**PCG Mazes:**

There are two general approaches to PCG, constructive and search based. In constructive PCG, the content is generated in one pass. While search based PCG (SBPCG) techniques generate and test content repeatedly until a solution is found which satisfies predefined constraints.

The three key elements of SBPCG:

* A search algorithm which defines the constraints that can be tailored to a defined content representation.
* A content representation, which defines how a solution is represented. This can also define any limits or constraints that a particular valid solution may have.
* A fitness or evaluation function. A way to determine the overall quality of a particular solution generated by the algorithm.

One common method of SBPCG is evolutionary algorithms, which are loosely based on the Darwinian theory of evolution. One must keep a population of individuals, called Candidate solutions, which are evaluated using a fitness function at each generation. The fittest individuals in a generation get a chance to reproduce to produce child solutions in the next generation. Thus, there are two stages to a generation, selection and crossover.

When using an evolutionary algorithm, the population is made up of two components μ and λ. μ represents the fittest individuals that are passed between generations, called the elite. λ is the part of the population that is made up of new solutions produced through selection and crossover.

The steps of an evolutionary algorithm are as follows:

1. Initialise a population of random individuals.
2. Evaluate all solutions by a fitness function.
3. Sort the population in order of fitness.
4. Remove the least fit individuals.
5. Create new individuals to fill removed ones through selection, crossover and/or mutation of elite solutions.
6. If the population contains an individual of sufficient quality or the maximum number of generations is reached, end the search, and return fittest individual. Else, return to step 2.

Content representation is one of the most important issues to evolving game content. It plays a key part in the efficiency of the algorithm and defines the constraints on the search parameters. The solutions encoded in the algorithm during the search are encoded in what is called a genotype. This provides efficiency as, in the context of PCG, they do not need be created into the physical 3D space. A genotype can later be converted into what is known as the phenotype, which is they content one is trying to create. An example of a representation, which is used to generate levels in Infinite ‘Mario Bros’, is to represent a solution as an array of integers, which represents a combination of features to optimise.

Each candidate solution in a generation much be evaluated using a fitness function. This assigns a score or fitness value to the candidate solution. This should be designed to achieve characteristics in the content that is desirable to the creator. This therefore depends greatly on what one is trying to create with the PCG content. However, one must understand the objective of the search, and how it can be represented and evaluated to achieve it.

Search based method have emerged as a key topic in recent times for research in to PCG. However, in the context of mazes, one must understand the current or generation algorithms and how they work to know how to represent and evolve solutions.

Depth First Search (Recursive Backtracker)

This algorithm starts with a grid of maze cells, where each cell has walls on every side.

1. A cell is chosen as the starting point.
2. A random unvisited neighbouring cell is chosen, and the walls are destroyed between them.
3. From the next cell another unvisited neighbouring cell is chosen, and the process is repeated.
4. Once we reach a cell that has no unvisited neighbours, the algorithm backtracks until it finds a cell which has unvisited neighbours, and the algorithm continues.
5. The algorithm terminates when all cells have been visited.

Prim’s Algorithms

The algorithm also starts with a grid of maze cells, as with the Recursive Backtracker method.

1. One defines two lists, one for visited cells, the other for unvisited cells. All cells start in the list of unvisited cells.
2. A cell is chosen as the starting point and is removed from the list of unvisited cells and added to the list of visited.
3. One then selects a random cell from the list of visited and then detects a random neighbour, destroys the wall between them and adds the next cell to the list of visited cells.
4. We repeat step 3 until all cells are in the list of visited cells.

Recursive Division

The algorithm starts with a gird of cells with no walls except the outer walls of the maze.

1. The grid is divided randomly by a wall and a pass is put between them.
2. One then selects the smallest part of the gird while the other is put in a list.
3. Repeat from step 1.
4. The algorithm terminates when the width or heigh of each part is one.

**References:**

Noor Shaker, Togelius, J. and Nelson, M.J. (2016). *Procedural Content Generation in Games*. Cham Springer International Publishing Imprint: Springer.

Wikipedia Contributors (2019). *Maze generation algorithm*. [online] Wikipedia. Available at: <https://en.wikipedia.org/wiki/Maze_generation_algorithm>.

Kozlova, A., Alexander Brown, J. and Reading, E. (2015). Examination of Representational Expression in Maze Generation Algorithms. *IEEE CIG 2015*.