CS312 Artificial Intelligence Lab Assignment 6

Prepared by Mehul Bose (170030010) and K. Sai Anuroop (170030035)

Description of the Domain:
When a problem can be divided into a set of sub problems, where each sub problem can be solved separately and a combination of these will be a solution, AND-OR graphs or AND - OR trees are used for representing the solution. The decomposition of the problem or problem reduction generates AND arcs. One AND are may point to any number of successor nodes. All these must be solved so that the arc will rise to many arcs, indicating several possible solutions. Hence the graph is known as AND - OR instead of AND. Figure shows an AND - OR graph.

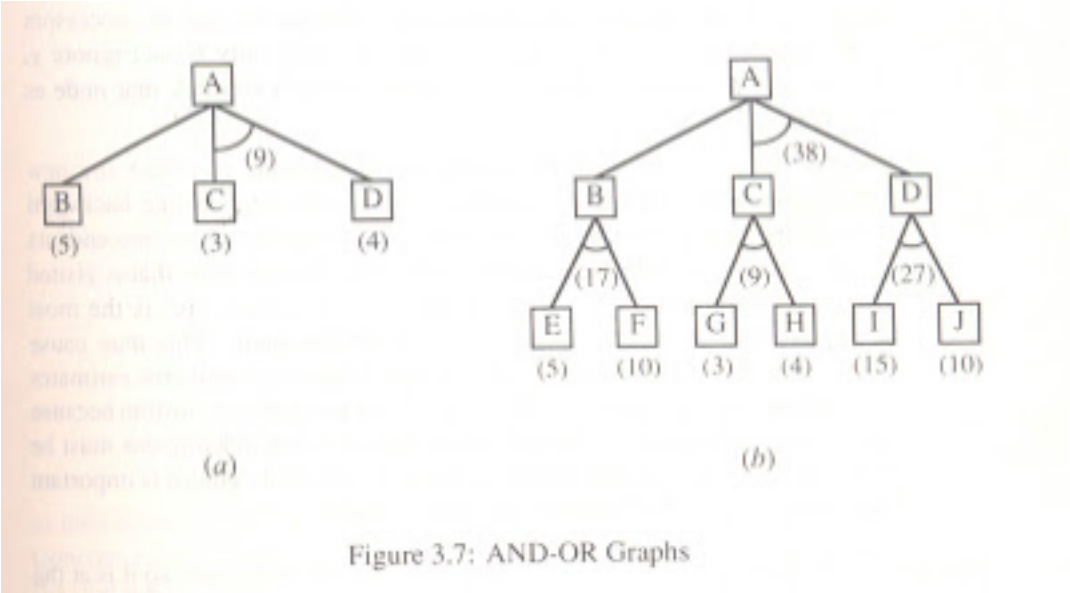


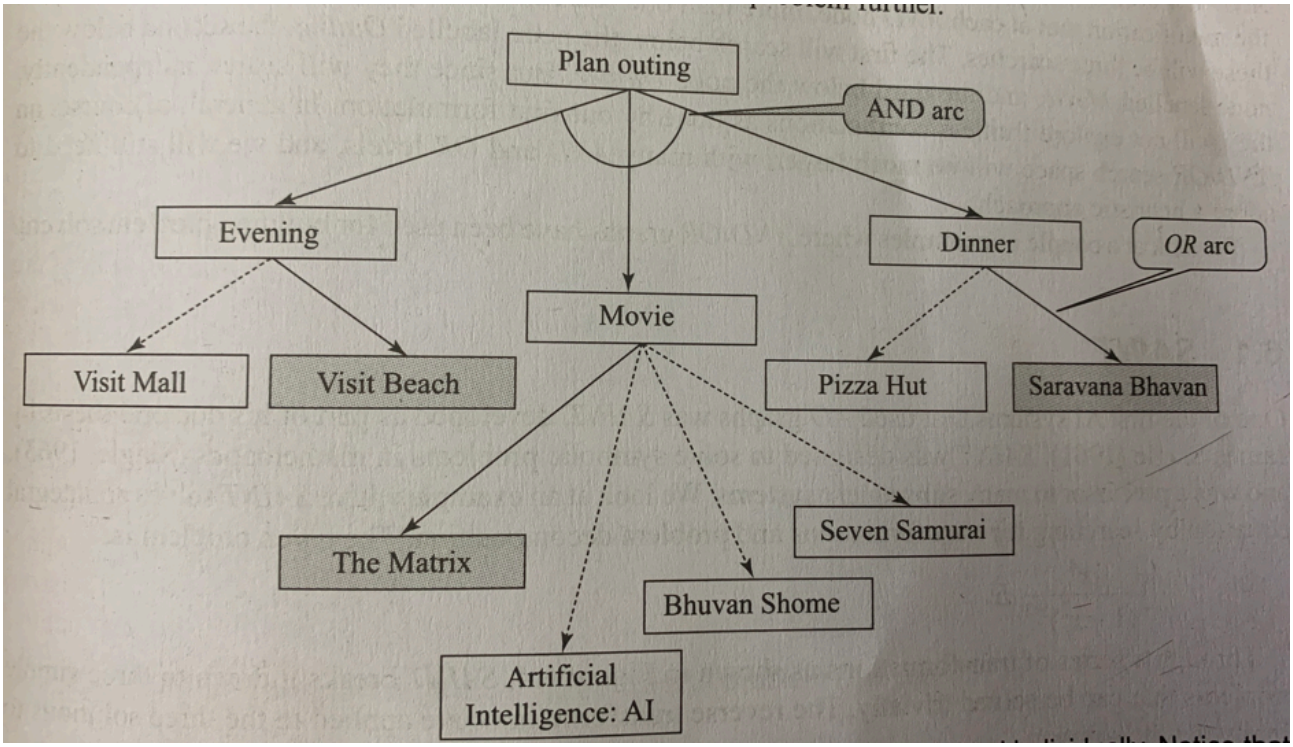
Figure 3.7: AND-OR Graphs

Heuristic Functions considered:
Function 1 (Underestimation): Edge Cost = highest cost of a single vertex
Function 2 (Overestimation): Edge Cost = lowest cost of a single vertex

Why does this work?
This particular set of functions works for the input graph which we have provided as a test case. In this case, Function 1 underestimates the cost of the distance as the distance to each node is always more than the value which the node holds since edge cost is the highest cost of a single node in all nodes.
In a similar manner, Function 2 overestimates the cost of the distance as the distance to each node is always lesser than the value which the node holds since edge cost is of the lowest cost of a single node in all nodes

Implementation:
We used the NetworkX module to create the graph and allow us to create the necessary constraints on the edges (for and-edges). Using this, we were able to take any graph as an input with the input format being as follows:
<number of nodes>
<start node>
<goal nodes i.e. nodes with cost 0 other than start node>
{for each node}
<node><SPACE><cost>
<number of edges>
{for each edge}
<node 1><SPACE><node 2>
{for each and set}
<parent node><SPACE><child 1><SPACE><child 2>

AO* algorithm analysis and observation:
In every cycle of refinement, the algorithm starts at the root. It follows the marked best option at each choice point, until it reaches some yet unresolved node or a set of nodes. It will refine one of them and then propagate the new values back up again. In the process if the subproblems of a given node are SOLVED nodes then it may also propagate SOLVED label back. The algorithm will terminate when the SOLVED label is backed up right up to the root.
We observed that the algorithm ran exactly as was given in the textbook. This was verified with the example which was given.



Use of AO* algorithm:
AO* algorithm can be used in a variety of places, for instance in evening planning (as mentioned in reference textbook) and also in the boolean 3-SAT problem.
In 3-SAT, we can theoretically use it to make decisions like set one variable as true, causing others to be necessarily false. These can be represented in the AND-OR graph with the use of and arcs. A solution of this problem would not be a path but in fact a subtree.
Similarly, we can use it to solve the evening planning problem, by having an and arc on the root of the tree. Once again, the solution is a subtree