Pre-conditions, Post-conditions, a Name, a Probability, and a String. The Name, String, and Parameters are similar to how they appear in Conditions; a Name describes what the Action represents, the String is that Action and its Parameters in plain English, and Parameters are details of the Action passed as arguments. Probability refers to the likelihood of an Action being chosen at some point during the story – this numeric value is constantly recalculated over the course of the story for every Action. These Parameters include not only any Characters or Objects involved in these actions, but also their intricacies– a Character's Attributes, for instance. Parameters will often affect the likelihood of an Action taking place, as an Action with the String "Character A lifts Object B" will be more likely to occur the closer the value of Character A's Strength Attribute is to 100. Pre-conditions are Conditions that must be in the story world in order for the Action to happen, and Post-conditions are Conditions that will exist in the story world after the Action has taken place. This module allows the program to model progression of the story and keeps track of changes in its world.

Finally, there are the Objects, the simplest module. Typed constants are the equivalent of this module digital storytelling terminology. The key difference between Objects and Characters is that Objects lack agency, having no Attributes, Relationships, or Goals. Objects merely have Names, describing what they are, and Parameters, which are Conditions describing how the agents can interact with the object. For example, an Object with the Name "pizza" may have the Parameter "edible," which would allow it to be used in the "eat" Action. This module is included for completeness, as Characters in a story should be able to interact with story objects besides other characters.

All Actions, Conditions, Characters, and Objects are stored in AESOP's knowledge base, a collection of files containing everything the program "knows". Having a diverse knowledge base is tantamount to creating interesting stories; the number of possible stories increases with each newly added item to the knowledge base.

To understand our approach, one must understand the Expanded Story Space Model. Also known as the "Rotten Egg" model, it describes how we conceived the creation, selection, and placement of Actions as the story progresses. This model is based on the work of Jantke and Knauf (2005). It is shown in Figure 2.

In this model, Actions exist in either the Story Space, the Possibility Space, the Current Space, or the Fabula. The Story Space contains every possible Action in the story. If there is any chance that an Action could appear in the story at any point, it is contained in the Story Space. The Possibility Space contains every Action that is possible, given the current story Conditions. If an Action's Preconditions align with the Conditions of the story thus far, that Action is in the Possibility Space. The Current Space contains the Action that has been chosen for the current part of the story. If an Action ends up here, it is then added to the Fabula, a list of all Actions so far. These distinctions exist to best demonstrate how story Actions are selected as the story progresses.

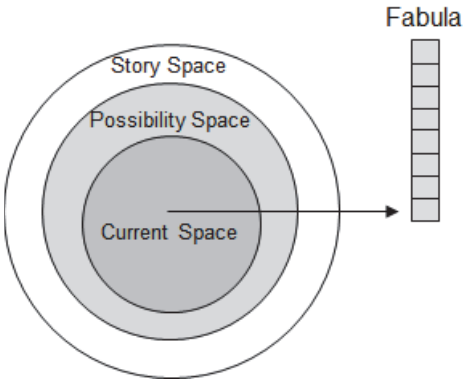

Figure 2. *The Expanded Story Space Model.*

With these central modules, we can describe the processes of our story program in much finer, clearer detail (see Figure 3). To trace through how our program produces Actions piece by piece in a story, we show three conceivable example Actions in AESOP's knowledge base involving Little Red Riding Hood and how these Actions are pruned throughout the progress of a hypothetical story generation. In addition, the flow of information will be demonstrated using the Expanded Story Space Model. Our three actions are as follows: a) Little Red eats an apple; b) Little Red attacks a wolf; and c) Little Red picks up an apple.

First, AESOP stores and remembers its Characters, Objects, and any starting Conditions that describe what the story world is currently like. Then, it builds the Story Space – it selects every known Action and permutes it with every relevant Character and Object, ensuring that each Action has produced different versions with each possible Character and/or Object as its Parameters. These permuted Actions make up the Story Space, and they are saved so that they need not be recreated every iteration. We cannot see an Action such as "Little Red flies" because it is not in our system's knowledge base of possible Actions, only in our imaginations as observers. Therefore, the Action "Little Red flies" cannot be created, but all other example Actions can be found in the Story Space.

Second, now that AESOP knows every possible Action in its Story Space, it looks through those Actions' Preconditions and XNORs them with its list of the world's Conditions to prune down to its Possibility Space. Our Action "Little Red eats an Apple" has the Precondition "Little Red is holding an apple." Our story has just begun, and this Condition was not listed in AESOP's starting Conditions, so this Action is pruned, and it is not found in the Possibility Space.

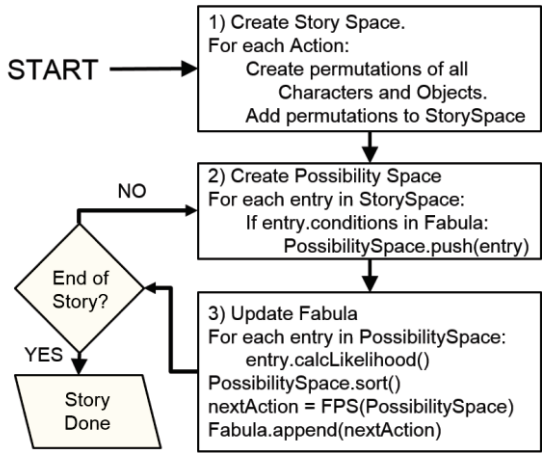

Figure 3. *Story generation through the AESOP Algorithm.*

Third, AESOP calculates probabilities of each Action happening in order to create its Fabula. We are left with two Actions, and while "Little Red attacks a wolf" is possible, the Parameters of that Action dictate that it is more likely to happen when Little Red's Strength Attribute has a value of 100. Little Red, being a weaker Character, only has a Strength value of 20, so this Action has its relative probability diminished. Other factors increase or decrease Action probabilities as well – for instance, if Little Red had a Goal of "Little Red and a wolf are fighting," the aforementioned Action would have its probability boosted. Finally, Fitness Proportionate Selection (FPS), where probability is analogous to fitness, is used to stochastically select the next action. Let us say that this pseudorandom function selects "Little Red fights the wolf" as the next Action, despite it having a lower relative probability. This Action is added to the Fabula, and no other Action is selected in this iteration.

Once an Action is selected, AESOP looks at its Post-conditions, updating its own Conditions list. What makes the AESOP unique is the Characters themselves change based on chosen Actions. In this example, Little Red will have a greatly boosted Strength Attribute; also, the Wolf and Little Red will have mutually diminished Friendship Relationships. With these alterations, AESOP restarts at step two of its algorithm, recreating the Possibility Space.

After some limit has been reached – such as a maximum number of Actions in a story or a certain character's Goal is met, the program ends, no longer selecting new actions. This

process facilitates the production of stochastic stories that follow character qualities, combining the best elements of unpredictability and logical consistency.

Test Plan

To test the effectiveness of AESOP'S modules, we generated stories, randomized their order, and distributed them amongst human test subjects for anonymous ranking. We define three fictional characters in the stories: Harriette, Gary, and a Witch, whose interactions form the basis of the test stories. These stories were generated under the same knowledge base (i.e., set of facts about the world). For example, in each story, Characters could choose to move between three locations - the 'RedHouse', 'WitchHouse', and the 'Forest'. This is important to consider, as Characters can only interact with each other if they are in the same location. Additionally, Characters were initialized with certain Attributes and Relationships, which affect the Characters' probabilities of performing an action; see the table below. In the interest of preserving space, the Relationship tables are omitted; however, the stories were generally such that Gary and Harriette were less likely to get along with the Witch.

	Harriette	Gary	Witch
Intelligence	25	60	5
Strength	75	80	85
Morality	25	25	0

Figure 4. *The test Characters' Attributes.*

Each story was generated under certain rulesets (i.e., selected story generation modules were toggled). A set of 40 stories were generated and given to the reviewers to rate for their coherence, consistency and desirability. Seven test subjects served as reviewers, all of whom are associated with the project. As such, the results cannot be considered statistically significant due to small sample size and the absence of test subject objectivity. However, a relative comparison of the average ranking of each category of test story generation ruleset used can provide us with valuable insights into the effectiveness of the different modules used.

A story earned a rank of 0 if the reviewer found it inconsistent with itself and illogical. This rank was meant to represent directionless stories which most children would not find enjoyable. A rank of 1 represented a story which the reviewer found coherent and consistent, signifying that the reader believed some children would like this story. Finally, a rank of 2 was earned if the reviewer believed the story was coherent and consistent, and had a definable beginning and end. This rank also meant the reviewer believed that most children would enjoy this story. Every story, except for those in the Context ruleset, terminated after 20 Actions. Below are the rulesets we used for our testing.

Random: (5 test cases) Under this ruleset, the sections of code that filtered impossible and improbable episodes were disabled; stories are entirely random. We hypothesize that these stories will rank the lowest, as they are expected to be nonsensical. These test cases act as the control group.

Planning: (5 test cases) This ruleset represents the most basic operation of the AESOP algorithm. Impossible episodes are filtered out, and Actions which fall more in line with Character capabilities are more likely to be chosen.

Character Relationships: (5 test cases) Relationship matrices were altered to observe the effects on the story. In this story set, the Witch is in love with Harriette.

Without Character Input: (10 test cases) This ruleset is similar to the Random ruleset; however, impossible episodes are eliminated from the selection pool. Character Attributes have no effect on which Actions are selected.

Goals: (5 test cases) Each character has a Goal that they "want" to accomplish, and Actions that create these Conditions will be more likely to happen. In these stories, Gary wants to fight Harriette, the Witch wants to be in love with Gary, and Harriette wants to kill the Witch.

Context: (10 test cases) Stories generated under this ruleset set are identical to those in the Planning ruleset except with the context module activated. The context module acts as a system of author goals, ensuring that certain Actions occur in a specific order during the course of a story; however, the specific Actions which are actually selected are generated using the AESOP algorithm. Each context-controlled story is governed by a linear Contextual Graph (CxG), or storyboard, created a priori. Each storyboard can be imagined as a series of steps where progression to the next context occurs when specific events have happened. Unlike other stories, these stories terminate when the storyboard is complete. In the Revenge storyboard (5 test cases), first Harriette agrees to marry the witch, then Gary kills the witch, and finally Harriette kills Gary. In the Change Of Heart storyboard (5 test cases), Harriette and Gary first fight, then become friends, and finally join forces to kill the Witch.

Results

The test story rated highest by the reviewers reads as follows: "Witch reads Book. Witch meets Harriette. Harriette antagonizes Witch. Harriette becomes an enemy of Witch. Witch seduces Harriette. Harriette becomes a friend of Witch. Harriette asks for Witch's hand in marriage. Harriette eats Cookies. Witch agrees to marry Harriette. Harriette seduces Witch. Harriette befriends Witch. Harriette attacks Witch. Harriette kills Witch. Gary moves to Forest from RedHouse. Gary moves to WitchHouse from Forest. Harriette meets Gary. Gary becomes a friend of Harriette. Harriette becomes a friend of Gary. Gary antagonizes Harriette. Gary becomes an enemy of Harriette. Gary attacks Harriette. Gary attacks Harriette. Harriette attacks Gary."

On the other extreme, the lowest rated test story reads as follows: "Harriette eats Cookies. Witch meets Harriette. Harriette seduces Witch. Witch antagonizes Harriette. Harriette antagonizes Witch. Witch attacks Harriette. Witch befriends Harriette. Harriette falls in love with Witch. Witch falls out of love with Harriette. Harriette attacks Witch. Harriette falls out of love with Witch. Harriette kills Witch. Gary moves to Forest from RedHouse. Gary moves to WitchHouse from Forest. Harriette meets Gary. Harriette antagonizes Gary. Harriette attacks Gary. Gary attacks Harriette. Gary kills Harriette."

The averages of the reviewers' opinions are shown below.

Figure 5. *Average rankings for test sets by reviewers.*

Clearly, the two sets of stories with the Context modules performed the best; the Change-of-Heart storyboard scored an average of 1.14, while the Revenge storyboard scored nearly 1.4. Though we would have liked to see higher average ratings, the results point to the value of contextual inclusion. Other observations derived from the results:

- Random stories ranked lowest: average rating of 0.08.
- Planning stories, the next highest group, ranked nearly three times higher with 0.23, although still low.
- Stories with and without character relations and with goals all performed nearly identically.

Conclusions and Future Work

Narrative is the embodiment of human experience, capturing how people interact with others and react to the world; we developed AESOP to explore the possibilities of this medium. AESOP's stories are based on the notion that interesting stories are driven by characters and how they change. By allowing stories to be guided with both character and author goals, we can generate stories with real direction. Furthermore, by altering our characters' statistics throughout the story, we can create story agents that can grow and develop.

However, AESOP is limited: it only generates a fabula, a skeleton which fails to prune out insignificant story events and excludes crucial details that bridge story events. Consequently, several of AESOP's stories involved excessive character movement between locations before they could interact with the other characters; in addition, many story moments were interrupted by less interesting story events, such as eating cookies after being asked to marry. Furthermore, the small number of story events in AESOP's knowledge base caused stories to be repetitive. With the constraints and shortcomings of our system, we leave to future research the development of a "syuzhet module" that can flesh out the story from a given fabula as generated by AESOP.

Many possible extensions to AESOP exist. One extension is allowing the user to influence the story's direction as it is being generated, as seen in Hollister's work. The user could undo events and change Character variables to see where the story will go under different rules. Additionally, we hope to expand the library of possible events and more fully incorporate other modules in progress such as automatic acquisition of story content, flashbacks, foreshadowing, and the ability of Characters to deceive one another.

AESOP is unfinished, and as such, not yet publicly available. It is far from perfect, but it has proven to be a framework capable of generating a fabula through directed probability, where characters change and act unpredictably in pursuit of established character and author level goals. Though it is currently a limited prototype, AESOP has the capacity to be a truly modular and intelligent storyteller.

References

Bae, B. C., & Young, R. M. (2014). A computational model of narrative generation for surprise arousal. IEEE Transactions on Computational Intelligence and AI in Games, 6(2), 131–143.

Hollister, J. (2016). A Contextual Approach to Real Time, Interactive Narrative Generation. University of Central Florida.

Jantke, K.P. & Knauf, R. (2005). Didactic design through storyboarding: Standard concepts for standard tools. Computer Science Press, Trinity College Dublin, Ireland.

Kruizinga, E. E. (2007). Planning for Character Agents in Automated Storytelling.

Mateas, M. &, & Sengers, P. (1999). Narrative Intelligence. AAAI Fall Symposium on Narrative Intelligence, 1–10.

Ontañón, S., & Zhu, J. (2010). Story and Text Generation through Computational Analogy in the Riu System. In Proceedings of the Sixth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE-10) (pp. 51–56).

Pérez y Pérez, R. (2007). Employing emotions to drive plot generation in a computer-based storyteller. Cognitive Systems Research, 8(2), 89–109.

Riedl, M. O., & Young, R. M. (2010). Narrative Planning: Balancing Plot and Character. Journal of Artificial Intelligence Research, 39, 1–49.

Turner, S. (1992). MINSTREL: a computer model of creativity and storytelling. Ph.D. Thesis, Computer Science Department, University of California, Los Angeles. Technical Report CSD-920057.