Procedural Maze Game Generation

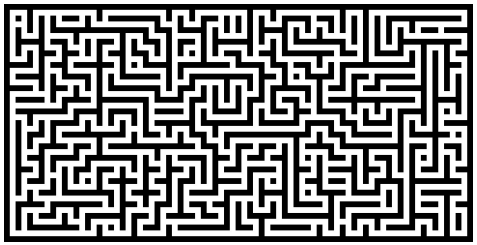
Jason Wang

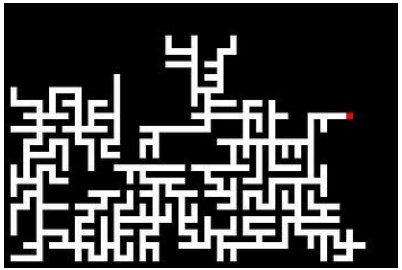
https://puzzlescriptgallery.tumblr.com/

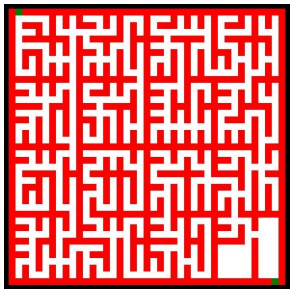
Maze Generation

- DFS Based Algorithm O(n + m)
  - Pros: Fast, easy to implement
  - Cons: Creates biased mazes with very long corridors
- Randomized MST Based Algorithm O(mlgn)
  - Pros: Fast, easy to implement
  - Cons: Creates biased mazes with short corridors
- Recursive Based Algorithm (runtime depends on implementation)
  - Pro: Creates non biased mazes
  - Cons: Mazes tend to look like grids

https://en.wikipedia.org/wiki/Maze_generation_algorithm

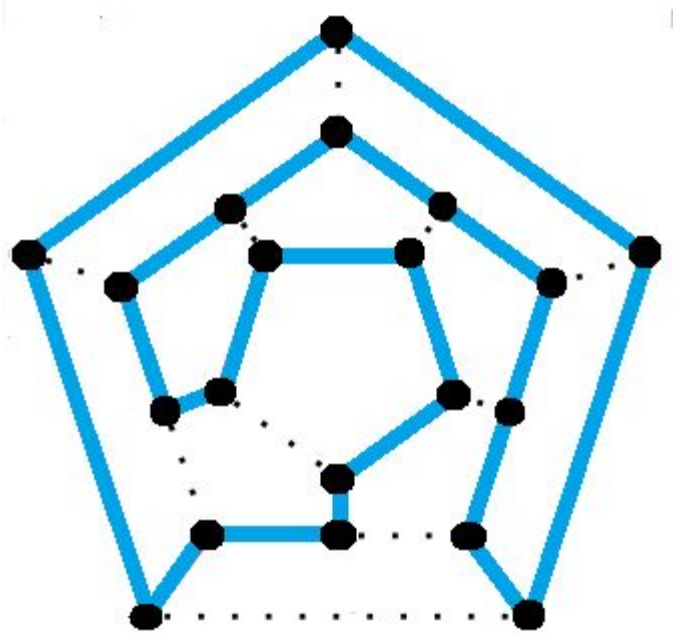


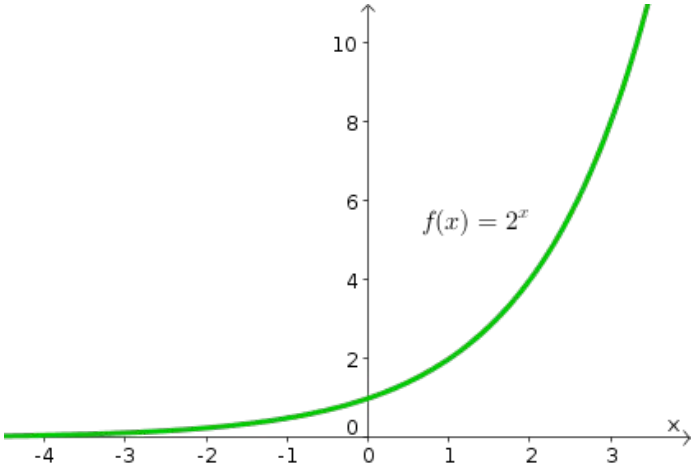




https://www.youtube.com/watch?v=9b67y24r6w4

The not so simple problem of simple paths...





Flow Free

https://en.wikipedia.org/wiki/Flow_Free

https://en.wikipedia.org/wiki/Numberlink

However, this problem is still NP-Complete

Example Restrictive Algorithm:

https://stackoverflow.com/questions/12926111/what-to-use-for-flow-free-like-game-random-level-creation

https://www.youtube.com/watch?v=hoCfdfRttT8

Algorithm

1. Generate a (m x n) grid that procedurally places boulders on the grid with some defined threshold (t)
2. Generate a random starting point (s) and ending point (e) on the grid
3. Run a modified BFS starting from (s) until it reaches point (e)
    a. Modified BFS travels edges in one direction until a boulder or a wall is reached
4. If (e) is unreachable from (s), repeat steps 1 - 3 until a valid map is generated

To make the map more interesting, we can save the path generated from the algorithm and add additional boulders to defined ‘safe spots’ on the map

Notes

- Runs in O(kx) where x = number of grid cells and k is the number of tries needed to make a valid map
- Algorithm is technically not deterministic and can run forever
  - The value of k can be infinite
  - However for small grids (m, n < 20) and a small boulder threshold (threshold < 15%) this should not pose a problem on runtime as there would be many possible solutions
- Assume k is small, algorithm does not run in exponential time since we do not need to deal with the issue of simple paths

Stretch Goals

Questions?