Playing with Data: Procedural Generation of Adventures from Open Data

Gabriella A. B. Barros[1], Antonios Liapis[2], and Julian Togelius[3]
[1]Tandon School of Engineering, New York University, New York, USA,
gabriella.barros@nyu.com
[2]Institute of Digital Games, University of Malta, Msida, Malta,
antonios.liapis@um.edu.mt
[3]Tandon School of Engineering, New York University, New York, USA,
julian@togelius.com

ABSTRACT

This paper investigates how to generate simple adventure games using open data. We present a system that creates a plot for the player to follow based on associations between Wikipedia articles which link two given topics (in this case people) together. The Wikipedia articles are transformed into game objects (locations, NPCs and items) via constructive algorithms that also rely on geographical information from OpenStreetMaps and visual content from Wikimedia Commons. The different game objects generated in this fashion are linked together via clues which point to one another, while additional false clues and dead ends are added to increase the exploration value of the final adventure game. This information is presented to the user via a set of game screens and images. Inspired by the *"Where in the World is Carmen Sandiego?"* adventure game, the end result is a generator of chains of followable clues.

Keywords

Data games, procedural content generation, adventure games, open data, Wikipedia.

INTRODUCTION & BACKGROUND

Adventure games, in their several variations and incarnations, are perennially popular. By adventure games, we refer to games where the player guides a protagonist through a quest with narrative elements, where the gameplay includes both exploration and puzzle-solving (Adams and Rollings 2006). Typically, the protagonist moves around in a physical space and solves problems by talking to non-playable characters, collecting items and using them in the right place, and gathering information about the game world which is used to progress the narrative. These games are typically single-player, but otherwise vary greatly in graphical style, input mode and playtime. Like most game genres, the boundaries of this genre are not clear-cut (Karhulahti 2011), but famous examples of adventure games include classic text adventures such as the *Colossal Cave Adventure* (Crowther 1977), point-and-click adventures like *Maniac Mansion* (LucasFilms Games 1987) and *The Whispered World* (Telltale, 2010), action adventures such as *Tomb Raider* (Core Design 1996) or *Metroid* (Nintendo 1986), and interactive drama adventures like *Heavy Rain* (Quantic Dream 2010) and *The Walking Dead* (Telltale, 2012).

Game design and development in general is a lengthy and labor-intensive process requiring deep expertise, and the situation is no different for adventure games. And just as for other

Proceedings of 1st International Joint Conference of DiGRA and FDG

types of games, there would be much to gain from automating all or part of the adventure game design and development process: simplifying the game development process, allowing people with fewer resources to create games, creating infinite games, creating player-adaptive games, and so on. In the past few years, a research community has addressed this problem cluster. Much of this work is referred to as procedural content generation (PCG), the decades-old practice of generating some parts of games algorithmically which has recently received new impetus from the introduction of new methods such as search-based (Togelius et al. 2011) and solver-based PCG (Smith and Mateas 2011). There are now several good methods for generating, for example, levels for platform games or maps for strategy games. Some forms of PCG attempt to generate complete games (Togelius et al. 2014), including the game rules themselves; this has been done successfully for board games (Browne and Maire 2010) and so far limited success for arcade games (Cook et al. 2012; Lim and Harrell 2014; Togelius and Schmidhuber 2008; Nielsen et al. 2015).

In this paper, we consider the problem of generating adventure games — or perhaps more properly, adventures for adventure games. Compared to arcade games, the nature of the content that needs to be generated is somewhat different for adventure games. Many adventure games rely on places (to explore), people (to interact with), social relationships (to explore and/or exploit) and items (to move or use) for their fundamental content. This type of content is frequently both textual and graphical in nature, e.g. a person needs both an image and textual description or dialogue. Building a system capable of creating believable and interesting non-player characters, items and places from scratch would be a tall task; one can easily compare with the many generators that have tried this, such as the world generator in *Dwarf Fortress* (Adams 2006), where the formulaic nature of names and characteristics quickly becomes apparent.

Several systems exist for creating stories in existing games. Examples include systems focused on authored stories are *Haunt 2* (Magerko 2007), *PaSSAGE* (Thue et al. 2008) and the Prefix-based Collaborative Filtering (PBCF) (Yu and Riedl 2012). There is also some work on creating puzzles for adventure games such as the Puzzle-Dice system for narrative puzzle generation (Fernández-Vara and Thomson 2012). However, these systems generally address the problem of creating new arrangements of and relations between existing objects, characters and locations rather than creating new artifacts of these types.

Luckily, we do not need to generate everything from scratch. Many adventure games feature locations and persons taken from (or heavily inspired by) the real world. An adventure game generator could do the same thing: build on real world places and characters. Information about places, persons and their relations is available online on an unprecedented scale, so the question becomes one of selecting accessible and appropriate data and integrating it into the content generation algorithm. This connects to work in data games (Friberger and Togelius 2012a), which use real-world information for creating their content. Data is incorporated into game content under the assumption that players should view, learn and interact with such data during gameplay (Friberger et al. 2013; Cardona et al. 2014). However, data in its raw form is usually not suitable for direct use in-game, and need to be transformed. Transformation of data into content appropriate for gameplay includes *data selection* (i.e. selecting which parts of the data are useful to content generation) and *structural transformation* (i.e. adapting the data in accordance to the game). Published work in data games often

revolves around direct conversions of existing games such as *Monopoly* (Parker Brothers 1935) boards (Friberger and Togelius 2012b), *Top Trumps* (Dubreq 1977) decks (Cardona et al. 2014) or *Civilization* (MicroProse 1991) maps (Barros and Togelius 2015) to include real-world data. A number of game-like activities based on the open encyclopedia Wikipedia have also been developed, for example the "Wiki Game", "5 Clicks to Jesus" etc.[1]

By developing ways to automatically (or pseudo-automatically) generate adventure games from open data, we hope to be able to facilitate the process of designing and developing adventure games, but also to make such games more personalizable (a player might be more interested in playing a game involving characters or places they have a connection to) and enhance replay value, as the games could be made to be effectively infinite. The process of designing such games, which is user-directed, could also be seen as a novel way of playful data exploration and visualization (and could possibly provide a form of edutainment). On a more ambitious note, we are interested in exploring the aesthetic induced by data-generated adventure games; the first examples of data adventure games shown in this paper, coupled with (Cook and Colton 2014), can be the basis for a genre of "living", contextual games which are playable e.g. only concurrently with the news or data that create them, or only by individuals with a deep knowledge of the domain (e.g. science, politics) where the data originates from. More generally, the adventure game generator which forms the end-goal of the work presented here would constitute an exemplar of computational game creativity (Liapis et al. 2014), as the computer must find ways of exploring the vast volumes of data, find which data can be combined together and finally transform the game's rules and aesthetics appropriately to the data at hand: this draws direct parallels to the exploratory, combinatorial, and transformational capacities of computational creativity (Boden 1992).

In the current paper, we describe a mechanism for generating a particular kind of adventure game content: non-player character groups suitable for the search for a missing person, or "manhunt". We are specifically inspired by the classic adventure game *"Where in the World is Carmen Sandiego?"* (Brøderbund Software, 1985) which focuses on finding the titular game character by traveling the world, talking to non-player characters and collecting clues. The system described in this paper is a functional "horizontal slice" of an adventure game focused on finding a missing person through the simple mechanics of traveling to places, talking to non-player characters and consulting items. Every time the game is started, the player can decide on the characters used as starting and finishing points, and a complete game is generated by selecting characters and places from the real world and constructing items. All the information is drawn from Wikipedia, Wikimedia commons and OpenStreetMaps.

The paper expands threefold on earlier preliminary work (described in Barros et al. (2015)), it provides an in-depth analysis of the processes for generating game objects and their visuals, describes the current state of item generation and how game flow is created through clue set-up, and finally presents an experiment which quantitatively evaluates part of the system. The robustness and expressiveness of the generator are tested by using a fairly broad corpus of data as seeds and observing the generated adventures.

1. The listed games and more are found at `http://thewkigame.com`.

GAME DESIGN

Adventure games are a popular genre of games where players interact with virtual environments in order to solve conflicts in a given context (Cavallari et al. 1992). Adventure games vary greatly, as they have been in circulation for over 30 years, yet a trait they share is a strong story-driven experience. It is thus necessary to focus on how the story is conveyed, either by dialogues, scenarios, visuals or any other form of expression. In this project, the story will be represented mostly by dialogues and description texts, in a similar fashion to classical adventure titles.

The game is defined by a pair of Wikipedia articles about two people, which become initial and final NPCs. Using these, other articles that lead from one to another can be found, and can refer to people, locations, categories, etc. Each article would act as a plot point, and in this sense the story would be the full path connected by plot points. The player's goal is to find the location of the final NPC, at which point the game ends. To do so, the player must travel between different cities and countries in search for clues. These clues can either be a piece of dialogue or an object (e.g. a note with a list of NPCs) that will direct the player towards the next clue. Since the experience is about freely exploring data, there is no losing condition.

The game draws inspiration for its visuals (as well as for other aspects) from the original *"Where in the World is Carmen Sandiego?"*, an adventure game released in 1985. In that game, the player is a detective searching for criminals around the globe. The player uses a simple interface to travel between different cities, issue search warrants used to interrogate people and learn clues, which appear as puns in the dialogue. The gameplay consists of clicking buttons, thus making Carmen Sandiego's mechanics simple compared to other adventure games. For instance, instead of having to walk to a door to leave one place and then move to the next, the player only needs to click on the map to choose the next location. Our game is built similarly: the interface contains dialogue screens, menu options and static images. While it is planned for future work, at the moment there is no inventory or puzzles, which are common traits of adventure games.

Our game starts at the world map screen, where the player can travel between cities. Once in a city, the player can travel to different buildings, such as hospitals, schools or houses. In a building, it is possible to view information about the NPCs and objects therein, as well as to select an NPC to talk to, which leads to a dialogue screen. A more elaborate example of the game progress and the interface is provided in the Results Section.

IMPLEMENTATION

In order to create a game from open data, several steps are necessary: the acquisition of data, the selection of appropriate data, the transformation of raw data into playable content and the presentation of content to the player. In this project, the adventure game is generated via the following steps:

1. Collect data to establish a plot defined from the links between two people.
2. Generate NPCs, objects and locations from nodes in the plot, transforming data into game objects.

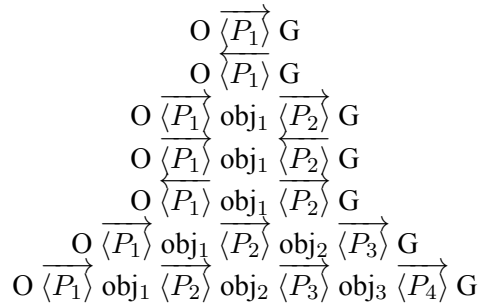


Figure 1: Examples of possible relations discovered by the crawler. The difference between the predicate direction (e.g. "Albert Einstein has a notable student Leó Szilárd" versus "Leó Szilárd is a notable student of Albert Einstein") is indicated by the arrow above the predicate.

3. Generate clues and set up preconditions and effects for creating a game progression.

Plot Mining

In the current version of the generator, the game's plot is constructed starting from two real people (with their own Wikipedia articles). The steps connecting one person to another are obtained from Wikipedia via the DBpedia endpoint. DBpedia is a project which extracts information from Wikipedia in a structured manner[2]. In this context, an endpoint is a service in the DBpedia server which allows communication between our system and DBpedia's database. Possible links between these individuals are obtained via a crawler heavily inspired by the *RelFinder* tool (Heim et al. 2010), a visualization and exploration tool for the web-semantic data.

Given two people, an *origin* O and a *goal* G, the crawler creates multiple queries and passes them through the DBpedia endpoint. The queries seek different relations between the goal and the origin, as shown in Fig. 1: P$_i$ are predicates between two subjects (e.g. "reside in" or "affiliation"). The objective is to find different configurations of connections. The results are represented as directed paths, which are then evaluated; the best one is used to flesh out the game's plot. An example path between Albert Einstein and Margaret Thatcher, found by the crawler, is shown in Fig. 2.

As the path is a direct representation of the story, we want to prioritize larger paths, thus providing a longer experience. To do so, we need to use a brute-force approach. First, we need to query DBpedia for the paths. We create every possible combination of path queries, up until a certain threshold, which represents the maximum length of the longest possible path. Each query represents one relation, as shown in Figure 1. For example, for 2 given nodes O and G, one query would search for relations that fit $O\overrightarrow{\langle P_1 \rangle}G$, while another would search for $O\overleftarrow{\langle P_1 \rangle}G$. Given a path with n nodes in the path and $n-1$ edges, the possible combinations of relations amount to 2^{n-1}. However, we do not want to exclude the chance of selecting a smaller, more interesting path. Therefore, the number $q(n)$ of queries for n nodes is $\sum_{i=1}^{n-1} 2^i$.

2. http://dbpedia.org/

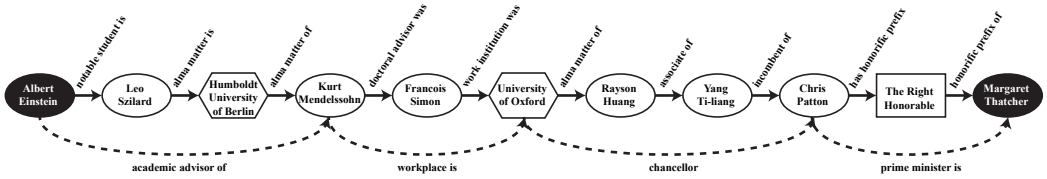


Figure 2: One of the possible paths between Albert Einstein and Margaret Thatcher. Dotted arrows represent major paths, which are connected indirectly with minor paths (black arrows). People are placed in a circle, locations in a hexagon, and categories in a square.

Nodes	2	3	4	5	6	7	8
Time	0.08	0.26	0.5	1	140	469	249
Queries	2	4	6	8	10	12	14

Table 1: Average amount of time, in seconds, taken to search all possible queries with n nodes from Alan Turing to Nikola Tesla.

Due to limitations in the DBpedia endpoint, it is not practical to make very large queries. Table 1 shows the time required for querying all combinations of nodes for paths of size between 1 and 7 nodes between Alan Turing and Nikola Tesla (chosen arbitrarily). To avoid large queries, we separated the crawling process into two steps. Firstly, we search for a path p_{major} between O and G with up to 5 nodes. We choose this value because it does not take as long to compute but still gives us reasonable paths that can be extended in the next step. Secondly, for each pair of nodes in p_{major}, we search another path p_{minor}, size 4, which will be incorporated into the main path (see Fig. 3). Path sizes were chosen empirically.

Evaluating paths discovered by the crawler is not straightforward: two desirable properties are *length* and *uniqueness*. The paths returned by the crawler are used to flesh out the game's plot, to instantiate the NPCs, locations and clues and ultimately to determine the game's progression. Favoring longer paths leads to more NPCs and locations, and makes for a longer, arguably more challenging game session.

Moreover, the links between game elements (objects, NPCs) should be somewhat obscure — to challenge the player in discovering these links — but also specific enough to avoid guesswork. Therefore, discovered paths are evaluated on their uniqueness, both for nodes and links. Uniqueness of links is evaluated based on how often they appear in all discovered paths between origin and goal; links that are common in most paths are less favored. As an example, in paths between Alan Turing and Nikola Tesla, many paths include links such as "influenced by" and "influenced" (scientists influencing each other), while far fewer paths include links such as "thesisYear" (scientists who handed in their PhD thesis on the same year); thus links such as "thesisYear" are more 'unique' and therefore would provide a more specific, if obscure, clue. Similarly, uniqueness of nodes is evaluated on how often specific nodes appear in all discovered paths; for instance, nodes such as "Austrian Empire" are less unique than "Mathematicians who committed suicide" (evidently, many individuals lived or died in the Austrian empire). Given a node i and a path p, let X^i be the amount of times

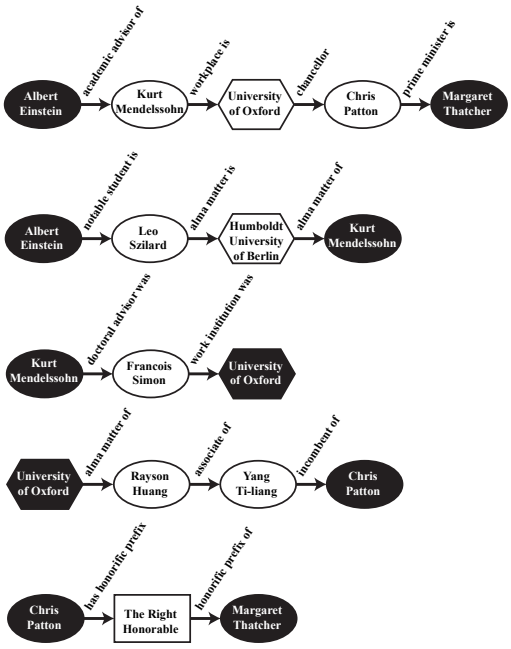


Figure 3: The process of expanding a major path of depth 5 (top) connecting Albert Einstein with Margaret Thatcher with minor paths. Each node in the major path becomes a start- and end-point for the crawler, which returns the 4 minor paths (at the bottom). These are combined to create the plot of Fig. 2.

that node appear in all paths, and x_p^i the amount of times that node appears within path p. The uniqueness value u of a path p is calculated as shown in Equation 1, where l_p is the length of p (the number of nodes in the path) and n is the total amount of nodes.

$$u(p) = \frac{1}{l_p} \sum_{i=1}^{n} \left(\frac{x_p^i}{X^i} \right) \tag{1}$$

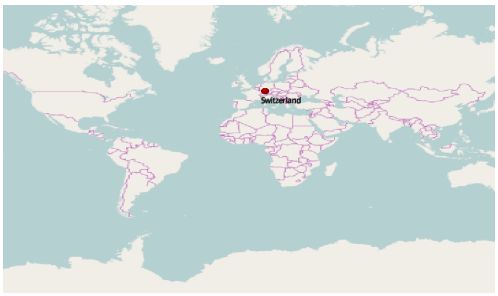
Calculation of uniqueness of links is done similarly, only changing the variables from nodes to links. The uniqueness of paths and the uniqueness of links are normalized and added to the path length (normalized to the maximum possible length) in order to evaluate the path. The highest scoring path is used to flesh out the components of the game, i.e. its locations, NPCs and objects.

Location generation

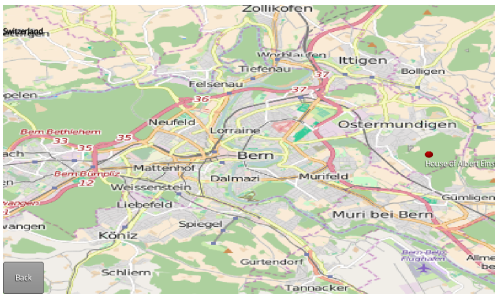
Similar to *"Where in the World is Carmen Sandiego?"*, the game revolves around travelling the globe in order to discover clues. Therefore, both NPCs and objects in the generated adventure game need to be placed in locations. Locations in the game are cities and buildings within those cities.

Locations are generated from nodes in the path discovered by the crawler. Each node may have coordinate information (i.e. latitude and longitude) or may be linked to data with coor-

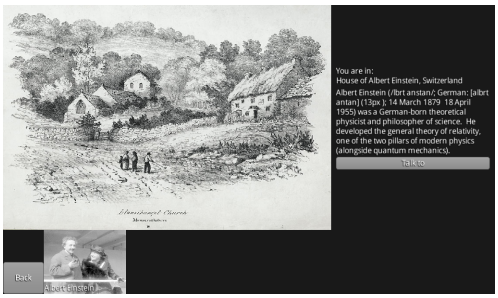
(a) Snippet of the DBpedia entry of Einstein, showing Switzerland under 'citizenship'.



(b) World screen when Einstein is added as a start NPC: Switzerland is placed at the right coordinates in the world map.



(c) City screen of Switzerland "city": the city of Bern from OpenStreetMaps is used as visual backdrop, and Einstein's house is placed as a building in it.



(d) Building screen for Einstein's house: Albert Einstein (with an appropriate portrait) can be interacted with via the icon at the bottom of the screen.

Figure 4: The process of generating locations (world locations, city locations, and building descriptions) for the Albert Einstein NPC of the adventure in Fig. 2.

dinate information, for instance via a "placeOfBirth" link (see Fig. 4(a)). If such coordinate information is a city or state (based on the DBpedia ontology), it instantiates a city location. The city is placed on the world map via its DBpedia coordinates (see Fig. 4(b)), and a map of the city itself is created via JMapViewer, a Java Swing component that renders OpenStreetMaps[3] (OSM). The map is centered at the DBpedia coordinates given, exported as an image and stored in the game for re-use (see Fig. 4(c)). If the node does not contain coordinate information or its coordinates are not a city, state or country, the node creates an in-game building, which is a generalized location (e.g. a plaza, a university, a hospital or a house). Buildings are placed in existing cities; NPCs and objects generated from the node in question are placed in their respective locations (if an NPC has a city, they are placed in a random building within that city). Images for buildings (see Fig. 4(d)) are obtained using Spritely[4](Cook and Colton 2014), a tool that automatically searches within Wikimedia Commons for images, ripping and compressing them – in this case, it searches for the location's name or type (e.g. "university" or "house").

3. http://wiki.openstreetmap.org/wiki/JMapViewer
4. http://www.gamesbyangelina.org/2012/12/spritely/

NPC & item generation

Transforming nodes in the path discovered by the crawler into NPCs or items is similar to the process for location generation. If the node is of type "Person" (based on the DBpedia ontology), an NPC is generated in-game and some of its stats (e.g. name, date of birth) are instantiated from its DBpedia entry. NPCs are placed in new buildings within cities which match their residency information, place of birth or place of death. In the current version of the generator, an image of the real person is obtained using Spritely (see Fig. 4(d) for an image of the NPC instantiated from Albert Einstein). If no image is found, a picture of a random man or woman is used.

Nodes which are not of type "Person", e.g. categories such as "The Right Honourable" in Fig. 2, are transformed into items. Items in the current version of the generator are limited to books. Pieces of information linked to the node (e.g. "subject of" or "abstract" in DBpedia) are added as readable content for the book. Generated items are placed in new buildings within existing cities, chosen randomly.

Setting up clues and creating the story flow

Once all game objects (NPCs, locations and items) have been generated, it is necessary to guarantee that they are presented in a sequential order, according to the rudimentary plot discovered via DBpedia. For each node in the plot's path, a clue and a condition are added: if the node is a person, the system instantiates a template dialogue tree for that NPC, adapting it to point to the game object of the next clue. If the node is a location, a new NPC (with random characteristics and picture) is placed in that location and a template dialogue tree is added. If the clue is neither of the above, the clue is added to the text within the book item. For instance, if the book represents a category, such as "Mathematicians in the 20th century", the book text could be something like: "Mathematicians in the 20th century: ...", followed by a list of names found in that DBpedia entry. Finally, a condition is added so that the game object of the next node on the path can only be accessed after the current node's clue has been seen, either in the NPC dialogue or in the book.

In addition to clues which point to (and unlock) the next node in the plot, the clue set-up can also provide false clues which confuse the player and enhance the exploration element common in adventure games. There is a 50% chance of giving a vague or false clue that can lead to a dead end. If the next node in the plot is an NPC, the current clue may point to a location that contains the right NPC but also other random ones who provide no useful information. If the next node in the plot is a building, the clue can be the name of the city that building is in, and new random buildings in that city are added with random NPCs (or no NPCs at all). However, if the next node is a city, no false clue can be given.

RESULTS

To test the acquisition and transformation algorithm, we used the list of 100 Most Important People of the 20th Century by Time Magazine (TIME 1998c, 1998b, 1999b, 1998a, 1999a). Albert Einstein, Franklin Roosevelt, Aretha Franklin and Anne Frank are examples of names in this list. The game was executed 851 times using randomly selected pairs of people in the list, with no repetition. However, the article sometimes considered multiple people as

Result	Number of runs
Runs with Errors	32
Runs with no Errors	819
Playable runs	827
Type of error	**Amount of errors**
No path found	10
Search error	9
Parsing error	12
Other errors	1

Table 2: Results of game generation runs.

one (e.g. The Beatles or the Kennedy Family). In such cases, we use only one individual of the group (e.g. John Lennon for The Beatles). This person was chosen based on the first reference in the group's Wikipedia article (e.g. The Beatles's article). The choice of people is not as important for this test's purposes, as long as they are reasonably "famous" and thus well-referenced by Wikipedia. One entry on the list, "American GI", was removed because we could not choose a single person to represent it. Therefore, pairs were built from a list with 99 people.

Table 2 shows the number of playable games: 96% of runs generate a complete adventure game without errors. Of the failed ones, 31% were due to no path found for the specified distance. This does not necessarily mean that there is no connection between them, but rather that no connection within 4 nodes can be found. Most errors (37%) occur during parsing (i.e. the transformation between data and game objects), which sometimes can be traced back to lack of information in DBpedia. For instance, creating an NPC depends on the data type "Person" in the DBpedia page; however, some pages (such as the one on Kurt Gödel) are incomplete and lead to parsing errors. Search errors include any other error not caused by the program itself during the search process, such as a bad gateway error caused by the DBpedia endpoint being under maintenance or offline. It should be noted that some errors are not catastrophic, leading to playable games as other queries are still able to find a plot.

Table 3 shows information about the objects created using data from the paths. An interesting finding is that most NPCs are based on real people from Wikipedia articles, which is desirable from a design perspective as more data finds its way into the game (compared to randomly generated, and thus inherently less interesting, NPCs). Overall, the results show that several locations must be visited, and NPCs interacted with, in order to discover the goal NPC. The amount of game content generated (buildings, items, NPCs) suggests that playing through such adventure games would be an involved, prolonged process; however, user tests with human players will validate these hypotheses in future work.

Examples of generated games

To better understand the results of the data-based adventure generation algorithm, this section provides two paths and some images from the final generated games of the previous experiment. The first path was calculated between Albert Einstein and Margaret Thatcher,

	Average (sd)
Number of cities	3.82 (1.39)
Number of buildings	7.86 (2.36)
Number of NPCs	5.33 (2.32)
Number of items	4.14 (2.36)
Ratio of real NPCs (over all NPCs)	74% (20%)
Average path length	8.34 (2.45)
Minimum path length	3
Maximum path length	13

Table 3: Top: Average amount and standard deviation of game objects generated. Bottom: Minimum, maximum and average length of paths selected by the crawler.

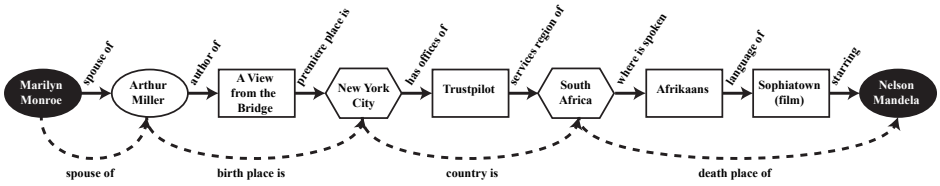


Figure 5: Path between Marilyn Monroe and Nelson Mandela.

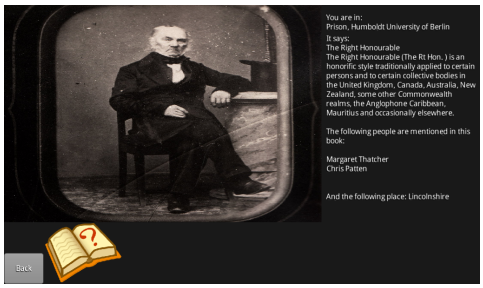
shown in Fig. 2. The game starts with a world map screen containing one city, "Switzerland", where the player can visit the house of Albert Einstein and talk to him (see Figure 6(a)). Talking to Einstein leads the player to a police station in Graham, where a book about Leó Szilárd can be found. This book cites the Humboldt University of Berlin, and takes the player to a small bar in Switzerland (Fig. 6(c)), where another book about this university is, with data about Kurt Mendelssohn — who can be met in his house in the United Kingdom. This continues until the player finally meets Margaret Thatcher, in Grantham (Fig. 6(d)). This playthrough included 4 cities, 8 buildings, 4 NPCs (all based on real people) and 4 items.

Another game was generated between Marilyn Monroe and Nelson Mandela. The path generated is shown in Fig. 5. The path contains misinformation as it states that New York City is part of South Africa. This can be traced back to the DBpedia resource page[5] for New York City, where several places exist in the field "dbp:country", including not only United States and South Africa, but also Israel, Italy and Egypt.

Note that the current version of the generator generates games that have only rather shallow dead ends; individual NPCs that are not tied to the story exist, but there are for example no subplots or hidden items or places. Essentially, the generated games can be played and won by simply visiting all places and talking to all NPCs. However, it is not possible to find the target NPC without having talked to the other NPCs first, meaning that any generated game has a given minimum play length.

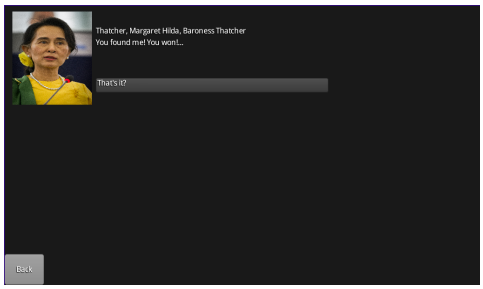
5. http://dbpedia.org/page/New_York_City

(a) Screenshot of a building (Albert Einstein's house), with an NPC (Albert Einstein). On the right, a description of Einstein and a button that leads to a dialogue screen.



(b) Screenshot of the book titled "The Right Honorable" (which covers the same category as that in Wikipedia), pointing to Lincolnshire, where players can find Thatcher.

(c) Another building (a bar), where there is a book about the Humboldt University of Berlin.



(d) The NPC dialogue screen at the end of the game, when the player finds Margaret Thatcher and wins the game. Unfortunately, the image of her NPC is clearly not Thatcher.

Figure 6: In-game screenshots of the adventure of Fig. 2 connecting Einstein with Thatcher.

DISCUSSION

The processes described in this paper choose data from vast repositories of open data (primarily Wikimedia projects) and transform them into game elements which can be interacted with and, after being experienced sequentially, lead to a goal state. The data which form paths between starting and goal NPC are dependent on the quality and breadth of information available in Wikipedia, and can lead to associations which are obscure (e.g. Margaret Thatcher and Chris Patton sharing the title "The Right Honorable" in Fig. 2) or unintuitive (e.g. New York City in Fig. 5 referring to a South African city rather than the obvious U.S.A. city of the same name). Moreover, when generating pictures for NPCs or places based on their names, the crawler may not find appropriate images (or in the case of Margaret Thatcher in Fig. 6(d), use a picture of Aung San Suu Kyi). This leads to a varying degree of absurdity in the results, and one could argue that the generative system has several issues that need to be fixed; the next Section highlights necessary changes to improve the output of the generator.

However, the very decision of using open data which are freely edited and updated daily by millions of users (in the case of Wikipedia) or susceptible to bias from latest user searches and newly appeared content (in the case of Google searches) comes with a degree of inherent

absurdity in the outcomes. While the unintended or catastrophic absurd outcomes should be hedged against through careful engineering, it is both unavoidable and desirable that a degree of absurdity in the resulting adventures remains. As with several games which rely on transformations of rhetoric (Treanor et al. 2012) or concepts (Cook and Colton 2014) into game mechanics or objects, absurdity is a desirable side-effect which evidences the data used to instantiate them — the player *should* be feeling "like [they are] playing a videogame against The Internet" (Cook and Colton 2014). Absurdity on the modern worldwide web is a reality (and largely an appealing quality for message boards such as `reddit`) and therefore can not be (and should not be) absent from the data adventures which transform them into playable experiences. As noted in (Cook and Colton 2014), information collected from open data "is often tainted by popular belief, misconception, stereotype and prejudice, as opposed to purely factual information". This is a strength (as they provide contextual associations based on current events or current interests of the userbase) and a weakness (as they can be volatile or offensive or absurd); regardless of whether it is more a strength or a weakness, this fact can not be cut off from data games. With this in mind, however, there are several directions for improving the playability and entertainment value of the adventures generated by our system, highlighted in Future Work.

FUTURE WORK

At its current state, the gameplay of the generated adventure games is still too restricted. This is mostly due to the lack of diversity and interactivity of game objects. Therefore, our next steps involve implementing different and more engaging mechanics, favoring objects which can interact with their surroundings, instead of serving only as information source. Generating more subplots and dead ends is also a priority.

Another concern is how the game still lacks a cohesive story. Although there is a series of relations between NPCs, places and items, this relation is not yet conveyed to the player in a clear manner. Furthermore, it is never explained why the player is searching for the goal NPC — with no motive, the player is simply running around searching for someone for no reason. We intend to explore variations in the game mechanics which allow for motives to surface — perhaps the player is searching for a murder suspect, or perhaps looking for a prospective mate?

Finally, it is necessary to expand on the experiment in this paper in order to evaluate how the generator performs with less recognized people, or with other types of data (e.g. places) as the plot's origin and goal. An important direction for future work is conducting user studies where human players interact with generated adventure games: their interaction data will shed light on the effect of different types of content (e.g. books) and test how the criteria for selecting plots (e.g. uniqueness) affect the challenge and usability of the game. User studies will be of utmost importance when the current prototype is enhanced with more involved game mechanics and a coherent narrative, in order to evaluate the effect of those additions to gameplay.

CONCLUSIONS

This paper presents a system which uses open data to generate content for an adventure game. A crawler was developed to discover a path which connects two real people, using

Wikipedia and DBpedia. Such a path can represent the story of the game, and undergoes a parsing and transformation process to generate the game's locations, NPCs and items. We have a working prototype where a player can travel around the world and interact with NPCs and items to obtain clues that lead her to the goal NPC. However, we still see much room for improving this prototype. Our next steps include adding new mechanics to enhance the complexity of its gameplay, and improve the storytelling system in-game.

Acknowledgments

Gabriella Barros acknowledges financial support from CAPES and Science Without Borders program, BEX 1372713-3. Antonios Liapis was supported, in part, by the Horizon 2020 project CrossCult (project no: 693150).

BIBLIOGRAPHY

Adams, Ernest, and Andrew Rollings. 2006. *Fundamentals of Game Design.* Prentice-Hall, Inc.

Barros, Gabriella A. B., Antonios Liapis, and Julian Togelius. 2015. "Data Adventures." In *Proceedings of the FDG Workshop on Procedural Content Generation.*

Barros, Gabriella Alves Bulhoes, and Julian Togelius. 2015. "Balanced Civilization map generation based on Open Data." In *Proceedings of the 2015 IEEE Congress on Evolutionary Computation.* IEEE.

Boden, Margaret. 1992. *The Creative Mind.* London: Abacus.

Browne, Cameron, and Frédéric Maire. 2010. "Evolutionary Game Design." *IEEE Transactions on Computational Intelligence and AI in Games* 2 (1): 1–16.

Cardona, Andrew Borg, Aske Walther Hansen, Julian Togelius, and Marie Gustafsson. 2014. "Open Trumps, a Data Game." In *Proceedings of the 9th Conference on Foundations of Digital Games.*

Cavallari, Beth, John Heldberg, and Barry Harper. 1992. "Adventure games in education: A review." *Australasian Journal of Educational Technology* 8 (2).

Cook, Michael, and Simon Colton. 2014. "A rogue dream: automatically generating meaningful content for games." In *Proceedings of the AIIDE workshop on Experimental AI in Games.*

Cook, Michael, Simon Colton, Azalea Raad, and Jeremy Gow. 2012. "Mechanic Miner: Reflection-Driven Game Mechanic Discovery and Level Design." In *Proceedings of Applications of Evolutionary Computation,* vol. 7835, LNCS, 284–293.

Fernández-Vara, Clara, and Alec Thomson. 2012. "Procedural generation of narrative puzzles in adventure games: The puzzle-dice system." In *Proceedings of the The third workshop on Procedural Content Generation in Games,* 12. ACM.

Friberger, Marie Gustafsson, and Julian Togelius. 2012a. "Generating game content from open data." In *Proceedings of the Foundations of Digital Games Conference,* 288–295. IEEE.

Friberger, Marie Gustafsson, and Julian Togelius. 2012b. “Generating interesting monopoly boards from open data.” In *Proceedings of the IEEE Conference on Computational Intelligence and Games,* 288–295. IEEE.

Friberger, Marie Gustafsson, Julian Togelius, Andrew Borg Cardona, Michele Ermacora, Anders Mousten, Martin Møller Jensen, Virgil-Alexandru Tanase, and Ulrik Brøndsted. 2013. “Data Games.” In *Proceedings of the 8th Conference on Foundations of Digital Games.*

Heim, Philipp, Steffen Lohmann, and Timo Stegemann. 2010. “Interactive Relationship Discovery via the Semantic Web.” In *Proceedings of the 7th Extended Semantic Web Conference (ESWC 2010),* 6088:303–317. LNCS. Berlin/Heidelberg: Springer.

Karhulahti, Veli-Matti. 2011. “Mechanic/aesthetic videogame genres: adventure and adventure.”

Liapis, Antonios, Georgios N. Yannakakis, and Julian Togelius. 2014. “Computational Game Creativity.” In *Proceedings of the Fifth International Conference on Computational Creativity.*

Lim, Chong-U, and Fox Harrell. 2014. “An Approach to General Videogame Evaluation and Automatic Generation using a Description Language.” In *Proceedings of the IEEE Computational Intelligence and Games.*

Magerko, Brian. 2007. “Evaluating preemptive story direction in the interactive drama architecture.” *Journal of Game Development* 2 (3): 25–52.

Nielsen, Thorbjørn S., Gabriella A. B. Barros, Julian Togelius, and Mark J. Nelson. 2015. “General Video Game Evaluation Using Relative Algorithm Performance Profiles.” In *Applications of Evolutionary Computation,* 369–380. Springer.

Smith, Adam M, and Michael Mateas. 2011. “Answer set programming for procedural content generation: A design space approach.” *Computational Intelligence and AI in Games, IEEE Transactions on* 3 (3): 187–200.

Thue, David, Vadim Bulitko, and Marcia Spetch. 2008. “PaSSAGE: A Demonstration of Player Modeling in Interactive Storytelling.” In *AIIDE,* 227–228.

TIME. 1998a. “Time 100: Artists and Entertainers [Special issue]” (June).

———. 1998b. “Time 100: Builders and Titans [Special issue]” (December).

———. 1998c. “Time 100: Leaders and Revolutionaries [Special issue]” (April).

———. 1999a. “Time 100: Heroes and icons [Special issue]” (June).

———. 1999b. “Time 100: Scientists and Thinkers [Special issue]” (March).

Togelius, Julian, Mark J. Nelson, and Antonios Liapis. 2014. “Characteristics of Generatable Games.” In *Proceedings of the FDG Workshop on Procedural Content Generation.*

Togelius, Julian, and Jürgen Schmidhuber. 2008. “An experiment in automatic game design.” In *Proceedings of the IEEE Symposium On Computational Intelligence and Games,* 111–118. IEEE.

Togelius, Julian, Georgios Yannakakis, K. Stanley, and C. Browne. 2011. "Search-based Procedural Content Generation: A Taxonomy and Survey." *IEEE Transactions on Computational Intelligence and AI in Games* 3 (3): 172–186.

Treanor, Mike, Bryan Blackford, Michael Mateas, and Ian Bogost. 2012. "Game-O-Matic: Generating Videogames that Represent Ideas." In *Procedural Content Generation Workshop at the Foundations of Digital Games Conference.*

Yu, Hong, and Mark O Riedl. 2012. "A sequential recommendation approach for interactive personalized story generation." In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 1,* 71–78. International Foundation for Autonomous Agents and Multiagent Systems.