Thomas Fiorilla, Anthony Tiongson

Professor McMahon

CS440 Intro to AI

February 22, 2020

Assignment 1 Search and Genetic Algorithms

Environment

A randomly generated, n-by-n puzzle, where the value n can be 5, 7, 9, or 11, and there exists a legal move from every non-goal state. The row and column indices are $(r_{min}, \ldots, r_{max})$ and $(c_{min}, \ldots, c_{max})$, so for a puzzle where $n = 5$, $r_{min} = c_{min} = 1$ and $r_{max} = c_{max} = 5$. The starting position of the puzzle is the upper left most position $r_{min} \times c_{min}$, or 1×1, and the goal position is the lower right most position $r_{max} \times c_{max}$, or $n \times n$. There is a value on every position of the board representing a legal move, with the goal position always having a legal move value of 0. There is no guarantee that the puzzle is solvable.

There is one agent tasked to solve the puzzle. The puzzle is static once generated. The next state of the world is determined by the puzzle and the agent's action related to the available legal moves that agent's current position has. Every subsequent puzzle generated is an independent problem for the agent to solve.

Thus, this is a fully observable deterministic episodic static environment with discrete states for a single agent.

State Space

The puzzle is represented by the two-dimensional matrix $boardBuilt[i][j]$ where i represents a position in the row range r_{min} - $1 = 0$ to r_{max} - $1 = n$ - 1 and j represents a position in the column range c_{min} - $1 = 0$ to c_{max} - $1 = n$ - 1. The values stored in the array $boardBuilt[i][j]$ each represent a legal move an agent can make at that particular i-th

row and j-th position of the puzzle. A legal move in the puzzle is an integer value representing the amount of positions the agent can move in one direction either vertically or horizontally within the bounds of the puzzle, so the minimum amount of legal moves per position is always 1 and the maximum can be 4 if its position is not located at the edge of the puzzle. The value of the legal move ranges from 1 to n - 1 for positions located at $boardBuilt[0][0]$ to $boardBuilt[0][n - 1]$, $boardBuilt[n - 1][0]$ to $boardBuilt[n - 1][n - 1]$, $boardBuilt[0][0]$ to $boardBuilt[n - 1][0]$, and $boardBuilt[0][n - 1]$ to $boardBuilt[n - 1][n - 1]$; these are visually the outermost positions of the puzzle. For the positions representing the next inward layer of the puzzle, positions $boardBuilt[1][1]$ to $boardBuilt[1][n - 2]$, $boardBuilt[n - 2][1]$ to $boardBuilt[n - 2][n - 2]$, $boardBuilt[1][1]$ to boardBuilt$[n - 2][1]$, and $boardBuilt[1][n - 2]$ to $boardBuilt[n - 2][n - 2]$, the values of the legal moves range from 1 to n - 2. This trend continues for every subsequent layer inward, up until the center position $boardBuilt[\lfloor\frac{n}{2}\rfloor][\lfloor\frac{n}{2}\rfloor]$ is reached where the legal moves are within the range 1 to $\lfloor\frac{n}{2}\rfloor$.

The puzzle starting position is always at $boardBuilt[0][0]$ and its goal position is always at $boardBuilt[n - 1][n - 1]$; the goal position will always have a legal move value of 0 as to represent the final desired destination. If $\mathbf{S}$ is the set of all states, then any state $\mathbf{s}$ where $\mathbf{s}\in\mathbf{S}$ is the combination of the array $boardBuilt[i][j]$, the current position of the agent, and the available moves for that agent's position.

Actions

The agent is able to move in one direction horizontally or vertically from its current position in the puzzle to the number of positions its current position's value holds. The direction an agent moves is legal if the move can resolve within the bounds of the puzzle. For instance, if the agent is currently at $boardBuilt[1][1]$ with a value of 2, the agent can move two positions below or two positions to the right, or in other words

to $boardBuilt[3][1]$ or to $boardBuilt[1][3]$ respectively. For each subsequent action, the agent repeats this process until it can successfully reach the goal position.

If $\mathbf{A}$ is the set of all possible actions, then any action $\mathbf{a}$ where $\mathbf{a} \in \mathbf{A}$ is any legal move an agent can make at any position of the puzzle.

Perception/Observations

The input the agent receives is the array $boardBuilt[i][j]$. If $\mathbf{O}$ is the set of all possible observations for a given puzzle, an observation $\mathbf{o}$ where $\mathbf{o} \in \mathbf{O}$ is any observable path an agent can make from the starting position $boardBuilt[0][0]$ to any possible position on the puzzle including a path to the goal position $boardBuilt[n - 1][n - 1]$.

Transition Function

Any agent action $\mathbf{a}$ during any state $\mathbf{s}$ for a transition function τ gives a new state $\mathbf{s}'$, or $\mathbf{s}' = \tau(\mathbf{s}, \mathbf{a})$. The next state of the puzzle depends on which legal move the agent takes. Once chosen, the current position of the agent and its current available legal moves are updated to reflect the new state.

Evaluation Metric

The first evaluation an agent will perform is to verify if a particular randomly generated puzzle is solvable using a breadth-first-search analysis of the array $boardBuilt[i][j]$ to find a path from $boardBuilt[0][0]$ to $boardBuilt[n - 1][n - 1]$. If the puzzle is unsolvable, the agent will set the value of the puzzle as $-k$, where k is the number of cells not reachable from the start. If the puzzle is solvable, the agent will set the value of the puzzle to the minimum distance from the start to the goal. Puzzles can then be classified by these values: negative values are not solvable, low positive values are easy or simple, and high positive values are more difficult.