Block World Problem Report

Xinghao Chen
xchen785@gatech.edu

Abstract—This work attempts to solve the BlocksWorld problem with a heuristic method. Analysis and strategies are established to count how many subgoals a move can potentially fulfill. Full score is achieved in the test with this simple but efficient method.

1 HOW TO REACH THE GOAL?

Let us first discuss what moving a block is.

1.1 The prerequisites of moving block A onto B

- Nothing is over A. (A is movable)
- Nothing is over B. (B is movable)

1.2 The impact of moving block A onto B

- Allows the block below A, and **potentially the deeper underneath blocks** to be moved.
- Changes the position of A.
- Prevents B to be moved.

For the first consequence, we can score the impact of moving A by counting how many blocks beneath A needs moving, or needs to become the destination of a move.
For the following two types of impacts, we can carry out two rules in order to eliminate the impacts and reduce complexity. We can identify which blocks are already in their correct position (and do not need any more moves). We only pile new blocks onto exsiting stacks where all the blocks are in correct position. The new block onto the stack should also reach its final position and never be moved again. On the other hand, we only put blocks onto the table if a block has to be moved but no block can be put at the final position with this move.
Furtherly, with respect to the rule that only blocks on the top can be moved, and in order to reach the goal, we can deduce the following rules:

- Blocks that are closer to the table in the goal state must be correctly placed with priority.

- To finally put A on B, we shuold first make A movable, and also clear everything on B, which also means making B movable.

2 HOW DOES MY AGENT WORK?

My method is heuristic, which means that a move can be decided only with the current state and the goal. Only 2 types of strategies are involved.

- Find whether any block can reach its correct final position with this move. Take the move immediately if a block qualifies.
- Find a movable block that has the most number of blocks that are not in the final position, or must serve as a destination of a move. Move the block onto the table.

With the second strategy, we are actually trying to meet the prerequisites of the first strategy. The two strategies requires subgoals to indicate which blocks needs to be moved. Therefore, we should try to describe the final goal before we move anything. We should track a set of blocks that are in their final correct position, and the other blocks that needs to be moved. The must-move blocks are registered as subgoals, which are described as which block should finally be on which block. Then we follow the two steps above to decide a move. When no block can be placed onto the final position with the following move, we count the number of blocks that needs to be moved in each stack, and move the block on the stack with the most counts (or "score") onto the table.
From my personal perspective, the two rules do not stand as typical Generate & Test or Means-Ends Analysis, but the ideas of subgoals and testing the stack with the highest score truely originate from what had been taught in class.

3 PERFORMANCE AND ANALYSIS

Using a heuristic method with very simple rules, my agent got full score in the test, and did not struggle on any case. The time complexity is proportional to the number of stacks, the number of blocks, and the number of steps to reach the goal. The linear time cost does not produce overwhelming compitational burden when the problem has an extremely large scale. A flaw in my program is that I search for a given block by traversing all the stacks. This can be optimized by building an index of block postion.
My agent is not really like a human. It can be difficult for humans to manage many subgoals and compute the score of every stack.