Artificial Intelligence Assignment-1 Report

Jay Patel Net ID - jap751

October 26, 2016

0.1 UnInformed Search

1. **Breath First Search -** Surprisingly it was one of the better performing algorithms in this implementation. The average score was comes around 5000. It does not implement any of the heuristics still traverses through the entire level search for as many child nodes as possible and thus makes a decision to make a move in the Game space. It always looks out for all the node on the same depth level. Then moves on to the child node from that depth, sometimes it gets stuck at the extreme corner looking for other nodes on the same depth level for a long time. Reasons can be that the best move at this level may not lead to a way out it has to take two moves in the same direction to move ahead. The implementation is done with a Linkedlist and a FIFO logic.

2. **Depth First Search -** Gives expected result where the average score has been around 800. It randomly selects one child node and keeps skipping one depth level at a time searching for child nodes. Because of this short-sitedness it traverses to the end of the frame from where it starts and then retracts back/taking a different move. Thus it always has same limited path to traverse even it does it with multiple chances. The only difference in implementation from BFS was it implemented a LIFO logic. Interestingly we depth limit the algorithm the performance increases significantly where the average score comes out to be 2000.

3. **Iterative Deepening -** It is basically the DFS where it iterates every time the goal state is not achieved within that depth limit and increases the limit and carries out the same procedure from stratch. The average score sometime exceeds the score of BFS which should be the expected behaviour normal cases. While it gets stuck with a similar behaviour of DFS after sometime in the game. Implementation wise the only difference in logic with DFS is it starts deepening when the goal state is not achieved and the stack is empty. Its starts the procedure of DFS all over again.

0.2 Informed Search

1. **A * Tree Search -** It adds all the nodes in the frontier not depending upon the depth level. It depends the sum of the cost to reach that node and the Manhattan distance from that node to the goal state. It implements an evaluation function which deals with the Math and this function also takes into account the number of active pills, as well as power pills. However it does not consistently take that into consideration. The heuristics can be improved for a better performance. Implementation wise the only logic difference with BFS is that it implements a Priority Queue where this data structure handles bringing up the best out of the evaluation function implemented.

0.3 Optimization Search

1. **HillClimber Search -** Implementation of this algorithm is simple where it searches for the neighbors which would possess the possible moves calculates the best of the both and and makes that as the current node. It is not quaranteed to reach the best solution and is ofted trapped in the local maxima. HillClimber overall is a short sighted algorithm in that manner and thus in this implementation is reaches on end of the frame and does not travel further.

2. **Simulated Annealing Search -** It takes an action sequence into account and where the algorithm chooses the best from the range thus is able to achieve the global maxima unlike HillClimber. It then retrieves a random move from the values and compares with the action that is randomly selected from the sequence. Comparing the move and the action with the best scores would find the goal state eventually. It was difficult to implement in terms of the technique been used here.

0.4 Evolution

1. **Evolutionary Strategy -** Implementation relies of the data structure of the Priority queue where is sorts the population. The person class has a variable fitness which basically determines the survival of the population. Also gives the action sequence corresponding to the game. The performance of the algorithm depends heavily on the number of generations and the size of the population. The average score turned out to be 800.

2. **Genetic Programming -** Conceptually we have reproduction features in this algorithm while the performace remains similar to evolutionary strategy. If we can randomly mutate the selected population to maintain the diversity can help in the performance of the algorithm.