1 Introduction and Objective

This is the last tutorial for the second part of CS2010 (Graph Algorithms). After this, we will switch to the last part of CS2010 (Dynamic Programming/DP).

We will continue discussing the SSSP problem, especially the graph modeling aspects. We will revisit Dijkstra’s algorithm, both the Original and the Modified implementation variants.

The VisuAlgo page that is used in this tutorial is still http://visualgo.net/sssp.html
2 Tutorial 08 Questions

Standard Stuffs

Q1. In Lecture 09, you are presented with one more SSSP algorithm: Dijkstra’s algorithm, but in two implementation variants: the Original one (as defined by Dijkstra himself) and the Modified one (as commonly used in Competitive Programming world). First, the tutor will (re-)demonstrate the executions of both variants of Dijkstra’s algorithm on a small directed weighted graph using http://visualgo.net/sssp.html from a certain source vertex $s$. The tutor will re-explain what the Original Dijkstra’s can do and cannot do, followed with similar discussion with the Modified Dijkstra’s. The tutor may invite some students to do this live demonstration using different source vertex $s$ and/or using different graph. Finally, the tutor will discuss which variant that you should use and why?

This part is left to the tutor and should take some time as probably the similarities and differences are not clear for some students. Basically, both variants are perfectly equal if the edge weights are non-negative and both run in $O((V + E) \log V)$.

The key difference that have to be highlighted is that the Original Dijkstra’s cannot handle negative weight edges without generating wrong answers but the Modified Dijkstra’s cannot handle negative weight cycles without getting trapped in an infinite loop.

Which one that we should use? The answer is depends. We have seen in Lab Demo 07 that Modified Dijkstra’s is not an ‘all-conquering’ SSSP algorithm as it can also ‘be-killed’ with ‘that kind of test case’. The tutor will re-demonstrate ‘that kind of test case’ again.

Graph Modeling Exercises, Again

Q2. A salesman frequently needs to drive from one city to another to promote his products. Since time is of the essence, he wants to use the shortest route to get from one city to another. However in every city he passes he will have to pay a toll fee. The toll fee is the same for every city and it is a positive unit. Therefore, given two different routes of the same distance (positive unit) to get from city $A$ to city $B$, he will prefer the one which passes through fewer cities. An example is shown below:

Propose the best algorithm using what you have learnt so far (and a bit more), so that the salesman will choose a route from any source city $A$ to any destination city $B$ such that it has the shortest distance and also passes through the fewest cities. What is the running time for your algorithm?

Near complete answer: Modify the priority queue of Dijkstra’s algorithm a bit. From storing pairs $(d[v], v)$, we store triplets $(d[v], h[v], v)$, where $h[v]$ stores how many hops/vertices have been used in the shortest path so far. On the sample graph above, the execution of Original Dijkstra’s is shown below. Observe the last step involving vertex E.

1. $PQ = \{(0, 1, A), (\infty, \infty, B), (\infty, \infty, C), (\infty, \infty, D), (\infty, \infty, E)\}$ // process (0, 1, A)
2. $PQ = \{(7, 2, C), (10, 2, B), (\infty, \infty, D), (\infty, \infty, E)\}$ // process (7, 2, C)
3. $PQ = \{(10, 2, B), (17, 3, D), (\infty, \infty, E)\}$ // process (10, 2, B)
To get from A to E, route A,B,E is preferred over route A,C,D,E even though both have the same cost 25, since A,B,E goes through fewer cities.

**Figure 1: An Illustration**

4. \(PQ = \{(17, 3, D), (25, 3, E)\}\) // process \((17, 3, D)\), it will NOT? change \((25, 3, E)\) as \((25, 3, E)\) is better? than \((25, 4, E)\) from \((17, 3, D)\)

5. \(PQ = \{(25, 3, E)\}\) // do nothing

6. \(PQ = \{}\) // done

Now, is that implementation (that is correct for THIS tut08 Q2 and runs in \(O((V + E) \log V)\) as with the standard time complexity of Original Dijkstra’s) also 100% correct for **PS5 Subtask C**?

**Figure 2: One (of the Several) Corner Case for PS5 C, try query(0, 5, 5) = 26 vs query(0, 5, 4) = 27**

1. \(PQ = \{(0, 1, 0), (\infty, \infty, 1), (\infty, \infty, 2), (\infty, \infty, 3), (\infty, \infty, 4), (\infty, \infty, 5)\}\) // process \((0, 1, 0)\)

2. \(PQ = \{(7, 2, 2), (10, 2, 1), (\infty, \infty, 3), (\infty, \infty, 4), (\infty, \infty, 5)\}\) // process \((7, 2, 2)\)

3. \(PQ = \{(10, 2, 1), (17, 3, 3), (\infty, \infty, 4), (\infty, \infty, 5)\}\) // process \((10, 2, 1)\)

4. \(PQ = \{(17, 3, 3), (26, 3, 4), (\infty, \infty, 5)\}\) // process \((17, 3, 3)\), now do you think changing \((26, 3, 4)\) to \((25, 4, 4)\) is a better move or a worse move?
5. \( \text{PQ} = \{(25, 4, 4), (\infty, \infty, 5)\} \) \quad \text{// process(25, 4, 4)}

6. \( \text{PQ} = \{(26, 5, 5)\} \) \quad \text{// do nothing}

7. \( \text{PQ} = \{} \) \quad \text{// done}

Now do you realize that you somehow still need \((26, 3, 4)\) if the query is \(\text{query}(0, 5, 4)\) that requires shortest paths of \(K = 4\) hops/vertices? Think further... and attend Lab Demo 08.

Q3. Please download CS2010 Written Quiz 2 paper, 2013-14-S1-WQ2-medium.pdf, and solve a problem titled: Lego Mindstorms EV3 \((12 + 5\text{ bonus} = 17\text{ marks})\).

Vertices: \((r, c, \text{dir})\), Edges: two types: Move forward to \((r', c', \text{same dir})\) according to \((r, c, \text{dir})\) with cost: 3 units, or Turn right, stays at \((r, c)\) but change dir to the right side of current dir with cost 2 units. There are \(O(MN)\) vertices/edges as there are \(O(4MN)\) cells/vertices with at most 2 edges per cell or \(O(8MN) = O(MN)\) edges too. The problem is simply an SSSP on weighted graph with source vertex \((0, 0, \text{East})\), we report shortest path from source to destination vertex \((M-1, N-1, \text{any dir})\).

This graph problem is solvable with any Dijkstra’s algorithm variant with overall time complexity of \(O(MN \log MN)\).

Problem Set 5

Now the tutor will discuss Problem Set 5, Subtask A+B+C... The focus for today is to discuss what is/are so special about Subtask A+B. The tutor will only describe Subtask C vaguely and just say that it has very similar flavor to both of the graph modeling problems that we discussed in this tut08.

Subtask A+B basically ignores parameter \(k\), so both Subtasks turn into ‘simple’ stuffs (beware that the number of queries \(Q\) is still insane... have you seen such PS with insane number of queries before?). Subtask C has that parameter \(k\), and we think you should be able to link what you need to do with that parameter \(k\) after the discussion of Q2+Q3 above...

Online Quiz 2 Preparation

If there is still some time left, the tutor will discuss random question(s) from \texttt{http://visualgo.net/training.html?diff=Hard\&n=20\&t1=40\&module=graphs,graphtraversal,mst,sssp} to prepare you all for the upcoming Online Quiz 2.

This part is left to the tutor, just pick one or two interesting questions that you find in the VisuAlgo online quiz system and describe your solving strategy to your tutorial group.