RoleModelVis: A Visualization of Logical Story Models

Sherol Chen, Andrew Duensing, Peter Kong, Arnav Jhala, Noah Wardrip-Fruin, Michael Mateas

Expressive Intelligence Studio, University of California at Santa Cruz

sherol@soe.ucsc.edu, aduensin@ucsc.edu, pkong@ucsc.edu, jhala@soe.ucsc.edu, nwf@soe.ucsc.edu, michaelm@soe.ucsc.edu

Abstract

In this demo we present a visualization of formalized representations of story. Introducing the interactive to storytelling requires the management of experiences that a user creates by their decisions. These sorts of variations can have impact on not only the user, but also the retrievable content appropriate to present to the user. The overall contribution of this work is to identify the player impact of story variation by modeling supplementary variations, and systematically responding to player interaction through supplementary variation, while respecting the author's intentions by maintaining the integrity of the core story skeleton.

Introduction

Where do we find variations in story? Often, in games like *Mass Effect* or *Planescape Torment*, when the user is required to give a verbal response to another character, the game will, hopefully, respond appropriately to the selected statement. When one decision changes any aspect of the experience, there had to have been a variation that was designed into the experience. Decisions can also be on the game side, such as the probabilistically distributed enemies that you find in a dungeon of a *Final Fantasy* game or the drama managed response from Grace of Trip in *Façade*.

Variation, however, occurs even in the absence of intentionally interactive experiences as narratologists and psychologists have realized. Unavoidably, there will always exist variation at, at least, 2 points of a storytelling: the delivery of the author/teller and the reception by the audience. More intentionally, there are explicit variations that are designed to direct the flow of a story or guide the experience of the user.

From the subtle to the explicit, how can we determine whether or not one variation is more meaningful than another? Another way to present this question is: how do we know whether a particular variation is sufficiently meaningful and whether or not it is worth the effort to design?

RoleModel: Story Formalization

RoleModel is a novel story generator organized around explicit formal models of character roles. RoleModel expands the expressiveness of stories generated from arbitrary partial domain specification by using a formal model of roles within an abductive logic programming framework. Authorial goals in the system can be fully or partially specified as constraints in an abductive logic program. In particular, the RoleModel system focuses on representing and satisfying role constraints of the story characters.

Character roles and archetypes play an important part in storytelling by providing motivations for character actions and introducing clearly recognizable dramatic interactions. Expert storytellers exploit character roles and role changing situations to manipulate user's beliefs and expectations to bring about dramatic conflicts and resolutions. For example, in Kurosawa's *Rashomon*, several re-tellings of a dramatic situation are presented to the viewer. In each narration, from a different character's point-of-view, roles of participating actors (e.g. Victim, Aggressor, Moderator) are manipulated to create coherent variations of the situation. Specific roles provide affordances for characters to undertake particular types of actions within the story. For example, in *Rashomon,* the woman's role of being either the aggressor or the victim provides the author with an option to create interesting variations on the aggressive episodes within the story. For intelligent storytelling systems, a rich formal model of roles enables authors to partially specify the domain and character constraints without sacrificing consistency of character behaviors with respect to their roles.

RoleModel is a story generator that explicitly models roles to generate meaningful variations of story situations. Due to the complexity involved in authoring complete and consistent formal domains that generate an authorially desired story space, we investigate the use of abductive logic programming to create models of possible story variations from a partially specified domain. Such a system provides authors with the ability to explore the space of

possible variations given varying levels of story constraints.

Character properties include roles, traits, dynamic attributes, and sentiments towards actions, while action properties include a variety of contextual properties and causal constraints. The relationships among these constraints provide the background theory for the solver to use. For our prototype, generation involves asking the system to satisfy a list of additional story constraints (including no constraints) on top of the background theory. The system produces a collection of grounded predicates (an answer set), where each collection corresponds to a concrete story that satisfies the constraints given the background theory. In the prototype, actions are represented using the event calculus, supporting temporal inferences about actions.

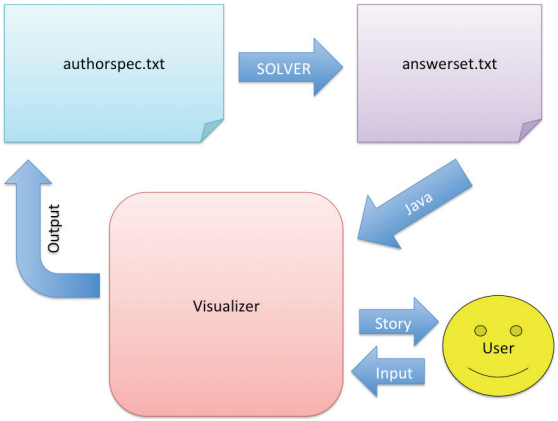
Given characters, role constraints, and goals, the RoleModel system aims to elaborate upon the initial story assertions (or problem constraints) to establish or amplify character roles through addition of preconditions on actions that fulfill character role constraints. In changing the role assignments, the system manipulates background knowledge and elaborates upon the story without breaking the initial story conditions as specified by the author. While many story generators emphasize the means by which they generate many different action sequences while maintaining causal consistency between actions, we were interested in how dynamic role assignment, and the implications that follow from roles, can be used to reframe similar action sequences to have very different meanings.

Story Visualization of Answer Sets

The output of RoleModel is in a logic programming format. For readability and user interactions, we've built a visualization tool in Java to translate the answer sets into graphical scenes, called vignettes. As the user changes the visually represented properties, a new *story specification* is created and sent to the solver. The Java program then revisualizes the returned story. This system is called RoleModelVis. The code below is represented in the screenshot on the right.

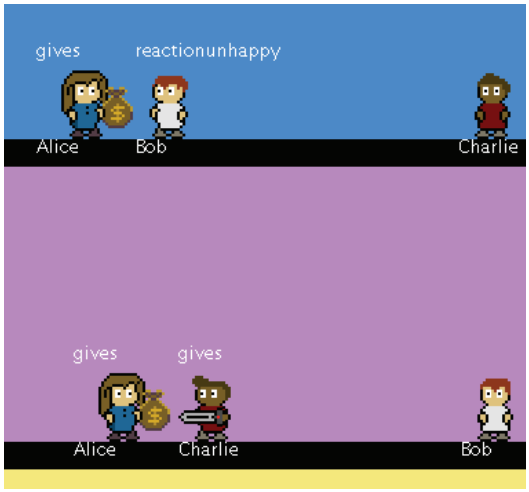
```
person(alice).
person(bob).
person(charlie).

scene(1,expectation(unwanted_gift,alice,bob)).
scene(2,expectation(premeditated_attack,alice,bob)).
scene(3,expectation(death,bob)).

vignette(1,action(gives,alice,bob,money)).
vignette(1,reaction(unhappy,bob)).

vignette(2,action(gives,alice,charlie,money)).
vignette(2,action(gives,charlie,alice,sword)).

vignette(3,action(kills,alice,bob)).
```

The interactions is represented in the figure below:



An author or story specification is first passed into the Clingo Answer Set Solver which returns a logical representation of a valid story. The answer set is then parsed through the RoleModelVis Java program and the rulesets are represented graphically in vignettes that correspond to the timeslots that were generated. The user can explore the current story and make adjustments that get reproduced into another specification file, which is then passed through the solver to return a new story with the user specified adjustments.

In the current version, only minor edits can be made to the story through the visualizer, RoleModelVis. In the future version, we hope to include the editing of the specification file as part of the visual interaction of the user. Below is an instance of RoleModelVis interpreting the example story answer set.



References

S. Chen, A. M. Smith, A. Jhala, N. Wardrip-Fruin, and M. Mateas, RoleModel: towards a formal model of dramatic roles for story generation, Intelligent Narrative Technologies III Workshop. 2010.