COMP2611

Artificial Intelligence

Assignment 1: Search Algorithm

A. Sliding Blocks Puzzle Search Investigation

A1(a) Puzzle Test Cases

**A1(b) Heuristics**

We designed the following two heuristics for our investigation and testing:

**Preprocessing Step**

First, we preprocessed the data by grouping blocks of the same color and determining their geometric centers. These geometric centers serve as anchor points for heuristic calculations.

**Manhattan Distance**

The Manhattan Distance heuristic calculates the sum of horizontal and vertical distances between each block and its target position.

Formula:

Algorithm:

*def manhattan\_heuristic(state, goal\_anchors):*

*initialize total\_distance to 0*

*compute state\_anchors from the current state*

*for each block in state\_anchors:*

*if block exists in goal\_anchors:*

*total\_distance += |p1.row - p2.row| + |p1.col - p2.col|*

*return total\_distance*

**Straight-Line Distance (Euclidean Distance)**

The Euclidean Distance heuristic calculates the direct straight-line distance between a block and its target position.

Formula:

Algorithm:

*def straight\_line\_distance(state, goal\_anchors):*

*initialize total\_distance to 0*

*compute state\_anchors from the current state*

*for each block in state\_anchors:*

*if block exists in goal\_anchors:*

*total\_distance += sqrt((p1.row - p2.row)² + (p1.col - p2.col)²)*

*return total\_distance*

This approach ensures efficient heuristic calculations based on geometric centers, improving accuracy while maintaining computational efficiency.