Greedy and A* search for Sokoban

Truong Minh Chau – 19521281

1. Heuristic

1.1. Zero Heuristic

This design always returns 0 and makes A* search become UCS, so it is admissible that A* search can give an optimal solution.

1.2. Manhattan Distance: Boxes and Goals

This heuristic design is to calculate the minimal sum of distance between boxes and goals. It is considered to be an admissible or consistent heuristic because the estimated cost is never greater than the actual cost of the shortest path from node n to a goal.

2. UCS

Uniform-cost search is uninformed search. It doesn't use any domain knowledge. It expands the least cost node, and it does so in every direction because no information about the goal is provided. So the runtime and memory took to solve problems are long and large.

3. Greedy search

In Greedy search, the evaluation function used is a heuristic function. This algorithm tries to expand the node appears to be closed to goal and not explores all possible states. Thus, it is not optimal and there is always the risk to take a path that does not bring to goal.

4. A* search

A* search is an optimal algorithm, that is, it always finds the optimal path between start states and targets if an admissible heuristic function is used in its evaluation function. Memory is still a big problem, although runtime is better than UCS. And with different heuristic, the time and memory may vary to solve Sokoban.

5. Comparison

In summary, if we want to get an optimal solution has a faster runtime than UCS, A* search is the great option. However, A* uses more memory than Greedy and

A* becomes impractical when the search space is huge. Furthermore, A* get a failure in finding an optimal solution if the heuristic is inadmissible. Which algorithm is the best up to context’s problem so we should consider carefully before opting for any algorithm.