CS213/293 Data Structure and Algorithms 2023

Lecture 19: Graphs - Shortest path

Instructor: Ashutosh Gupta

IITB India

Compile date: 2023-11-06

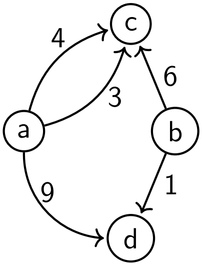
Labeled directed graph

Definition 19.1

A labeled directed graph $G = (V, E)$ is consists of

- *set V of vertices and*
- *set $E \subseteq V \times \mathbb{Q}^+ \times V$.*

For $e \in E$, we will write $L(e)$ to denote the label.



The above is a labeled graph $G = (V, E)$, where

$V = \{a, b, c, d\}$ and

$E = \{(a, 3, c), (a, 4, c), (a, 9, d), (b, 6, c), (b, 1, d)\}$.

$L((a, 3, c)) = 3$.

Shortest path

Consider a labeled directed graph $G = (V, E)$.

Definition 19.2

For vertices $s, t \in V$, a path from s to t is a sequence of edges $e_1, ..., e_n$ from E such that there is a sequence of nodes $v_1, ..., v_{n+1}$ such that $v_1 = s$, $v_{n+1} = t$, and $e_i = (v_i, \_, v_{i+1})$ for each $i \in 1..n$.

Definition 19.3

A length of $e_1, ..e_n$ is $\sum_{i=1}^{n} L(e_i)$.

Definition 19.4

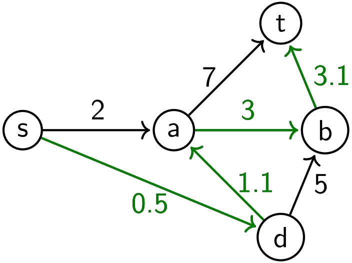
For vertex $s, t \in V$, a shortest path is a path s and t such that the length of the path is minimum.

Commentary: Does the above definition work for $n = 0$?

Example: shortest path

Example 19.1

The shortest path from s to t is 0.5,1.1,3,3.1.



Exercise 19.1

a. How many simple paths are there from s to t?

b. Show that there are exponentially many simple paths between two vertices.

Problem: single source shortest path(SSSP)

To compute a shortest path from s to t, we need to say that there is no other way to reach t.

We need to effectively solve the following problem.

Definition 19.5

Find shortest paths starting from a vertex s to all vertices in G.

Definition 19.6

Let $SP(x)$ denote the length of a shortest path from s to x.

Observation: relating SP of neighbors

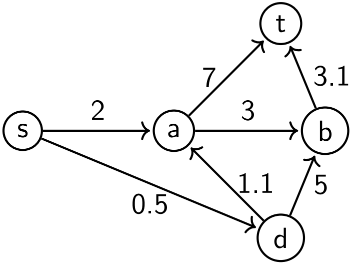
For $(v, k, w) \in E$, we can conclude

$$SP(w) \leq SP(v) + k$$

Observation: upper bounds of paths

Example 19.2

Considering only outgoing edges from s, what can we say about a shortest path from s to a and d?

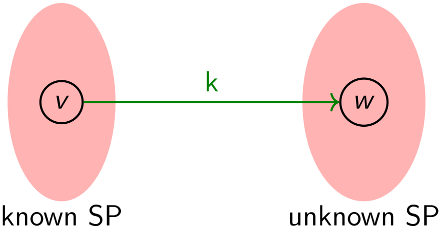


$SP(s) = 0$ $\quad SP(a) \leq 2 + SP(s)$ $\quad SP(d) = 0.5 + SP(s)$

Observation: Since we know SP to s, we can compute SP to the closest neighbor and upper bound SP for the other neighbors.

Can we lift the observation for a set of nodes?

Let us suppose we know SP for a set of vertices. What can we say about the remaining vertices?



$SP(w) \leq SP(v) + k$ holds for all edges that are in the cut between known and unknown.

Can we say something more about $SP(w)$ for which $SP(v) + k$ is the minimum among all edges on the cut?

Expanding known set

Consider labeled directed graph $G = (V, E)$.

Theorem 19.1

Let C be the cut for set $S \subset V$ in G.Let $d = min\{SP(v') + k|(v', k, \_) \in C\}$ and $(v, k, w) \in C$ achieves the minimum. Then, $SP(w) = d$.

Proof.

Let us suppose there is a path $e_1, .., e_n$ from s to w such that $L(e_1, ...e_n) < d$.The path has prefix

$$\underbrace{e_1 ... e_j}_{\in S} \underbrace{e_{j+1}}_{\in C} .....$$

Let $e_{j+1} = (v', k, w') \in C$. Therefore, $L(e_1, ...e_j e_{j+1}) \geq SP(v') + k$.

Due to the definition of d, $SP(v') + k \geq d$. Therefore, $L(e_1, .., e_n) \geq d$. Contradiction.

Therefore, $SP(w) \geq d$. □

Dijkstra's algorithm

Algorithm 19.1: SSSP(Graph $G = (V, E)$, vertex s)

```
Heap unknown;
for v ∈ V do
    v.visited := False;
    unknown.insert(v, ∞);
unknown.decreasePriority(s, 0);
sp[s] := 0;
while unknown ≠ ∅ do
    v := unknown.deleteMin();
    for e = (v, k, w) ∈ E do
        if ¬w.visited then
            unknown.decreasePriority(w, k + sp[v]);
            sp[w] := min(sp[w], k + sp[v]);
    v.visited := True
```

Example: Dijkstra's algorithm

Consider the following graph. We start with vertex s. $SP(s) = 0$.
The cut has edges 2 and 0.5.

The minimum path on the cut is $SP(s) + 0.5$. $SP(d) = 0.5$.

Now the cut has edges 2, 1, and 0.5.

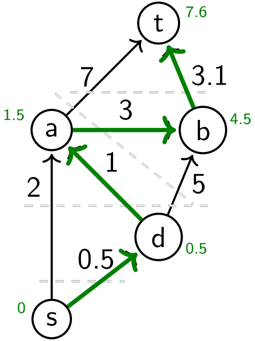
The minimum path on the cut is $SP(d) + 1$. $SP(a) = 1.5$.

Now the cut has edges 7, 3, and 5.

The minimum path on the cut is $SP(a) + 3$. $SP(b) = 4.5$.

Now the cut has edges 7 and 3.1.

The minimum path on the cut is $SP(b) + 3.1$. $SP(a) = 7.6$.



Exercise 19.2

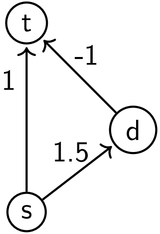
Modify Dijkstra's algorithm to construct the shortest paths from s to every vertex t.

Negative lengths

Dijkstra's algorithm does not work for negative lengths.

Example 19.3

On the following graph, Dijkstra's algorithm will return wrong shortest path.



The algorithm uses an argument that depends on monotonic increase of length.

Topic 19.1

Tutorial problems

Example: Counting paths

Exercise 19.3

Modify Dijkstra's algorithm to compute the number of shortest paths from s to every vertex t.

Example: Negative edges

Exercise 19.4

Show an example of a graph with negative edge weights and show how Dijkstra's algorithm may fail. Suppose that the minimum negative edge weight is -d. Suppose that we create a new graph G' with weights w', where G' has the same edges and vertices as G, but w'(e)=w(e)+d. In other words, we have added d to every edge weight so that all edges in the new graph have edge weights non-negative. Let us run Dijkstra on this graph. Will it return the shortest paths for G?

Example: Road network

Exercise 19.5

Let G(V,E) be a representation of a geography with V as cities and (u,v) an edge if and only if there is a road between the cities u and v. Let d(u,v) be the length of this road. Suppose that there is a bus plying on these roads with fare f(u,v)=d(u,v). Next, suppose that you have a free coupon that allows you one free bus ride. Find the least fare paths from s to another city v using the coupon for this travel.

Exercise 19.6

Same as above. Suppose w(u,v) is the width of the road between the cities u and w. Given a path pi, the width w(pi) is the minimum of widths of all edges in pi. Given a pair of cities s and u, is it possible to use Dijkstra to determine d such that it is the largest width of all paths pi from s to v?

Exercise 19.7

Same as above. For a path pi, define hop(pi)=max(d(e)) for all e in pi. Thus if one is traveling on a motorcycle and if fuel is available only in cities, then hop(pi) determines the fuel capacity of the tank of your motorcycle needed to undertake the trip. Now, for any s and u, we want to determine the minimum of hop(pi) for all paths pi from s to u. Again, can Dijkstra be used?

Topic 19.2

Problems

Example: Travel plan

Exercise 19.8

You are given a timetable for a city. The city consists of n stops V=v1,v2,...,vn. It runs m services s1,s2,...,sm. Each service is a sequence of vertices and timings. For example, the schedule for service K7 is given below. Now, you are at stop A at 8:00 a.m. and you would like to reach stop B at the earliest possible time. Assume that buses may be delayed by at most 45 seconds. Model the above problem as a shortest path problem. The answer should be a travel plan.

Example: Preferred paths

Exercise 19.9

Given a graph G(V,E) and a distinguished vertex s and a vertex v, there may be many shortest paths from s to v. What shortest path is identified by Dijkstra?

End of Lecture 19