Scott Hoge (1,3,5,7)
Dusty Argyle (2,4,6,7)
October 20, 2016
CS 4300 - Artificial Intelligence

A4-b

1. Introduction

Assignment A4-B was to get a working Hybrid Wumpus Agent. This agent uses resolution theorem proving to “solve” the Wumpus Worlds. A solvable board is one where the Agent can succeed by finding the gold, grabbing it, and making its way to the exit. The resolution theorem proving is used to determine what to do next. Some interesting questions are whether or not the Hybrid Wumpus Agent will be able to solve boards 1, 2, and 5?

2. Method

We assigned logical values as described:

- 1-16 Wumpus: where 1 is Wumpus in (1,1) the bottom left and -4 is (4, 1) not a wumpus in the bottom right
- 17-32 Pits: where 17 is Pit in (1,1) the bottom left and -20 is (4, 1) not a pit in the bottom right
- 33-48 Gold: where 33 is Gold in (1,1) the bottom left and -36 is (4, 1) not gold in the bottom right
- 49-64 Stench: where 49 is a stench in (1,1) the bottom left and -52 is (4, 1) stench in the bottom right
- 65-80 Breeze: where 65 is breeze in (1,1) the bottom left and -68 is (4, 1) not a breeze in the bottom right
- 81-97 Bump: where -81 is not a bump in (1,1) the bottom left and 84 is (4, 1) bump in the bottom right
- 97-113 Scream: where 97 is Scream in (1,1) the bottom left and -100 is (4, 1) not a scream in the bottom right

These numerical values are used in the knowledge base for the resolution theorem prover to make logical decisions. Initial rules are placed into the Knowledge Base everytime it is empty via the Initialize_KB method.

The first part of the Wumpus agent is to turn the percepts into sentences. This is done by taking the percept, the logical values described above and the agent’s current position to determine the values perceived by the agent. These values are then added to the Knowledge base via the Tell function. This function adds the sentence to the knowledge base if it isn’t already there.

The next part of the agent checks if the agent has stumbled across the gold. If the agent has found the gold, a plan is made to grab the gold and find a way out. This path is determined by calling A* search back to coordinate 1,1 on the board. Then the agent will climb out.

If the agent hasn't found the gold yet, the plan queue is checked for any remaining actions. If there is actions, they are popped from the queue and returned. If the plan queue is empty, the RTP portion of the agent is used to determine safety for potential moves on the board. The function Get_Fringe returns the coordinates around the current agent location with the unvisited locations prioritized first.

For each of these fringe locations, the KB is asked if there is a pit or a wumpus there. If there is not a possibility of an obstruction, the location is then explored. If there are no fringes that are known to be safe, the agent will have to choose a random action to execute between the three actions.

3. Verification of Program

For the verification of the program I did boards one and five by hand, see boards by hand compared to the ones from our agent.

4. Data and Analysis

Please see the appendix for the board images.

5. Interpretation

In the end our Wumpus ended us working just as planned. The Wumpus was able to do the first and second, but failed on the fifth.. This was because the fifth board did not provide the Wumpus with a full enough picture when he spawned. If the agent had known the board ahead of time then he could have succeeded, but we went in only knowing that death was eminent. It was cool to see him work on boards one and two, kind of a Frankenstein moment "It's Alive!"

6. Critique

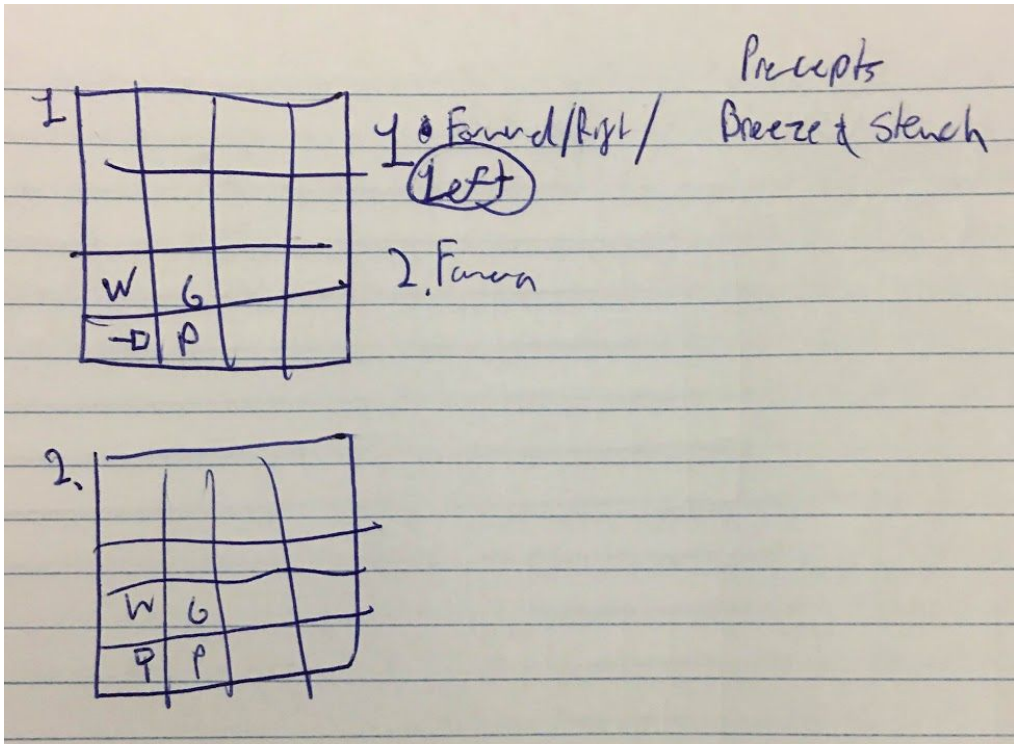
I gained a better understanding of the concept of agent implementations using logic. Logic agents can be really slow due to the constant querying of the KB. I learned that there can be many small improvements made to the agent implementation as well. While this may increase the efficiency of the implementation, it is still expensive to query the knowledge base and these queries are required if a plan is not currently being executed. There are ways to speed up the implementation by saving a known board as you retrieve information, from asking RTP questions. Using this board to do initial logical testing can save time if you have already established what is there. I found that I had to keep a visited board and a known game board so

that I could determine if I knew what was going on. I decided to keep it simple and only do those two boards. I could have kept a pit board or a wumpus board that I could check before to see if there was an obstruction in order to increase the performance, but I would rather have a working agent then a fast one.

7. Log

Dusty: 17 hours Algorithm, 3 hours report
Scott: 2 hours Algorithm, 2 hours report

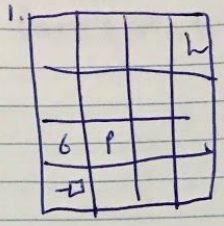
8. Appendix
Verification:



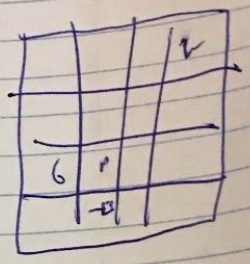
1. Forward
2. Left
3. Left
4. Forward
5. Right
6. Forward
7. Grab - glider
8. Left
9. Left
10. Forward
11. Climb

Precepts
none
~~Breeze~~

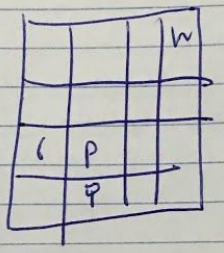
1.



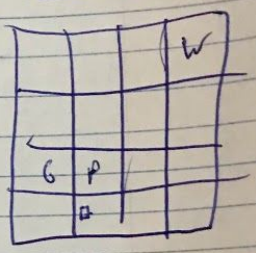
2.



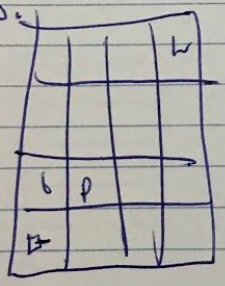
3.



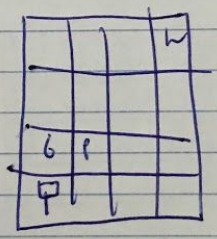
4.



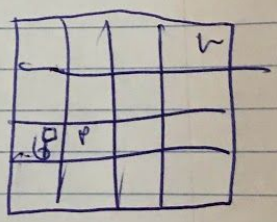
5.



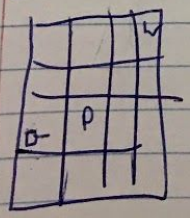
6.



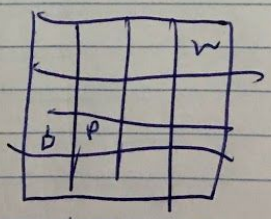
7.



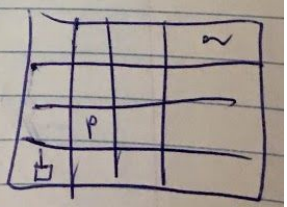
8.



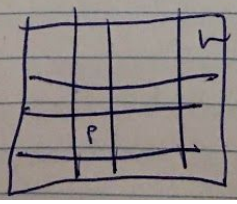
9.



10.

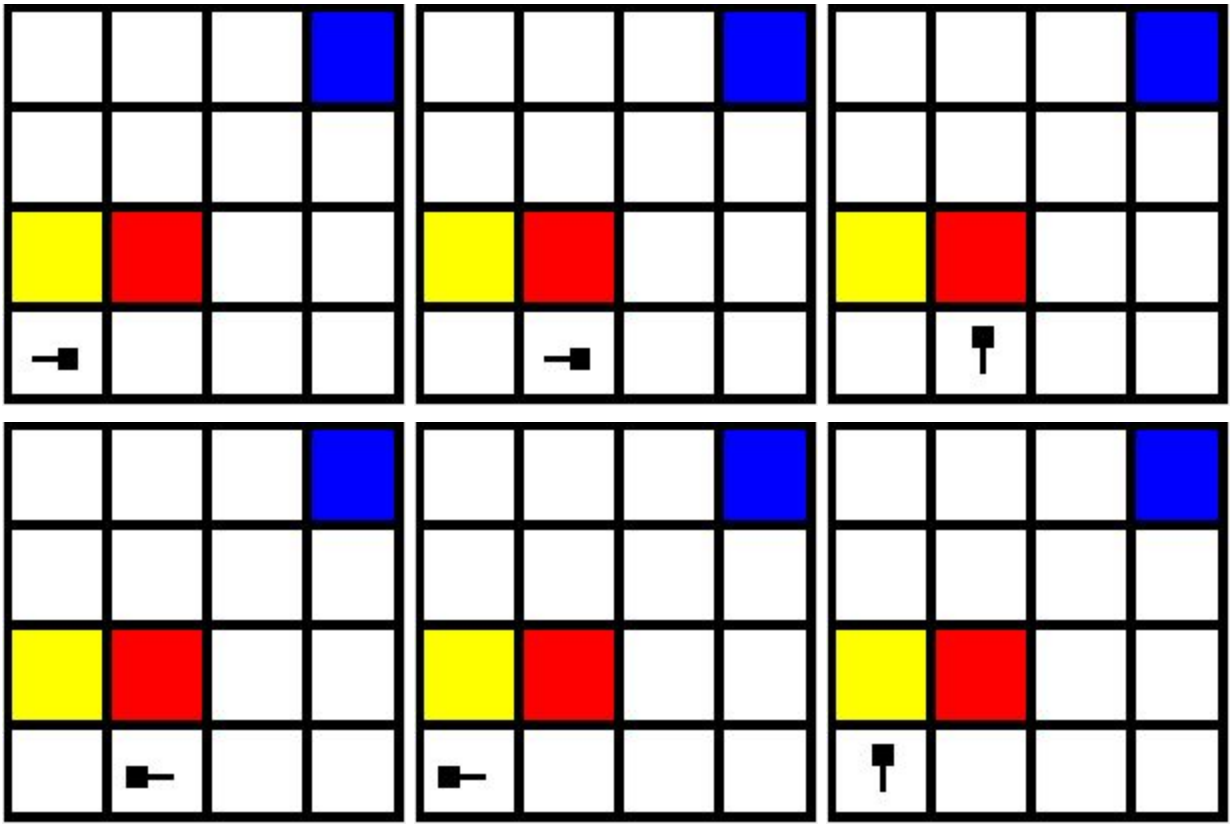


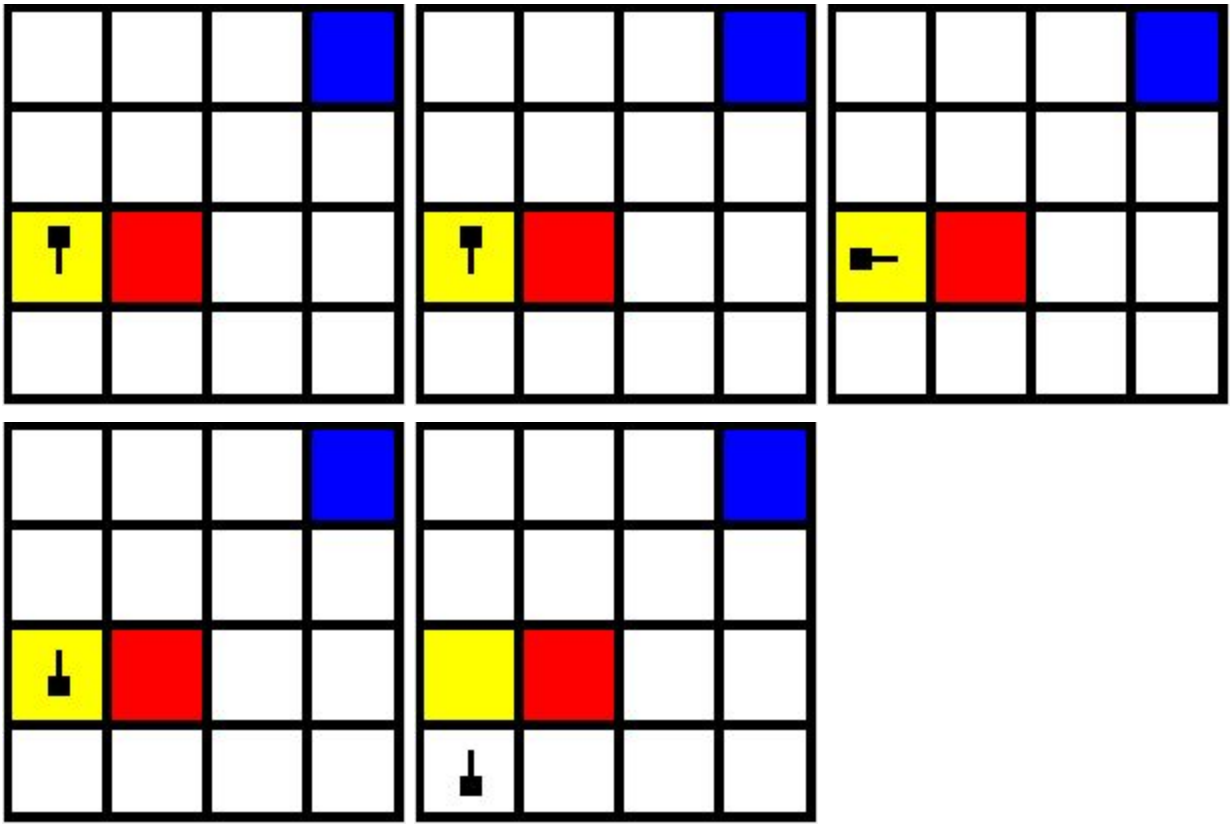
11.



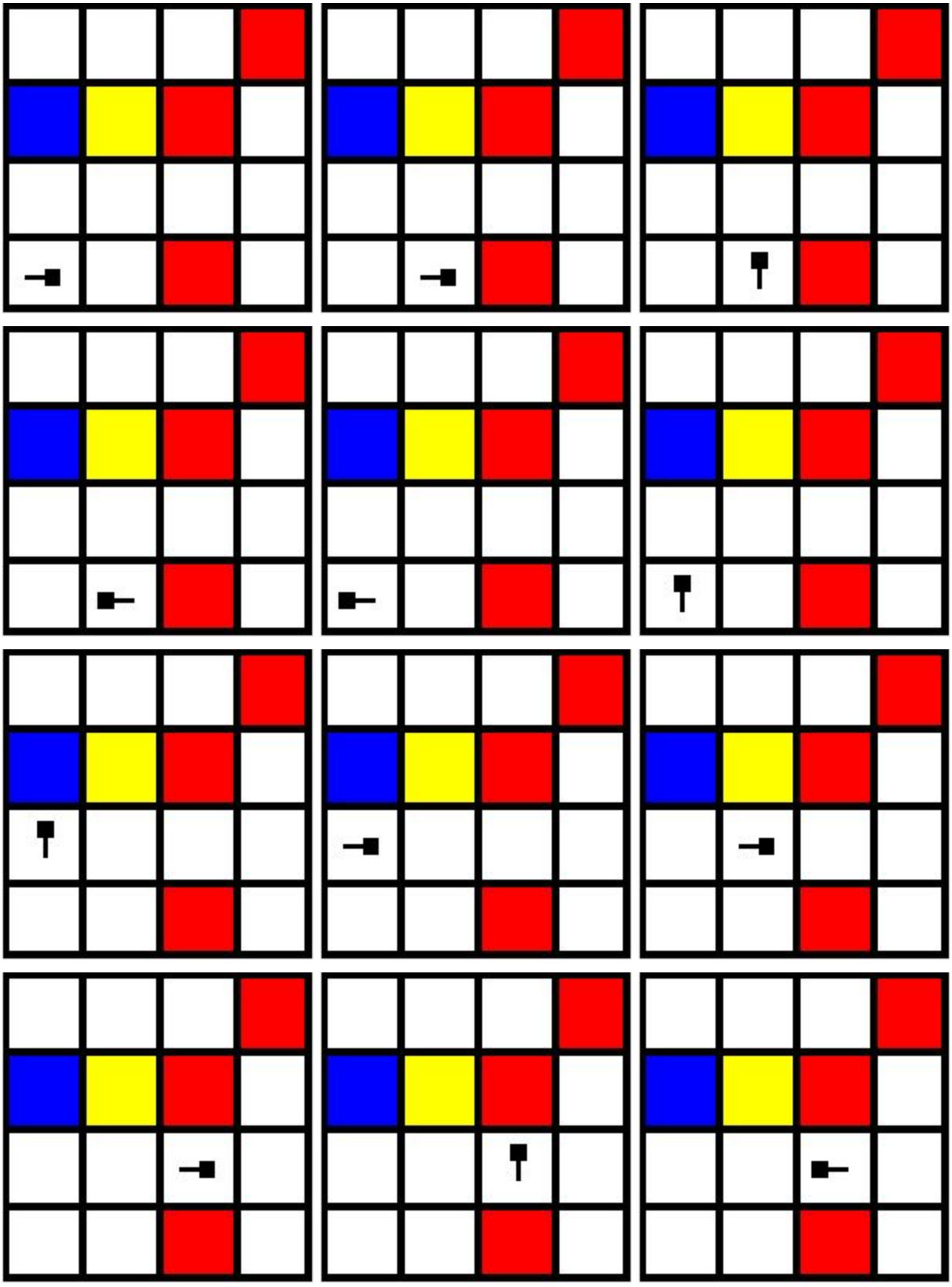
Data:

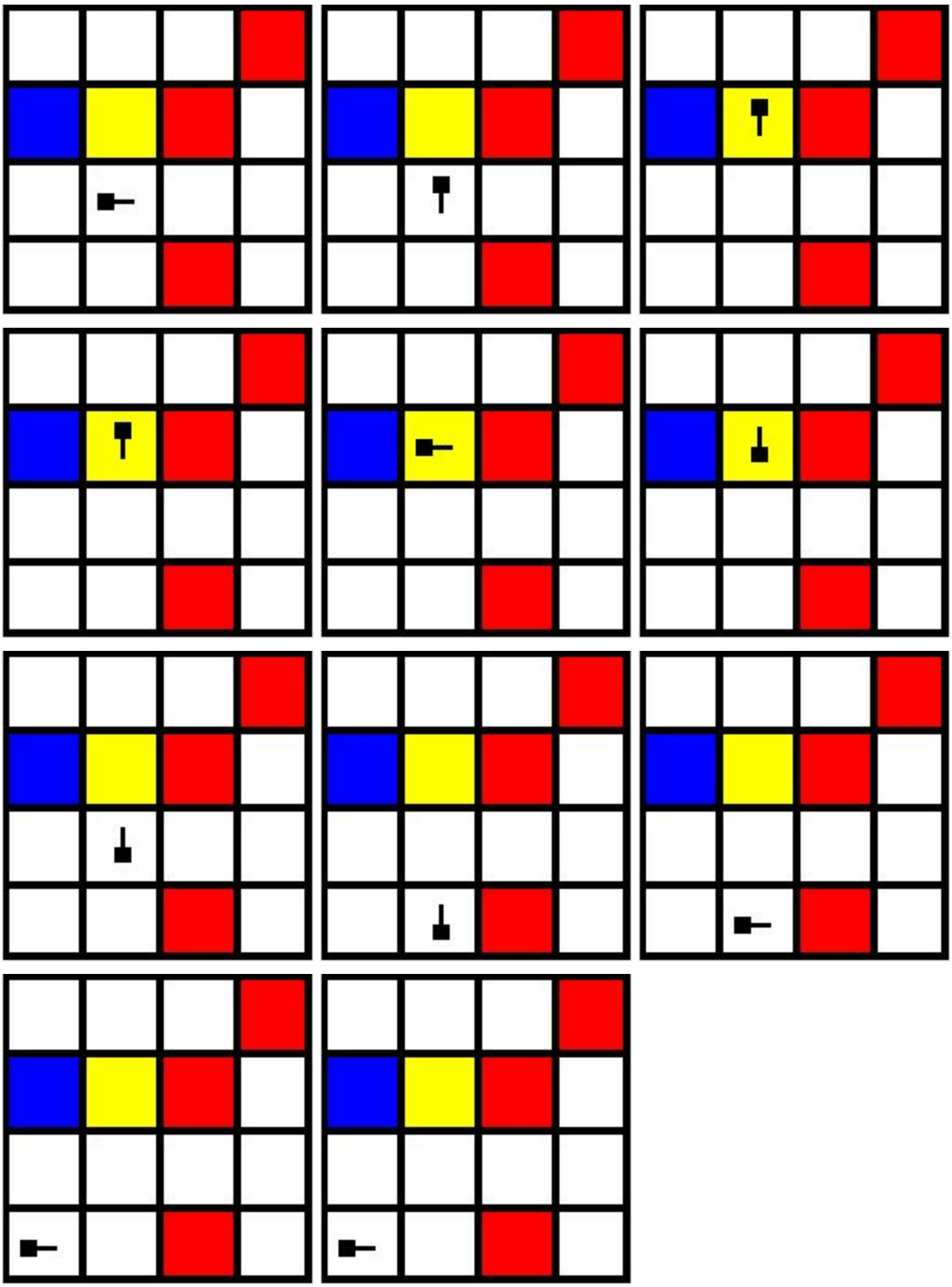
Board 1:





Board 2:





Board 3 (Random actions):