Crowdsourcing Interactive Fiction Games

Boyang Li, Stephen Lee-Urban, and Mark O. Riedl
School of Interactive Computing, Georgia Institute of Technology
Atlanta, Georgia, USA
{boyangli; lee-urban; riedl}@gatech.edu

ABSTRACT

Procedural generation of games has become an active research field. We present a system that automatically generates an interactive fiction (IF) by learning from crowdsourced corpora of example stories. We ask crowd workers from Amazon Mechanical Turk to write short stories about a given situation with simple language, from which a plot graph is learned, containing plot events, temporal precedence and mutual exclusion relations between the events. The plot graph describes an IF where players and non-player characters choose from executable events as determined by the plot graph. We demonstrate an IF learned from the domain of bank robbery.

Categories and Subject Descriptors

I.2.1 [**Artificial Intelligence**]: Applications and Expert Systems—*Games*

General Terms

Algorithms, Human Factors

Keywords

Procedurally generated games, interactive fiction, crowdsourcing

1. INTRODUCTION

Recent years have seen a growing interest in computer-generated games. Game generators can help simplify technically challenging aspects of game creation and improve replayability by dynamically adapting game content. Recent work explored generation of mechanics and rule sets (e.g., [6]) and content generation within story contexts (e.g., [1]).

One game genre especially suitable for automated generation is *Interactive Fiction* (IF): a type of game with a text interface, a strong narrative focus, and abundant opportunities to change the story [2]. The fiction unfolds as a combination of the designer's intent and the choices the player makes throughout. The core mechanic in IF is a *lock-and-key* progression in which certain scenes and actions only become possible after other scenes and actions have occurred. The lock-and-key mechanism can be modeled by a *plot graph* in which plot events—significant scenes that can occur in the game—are related by temporal precedence constraints [8]. For example, a plot event of the player opening a vault must be preceded by the event of acquiring the key [5]. A plot graph captures the designer's intended logical flow of events that can happen; it reflects the designer's beliefs about acceptable gameplay experiences.

This extended abstract describes the SCHEHERAZADE system, which generates playable interactive fiction experiences based on common sociocultural situations such as going to a restaurant, going to the movies, catching an airplane, or robbing a bank. SCHEHERAZADE addresses the critical research challenge of obtaining domain knowledge about the topic for which the game is to be constructed, including the plot events, temporal precedence and mutual exclusions between events. Without assuming a pre-existing ontology, we delegate this complex knowledge authoring task to a large number of anonymous workers via Web services, i.e. we crowdsource the plot graph. Crowdsourcing provides access to human memories and creativity; a surrogate for the lifetime of experiences held by a human author. Previously, Shaker *et al.* [7] crowdsourced aesthetics for Super Mario levels.

2. PLOT GRAPH LEARNING

To create an interactive fiction of a particular situation (e.g., a bank robbery), an automated query is sent to Amazon Mechanical Turk to solicit crowd worker input. Crowd workers are asked to provide linear archetypical narratives in natural language for the given topic, which is a natural mode of communication. To simplify natural language processing, crowd workers are asked to describe one event with one sentence containing a single verb, and to avoid complex sentence structures and pronouns.

Second, sentences from different narrative examples that are semantically similar are clustered together to create plot events. Because of the simplified language, the system can discover clusters that represent plot events with relatively high accuracy. For example, we identify plot events for the bank robbery scenario with 80.4% purity. We can also use a second round of crowdsourcing to ask crowd workers to improve the accuracy of NLP / clustering to 89.8%.

Third, we identify precedence between learned plot events. For all pairs of events e_i and e_j, we test the two statistical

FDG '13 Crete, Greece

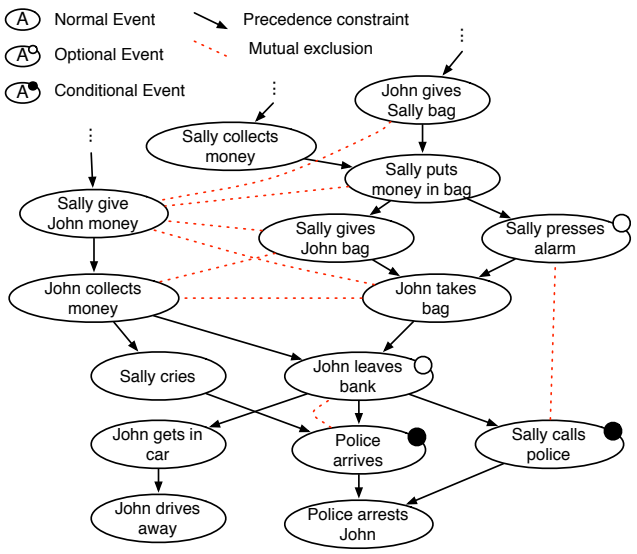

Figure 1: Part of a plot graph learned from the crowd.

hypothesis $before(e_i, e_j)$ and $before(e_j, e_i)$ based on the binomial distribution. The hypothesis with confidence greater than threshold T_c are accepted and added to the model. Both hypotheses may be rejected and the two events will be considered as parallel. Relations that fall slightly below the threshold may be recognized if adding the relation to the graph reduces an error metric, computed as the difference of the distance between two events on the graph and their average distance in the input data set. This effectively lowers the threshold locally and provides a flexible mechanism that caters to noisy input narratives.

Fourth, we identify mutual exclusion relations between plot events. As a generalization of OR-relations in plot graphs, mutual exclusions indicate when only one of two events can happen in the same game. Mutual exclusions occur when the *mutual information* between two plot events is high—indicating that one plot event predicts the existence (or non-existence) of the other—and co-occurrence is low. When a mutual exclusion relation is found between two plot events e_i, e_j that are temporally ordered, probably with other events in between, then e_i is marked as *optional* and e_j is *conditional* on e_i. For example, pressing the silent alarm precludes calling the police later in the game.Figure 1 shows part of a learned plot graph capturing several variations for a bank robbery.

3. PLOT GRAPH EXECUTION

A set of execution rules, used together with the plot graph, ensures that players always experience a coherent progression through the game, regardless of their choices. The execution rules determine what events are available for execution at any given point in time. A plot event is executable when all of its direct, non-optional predecessors have been executed, except those excluded by mutual exclusion. When an event is executable, its owner, either the player or a non-player character (NPC) has to decide whether to execute it. Race conditions can occur if both the player or NPCs have executable events; we implement a short delay on NPC

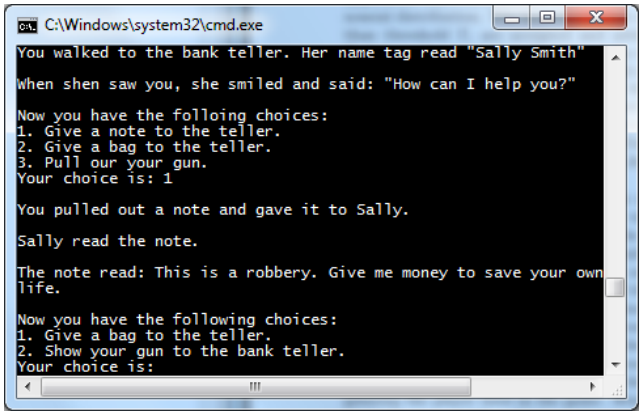

Figure 2: Playing the generated interactive fiction.

choices to give the player slightly more control over the story. Once an event is executed, SCHEHERAZADE removes the following events from the graph: events in mutual exclusion with executed events, events all of whose parents are excluded, and events whose descendants are executed. The game continues until one ending event is executed.

The entire plot graph learning pipeline and plot graph execution algorithms are fully implemented. We have demonstrated the system on a plot graph of a bank robbery situation. One manual step remains necessary: a dramatized textual description must be given for each plot event for display purposes. The game interface is shown in Figure 2. Due to space constraints, we refer readers interested in more details to earlier papers [3, 4].

4. ACKNOWLEDGMENTS

We gratefully acknowledge the support of the U.S. Defense Advanced Research Projects Agency (DARPA).

5. REFERENCES

[1] K. Hartsook, A. Zook, S. Das, and M. Riedl. Toward supporting storytellers with procedurally generated game worlds. In *Proc. of IEEE CIG*, 2011.

[2] K. Hayles and N. Montfort. Interactive fiction. In *The Routledge Companion to Experimental Literature*. Routledge, 2012.

[3] B. Li, S. Lee-Urban, D. S. Appling, and M. O. Riedl. Crowdsourcing narrative intelligence. *Advances in Cognitive Systems*, 2:25–42, 2012.

[4] B. Li, S. Lee-Urban, and M. O. Riedl. Toward autonomous crowd-powered creation of interactive narratives. In *Proc. of the INT5 Workshop*, 2012.

[5] M. Nelson and M. Mateas. Search-based drama management in the interactive fiction Anchorhead. In *Proc. of AIIDE*, 2005.

[6] M. Nelson and M. Mateas. An interactive game-design assistant. In *Proc. of Int'l Conf. on Intelligent User Interfaces*, 2008.

[7] N. Shaker, G. N. Yannakakis, and J. Togelius. Crowd-sourcing the aesthetics of platform games. *IEEE Trans. on Computational Intelligence and AI in Games*, 2012.

[8] P. Weyhrauch. *Guiding Interactive Fiction*. PhD thesis, Carnegie Mellon University, 1997.