Data Adventures

Gabriella A. B. Barros
Center for Computer Games Research
IT University of Copenhagen
Copenhagen, Denmark
gbar@itu.dk

Antonios Liapis
Institute of Digital Games
University of Malta
Msida, Malta
antonios.liapis@um.edu.mt

Julian Togelius
Department of Computer Science and Engineering
New York University
New York, USA
julian@togelius.com

ABSTRACT

This paper outlines a system for generating adventure games based on open data, and describes a sketch of the system implementation at its current state. The adventure game genre has been popular for a long time and differs significantly in design priorities from game genres which are commonly addressed in PCG research. In order to create believable and engaging content, we use data from DBpedia to generate the game's non-playable characters locations and plot, and OpenStreetMaps to create the game's levels.

1. INTRODUCTION

Much work in procedural content generation (PCG) is currently focused on generating content such as levels, maps and items for mechanics-heavy games, such as action games and puzzle games. Similarly, attempts to generate complete games have focused on simple arcade games or board games, with nothing in the way of story [12].

In a sense, most of PCG and game generation focuses on "silent" content, reflecting a formalist mechanics-first approach to games. At the same time, research in interactive narrative focuses on generating stories first, where gameplay is contingent on the stories. In this paper, we seek inspiration in an existing and successful narrative-heavy game genre, with strong genre conventions, and propose generating complete games in this genre following the structural constraints given by these genre conventions.

We look at the genre of *adventure games*, as exemplified by games such as *Maniac Mansion* (Lucasfilm Games 1987), *Day of the Tentacle* (Lucas Arts 1993) and in particular *Where in the world is Carmen Sandiego?* (Brøderbund Software 1985), as a challenging novel domain for game generation methods. This poses new challenges for content generation and game generation. In particular, the generated game needs to be meaningful in terms of its textual description as well as in terms of game mechanics. To take Carmen Sandiego as an example, the game content to a large extent consists of descriptions of existing cities and places within cities. Simply generating random names and descriptions is likely to lead to an underwhelming experience.

In order to generate such content, we turn to the practice of *data games*, i.e. generating game content based on open data [6]. The basic idea is that the real world already contains almost everything we need to know in order to (automatically) create adventure games; design is mostly a matter of selecting and ordering data so as to fit with genre conventions, and perhaps add a few extraneous elements. Hence, in order to generate an adventure game, we make use of open data to provide both the structure and the associations between game elements. Networks of information, as found in Wikipedia, can provide with the necessary links between dissimilar objects (people, locations, objects, topics). Such links can be used to create a rudimentary plot, which the player needs to discover. Adding more depth to these discovered links, images and maps from online databases can be used to enhance both gameplay and aesthetic appeal of the game. The first steps to implementing such a game are described in this paper.

1.1 Game generation and content generation

Procedural generation of game content is a common feature in several game genres and an active research field [10]. Content generation can be more or less ambitious in scope; the proposed project seeks to generate essentially the whole adventure game, while simply keeping some core mechanics fixed. Many different efforts have been made to generate complete games, using methods that are solver-based, constructive or search-based [12].

Search-based game generation uses optimization algorithms (e.g. evolutionary computation) to search a space for feasible games. *Ludi* generates board games through evolutionary computation, evaluating the games via a weighted sum of many heuristics [2]. A number of attempts have been made to generate similar simple mechanics-focused games [13, 4, 9]. It has even been argued that this type of games — single-player, short mechanics-heavy games with easily quantifiable properties — are ideal for generation as they can be evaluated easily through simulated playthroughs [12].

Instead, the proposed project aims to generate games which cannot easily be evaluated based on whether an algorithm playing them wins or loses: losing the game is not possible or not of much consequence. Instead, the game is focused on progression through exploration. We assume a very limited basic vocabulary of mechanics, with most of the options for action being implicitly defined by the content. This opens up a set of new challenges for game and content generation, most importantly what to base the content on so that it becomes meaningful and interesting to play with.

1.2 Data games

The term “data games” describes games which use real-world open data to generate in-game content. In such games, data can be explored during gameplay, which allows the emergence of several ways to learn and visualize information [6, ?]. In most cases, data from the real world must be transformed in order to be useful as game content. Transformation typically involves both structural transformation, where data format is changed to work with the game, and data selection, where parts or aspects of a dataset that is useful for generating game content are selected.

Examples of data games include Bar Chart Ball [11], where the player moves a ball atop a bar chart by choosing different demographic indicators which change the appearance of the chart. Other projects include Open Data Monopoly, which uses real-world demographic information to create a *Monopoly* (Parker Brothers 1935) board game [5], and Open Trumps, which evolves sets of cards for *Top Trumps* (Dubreq 1977) based on countries' data [3]. Similarly, data from OpenStreetMaps and resource maps have been used to create balanced *Civilization* (MicroProse 1991) maps [1].

1.3 Adventure games

Adventure games were extremely popular during the 1990s, with classic games such as *The Secret of Monkey Island* (LucasArts 1990) or *Day of the Tentacle.* Such games were hailed for their witty humor, their amusing dialogue and difficult puzzles. Despite a decrease in commercial interest during the 2000s, adventure games have resurfaced both as commercially viable products and as popular entertainment outlets. Contemporary adventure games such as *Tales of Monkey Island* (Telltale Games 2009) or *Broken Age* (Double Fine 2014) use episodic releases of the adventure's story, thus retaining player interest for longer periods of time.

While modern adventure games may be visually very different than their counterparts in the 1990s, certain common patterns of play and design connect all adventure games together. Adventure games often revolve around solving a mystery or overcoming a challenge through cunning and unexpected associations between pieces of information or unlikely objects. The core challenge is the discovery of such associations, which may require that players perform specific sequences of actions, combine objects in their inventory to create new objects, or pay close attention to the hints provided in dialogue with non-player characters (NPCs). Challenges in most adventure games are local: a player must solve a puzzle in order for the story to move forward and the next puzzle be presented. Some adventure games can include losing conditions, where a player needs to load or restart from a previous game state, although more often players can always revisit all previous states of the game and search for missing clues. Due to the carefully written end-game and plot progression, most adventure games are played once and have little replay value.

In this project, we are particularly inspired by *Where in the world is Carmen Sandiego?*, a series of popular educational adventure games by Brøderbund Software. The first game in the series was released in 1985. In this game, the player has to find the eponymous Carmen Sandiego by traveling around the world and collecting clues as to Sandiego's appearance and whereabouts. The destinations visited are real institutions and monuments in real cities, though (most) NPCs in the game are fictional. The use of real world places and phenomena suggests that the basic game structure could form the basis for a type of data game.

2. PROPOSED GAME DESIGN

Traditionally, creating an adventure game is a laborious process where clever dialogue and interesting storylines are carefully authored by one or more writers. Adventures are often driven by their story; therefore, plot, NPCs and dialogues are carefully tuned to provide a rich user experience. In contrast, using open data to drive the system of an adventure game, we come across the challenge of designing a game that needs to interpret and understand raw input, and can work in less controlled (or controllable) instances than traditional adventure games.

With this in mind, our goal is to create an adventure game focused mainly on discovery of visual and verbal clues and pieces of NPC dialogue. Fundamentally, the player's goal is to find the location of a certain person, starting from the location of another person or clue. To do so, players can travel between different countries and cities, and explore buildings in each city. Entering buildings within a city, the player can interact with NPCs and objects found there in order to get clues which can unlock new locations, NPCs or dialogue options. Some objects or NPCs might give false clues, and NPCs can give no clues at all. A clue can either be a line of dialogue or an object with some piece of information (e.g. a letter portraying a NPC that the player needs to talk to).

Similar to many adventure games, it is not possible to lose the game, and there are no action sequences or other timing-dependent mechanics. The game is won by finding the target person. While there might be some animation signaling e.g. player movement, most of the game interface is based on structured text, menu choices and static images.

3. IMPLEMENTATION

The game is implemented in Java using libGDX[1], a framework for multi-platform game development. It currently uses DBpedia and OpenStreetMaps. The former provides data for generating the plot of the game, while the latter is used to render maps. Currently, the main mechanics are talking to NPCs and traveling from one location to another.

The game's plot is defined via one of the possible relations between two real people. With these people as arguments, a path between them is traced using Wikipedia data gathered from DBpedia, a community project aiming to extract structured information from Wikipedia[2]. For instance, if the initial person is Alan Turing and the target person is Nikola Tesla, one of the possible paths is shown in Figure 1. Algorithmically, the path is represented as a directed graph, where each person, location, category, etc. is a node and their relations are the edges.

Figure 1 could be read as: **Alan Turing** was influenced by **Ilya Prigogine**, who also influenced **Friedrich Hartogs**. Hartogs was a **mathematician who committed suicide**, as did **Ludwig Boltzmann**. Boltzmann was a **fellow of the American Mathematical Society**, as is

[1]http://libgdx.badlogicgames.com/

[2]http://dbpedia.org/

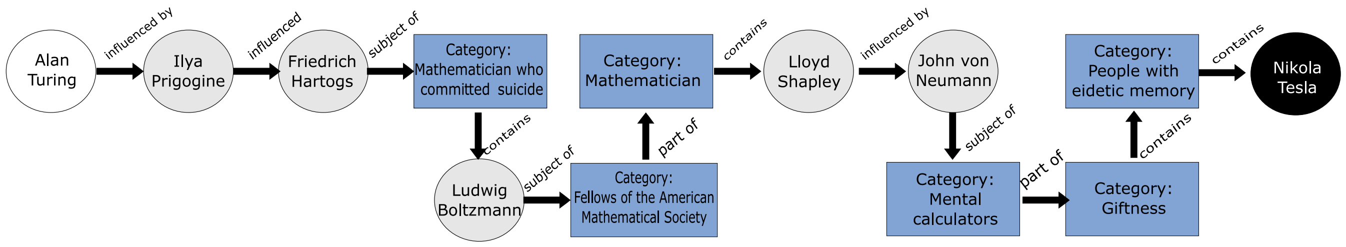


Figure 1: One of the possible paths between Alan Turing and Nikola Tesla, discovered via the DBpedia crawler.

origin ⟨P⟩ goal
origin ⟨P_1⟩ object ⟨P_2⟩ goal
origin ⟨P_1⟩ object1 ⟨P_2⟩ object2 ⟨P_3⟩ goal

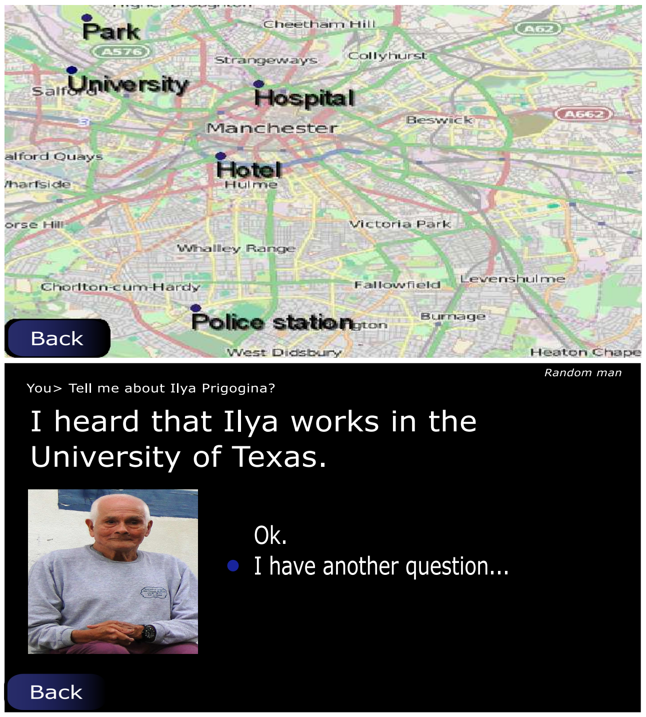
Figure 2: Possible relations discovered by the crawler.

```
queries ← all_queries_between(A,B);
paths ← search_dbpedia(queries);
sort paths by diversity;
path ← best in paths;
initialize final_path;
for $i \leftarrow 0$ to $path.size - 1$ do
    queries ← queries_between(path[i],path[i+1]);
    loc_paths ← search_dbpedia(queries)
    sort loc_paths by diversity
    loc_path ← best in loc_paths
    add loc_path to final_path;
end
return final_path
```
Algorithm 1: Crawler pseudo-code (A: origin, B: goal).

Lloyd Shapley, who was inspired by **John von Neumann**, a prodigy in many fields, like mathematics, language and **memorization**. Both von Neumann and **Nikola Tesla** were **gifted**, as Tesla had **eidetic memory**.

A crawler has been implemented to search for possible relations between two given subjects: the *origin* and the *goal*. It is heavily inspired by RelFinder [8], a visualization and exploration tool for web-semantic data. Our search is done by querying the DBpedia SPARQL endpoint. Several different queries are generated between the two main subjects, in order to find links connecting both of them. Figure 2 shows possible relations as responses to the queries, where P_i is the predicate that joins two subjects (e.g. "subject", "reside in" or "spouse"). Afterwards, one of the resulting paths is chosen and a new search is made for each pair of relations in this path. Algorithm 1 shows the pseudocode for this.

Selecting the best path is made according to the *uniqueness* of the path. There are multiple ways to relate two different subjects; for instance, two people may have both lived in the Austrian Empire, but so did millions of others. These two people may also have more specific things in common: a mutual acquaintance, an institution they both went to or a similar field of research. These less obvious choices are less general and can possibly create a more interesting plot, and intuitively better clues for the player. To measure this, we calculate the frequency of each predicate and node in all possible paths. A path is considered 'more' unique than another if its edges and nodes appear fewer times in



I heard that Ilya works in the University of Texas.

Ok.
I have another question...

Figure 3: Screenshots from the game. **Top:** the map of Manchester rendered with OpenStreetMaps. Buildings are positioned at random locations. **Bottom:** A part of a dialogue with a random NPC.

the search. For example, suppose a search finds 50 paths, where an edge tagged with "influenced by" appears 80 times (it can appear multiple times in a single path), another with "residence" shows up 20 times, and "work institutions" 15. If path A consists of edges "influenced by" and "residence" and path B contains edges "residence" and "work institutions", path B is 'more' unique than A, because its edges appeared less in the search results. Path length is also taken in consideration: longer paths result in more clues which need to be discovered and longer gameplay in general. The total number of edges and nodes in the path is added (with appropriate weighting) to the path's uniqueness score.

Each node found in the final path is expanded and transformed into a NPC, a location or a clue. The current version of the system does not contain objects, so all clues come in the form of dialogue lines. NPCs and locations are generated in a more straightforward manner than clues. People are instantiated as NPCs, and their personal characteristics (e.g.

age, race, hair color, etc) are included in the object. Locations are created as either cities or buildings within them — states are treated as cities to simplify the process.

Clues, on the other hand, are anything that can not be transformed into a NPC or a location (e.g. categories). For each clue, a small piece of text from its DBpedia resource page is gathered and added as a dialogue line in the NPCs' dialogue tree.

In the current state of the project, the game renders images in two manners. For NPCs and buildings, it uses a handcrafted database of faces and buildings to provide visual information. It is not identifying real images from people (e.g. it does not use a photo from Alan Turing for his NPC), nor does it use a real image of the building itself.

Maps, however, are rendered using JMapViewer, a Java Swing component that renders OpenStreetMap (OSM) maps data[3]. OSM is a community-based world mapping project [7]. The image is exported as a texture and stored for in-game use: an image of the world map is rendered and saved first, then one zoomed map for each city in the game. The coordinates of each city are acquired from DBpedia. Screenshots of a city map and a dialogue in the game are shown in Figure 3.

4. OUTLOOK

In this paper, we propose the use of open data to generate content for an adventure game. We use data from Wikipedia, obtained through DBpedia, to create the game's plot, NPCs and locations, and render the game's maps using OpenStreetMaps. This project is very much a work in progress. A next step is to obtain the images of people and buildings automatically (see Figure 4), possibly via APIs such as those provided by Google. Improving the plot generation is also imperative. At the moment, the story is linear, but adventure games usually require some sort of branching. We can achieve this with false clues, with dead-end links from DBpedia, with side-quests (e.g. with another target person) or with multiple paths towards the main plot's target person. We also intend to add objects to the game: at first only readable objects (e.g. books, letters), and eventually also objects that can provide different interactions with the environment (e.g. keys that unlock certain rooms, flashlights that allow to search in a dark place), or puzzles.

Acknowledgment

Gabriella Barros acknowledges financial support from CAPES and Science Without Borders program, BEX 1372713-3.

5. REFERENCES

[1] G. A. B. Barros and J. Togelius. Balanced civilization map generation based on open data. In *Proceedings of the 2015 IEEE Congress on Evolutionary Computation*, 2015.

[2] C. Browne. *Automatic generation and evaluation of recombination games.* PhD thesis, Queensland University of Technology, 2008.

[3] A. B. Cardona, A. W. Hansen, J. Togelius, and M. G. Friberger. Open trumps, a data game. In *Proceedings of the International Conference on the Foundations of Digital Games*, 2014.

[4] M. Cook and S. Colton. Multi-faceted evolution of simple arcade games. In *Proceedings of the IEEE Conference on Computational Intelligence and Games*, 2011.

[3]http://wiki.openstreetmap.org/wiki/JMapViewer

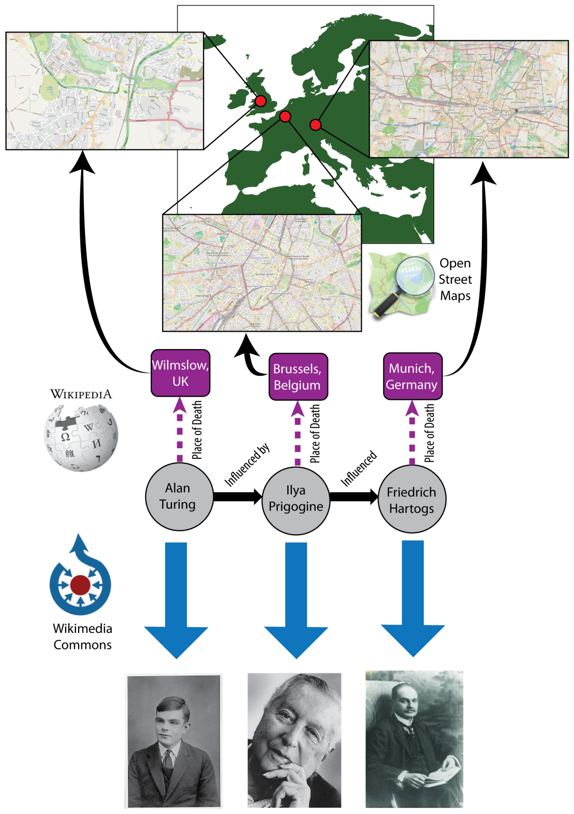


Figure 4: Finding data from different data sources to create maps and characters in the game from a subpath of Fig. 1.

[5] M. G. Friberger and J. Togelius. Generating interesting monopoly boards from open data. In *Proceedings of the IEEE Conference on Computational Intelligence and Games*, 2012.

[6] M. G. Friberger, J. Togelius, A. B. Cardona, M. Ermacora, A. Mousten, M. Møller Jensen, V.-A. Tanase, and U. Brøndsted. Data games. In *Proceedings of the International Conference on the Foundations of Digital Games*, 2013.

[7] M. Haklay and P. Weber. Openstreetmap: User-generated street maps. *Pervasive Computing*, 7(4):12–18, 2008.

[8] P. Heim, S. Lohmann, and T. Stegemann. Interactive relationship discovery via the semantic web. In *Proceedings of the 7th Extended Semantic Web Conference*, volume 6088 of *LNCS*, pages 303–317. Springer, 2010.

[9] T. S. Nielsen, G. A. B. Barros, J. Togelius, and M. J. Nelson. General video game evaluation using relative algorithm performance profiles. In *Applications of Evolutionary Computation*, pages 369–380. Springer, 2015.

[10] N. Shaker, J. Togelius, and M. J. Nelson. *Procedural Content Generation in Games: A Textbook and an Overview of Current Research.* Springer, 2015.

[11] J. Togelius and M. G. Friberger. Bar chart ball, a data game. In *Proceedings of the International Conference on the Foundations of Digital Games*, 2013.

[12] J. Togelius, M. J. Nelson, and A. Liapis. Characteristics of generatable games. In *Proceedings of the 5th Workshop on Procedural Content Generation in Games*, 2014.

[13] J. Togelius and J. Schmidhuber. An experiment in automatic game design. In *Proceedings of the Symposium on Computational Intelligence and Games*, pages 111–118. IEEE, 2008.