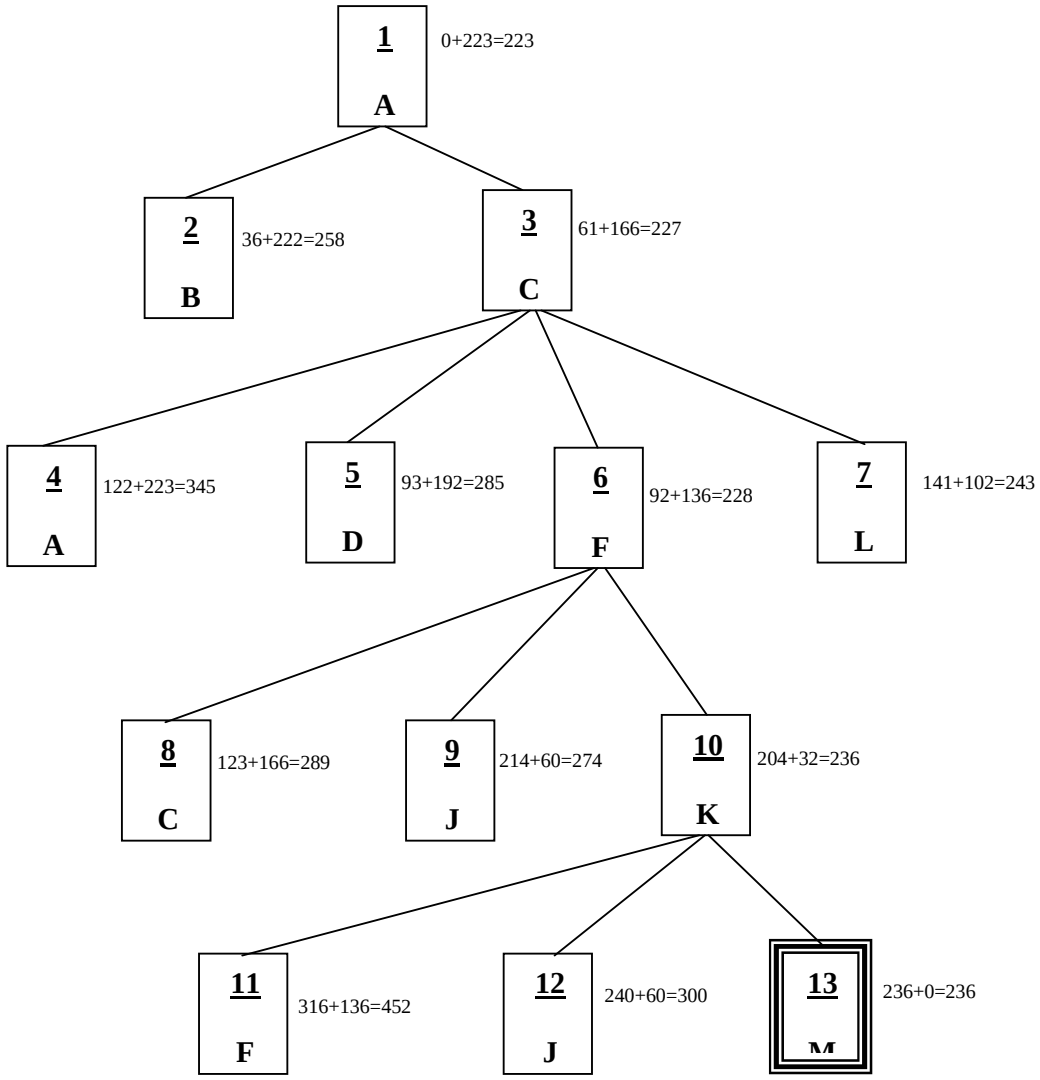
G5AIAI – Introduction to AI Exam
2002/2003

Question 1 – Model Answer

Using A* Algorithm

The Search Tree

The figures next to each node represent the G(n) and H(n) functions, where
G(n) = The cost of the search so far (i.e. distance travelled)
H(n) = The heuristic value (i.e. the straight line distance to the target town)



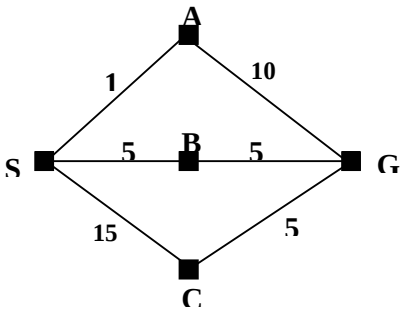
The route you would take is A, C, F, K, M at a cost of 236.

G5AIAI – Introduction to AI Exam
2002/2003

Show Uniform Cost Search operates.

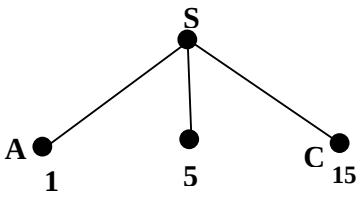
Breadth first search finds the shallowest goal state and that this will be the cheapest solution so long as the path cost is a function of the depth of the solution. But, if this is not the case, then breadth first search is not guaranteed to find the best (i.e. cheapest) solution. Uniform cost search remedies this.
It works by always expanding the lowest cost node on the fringe, where the cost is the path cost, g(n). In fact, breadth first search is a uniform cost search with g(n) = DEPTH(n).

Consider the following problem.

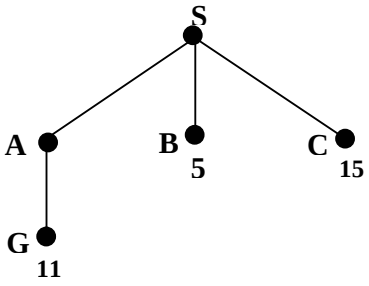


We wish to find the shortest route from S to G; that is S is the initial state and G is the goal state. We can see that SBG is the shortest route but if we let breadth first search loose on the problem it will find the path SAG; assuming that A is the first node to be expanded at level 1.
But this is how uniform cost search tackles the problem.

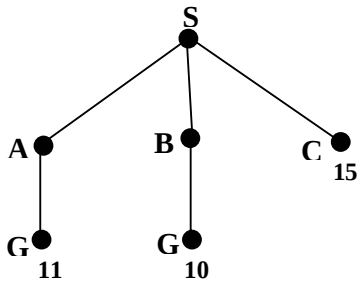
We start with the initial state and expand it. This leads to the following tree



The path cost of A is the cheapest, so it is expanded next; giving the following tree



We now have a goal state but the algorithm does not recognize it yet as there is still a node with a cheaper path cost. In fact, what the algorithm does is order the queue by the path cost so the node with cost 11 will be behind node B in the queue.
Node B (being the cheapest) is now expanded, giving



A goal state (G) will now be at the front of the queue. Therefore the search will end and the path SBG will be returned.

In summary, uniform cost search will find the cheapest solution provided that ***the cost of the path never decreases as we proceed along the path***. If this requirement is not met then we never know when we will meet a negative cost. The result of this would be a need to carry out an exhaustive search of the entire tree.