15-210 Assignment ThesaurusLab

Roy Sung
roysung@andrew.cmu.edu
Section E
3/3/2014

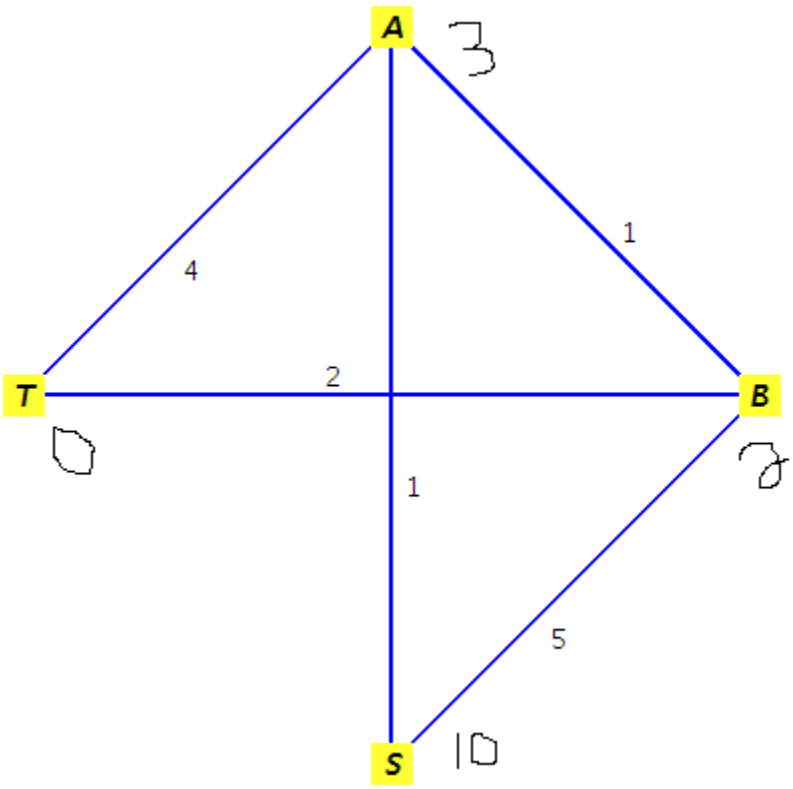
5: Path R Us

Task 5.1
So our graph would be $V = 1, 2, 3, E = ((1, 2), 1), ((1, 3), 2), ((2, 3), -5)$. So for the dijkstra's algorithm,it would first have 1 at the source node and that distance would be 0. Then it would insert $(2, 1), (3, 2)$. Where the first element is the node that it will travel to and the next element is the weight that it has to traverse. So the first thing to come out of the priority queue would be the $(2, 1)$ which then we would travel to node 2. Then we would insert $(3, -1)$ into the queue. The shortest paths we have so far would be $(1, 0), (2, 1)$. The queue would then have $(3, -4), (3, 2)$ and would pick the $(3, -5)$ as the next thing that gets retrieved from the queue. Then we would reach node 3 and since we have reached all nodes the algorithm would be terminated. This is where the error comes in as clearly the shortest path given to us by dijkstra's to go to node 2 is of distance 1 but we can see that we can go through node 3 to reach 2 and get a distance of -3. Thus dijkstra's gave use the wrong shortest path.

Task 5.2
We would just do the Bellman Ford's algorithm in that we would have to look for all the shortest path by the length of edges traversed. So in our previous example we would have to look for all the edges to traverse of length $1, 2$ This would make is so that we can so that the algorithm would check the path of $(1, 3, 2)$ which is the shortest path to node 2 from node 1.

Task 5.3
The Euclidean distance is admissible because we know that the shortest distance between two points is a straight line. which is what this heuristic is doing. So in the example graph, the heuristic will give the shortest distance possible thus it is guarented to give a shorter or of equal distance from some vertex to a destination vertex.
This heuristic is also consistent because again it gives the shortest distance between two points. Thus suppose that we have two arbitrary vertices (u, v) and there is an edge e between them with some weight. $h(u)$ will give us the shortest distance from some other arbitrary source vertex to u and the same thing for $h(v)$. We can note that if we travel from the arbitrary source vertex to v then to u through the edge that is a path from the arbitrary source vertex to u. Thus we can derive the inequality from this observation that $h(u) \leq h(v) + w(e)$ where the $w(e)$ returns the weight of the edge. This is because again h returns the smallest possible distance between some arbitrary source node to a target node and since we just observed that if we travel from the source vertex to v then to u that is also another path to u thus this path cannot have a distance any smaller than $h(u)$.

Task 5.4
The heuristic function would just return 0 for all inputs
Task 5.5
These are the heuristic values.
$(h(S) = 10), (h(A) = 1), (h(B) = 2), (h(T) = 0)$

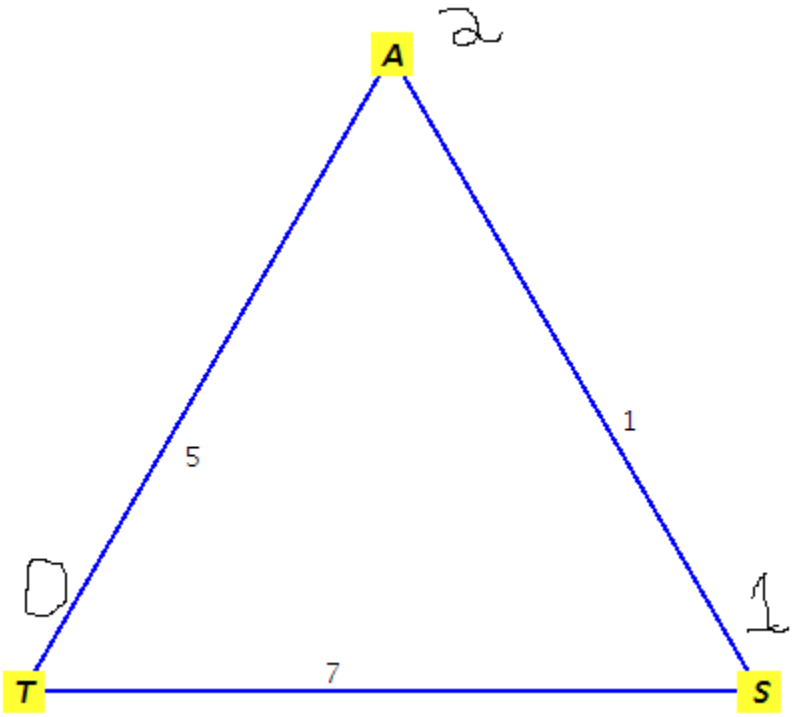


The algorithm fails as clearly the shortest path from S to T is the path S, A, B, T. The shortest path that gets outputted by A-star algorithm is the path S, A, T. The algorithm will always go through the vertex A then B. The next step is messed up as the heuristic value added up weight going through vertex B so it goes straight to vertex T from A. Thus the shortest path given will be (S, A, T) instead of (S, A, B, T).

Task 5.6

$(h(S) = 1), (h(A) = 2), (h(T) = 0)$

This graph is clearly not admissible. The algorithm fails when reaching the vertex T. This is because the heuristic value applied the vertex A, S. When the algorithm starts (at S) it will first reach vertex A. Then because of heuristic value of S and A, the algorithm will then reach the vertex T(destination) by going through A. Thus the A-star algorithm will output the shortest path to T by going through A.


A
5
1
7
T
S