1 Introduction

1.1 Motivation

People have long enjoyed puzzles. Crossword puzzles, Sudoku, sliding puzzles, and other brain teasers have engaged and intellectually stimulated people throughout time. We continue this tradition today with hyper-casual puzzle games. Digital games with bite sized levels that can be enjoyed by anyone anywhere. Many of these games offer thousands of levels and provide new levels every day to challenge players. This massive generation of content is no small feat. A designer must create the levels, test their engagement quality and difficulty, and then order them in a way that challenges the player appropriately. Doing this for thousands of level can be a time-consuming process.

A solution to this is procedural content generation (PCG), where algorithms are used to automate the generation process. This has been researched and used in many different types of games, both for aesthetics and game content. Still, puzzle games provide a unique problem as the generated levels must be solvable and appropriately challenging [4]. In most games, solvability can be verified or guaranteed through the generation method, but difficulty is more challenging to estimate. It requires a greater understanding of the game's design and how players interact with the game. At the same time, it is also crucial for generating engaging puzzles.

Recently, content generation for other media has progressed significantly through the use of deep learning methods. With these developments, work into using these methods for PCG has emerged, called procedural content generation via machine learning (PCGML)[14]. However, there are some challenges when applying machine learning to games. In the best case, games might have a few thousand levels for training data, but this is not enough for many methods. Games also require levels to be solvable or playable. For some games, even small changes can make a level impossible to complete. What details determine this can be difficult for a model to learn.

Since PCGML emerged, relatively little work has been done on puzzle games, but among the explored methods, procedural content generation via reinforcement learning (PCGRL) has shown promising results for generating levels for the puzzle game Sokoban [18]. PCGRL [8] describes the generation as a Markov decision process, where iterative actions create a level. This requires no training data, which solves one of the challenges with PCGML.

However, PCGRL still does not guarantee solvability and has little qualitative control for generating engaging levels.

So PCGRL could be improved by using generation processes that guarantee solvability and rewarding it for generating engaging puzzles. Hyper-casual puzzle games could be good candidates for exploring this further. Their limited states and actions, relatively high-level quantity, and large player count provide a reasonably sized problem and possibly sufficient data for solving it. One such game is Longcat, which Fancade developed. It has 250 levels, but thousands of users, meaning it is well suited for exploring this problem.

1.2 Assumptions on Engagement and Difficulty

In this work, we aim to explore the generation of engaging puzzles. To do that, we first need to define engagement. This might seem intuitive but the concept of engagement within games is quite fuzzy and has many definitions [3, 9]. It also encapsulate many parts of design such as usability, aesthetic appeal, novelty, involvement, attention. One key aspect is difficulty [9]. Players lose interest if a game is too easy, and players leave frustrated if a game is too difficult. There is a balancing act in design to challenge the player appropriately. As the player plays the game, they also learn its mechanics, meaning progressively more difficult levels are needed to challenge them. So, to create a series of engaging levels, we need to adjust difficulty progressively. This is called a difficulty curve [1].

To adjust this project's scope, we will focus on this aspect of engagement. We will work under the assumption that if we can control the difficulty of generated levels, we can create a series of engaging levels.

1.3 Aim and approach

In this work, we aim to explore whether the PCGRL approach can be trained with a difficulty predictor to create challenging and engaging levels, and how our novel solution-down generation process compares to the one previously used with PCGRL. We call this method solution-down procedural content generation via reinforcement learning.

To accomplish this aim, we will first implement a level difficulty predictor. This predictor will take a Longcat level and return a numeric value that estimates the level's difficulty. It will be used to reward the generators during training for making challenging levels.

We will then create and compare four generators. Two of them will select actions at random: the first, using a puzzle-up process, and the other, using the solution-down process that the authors designed. These two random generators will be used as a baseline. The other two generators will use PCGRL and be trained with the difficulty predictor to generate challenging levels. The first will also be a type of puzzle-up generator based on the Turtle method previously used with PCGRL, and the second will use the solution-down generation process.

The comparison will be done by generating 20,000 levels with each generator and comparing solvability, duplicates, and difficulty of the generated levels.

1.4 Research questions

This project aims to answer the following:

1. What attributes of Longcat levels are relevant for predicting difficulty?
2. What percentage of Longcat levels generated with PCGRL and Puzzle-Up are solvable?
3. What are the differences in the distribution of difficulty for Longcat levels generated with SD-PCGRL compared to those generated with a random generator?

4. What percentages of Longcat levels generated with SD-PCGRL are duplicates, and how does it compare with a random generator?

5. What percentages of Longcat levels generated with SD-PCGRL have a non-trivial difficulty, and how does it compare with a random generator?

1.5 Contributions

This work builds on the concept of PCGRL created by Khalifa et al.[8]. This works contribution is to explore if these models can learn aspects of level design, such as difficulty, which could be used to make more engaging levels. Additionally, this work proposes a different type of generation method than the one used by Khalifa et al. called Solution-Down generation that could, in some aspects, be more effective. This work also tests both methods on the game Longcat, which had not previously been tested.

1.6 Outline

This sections describes the outlining structure of this report. Chapter 1 is an introduction where the motivation for the work is establish as well as its aim and the research questions it aims to answer. Chapter 2 summaries the theory and previous work that the reader needs to understand the work, and also establish what work it is based on. Chapter 3 describes the analysis phase of this work. Before the work could begin some initial analysis of the game Longcat was needed to understand how levels could be interpreted to find their difficulty. Chapter 4 describes the development of the difficulty predictor used in the rest of the work. Chapter 5 describes the work done around the generators. What generators where tested and how they work and differ from each other. The methodology used to test the generators, and the result of these tests. Chapter 6 discusses the methods and results found in all the three previous chapters. Chapter 7 gives the conclusion of the work and clearly answers the research questions.