Department of Computer Science

Submitted in part fulfilment for the degree of BEng.

Networks and Dragons: A data-driven approach to procedural dungeon generation

Angel Simeonov

2020-January-27

Supervisor: Dr. Rob Alexander

<insert dedication>

Acknowledgements

<insert acknowledgements>

Contents

List of Figures

List of Tables

Executive Summary

>At most two (2) pages, aimed a non-specialist, knowledgeable authorial peer. The summary must: –state the aim of the reported work, –motivate the work, –state methods used, –state results found, and –highlight any legal, social, ethical, professional, and commercial issues as appropriate to the topic of study (if none, then this should be explicitly stated).

1 Introduction

1.1 Role-playing games

- Nature and goal RP is trying to create a compelling narrative in which players can be involved in.
  - Contextualise RPs by comparing different genres. Digital vs Tabletop.
- Interaction mechanisms. How the game is played and what is the role of the DM
- Problems: DM has to create the narrative. Can we devise an algorithm that helps DMs create a compelling narrative?

Role-playing game (RPG) is a broad term encompassing a multitude of different games with often distinct mechanics and platforms of interaction (the media). The common factor between all RPGs is that the player(s) portray a fictional character and is involved in a fictional world (or a subset of one). The interaction of the players with this world is governed by rules, defined by the media. To better understand the structure of RPGs, we can view it in terms of two sets:

1. Rules defining how we play the game. Describe the allowed actions for the player at any given moment in the game.
2. Narrative elements. Provide the *purpose* of the players to interact with the fictional world.

The first set can be viewed as "How I interact with the environment" (functional) and the second as "What is the meaning of the environment" (narrative).

Computer RPGs like the Action RPG (ARPG) Legend of Zelda, Diablo, Fallout and many alike have both sets of rules defined by the game designers. A functional rule in an RPG like Skyrim is that you can attack with the left mouse button and a story element is that you are a Dragonborn with a quest. This quest is defined by the writers and designers of the game and as such exploring the narrative in a digital RPG can be comparable to an interactive reading of a fiction book. Good computer RPGs often have the ability to relax the narrative [1], allowing for the player to have more freedom in the exploration of the world and as such create the sensation that the player has some impact on this pre-programmed fictional environment.

The ability for a player to influence the narrative is one of the defining features of tabletop RPGs (TRPG a.k.a pen-and-paper PnP) like Dungeons and Dragons [2]. In them, the functional rules are usually defined by a rulebook (like the Player's Handbook) and players verbally describe their interactions with the environment. The narrative is an ever evolving amalgam between the input of players and the Dungeon Master (DM). The DM's task is to create a narrative outline and guide the player interaction. Because of the verbal nature of the game, the narrative does not suffer the limitations of its digital counterparts. But because there are no hard constraints to how the narrative is told, the DM has the non-trivial task of introducing consistency and outlining a structure for the story that would result in a compelling and ideally immersive experience for the players. This is often achieved by focusing the adventure's act on a particular and detailed location. These locations are often referred to as Dungeons and in practice can be anything from the villain's mansion or a beast's cave to a city under siege. A good dungeon design is crucial for creating a compelling narrative. The creative task of making the Dungeon is a laborious process. To facilitate that, academics and various gaming communities have been exploring different ways of automating Dungeon creation. We will refer to this process as Procedural Dungeon Generation (PDG). Arguably the greatest problem posed by automating PDG is answering the question of "What is a compelling Dungeon?". In the next section we will review different generative methods for Dungeons and their associated limitations in an attempt to answer this question.

1.2 State of the Art Generative methods

- Review various different algorithms for PDG and discuss their take on solving the "is this dungeon interesting?" problem.
  - Non-digital: [[the Advanced DnD DM design kit book 3: Adventure Cookbook]]. Using dice tables to create different elements of the dungeon. Laborious process which requires the DM to remove any inconsistencies. Does not provide an actual Dungeon structure.
  - Cellular automata: donjon [3]. Issues associated with completely

random topology and random content generation. Evaluated only on if the level is solvable (no evaluation of semantic content).

- Constraint Propagators and their related issues when formalising narrative as constraints.
- BPTs + semantic additions[[citation needed Thrall and Brown]]. Difficulty proving the 'goodness' of the generator. Discuss difficulties with an objective HCI evaluation for PnP (i.e. difficult to get a decent sample size, because playing a game is time consuming).
- Formal language approach. Grammars [4]–[6] as a natural way of describing narrative structures.
- Data-driven approaches. [7]–[9] Relate the notion of learning from human-made dungeons as a way to create good structure + narrative. Note the lack of data. Expand more in the Mission and Space –> Aim sections

1.2.1 Non-digital Generators

The designers of the classic PnP modules acknowledge the issue of Dungeon creation and often incorporate *Loot* or *Encounter* tables in the rule-books. The tables are a reference over a set of treasures or monsters respectively, which can be chosen by rolling the specified die **need figure of loot/encounter table**. As noted before, a PnP RPG can evolve rapidly outside the planned narrative structure prepared by the DM and a simple way to quickly define new adaptive story elements via a loot table can be useful. Some early modules even provided a step-by-step guide for creating a full campaign from the plot, villains' obsessions and story setting to the various encounters and and treasures [10] entirely based on randomised content tables. An issue arises when using loot tables when a randomly selected element from one table is contradicted by an item selected from another. **show an example from the ADND module**. It is up to the DM to resolve such disparities. As can be seen, this process although providing some degree of creative assistance, it does little to facilitate the laborious nature of creating elements for a compelling narrative. Furthermore, these methods pay no attention to providing guidance of what would mean a good Dungeon as they are occupied with solving global narrative questions. **Argue that DMs know what they want to do, they just need a facilitator that will create a dungeon based on their requirements ? citation needed for what people use dungeon generators for.**

1.2.2 Digital Generators

It is important to note that PnP RPGs and the various genres of computer RPGs although differing in the mechanisms of interaction, they share the same narrative goal [11]. Therefore we will not limit ourselves to looking only at existing tabletop solutions. Unlike the original non-digital generators, computerised ones rarely allow for human input in the middle of the process. A generative algorithm would provide a DM with an interface that takes a set of parameters and produce a template of a game. The degree of complexity of this template is naturally dependent on the complexity of the algorithm itself. Because they aim to exclude the human designer from the process, the digital generators tend to focus on creating elaborate Dungeon spaces and struggle providing coherent semantic content [12], [13].

Cellular Automata

Cellular automata (CA) utilises a procedural generation mechanism in which a $N \times M$ grid space is mutated incrementally by a set of agents**[citation needed for CA and fig. showing agents in action?]]**. The implementation of these mutation operators define if the CA will simulate erosion [14] or man-made structures such as rooms and corridors. The latter is the case of the donjon dungeon generator [3]. Its simplicity and degrees of customisations of both layout and style has made donjon a popular generator choice with more than 2500 generated dungeons in the last 12 hours and 100 donations in the last 3 months **[[citation needed?]]**. Despite its popularity, donjon (as well as other CA) implores purely random generation techniques which are only evaluated based on a notion of the solvability of the level. The criteria for donjon's level is only if each room is reachable (i.e. there is a path to each room). As it can be observed, although we have control over the input parameters for the topological randomness, the semantic elements (the contents, rather than the structure) of the dungeon are entirely arbitrary and often contradictory **[[citation of brown and thrall or elaboration?]]**. In terms of the aforementioned solvability, when accounting for reachability donjon does not take in account the randomly allocated locked doors as the key allocation is left at the DM's discretion. That ultimately limits the usefulness of the tool and only provides a skeleton for the dungeon. All information generated about the monsters, treasures and other quest elements are discarded by the DM **[[citation needed on how people use dungeon generators]]**

Constraint Propagators

Constraint Satisfaction Problems (CSPs) define a discrete set of variables with corresponding constraints that restrict the possible variable instantiations **[[cn of CSPs]]**. Dungeon generation has been described as a CSP on occasions where the variables and constraints define the layout **[[cn]]**, contents **[[cn]]** or both layout and content**[[cn]]** of a dungeon. Constraint Propagation (CP) is a particular method proven useful for solving CSP in the context of dungeon level generation. The algorithm selects a set of possible values for a particular variable and *propagates* the result to the rest of the variables, adjusting their possible values accordingly. If the selected value narrows another variable's possible values to the empty set, we infer that this instantiation is suboptimal and we backpropagate to the last viable solution and retry. Furthermore it can be generalised that if an instantiation for all variables that does not narrow any variable to the empty set exists, the level is solvable. Utilising this formalisation of the *solvability* of a level we can populate a level with varying degrees of complexity from ensuring that keys are always spawned before the door that they must unlock to adjusting difficulty by measuring survivability metrics between rooms **[[cn]]**. Even thought CPs have solved some problems of the completely random generators, formulating a numerical constraint for a narrative element can be difficult. The complication is both in the translation of a story element to a numerical representation and in the computational expense that comes with computing large amounts of permutations.

Binary Partitioning

Formal Languages

Data-driven approaches

1.3 Mission and Space

- Introduce the notion of Mission and Space
- The reason for separating Mission and Space. We can model player experience better [4], [7].

An attentive reader would have noted that distinguishing between the level layout and contents is a common occurrence in popular PDG algorithms. This discrimination was formalised by Dormans[4] in his definitions of *Mission* and *Space* in relation to the ARPG Legend of Zelda game series. The *Space* of a dungeon embodies it's topological structure and can be encoded as an undirected cyclic graph where each vertex is a room and each edge is a door or corridor. The *Mission* is the set of narrative tasks the player must accomplish in order for him to complete the dungeon. It can be encoded as a directed graph where each vertex is the objective and the edge directions show the required sequence of completion. It is clear to see that the gameplay experience is defined in the interaction of the two graphs. As noted by Dormans, a *Mission* mapped on different layouts will result in drastically different exploration patterns and it is by understanding the necessities of the two separate entities that we can model player experience better. Due to this separation we have seen advancements in generating adaptable to the player game environments [15] and even reverse engineering the creation of *Missions* and *Spaces* by learning structure from data [7]

1.4 Aim

The dungeon's topology (Space) and contents (Mission) are correlated and embody the narrative of the game. As we have seen, dungeon generators usually implore bottom up approaches in which they apply rules for topology and then introduce dungeon content in an attempt to provide the structure for a captivating narrative. The various algorithmic methods have clear strengths and shortcomings in achieving the complex goal of creating an interesting story. I want to argue that from these observations we can affirm that the ultimate *dungeon generator* is the human designer. Dungeons and therefore narratives produced by humans are the most compelling out of all created dungeons. As noted by others **[[cn]]**, if we implore a top-down approach of analysing what makes a good man-made dungeon, we could potentially achieve higher levels of narrative automation.

Deery made the first steps by manually analysing submissions to the One Page Dungeon OPDC competition [[OPDC]] to extract a graph grammar for Mission generation [5]. The Mission graph was then mapped to a physical Space in a 1:1 ratio. The result was that each room was limited to a single Mission element. It is trivial to see that the originals in OPDC do not impose such a restriction. One room can have multiple Mission elements (e.g. the key to unlocking the door is on the bandit's waist, Key + Encounter). Furthermore, Deery's approach was to manually look at 10 competition winners and heuristically extract the grammar rules, which he highlights that they do not capture all the possible patterns. An automated approach to learning would potentially solve that issue. Programmatically learning a level from data has been an object of interest for computer based RPGs [7], but has not been applied to PnP RPGs, presumably because of the lack of a consistent dataset.

In this paper we will investigate if applying a data-driven approach to learning the Space of the dungeon can produce a map that is undistinguishable from human made dungeon topologies. We will define a meaningful set of features to be extracted from the OPDC dataset and use those features as random variables defining an Inference (Bayesian) Network. Subsequently we compare different network structures and learning algorithms and internally assess which model has the greatest statistical capabilities using various scoring rules.

2 Bayesian Inference Networks

What are Bayesian networks? Bayesian Networks *BNs* [16] (also referred to as Inference Networks, Belief Networks, Directed Acyclic Graph (DAG) Models and others) are a graphical probabilistic model in which vertices are random variables *RVs* whose edges indicate causal beliefs. BNs assert that our domain is defined in terms of a joint probability distribution $P(X_1, \ldots, X_n)$ over a set of RVs X, each of which has a set of potential values it can take $X_i(\Omega) = \{x_1, \ldots, x_j\}$ with an associated probability distribution *PD*. The network structure shows how the full join distribution factorises given the causal dependencies. This property gives us a way to encode our prior domain knowledge about the dependencies between our RVs. Once we have encoded our knowledge about the RV interaction, using a BN as an inference tool is a matter of querying with a conditional set of RVs and noting how the CPTs change given our new knowledge. In practice, conditioning is done by *observing* (i.e. instantiating) an RV to be a particular value.

To contextualise this, we can look into Summerville's transcription of the Legend of Zelda dungeons [7]. After annotating the dungeon maps with elements of both Mission and Space (total number of rooms, treasures, monsters, critical path CP, etc.), he extracts features for the BN by parsing them as RVs. If we condition on total number of rooms (i.e. observe that RV), the PD for the possible CP would reflect our new knowledge. Most likely the CPs that are greater than the total number of rooms would be impossible $P(CriticalPathLength > n \mid NumRooms = n) \to 0$, which is logical. Then by sampling the RV for the CP length we can fix the parameter and get a point estimate for critical path length given that number of rooms in the dungeon. This inference process can take various degrees of complexity

Why use Bayesian networks? The reasoning behind choosing BNs is that the probabilistic causal structures have been shown to perform well in capturing properties from data for PDG tasks in ARPGs [7], [9]. Furthermore they give us a natural description about the dependencies in our model, which can be used as a supplemental analysis tool for investigating human made dungeons. As a PDG tool itself, the *observe-then-infer* pattern gives the user precise control over the desired properties of his dungeon without loss of automation. If a user observes a dungeon with 5 rooms and critical path of length of 3, they will be guaranteed in the final product. This solves the problem previous graph grammar data-driven approaches encountered, whereby the dungeon size input was used as an approximation for the actual number of rooms in the dungeon [5].

The downside of BNs is that their performance and generalisation capabilities are highly dependent on the availability of data. If conditioning on an undefined set (e.g. there was never a data entry that showed a 5 room dungeon with CP of 3), the inference will fail. A proposed way to solve this issue is by creating artificial data entries for the undefined cases. We will use linear interpolation due to its simplicity and the fact it has been shown to produce good results in general [17] and exceptional results on small datasets [18].

What we need to do to use Bayesian Networks? In order for us to create an inference network that reflects our beliefs we need to define two key attributes of the BN:

1. The causal structure
2. Conditional probability tables *CPTs* for each RV

3 Methods and Implementation

3.1 Data acquisition

3.1.1 Data selection

3.1.2 Extracting Features

The final dataset consists of **[[size]]** organic entries and **[[synthetic]]** linearly interpolated entries for a total of **[[dungeon params]]** and **[[room params]]**.

3.2 Model selection

3.2.1 Bayesian Network structure

Summerville's Extreme Sparse Model

TAN

3.2.2 Parameter Learning

Count

Expectation Maximisation EM

Gradient Descent GD

3.3 Implementation

Dungeon Sampling We can define characteristics of the desired dungeon by leveraging the inference nature of BNs. Generating a dungeon with five rooms is a matter of fixing (*observing*) the `NumRooms` parameter.

The user can choose virtually any permutation of parameters to be fixed or inferred. For consistency purposes we are observing only the size of the dungeon via `NumRooms`.

Room Sampling

Converting from samples to Space

4 Results

4.1 Internal Validation

Table 4.1: Structure and parameter learning comparison.

	Count			EM			
	log loss	quadr loss	sphere pay	log loss	quadr loss	sphere pay	lo
Summerville's ES	-	-	-	-	-	-	
TAN	-	-	-	-	-	-	

4.2 External Validation

5 Discussion

5.1 Critique

1. Data
   a) availability and extraction.
      i. Could extend to include even non-winning entries for the sake of having a greater sample size. More datasets can be considered. We've use OPDC due to its CC license, but paid modules exist.
      ii. Data extraction has been a manual process. Although due diligence is paid, noise introduced from human error is inevitable.
2. Method
   a) Cannot generalise to unseen cases. We can interpolate for missing data, but we cannot extrapolate. Argue that this is an issue of all ML approaches, not just BNets, because we're trying to capture properties of One-Page dungeon.
   b) This paper is considering only Space gen. A more robust approach would be needed to extract consistent and meaningful parameters for Mission. For Legend of Zelda there is a discrete subset of things you can do so that is why Mission extraction is possible. Deery has shown that formalising Mission in PnP RPG's is difficult due to the variant and creative nature of the human made dungeons.
3. Validation
   a) We have decided to approach the external evaluation in a quantitative, rather qualitative measuring. That is due to the fact that conducting a study with a high environmental index is difficult due to multiple confound factors that are due to the nature of a tabletop RPG session. Not only does a one-session adventure usually take around 3 hours **[[cn]]**, but they are extremely variant between groups of players and DMs. An experiment that analyses how players use the generated dungeon rather

than just discriminating between different dungeons topologies would give us more insight in the success of the recreation of a human-made dungeon.

6 Conclusion

A Some appendix

Use this section for graphical showing of the models. Nets, result tables (or tables should be inline?)

B Another appendix

Use this section for questionnaires and external validation support

Bibliography

[1] A. Tychsen, M. Hitchens, T. Brolund and M. Kavakli, 'The game master,' in *Proceedings of the Second Australasian Conference on Interactive Entertainment*, ser. IE '05, Sydney, Australia: Creativity Cognition Studios Press, 2005, pp. 215–222, ISBN: 0975153323.

[2] G. Gygax and D. Arneson, *Dungeons and dragons*, 1974.

[3] *Donjon 5e dungeon generator*, https://donjon.bin.sh/5e/dungeon/, Accessed: 27/01/2020.

[4] J. Dormans, 'Adventures in level design: Generating missions and spaces for action adventure games,' in *Proceedings of the 2010 Workshop on Procedural Content Generation in Games*, ser. PCGames '10, Monterey, California: Association for Computing Machinery, 2010, ISBN: 9781450300230. DOI: 10.1145/1814256.1814257. [Online]. Available: https://doi.org/10.1145/1814256.1814257.

[5] C. Deery, 'Generating dungeons dragons dungeons that make visible sense,' Master's thesis, University of York, 2019.

[6] H. Cadogan, 'Procedurally generating dungeons and dragons content using context-free grammars,' Master's thesis, University of York, 2019.

[7] A. J. Summerville, M. Behrooz, M. Mateas and A. Jhala, 'The learning of zelda: Data-driven learning of level topology,' in *Proceedings of the 10th International Conference on the Foundations of Digital Games*, Jun. 2015.

[8] A. Summerville, S. Snodgrass, M. Guzdial, C. Holmgård, A. Hoover, A. Isaksen, A. Nealen and J. Togelius, 'Procedural content generation via machine learning (pcgml),' *IEEE Transactions on Games*, vol. PP, Feb. 2017. DOI: 10.1109/TG.2018.2846639.

[9] A. J. Summerville and M. Mateas, 'Sampling hyrule: Sampling probabilistic machine learning for level generation,' 2015.

[10] H. Johnson and A. Allston, *Dungeon master's design kit (1e)*, Reading, Massachusetts, 1988.

[11] A. Tychsen, 'Role playing games: Comparative analysis across two media platforms,' in *Proceedings of the 3rd Australasian Conference on Interactive Entertainment*, ser. IE '06, Perth, Australia: Murdoch University, 2006, pp. 75–82, ISBN: 869059025.

[12] H. Thrall, 'Procedural generation of a dungeon for dungeons and dragons,' Master's thesis, University of York, 2017.

[13] S. Brown, 'Prototyping software for the complex human task of creating dungeons and dragons dungeons with narratives, using user centred design and procedural generation,' Master's thesis, University of York, 2018.

[14] *Cellular automata method for generating random cave-like levels*, http://roguebasin.roguelikedevelopment.org/index.php?title=Cellular_Automata_Method_for_Generating_Random_Cave-Like_Levels, Accessed: 27/01/2020.

[15] J. Dormans and S. Bakkes, 'Generating missions and spaces for adaptable play experiences,' *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 3, no. 3, pp. 216–228, Sep. 2011, ISSN: 1943-0698. DOI: 10.1109/TCIAIG.2011.2149523.

[16] J. Pearl, 'Bayesian networks: A model of self-activated memory for evidential reasoning,' in *Proceedings of the 7th Conference of the Cognitive Science Society, University of California, Irvine, CA, USA*, 1985, pp. 15–17.

[17] P. H. Ibargüengoytia, U. A. García, J. Herrera-Vega, P. Hernandez-Leal, E. F. Morales, L. E. Sucar and F. Orihuela-Espina, 'On the estimation of missing data in incomplete databases: Autoregressive bayesian networks,' in *ICONS 2013*, 2013.

[18] J. Yu, V. A. Smith, P. P. Wang, A. J. Hartemink and E. D. Jarvis, 'Advances to bayesian network inference for generating causal networks from observational biological data,' *Bioinformatics*, vol. 20, no. 18, pp. 3594–3603, 2004.