Exploring the Possibility Space of 1 Billion Spells

Oliver Withington
Queen Mary University of London
London, United Kingdom
o.withington@qmul.ac.uk

Abstract

In this short paper we introduce the first version of a 'spell simulator' for 1 Billion Spells, an in development action game, and explore the simulator's utility for conducting large scale exploration and evaluation of that game's spell creation system. Using this simulator we conduct initial experiments to explore specific developer concerns about the game which would be challenging to explore with conventional playtesting. We argue that our initial exploration demonstrates the potential of this approach in evaluating content generation systems for games that have large player explorable possibility spaces.

CCS Concepts

• **Applied computing** → *Media arts*; • **Human-centered computing** → Human computer interaction (HCI).

Keywords

procedural content generation, evaluation, games

ACM Reference Format:
Oliver Withington. 2025. Exploring the Possibility Space of 1 Billion Spells. In *International Conference on the Foundations of Digital Games (FDG '25), April 15–18, 2025, Graz, Austria.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3723498.3723783

1 Introduction

Within the research community concerned with procedural content generation (PCG), there is a consistent preoccupation with what Togelius and Yanakakis term 'The Gap' [12] - the disconnect in practice and priorities between researchers and commercial game developers. While there have been efforts to reduce it [3], Lai et al argued in 2020 that it was larger than ever [4]. We argue that an avenue for bridging this gap is to center the goals of a real world game developer to see how effective PCG research techniques are in addressing these goals.

The goal for this work is to do just that, and bring research techniques and practice directly to an in development commercial game: 1 Billion Spells (hereafter referred to as 1BS). 1BS is an action game being developed by Icedrop Games (https://icedropgames.com/), with a node based spell creation system as its central mechanic. This system supports the creation of large numbers of unique spells, which the developers hope will produce an experience that players will want to replay to explore alternative options within the spell possibility space. However the size of this possibility space makes the game hard to test and balance.

Figure 1: Screenshot of 1 Billion Spells, showing the player character evading a hoard of enemies

In this paper we introduce a spell simulator for mapping this possibility space using PCG research techniques, including a generate-and-test genetic search algorithm for generating spells, and Expressive Range Analysis (ERA) [7]. The goal is to be able to produce data to address developer questions and concerns about the spell making system. The primary contribution of this work is the initial exploration of whether and how these techniques are practical for producing useful information for game developers.

2 Background

In this section we introduce and discuss the context needed to understand the work done in this paper, both on the pre-existing research that informs this, but also on the game that we are analysing.

2.1 PCG Evaluation

While PCG for games has been a popular research domain for almost two decades now, with hundreds of new works each year, the evaluation of new research has never been successfully standardised. Other AI fields have made extensive use of standardised benchmarking, such as the EMNIST dataset for image recognition [1], and have arguably benefited significantly from being able to directly compare novel systems to prior ones (though we should note there has been significant recent pushback against the value of benchmarking in other AI fields such as LLM research [6]).

It has often been argued that PCG as a field needs something similar [2, 5, 9], but no consensus has emerged. A 2024 survey of evaluation practice within PCG for game level generation reinforced this perception by showing both the diversity of techniques being used and the lack of reference to prior work in new ones

FDG '25, Graz, Austria

ACM ISBN /25/04
https://doi.org/10.1145/3723498.3723783

[11]. However, this is in many ways unsurprising. As argued by Whitehead [10] PCG for games is an aggregate field which draws on fields as disparate as art, engineering and computer science. Finding evaluative practice which can unify the needs of these different disciplines is obviously challenging, and maybe infeasible.

However, a potential solution is to instead base PCG design and evaluation in the needs and concerns of developers [4], an approach that this paper aims to be an early exploration of.

2.2 Expressive Range Analysis

Expressive Range Analysis (ERA) is one of the dominant techniques used for evaluating Procedural Content Generation (PCG) systems [11]. Introduced by Smith and Whitehead in 2010 [7] its basic operation is appealingly simple. First a pair of quantifiable Behavioural Characteristics (BCs) are chosen which describe the artefacts produced by the PCG system. These BCs are then calculated for a large generated sample of content and then this set of BC pairs can then be visualised on a 2D plot. These can then be used to interpret what types of content are likely and unlikely to be produced by a given PCG system in terms of these BC values. While ERA has been iterated upon many times in the intervening 15 years [2, 8] it is commonly used in its original form to describe the output diversity of PCG systems.

2.3 1 Billion Spells

1BS in its current form is a 2D action game in which a trainee wizard is tasked with evading and dispatching waves of enemies. A key game mechanic is spell making. It uses a click and drag node based system to allow players to create their own unique spells (See Figure 2 for a screenshot). These nodes are chosen at the end of each enemy wave by the player from a randomly selected set.

In the current version of the game there are over 50 node types and in a given playthrough of the game the player can choose a dozen or more from this pool. This means there is a huge possibility space of spells that could be produced, with a potentially large amount of variance in the viability of the spells within the possibility space. For 1BS' developers, trying to get a high level view of this space to know whether their design goals are being met is a serious challenge.

The high level designer goals that we look to explore with the initial experiments presented in this paper relate to both game balance and diversity of play. They are as follows:

(1) For all nodes to be viable options
(2) For it to be not overly easy to make bad spells to avoid new player frustration
(3) For the system to not be biased towards linear chains of nodes

3 Spell Simulator System

The goal for this system is to be able to gain useful insights into the possibility space of makeable spells in a way that supports the game designers in deciding what changes need to be made to the underlying spell system. To do this we developed a 'spell simulator', a lightweight Python system which allows for the generation of hundreds of thousands of spells with limited time and computing resources required. Below we describe the spell simulator system at a high level to support an understanding of this work, though we elide many of the specifics and details of how it operates for reasons of space and relevance. For more specifics on how it works please contact the authors or consult the Github repository at github.com/KrellFace/1BilSpells_Sim.

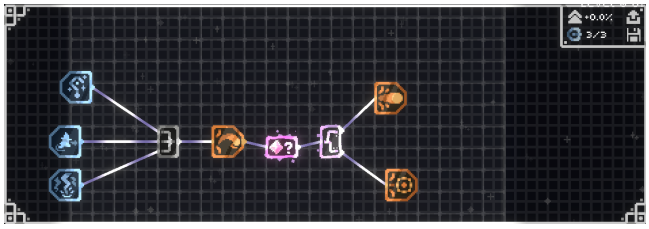

Figure 2: Screenshot showing the in game node based spell creation screen. Energy nodes are colored in blue, damage nodes in orange and modification nodes in grey & purple

3.1 Abstractions

As this simulator is a completely separate system to the actual game engine it has no model of the game world or the enemies present, and there are abstractions that we make about spells when we are simulating them. The primary ones are that we assume:

- Every damage instance will connect with at least one enemy.
- Every energy source will trigger at its maximum refire rate.

However in terms of the nodes available, how they can connect with each other and the effects they have in game; the simulator aims to closely adhere to the in-game reality.

3.2 Spell Modelling

Spells are modelled as networks of connected nodes, similar to their in-game representation in the spell maker menu (See Figure 2). All nodes are partially defined by the maximum number of parent and child nodes they can have, but there are also three main node subtypes which have their own important features:

- **Energy Nodes** - Which produce energy pulses. These are primarily defined by the frequency with which they produce energy pulses and the size of these pulses, as well as whether these pulses are triggered directly by the player or triggered by an in-game event such as an enemy's death
- **Damage Nodes** - Which convert energy pulses into damage effects. These are primarily defined by their conversion rate of energy into damage, and the size of the area of effect produced.
- **Modification Nodes** - Which change the properties incoming energy pulses and all child nodes that receive pulses from it, such as adding movement to projectiles

Time can then be simulated by generating energy pulses at a rate and frequency dictated by their values, represented as simple data tuples, from energy nodes and then passing them down to child nodes. On arrival at a damage node the information in the energy pulse data packet is used to generate information about the in game effect.

3.3 Spell Generation

The current version of the simulator uses a Generate and Test system to make spells. The steps involved are:

1: Select node pool P for next spell
2: Copy P as P‘
3: Select node X from P‘
4: Attempt to add node X to spell. Remove from P‘ if successful
5: Repeat steps 3 and 4 until P‘ is empty or failed attempts reaches threshold
6: Calculate quality heuristic for spell. If higher than current best for P‘, store spell
7: Repeat steps 2 to 6 a fixed number times to find the best spell for pool P
8: Repeat Steps 1 to 7 until the desired number of spells are generated

4 Experiments and Initial Results

Here we present the initial experimentation and results using this system, primarily to illustrate the early use cases of the system, as well as to inform discussion of its limitations.

4.1 Spell Features Calculated

The features we calculated for each generated spell are:

- **1DPS** - The theoretical maximum damage that could be done to a single enemy unit per second.
- **Linearity** - Heuristic for the amount of branching within a spells node web. Calculated by dividing the length of the longest parent-> child chain by the number of unique nodes.

4.2 Spell Generation Parameters

For our spell set to evaluate we generated 100,000 individual spells, with 1DPS as the quality heuristic for deciding which spell to store. A pool of 9 nodes was selected for each spell, evenly selected from Energy, Modification and Damage nodes. The system was given 20 attempts to generate the spell with the highest 1DPS for each pool.

Each spell was simulated for 30 seconds to extract information about the damage it could do in this time period. The spell generation and evaluation process took 45 minutes in total on a Dell laptop with an i5-10310U CPU with 16.0 GB of RAM.

As the current system cannot guarantee that all nodes will be placed in the final spell, we then filtered this set of 100,000 spells down to only spells that used the full 9 nodes made available to them to make the later analysis of them fairer. This left us with a set of 1,855 spells to analyse.

4.3 Analysis Conducted

The first round of analysis we conducted was looking at the correlation between the appearance of individual nodes within a spell and its 1DPS. To do this every spell was assigned a binary flag of 0 or 1 for every node type, where 1 indicated its presence in the spell. We then evaluated the Pearson correlation coefficient for the correlation between the values for each node type and 1DPS. This gives us a value between -1 and 1 with higher values indicating a stronger relationship between that node's presence in a spell and its damage (Table 1).

	Node	Pearson (3dp)
Highest	WandCastNorm	0.387
	ProjectileWithOutput	0.373
	OnDashStart	0.221
Lowest	CrystalNode	-0.200
	OnManaCollect	-0.214
	OnHitNode	-0.224

Table 1: Table showing the three nodes most and least correlated with 1DPS

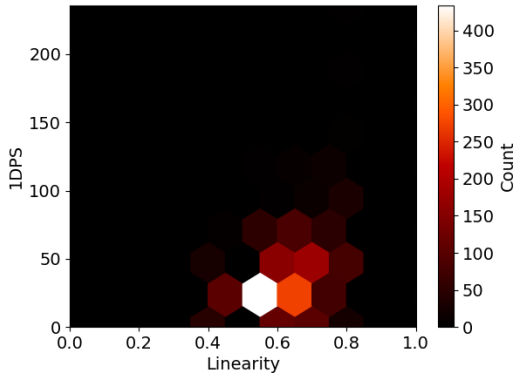

Figure 3: ERA plot of linearity against 1DPS

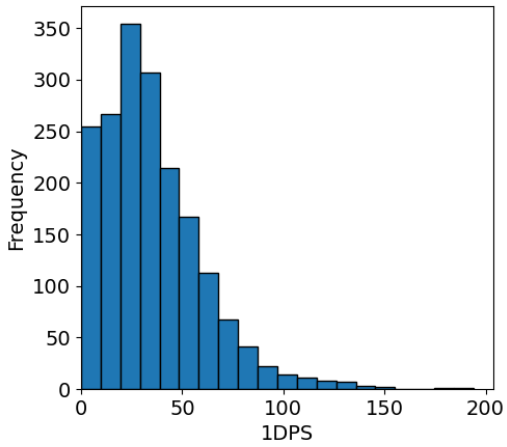

Figure 4: Histogram showing the distribution of 1DPS values

We conducted ERA on the pairing between spell 1DPS and Linearity to illuminate the distribution of the two values (Figure 3). We also calculated the Pearson correlation coefficient between the two to explore the 1BS' developers' concerns that linearity was correlated with spell power.

5 Discussion

Though it is early in the spell simulator's development, the insights that can be drawn from it are already interesting. If we look at Table 1 the level of disparity between the strongest and weakest nodes in this version of the game is interesting, and a potential cause

for concern for the developers of 1BilSpells for whom balance is a priority. The two strongest nodes are significantly stronger than all others, and vastly more so than the bottom of the scale.

The spread of 1DPS (Figure 4) was also informative, showing heavy clustering around 20 to 40 1DPS. It also shows that it is easy to make a spell that produces less 1DPS. This is a cause for concern as 1BS' developers wanted to limit the availability of underpowered spells and for reference the basic spell that every player starts with already produces 20 1DPS. The system also produces large numbers of spells that produce 0 or close to 0 damage, but this represents a limitation of our spell generator, not of the spell system itself, as we discuss further shortly.

On the correlation between spell potency and the spells linearity we were able to somewhat allay the developers concerns as we found a negative correlation between linearity and 1DPS of -0.178. However, the ERA plot in Figure 3 does show a potentially troubling clustering, indicating that while complete linearity is not optimal, there is a more optimal linearity to aim for which could indicate a lack of balance and diversity in optimal strategies.

On the overall practicality of the approach, 1BS's developers were surprised at the speed with which this analysis could be done and optimistic about its potential utility. The simulator and this analysis were completed in approximately 120 hours by a single developer. Much of this time was spent on comprehending the existing codebase for 1BS which was not built to support this kind of analysis. Planning for this kind of evaluation from the start would significantly accelerate this kind of approach. Both sides of this project, researchers and developers, felt the approach was promising but that more work and analysis is required to know how valuable it would be as part of the game's production.

5.1 Limitations

A limitation with this system is the simplifications and abstractions, the biggest of which being the assumption that all projectiles will hit a target. This is an issue, as the downside for many nodes which their strengths are meant to offset is reducing your odds of being able to hit the enemies, such as the 45°node which randomly perturbs the launch direction of projectiles in exchange for a power boost.

There are also the issues with the spell generator. Our generate-and-test implementation which creates spells purely stochastically does not necessarily correlate with the decisions a human player would make. Whether it's generating spells that produce no damage by including no damage nodes, or making decisions like splitting the output of an energy node and then only utilising one of the split outputs, the system often makes spells that no human with understanding of 1BS ever would. This could be partially addressed by adding more constraints to the generative process, but this would have tradeoffs in maintainability.

6 Future Work

We have many ideas for how to improve the system in future, informed first and foremost by the needs and interests of the 1BilSpells development team. We aim to expand the modeling of spells by supporting the calculation features like spell range and the change of spells to connect.

The most important though, is confirming with 1BS's developers that the insights produced by this system are useful and informative. Ideally this will take the form of a qualitative follow up study on the insights from the simulator and their utility in supporting 1BS' development.

7 Conclusion

In this paper we introduced a spell simulator system for exploring the spell design space of the upcoming action game, 1 Billion Spells. We introduced the goals that the game's designers have for the spell making system, and how we translated these into forms of evaluation the simulator could conduct. While this system has significant limitations, it produces insights which would be hard to get through conventional play-testing.

Acknowledgments

This work was supported by the EPSRC Centre for Doctoral Training in Intelligent Games & Games Intelligence (IGGI) [EP/S022325/1]. We are also grateful for the helpful and informative feedback we received from our reviewers on the initial version of this work.

References

[1] Gregory Cohen, Saeed Afshar, Jonathan Tapson, and André van Schaik. 2017. EMNIST: an extension of MNIST to handwritten letters. (2017). doi:10.48550/ARXIV.1702.05373 Publisher: arXiv Version Number: 2.

[2] Michael Cook, Jeremy Gow, Gillian Smith, and Simon Colton. 2021. Danesh: Interactive Tools For Understanding Procedural Content Generators. *IEEE Transactions on Games* (2021). https://ieeexplore.ieee.org/document/9426419

[3] Matthew Guzdial, Nicholas Liao, Jonathan Chen, Shao-Yu Chen, Shukan Shah, Vishwa Shah, Joshua Reno, Gillian Smith, and Mark O. Riedl. 2019. Friend, Collaborator, Student, Manager: How Design of an AI-Driven Game Level Editor Affects Creators. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. ACM, Glasgow Scotland Uk, 1–13. doi:10.1145/3290605.3300854

[4] Gorm Lai, William Latham, and Frederic Fol Leymarie. 2020. Towards Friendly Mixed Initiative Procedural Content Generation: Three Pillars of Industry. In *International Conference on the Foundations of Digital Games*. ACM, Bugibba Malta, 1–4. doi:10.1145/3402942.3402946

[5] Antonios Liapis. 2020. 10 Years of the PCG workshop: Past and Future Trends. In *International Conference on the Foundations of Digital Games*. ACM, Bugibba Malta, 1–10. doi:10.1145/3402942.3409598

[6] Timothy R. McIntosh, Teo Susnjak, Nalin Arachchilage, Tong Liu, Paul Watters, and Malka N. Halgamuge. 2024. Inadequacies of Large Language Model Benchmarks in the Era of Generative Artificial Intelligence. doi:10.48550/ARXIV.2402.09880 Version Number: 2.

[7] Gillian Smith and Jim Whitehead. 2010. Analyzing the expressive range of a level generator. In *Proceedings of the 2010 Workshop on Procedural Content Generation in Games - PCGames '10*. ACM Press, Monterey, California, 1–7. doi:10.1145/1814256.1814260

[8] Adam Summerville. 2018. Expanding Expressive Range: Evaluation Methodologies for Procedural Content Generation. In *Fourteenth Artificial Intelligence and Interactive Digital Entertainment Conference*. https://www.aaai.org/ocs/index.php/AIIDE/AIIDE18/paper/view/18085

[9] Julian Togelius, Georgios N. Yannakakis, Kenneth O. Stanley, and Cameron Browne. 2011. Search-Based Procedural Content Generation: A Taxonomy and Survey. *IEEE Transactions on Computational Intelligence and AI in Games* 3, 3 (2011), 172–186. doi:10.1109/TCIAIG.2011.2148116

[10] Jim Whitehead. 2017. Art and science of engineered design: what kind of discipline is PCG?. In *Proceedings of the 12th International Conference on the Foundations of Digital Games*. ACM, Hyannis Massachusetts, 1–3. doi:10.1145/3102071.3110571

[11] Oliver Withington, Michael Cook, and Laurissa Tokarchuk. 2024. On the Evaluation of Procedural Level Generation Systems. In *Proceedings of the 19th International Conference on the Foundations of Digital Games*. ACM, Worcester MA USA, 1–10. doi:10.1145/3649921.3650016

[12] Georgios N. Yannakakis and Julian Togelius. 2018. *Artificial Intelligence and Games* (1st ed. 2018 ed.). Springer International Publishing : Imprint: Springer, Cham. doi:10.1007/978-3-319-63519-4