CS2010 PS5 - The Onset of Labor v2

Released: Tuesday, 09 October 2012
Due: Saturday, 20 October 2012, 8am

Collaboration Policy. You are encouraged to work with other students or teaching staffs (inside or outside this module) on solving this problem set. However, you **must** write the Java code **by yourself**. In addition, when you write your Java code, you **must** list the names of every collaborator, that is, every other person that you talked to about the problem (even if you only discussed it briefly). This list may include certain posts from fellow students in CS2010 IVLE discussion forum. Any deviation from this policy will be considered cheating, and will be punished severely, including referral to the NUS Board of Discipline. It is not worth it to cheat just to get 15% when you will lose out in the other 85%.

R-option. There is no R-option in this PS. Time to cool down until PS7 =).

Special for e-Learning week. Actually, Steven wants to make PS5 harder than last year. However, due to e-Learning week on Week08, many students do not read/watch Lecture08 (Dijkstra's algorithm) during the stipulated lecture time. In order to 'force' majority of students to learn Lecture08 material as soon as they can, Steven changes the weightage of Subtask1-2-3 down to 60 points only, and make Subtask4 worth additional 40 points. There is no secret that this Subtask4 requires Lecture08 knowledge.

Last Year's Story (Updated). After $\approx$ 40 weeks, $\approx$ 9+ months, and 3 trimesters, a baby is born. Jane's EDD (Expected Date of Delivery) was 24 October 2011 (NUS Week11 last year). But as many parents will testify, babies rarely born *precisely* on their EDD, but they can be born +/- 2 weeks from the EDD.

When you seniors first read this PS (4 October 2011), Jane was actually just three weeks away from birth (she was born on 24 October 2011, exactly on her EDD). By solving this PS, your seniors had a chance to help their lecturer (me =D) did something super important in his life: Accompanied his wife during *the onset of labor*[1], escorted her from home to our chosen hospital as fast as possible, so that she went through the 'challenging' three stages of labor[2] in a proper delivery suite in the hospital.

Now, it is your chance (CS2010 S1 AY2012/2013) students to do the same =D.

[1]The cues that birth is imminent, like the ruptured water bag, contractions, etc.
For more details, see: `http://www.babycenter.com/0_signs-of-labor_181.bc`

[2]See PS2 story: 'Scheduling Deliveries'. Now imagine that Steven's wife, Grace, is now really one of the women in the delivery room.

The Actual Problem. Given a map of Singapore (as a directed weighted graph), estimated time[3] to travel through Singapore roads (as *non-negative* weights of the corresponding directed edges – in minutes), Steven and Grace's home (always vertex 0 on that graph), their chosen hospital (always vertex 1 on that graph), determine the *shortest path* to go from Steven and Grace's home to their chosen hospital and report the shortest path weight: The sum of edge weights along the shortest path. Steven will call for taxi and then instruct the taxi driver to take this path. It is guaranteed that there will be at least one path from vertex 0 to vertex 1 in the given graph.

Steven and Grace needs the shortest path/quickest way, because once the contractions start (**the onset of labor**), Grace does not want to spend *too much time* on the road because – as you may have guessed it correctly – it is 'not without pain'... Similarly, Steven does not want to be *overly anxious* and wants to have Grace taken care by the doctor and nurses/midwives *as soon as possible*. The first stage of labor usually last for *a few hours*.

Now, let's go back to the problem. For example, suppose Singapore[4] a directed weighted graph as shown below (all edges are directed, except edge 4-6 is bidirectional, or we can view it as having two directed edges $4 \to 6$ and $6 \to 4$):

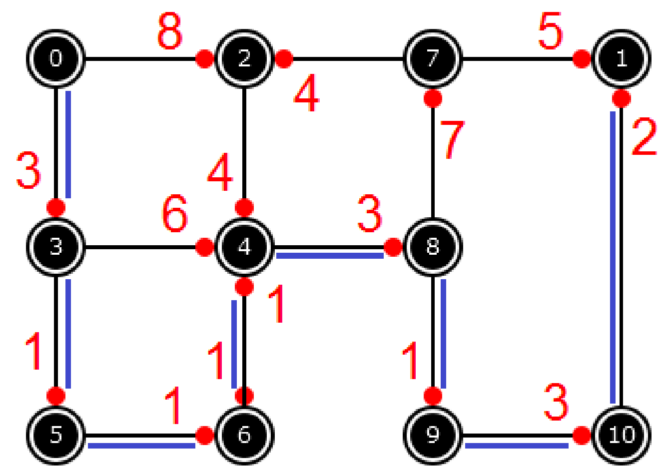

Figure 1: A Sample Singapore Map

If Steven and Grace's home is at vertex 0 and their chosen hospital is at vertex 1, then the quickest way is this path: `0` $\to$ `3` $\to$ `5` $\to$ `6` $\to$ `4` $\to$ `8` $\to$ `9` $\to$ `10` $\to$ `1` with total estimated traveling time of: 3+1+1+1+3+1+3+2 = 15 minutes. The skeleton program `Labor.java` is already written for you, you just need to implement one (or more) method(s)/function(s):

- `int Query()`
  Query your Adjacency List data structure[5] where the (non-negative) weight of each road (edge) is stored in the Adjacency List itself, return the shortest path weight (in minutes) from vertex 0 to vertex 1. There will be at least one path from vertex 0 to vertex 1 in the given graph.

- If needed, you can write additional helper methods/functions to simplify your code.

[3]To simplify this problem, let's assume that this time estimation is accurate and there is no traffic jam in Singapore.
[4]Yes, Singapore map does not looks like this, but let's just assume it is.
[5]Already implemented in `Labor.java`.

Subtask 1 (20 points). On an 'impossible case' that simplifies this problem, the road network in Singapore is a directed weighted tree and $1 \leq V + E \leq 10$.

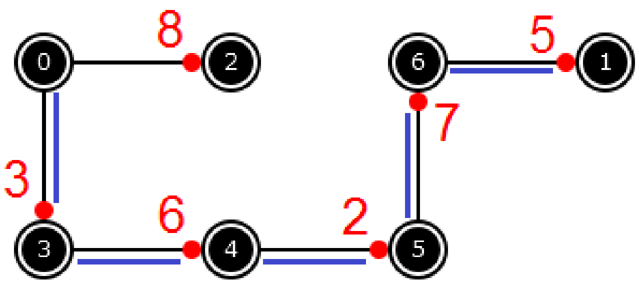

Figure 2: A Simplified Singapore Map (Tree)

The quickest way for the tree above is path: `0` $\rightarrow$ `3` $\rightarrow$ `4` $\rightarrow$ `5` $\rightarrow$ `6` $\rightarrow$ `1` with total estimated traveling time of: 3+6+2+7+5 = 23 minutes.

Subtask 2 (Additional 20 points). On another 'impossible case' that simplifies this problem, all roads in Singapore is a directed weighted graph but somehow require exactly 7 minutes to traverse and $1 \leq V + E \leq 100$.

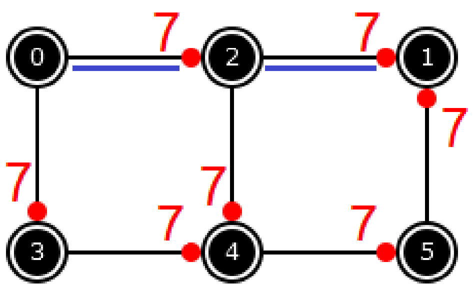

Figure 3: A Simplified Singapore Map (Similar Weight)

The quickest way for the graph above is path: `0` $\rightarrow$ `2` $\rightarrow$ `1` with total estimated traveling time of: 7+7 = 14 minutes.

Subtask 3 (Additional 20 points). The road network in Singapore is a directed weighted graph, the time to traverse Singapore roads varies, and $1 \leq V \times E \leq 1000000$. In fact, the sample test case shown above fits this description.

Subtask 4 (Additional 40 points). Same as Subtask 3, but now $1 \leq V + E \leq 250000$.

Note: The test data to reach 40 points: `Subtask1.txt` and `Subtask2.txt` are given to you. The `Sample.txt` contains *a small subset* of our official test data for Subtask 3 and 4. You are allowed to check your program's output with your friend's. You are encouraged to generate and post additional test data in IVLE discussion forum. Moreover, starting from PS5, you are officially given a test data verifier program to test the validity of test data posted in discussion forum. For PS5, see `LaborVerify.java` that will check if the edge weights are non-negative, check if there is at least one path from vertex 0 to vertex 1, and finally check the suitability of test data w.r.t Subtask 3 and Subtask 4 constraints.