COM S 572
Principles of Artificial Intelligence
Report
Lab 2

Name: Monoshi Kumar Roy **Email:** monoshi@iastate.edu
Student Id: 375360043

Question 6: To answer the question, I have implemented two AI agents to play Othello using weighted piece counter heuristic[2,3] and the two AI agents played the game of Othello using the same evaluation function. Now, for this function, using the alpha beta cutoff search, I have found that by increasing the search depth, the performance of the agent can be improved. Here the performance means winning against the other agent opponent. However, there is an obvious tradeoff between time and winning the game. For increasing depth, the time for generating a move increased exponentially. So, it takes much longer to play the whole game if we increase the search depth. Table 1 shows the summarized result for different depths.

Table 1: Performance of two AI agents on different search depth

Agent 1 (Black)		**Agent 2 (White)**		**Winner**
Depth	**Time(sec)**	**Depth**	**Time(sec)**	
3	2.62	3	2.55	Agent 2
3	3.92	4	13.47	Agent 2
6	269.27	3	3.14	Agent 1
3	4.30	6	260	Agent 2
7	1432.45	3	3,58	Agent 1
7	1719.88	4	19.57	Agent 1
4	19.21	7	1718.22	Agent 2
8	16628	3	4.27	Agent 1

Table 1 shows that increasing search depth improves an agent's performance and helps the agent win the game. This improvement happens because it allows the agent to explore more possible game moves and gives a better-informed understanding of which move to take next. However, we can see from the table that by increasing the depth, the time of the agent to make a move is also increasing, which may not be feasible in playing a real-time game. We can generate the results by running the "**Othello_Comparision_6.py**" file, and for convenience, all my outputs are saved in the "**OutputFor6.txt**" file.

Question 7: To compare two evaluation functions, I have implemented "Number of Piece Differences" and "Weighted Piece Counter Heuristic". Some places in the Othello board are strategically valuable than other places, and occupying those places may ensure the win for an agent. For example, corner places are difficult to flip for the opponent and taking these places should be given more emphasis than other places. That's why "Weighted Piece Counter Heuristic" assigns weights to all 64 cells in terms of their strategic value. This heuristic works better than other simple heuristic which only counts and subtract the total different pieces (white and black) on the board. Because, "Number of Piece Differences" function doesn't consider the impact of choosing any strategically significant place. So, weighted heuristic is theoretically better and this was evident also after the implementation. I gave weighted heuristic to one agent and made another agent play with the normal difference function. Using the alpha beta cutoff search for the same depth, the agent with weighted heuristic function always beats the agent with normal difference function. Again, performance and time tradeoff exist here. Weighted piece counter heuristic requires more time to generate moves. However, here, I have mentioned the same search depth. If I increased the search depth for the difference heuristic function agent, it beat the agent with the better weighted heuristic function. So, it also supports my argument in Question 6. We can generate the results by running the "**Othello_Comparision_7.py**" file, and for convenience, all my outputs are saved in the "**OutputFor7.txt**" file.

References

1. https://github.com/aimacode/aima-python
2. https://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=10AB4B0966FEE51BE133255498065C42?doi=10.1.1.580.8400&rep=rep1&type=pdf.
3. https://courses.cs.washington.edu/courses/cse573/04au/Project/mini1/O-Thell-Us/Othellus.pdf