Assignment – 2

Foundation of Artificial Intelligence – CS2701

Name: Khushboo Thaker

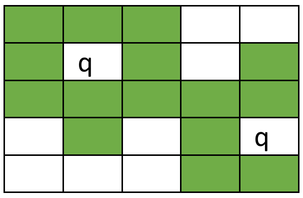
ID: 4147510

Email: kmt81@pitt.edu

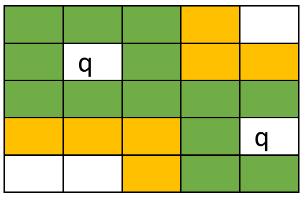
1. The start of the game has many possible moves. How does your program deal with the huge branching factor?

To deal with huge branching factor I have generated successor nodes in step:

1. First of all, the blocks adjacent to other player queens will be used to generate the moves. Also the program considers the blocks which can block the path of two of other player's queens together



2. If these child nodes are not enough (number of childnodes < n) (for example 20) (configurable)), then the blocks adjacent to the locations in first step will be included to generate child nodes



This will go iteratively till we get the enough child nodes

NOTE: Assumption is the blocks near the opposite queens are of prime importance

2. There is a time limit for the turn. What does your program do to make the most use out of the time you have?

To deal with time limit my program has modified minmax algorithm for iterative deepening – level based. And there is a timer which signals the minmax program at the end of 14 seconds and the program returns the latest minmax result in case it is not done with all the levels

3. How well does your automatic player compete against a human player?

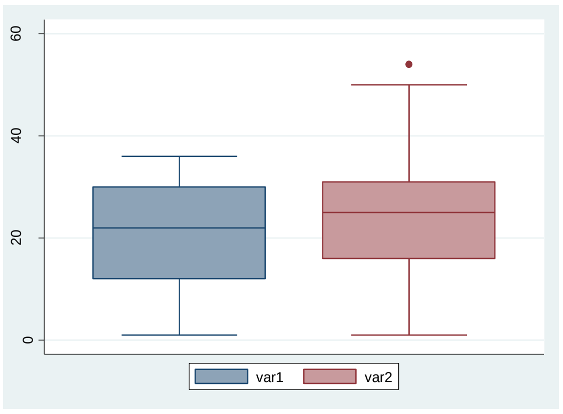
I tried playing with my game. But most of the time I won the game except the times if I missed an important point and played a wrong step. I believe, this is because my strategy is to not let the opposite player's queens come in the second half of the board towards my queens. (which heuristic is not able to take care of)

4. What different heuristic functions have you tried before settling on your final choice? What ideas worked? What didn't? Be sure to describe your final heuristic function in sufficient details.

The different heuristics I tried are as below:

1. Number of blocks reachable by the opposite queens
2. 1/ Number of blocks reachable by my queen
3. Number of adjacent blocks open for my queens
4. Number of adjacent blocks open for opposite player queen
5. 1/ (Number of blocks I am reachable – Number of black opposite player is reachable)
6. Random - surprisingly if the random queen's take too long steps they cause the heuristic to not work

I tried playing all the heuristics against random player for 100 runs and I found that the No of adjacent blocks heuristics wins most of the time although sometimes with large number of moves and very few blocks left (less margin). So I selected it to be the final heuristic



As can be seen in the left figure I have plotted the box plot with var1 being the win margin for heuristic 1 and var2 being the win margin for heuristic 4 and heuristic 2 has slightly better mean as well as good distribution towards higher values above 25.

Also with heuristic 1 my program wins against random player 90% time, while with heuristic 2 for 94% times

My final heuristic is 4) No of adjacent blacks open for opposite player queen. As my code is based on prime locations this heuristic makes more sense

Below is the example: Table 1

		q	q						
			Q						

For example, if at some time we have this condition on the board as shown in the left side. (I haven't displayed the remaining queens and the crosses as it is only for illustration purpose)

Possible Move1

		q	q						
		X	Q						

Possible Move 2

		q	q						
	x				Q				

So as the move 1 helps in blocking queen this would be given priority over the move 2

One of the heuristic can be to divide the board into halves and control the movement of the opposite queens in their halves but I think that is much more dependent on the initial state of the board so I did not select that heuristic. For example, all the queens are in same half or same line in extreme columns or rows

References:

Lecture Notes

Class Notes

https://en.wikipedia.org/wiki/Game_of_the_Amazons
Jens Lieberum, An evaluation function for the game of amazons, Theoretical Computer Science, Volume 349, Issue 2, 2005, Pages 230-244, ISSN 0304-3975, http://dx.doi.org/10.1016/j.tcs.2005.09.048.