CSE 4633 Programming Assignment 1

Naveen Neelakandan (nn149)

Part 2:

IDA* is a variant of the A* search algorithm that eliminates the memory constraints of A* without sacrificing solution optimality. It is an informed search version of the iterative deepening depth- first search algorithm and utilizes iterative deepening to keep memory usage lower than in A*. Its memory requirement is linear with respect to the maximum search depth. Unlike iterative deepening depth-first search though, IDA* uses f-costs as the next limit for the search. In each iteration, nodes whose f- value exceeds the cost threshold for that iteration are cut off. The cost threshold is increased to that of the lowest cost node that was pruned during the previous iteration and the search is repeated. My implementation of IDA* uses depth- limited search as a subroutine. This subroutine takes a cost as a limit and performs a depth-first search till that limit. The IDA* algorithm then repeatedly calls this subroutine with increasing f-costs. It is observed from the implementation of IDA* that increasing the search depth increases the search time/ memory by an exponential amount. This is because the branching factor of the 8-puzzle is around 2.5 and so each depth of the tree has 2.5 times as many nodes as the previous. This causes an exponential increase in the amount of nodes generated/expanded and thus the time/ memory consumed is also vastly greater. Thus this makes IDA* an algorithm suited to solving the n- puzzle. As most of the nodes are at the last iteration, using depth first search drastically reduces the memory requirements. From this, it also follows that the 15-puzzle is vastly more difficult to solve than the 8-puzzle as the average search depth of 15-puzzle is far greater. Hence, in order to not run out of memory on the 15-puzzle, certain enhancements are required such as parent checking where you do not generate a successor node that is the same as a parent node. This helps in pruning out a great amount of nodes and thus saves memory. From the implementation of IDA*, it is also noticed that the cost threshold is automatically increased by 2 most of the time. This could be reasoned from the fact that only half of the states in an n-puzzle are reachable from any one state and so the solution lengths are all always either odd or even.