ECE356 Lab 3: Negative Weight Discussion

Andrew Shim

December 5, 2012

Dijkstra's Algorithm for Directed Negative Weighted Graph

Step	N'	D(B), p(B)	D(C), p(C)	D(D), p(D)
1	A	1, A	0, A	99, A
2	AC	1, A	0, A	99, A
3	ACB	1, A	0, A	99, A
4	ACBD	-201, D	0, A	99, A

1

This depends on how one defines the meaning of the negative weight edge. If the negative weight edge is meant to label an edge as invalid, then no, the result calculated by Dijkstra's does not represent the shortest path from A to every other node in the graph. Specifically, the calculated path from A to B is invalid. However, if the negative weight edge has valid meaning (traversing that edge reduces the cost in a meaningful way), then I suppose we can say that all calculated paths are the shortest paths. However, if we pay close attention to the formulation of Dijkstra's, we observe that the condition that all edges have non-negative weights is required, and that any guarantee for optimally becomes lost if Dijkstra's is run on a graph containing negative weight edges.

2

The negative weight basically tricks Dijkstra's into thinking that there is path from A to B through D that is the valid shortest path. In other words, assuming that it is impossible to actually reduce the cost of a path, Dijkstra should choose the path from A to B that simply goes through the edge from A to B with cost 1. However, because of the negative weight edge in the path A - D - B, Dijkstra's thinks that it is a better idea to first traverse the 99 cost edge from A to D.

3

The first incorrect event that will occur in the undirected graph (let's simply take Figure 2 and make the edges undirected) is Dijkstra's will believe that there is a path from A to D through B that is of cost -299. Then, if we assume that the end condition for Dijkstra's is that the path costs converge to their minimums, the algorithm will essentially believe that the cost to any of the other nodes is negative infinity, because it can forever traverse the edge connecting B and D in order to continually decrease the cost of any path. However, in the way the algorithm was described in lecture, Dijkstra's will stop once it knows a path from the source to every other node; in this case, Dijkstra's will stop but the calculated paths will be invalid.