Name: CHAN King Yeung

SID: 1155119394

CSCI3230 Assignment 2

1. Genetic algorithm is an example of stochastic search algorithm. It evaluates an optimal point for an objective function. It initialises a set of individuals randomly and generates next generation repeatedly. For every generation, the population will experience selection, crossover and mutation (each of which consists of probabilities) to search through the whole state space. It stops at some reasonable generation or the fitness function (function to evaluate the objective function) no longer change substantially.
2. is a function which records the total cost spent from the first node to node.

is a heuristic value function which estimates the (minimal) cost from node to the goal node.

1. Both depth-first search and depth-limited search favour the Minimax algorithm.
2. Breadth-first search (BFS)

With limited nodes in the search tree, if there is a solution in it, BFS eventually finds its depth. Thus, BFS is complete. In the view of worse case, BFS needs to go through all the depth for searching a solution. If there are depths in the search tree, the time complexity will be , that is . Moreover, BSF memories every nodes while searching the solution. The space complexity in the worse case is also . In general, BFS is not optimal since BFS does not consider the cost when replacing the fringe. However, if all the step cost is the same, BFS becomes uniform-cost search which guarantees optimality.

Depth-first search (DFS)

Similar to BFS, if the depth is finite, DFS eventually reaches to a solution. Yet, if there are cycles in the depth space, the depth space becomes infinite. DFS may get stuck in the looping the cycles and DFS no longer complete. Given a search tree has maximum depths, DFS goes through all the depth to finds a solution in the worse case, and thus, the time complexity is . DFS searches a branch of subtree at a point in time. The space forms a linear space and the space complexity is . DFS ignores the step cost to search for a solution in the leftmost subtree. Hence, DFS is not optimal.

1. Suppose is consistent, where is a goal node and is the actual cost. We have

where is the cost from node to node

given we know at the goal node

given we know

If the fact holds for any arbitrary node , consistency implies admissibility by definition. Thus, we have

from the proof above

Thus, consistency implies admissibility.

On the other hand, admissibility does not imply consistency. Consider a single way out path which has nodes, that is where is the first node and is the goal node. Suppose for every step and , we know is admissible if we let and . In the situation, , hence, is admissible without doubt. However, we know that

Thus, consistency implies admissibility but not the other way around.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Diagram  Description automatically generated |  | Diagram  Description automatically generated |
|  | Diagram  Description automatically generated |  | Diagram  Description automatically generated |
|  | Diagram  Description automatically generated |  | Diagram  Description automatically generated |
|  | Diagram  Description automatically generated |  | Diagram  Description automatically generated |
|  | Diagram  Description automatically generated |  |  |

Therefore, the shortest path will be SHHC CC WSYC SC.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Diagram  Description automatically generated |  | Diagram  Description automatically generated |
|  | Diagram  Description automatically generated |  | Diagram  Description automatically generated |

Therefore, the shortest path will be SHHC CC WSYC SC.

1. Pruning in minimax algorithm aims to eliminate a branch of a search tree if we have enough information about some nodes to draw the conclusion. Without checking all the nodes in the search tree, it reduces the exponent in time complexity by half. The beauty of pruning is that the final result remains the same even cutting branches. There are 2 parameters in - pruning, which are and respectively. consists of the maximum value it has found and records the minimum value it has found. When we found a value which is the best choice in the corresponding subtree, that is , we can ignore the remaining nodes in that subtree. Because we always select the best choice before pruning, ignoring nodes will not affect the final decision and makes the algorithm runs faster.

Diagram

Description automatically generated