CS440 MP2 Report

Andong Jing ajing2 - 3 credit In charge of

Siping Meng smeng10 - 3 credit In charge of

Siyu Tao siyutao2 - 3 credit In charge of

# 1. Smart Manufacturing

## 1.1 Planning Using A\* Search

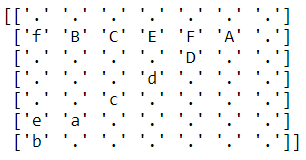
## 1.2 Planning Using a Planning Graph

## 1.3 Extra Credit

# 2. Game of Gomoku (Five-in-a-row)

## 2.1 Reflex Agent

* Reflex vs. reflex result



## 2.2 Minimax and Alpha-Beta Agents

* Implementation of minimax and alpha-beta search

1. **Evaluation Function**

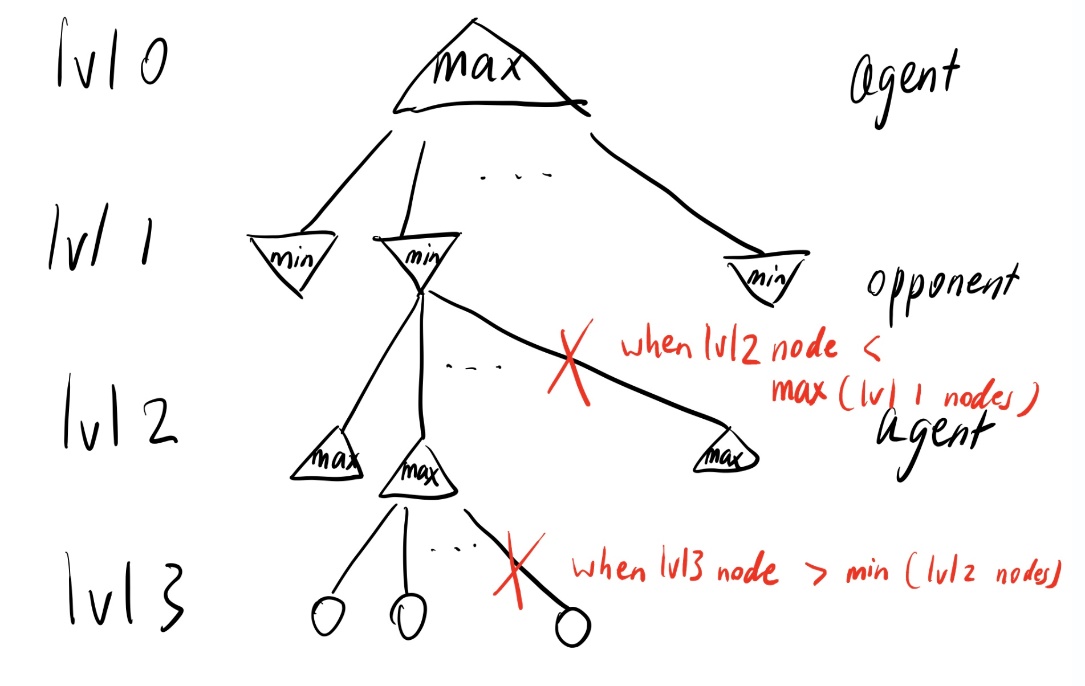
The evaluation function of our group searches for every “winning block”, as defined in rule 4 of the reflex agent, for both the agent side and the opponent side. An empty board has 21 (horizontal) + 21 (vertical) + 18 (diagonal) = 60 such winning blocks for each side.

For each agent’s winning block, if a winning block contains 1 agent’s stone, 1 point is assigned. If a winning block contains 2 agent’s stones, 10 points are assigned. If a winning block contains 3 agent’s stones, 100 points are assigned. If a winning block contains 4 agent’s stones, 1000 points are assigned. If a winning block contains 5 agent’s stones, 10000 points are assigned. For each opponent’s winning block, if a winning block contains 1 opponent’s stone, -1 points is assigned. If a winning block contains 2 opponent’s stones, -10 points are assigned. If a winning block contains 3 opponent’s stones, -100 points are assigned. If a winning block contains 4 opponent’s stones, -1000 points are assigned. If a winning block contains 5 opponent’s stones, -10000 points are assigned. An easy-to-read table is included as follows.



2. **Alpha-beta pruning**

Our team recognized two opportunities of alpha-beta pruning in three levels of minimax search.



As shown by the diagram above, a level three node could be pruned if its value (according to evaluation function) is larger than the minimum of all level two nodes since each level one node is going to minimize its collection of level two nodes. In addition, a level two node could be pruned if its value is less than the maximum of all level one nodes since the root node is going to maximize all level one nodes.

For level three pruning, we keep track of the minimum of all level two nodes each time a node gets added to level two. Once we discovered a level three node that’s larger than that value, we stopped current level three node expansions and continue with the next node in level two to expand. Level two pruning is conducted in a similar manner.

* Match-up results

1. **alpha-beta vs. minimax**



2. **minimax vs. alpha-beta**



3. **alpha-beta vs. reflex** (alpha-beta won, igfhj)



4. **reflex vs. alpha-beta** (alpha-beta won, HDCIJ)



5. **reflex vs. minimax** (minimax won, HDCIJ)



6. **minimax vs. reflex** (minimax won, igfhi)



* # of nodes expanded (minimax vs. alpha-beta)





* Relationship of # of nodes expanded (minimax vs. alpha-beta)

From the tables above and our group’s experimentation with different combinations of matchups, we discovered that at any particular move (say move 2), the minimax agent always expands the same number of nodes. It is because when there are n available positions on the board, the agent always expands n\* (n-1) \*(n-2) nodes. As a result, the number of nodes expanded decreases as the game goes on. The alpha-beta agent always expands less nodes than (strictly less than or equal to) minimax agent at any particular move. The number of nodes expanded doesn’t necessarily decrease as the game goes on. It’s because sometimes early on when there are more possible agent-opponent-agent combinations on the board, the pruning might be more effective, cutting more unnecessary nodes to expand. However, alpha-beta is still always better than minimax in terms of efficiency, while still yielding the same results.