Move the Block Problem

# Problem Definition

The problem is made up of 3 components,

## Goal Block:

The block that needs to reach the goal state.

## Hurdle Blocks:

The blocks that are in the way of Goal Block and Goal State.

## Empty Spaces:

The spaces in which blocks can be moved.

# Environment

The environment of the game is fully observable and deterministic in nature.

The Environment can be represented through a grid. Where each block is 1x2 in grid size, either horizontally or vertically.

# Gameplay

To define the gameplay for the agent,

* We would start with an initial state and a goal state.
* Each slide would have a cost of 1.
* There would be a goal test function that would see if the goal block is at the goal state.

# Heuristic

We can use the Manhattan distance heuristic for this problem, which will see the distance that goal block is from the goal state and use that to determine what the best moves would be in an informed search.

# BFS and UCS:

* The algorithm dequeues a node from the frontier.
* For each valid action in the current state, a new state is generated, and the resulting node is enqueued with an increased cost and updated action sequence.
* The algorithm continues to explore nodes level by level, considering all possible actions in the current state before moving on to the next level.

# DFS:

* The algorithm pops a node from the top of the stack.
* For each valid action in the current state, a new state is generated, and the resulting node is pushed onto the stack with an increased cost and updated action sequence.
* The algorithm continues to explore as deeply as possible along each branch before backtracking.

# Greedy Search:

* The algorithm iteratively selects the node with the highest priority (lowest heuristic value) from the priority queue.
* For the selected node, valid actions are considered, and new states are generated by applying these actions to the current state.
* Each new state is assigned a priority based on the heuristic function, and the resulting nodes are inserted into the priority queue.

# A-Star Search

* The algorithm iteratively selects the node with the lowest priority (lowest total cost) from the priority queue.
* For the selected node, valid actions are considered, and new states are generated by applying these actions to the current state.
* Each new state is assigned a priority based on the sum of the cost to reach that state and the estimated cost from that state to the goal, and the resulting nodes are inserted into the priority queue.

# Most Optimal Algorithm:

In the case of this specific problem, the BFS and UCS algorithms would have the same cost and nodes explored because the cost of each action is the same.

* The answer would be complete but not optimal.

Depth First would reach a solution with a low cost but would explore the most nodes.

* The answer would be complete but not optimal.

Greedy Search would reach an answer relatively quickly with cost less than BFS and UCS and exploring lesser nodes than DFS but the heuristic, if changed, may cause a lot of backtracking.

* The answer would be complete and optimal.

A-Star Search would likely reach the same conclusion as the greedy search because cost remains same and only deciding factor is the heuristic.

* The answer would be complete and optimal.