12.0

Last week we gave Kruskal algorithms by Kruskal and Prim for constructing a MST

We then started discussing Single Source Shortest Path problem in which we are given a weighted directed graph an a vertex as input and we have to compute, for each vertex, the length of the shortest directed path from the input vertex.

Important observation:

Optimal substructure: If $p = \langle v_0, \ldots, v_k \rangle$, $s = v_0$, is a shortest path from v_0 to v_k then $\forall\ 0 \le i < j \le k$ $\langle v_i, \ldots, v_j \rangle$ is a shortest path from v_i to v_j.

Negative cycles: If $\exists$ a negative cycle then no solution.

```
Relax(u,v,w):
1. If v.d > u.d + w(u,v)
2.     v.d = u.d + w(u,v)
3.     v.π = u
```

```
Initialize-Single-Source(G,s)
1. For each v ∈ G.V
2.     v.d = ∞ , v.π = NIL
3. s.d = 0
```

```
Bellman-Ford alg: Bellman-Ford(G,w,s)
1. Initialize-Single-Source(G,s)
2. For i=1 to |G.V|-1
3.     For each (u,v) ∈ G.E
4.         Relax(u,v,w)
5. For each edge (u,v) ∈ E
6.     If v.d > u.d + w(u,v)
7.         Return False (neg. cycle exist).
8. Return True
```

Bellman-Ford Run time is $O(|V| \cdot |E|)$

Analysis.

Main Lemma:

If $p = \langle v_0, \ldots, v_k \rangle$ is a shortest path from v_0 to v_k and we relax edges of p in the order $(v_0, v_1), \ldots, (v_{k-1}, v_k)$ then $v_k.d = \delta(v_0, v_k)$ (the length of the shortest path). This remains true even if other relaxation steps occur. (Assuming no negative cycles).

Pf:

By induction on k. For $k=0$ (or $k=1$) this is trivial. Also easy to see for $k=1$.

For $k = m+1$ by induction $\langle v_0, \ldots, v_m \rangle$ is a s.p. caused by ind. hyp. $v_m.d = \delta(v_0, v_m)$.

Regardless of other relaxations, when we relax (v_m, v_{m+1}) we get that $v_{m+1}.d = v_m.d + w(v_m, v_{m+1})$

$= \delta(v_0, v_{m+1})$. ∎

Comment: We should have also argued by induction that $v.d \geq \delta(v_0, v)$ for all v.

Corollary: if no negative cycles, since each path of length $\leq n-1$ after $n-1$ steps of relaxing all edges all distances are computed correctly.

Also need to show that we get a tree of shortest paths.

Corollary: if any cycle exists we will catch it.

pf: if cycle is $\langle v_0, \ldots, v_k \rangle$ then $\sum_{i=1}^{k} (v_i, v_{i-1}) < 0$.

if we fail to find it in lines 5-7 then

$v_i.d \leq v_{i-1}.d + w(v_{i-1}, v_i)$

$\Rightarrow \sum_{i=1}^{k} v_i.d \leq \sum_{i=1}^{k} v_{i-1}.d + \sum_{i=1}^{k} w(v_{i-1}, v_i)$

As $v_0 = v_k$ we get a contradiction ∎

12.2

Next we are going to see another classical alg. due to Dijkstra in the case when all the weights are non negative

The basic idea is at each step to resolve the distance to one vertex and adjust the rest of v.d accordingly.
For example at the first step we are assured that for some neighbor of s, v vertex $\delta(s,v) = w(s,v)$ etc.
We shall use a min-priority-queue again, sorted according to v.d.

Dijkstra(G,s,w)
1. Initialize-Single-Source(G,s)
2. Let Q = G.V min-priority-queue
3. while Q ≠ ∅
4. u = Extract-min(Q)
5. For each v ∈ G.Adj[u]
6. Relax(u,v,w)

The running time is $O((|E|+|V|)\cdot \lg|V|)$. Indeed, we need $O(|V|)$ at line 2 and for line 1. For each edge we have to decrease key which costs $O(\lg|V|)$. For each v we also have Extract-min which costs $O(\lg|V|)$.

Analysis: At each iteration, all vertices in G.V \ Q satisfy v.d = δ(s,v). ~~about~~ each v ∈ Q v.d is its minimal ~~distance~~ distance to G.V \ Q (distance measured by nearest neighbor)
Since at the end Q = ∅ this ~~proves~~ gives correctness.

12.3

Proof: The invariant holds after first iteration.

Assume it does not hold at some point and let u be first such vertex.

By assumption, for all $v \notin Q$ $v.d = \delta(s,v)$.

Consider the minimal path from s to u and let y be the first vertex there from Q (maybe y=u).

Let x be y's predecessor on that path. Thus the path is $s \rightsquigarrow x \to y \rightsquigarrow u$

We now note that we must have $y.d = \delta(s,y)$ as we relaxed (x,y) after removing x from Q. (and by assumption $x.d = \delta(s,x)$).

By optimal-substructure $\delta(s,u) \geq \delta(s,y)$, but also $u.d \geq \delta(s,u)$ (as before)

hence $u.d \geq \delta(s,u) \geq \delta(s,y) = y.d \geq u.d$

Thus, we only have equalities and hence $u.d = \delta(s,u)$ and no contradiction. ∎

All pairs shortest paths

The problem of finding shortest paths between any two vertices in a directed graph.

Comments:

If graph has more than $|G.V|^2/\lg|V|$ many edges, we can just keep all info in an array (identifying vertices with the numbers $1, 2, \ldots, |G.V|$)

Extract-min takes $O(|V|)$ and we get a run time of $O(|V|^2 + |E|) = O(|V|^2)$

If we use Fibonacci-heap can reduce runtime to $O(|V| \lg |V| + |E|)$

12.4

All Pairs Shortest Paths (APSP)

The goal is to compute, for every u, v, the shortest path between them $(u \rightsquigarrow v, v \rightsquigarrow u)$.
We can run the SSSP alg $|V|$ times to get $O(V^3)$ or $O(VE \log V)$ time alg.
(or else $O(|V|^2 \log V + VE)$ using Fib. heap), if all wts ≥ 0.
If this is not the case we must run Bellman-Ford and get $O(|V|^2 |E|)$.

We will see an alg. that does better.
Here it will be convenient to assume the input is given by adjacency matrix W, where $W_{ij} = w(i,j)$, where we think of $G.V$ as $\{1, 2, \ldots, |G.V|\}$.
If there is no edge (i,j) we have $w(i,j) = \infty$.

The output will be represented by a matrix D where $D_{ij} = \delta(i,j)$ and a matrix Π where $\Pi_{ij} = NIL$ if $i = j$ or there is no $i \rightsquigarrow j$ path and else by j's predecessor in a shortest $i \rightsquigarrow j$ path.

We will present the Floyd-Warshall alg.
The idea is for every (i,j) and k to compute the s.p. from $i \rightsquigarrow j$ that only goes through vertices in $\{1, \ldots, k\}$. Then we increase k until we get them.
The reasoning is that if p is a minimal path $i \rightsquigarrow j$ using only $\{1, \ldots, k\}$ then it must be of the form $i \overset{k}{\rightsquigarrow} j$ or $i \overset{k}{\rightsquigarrow} k \overset{k}{\rightsquigarrow} j$.
Thus, it can be solved by using the information computed so far.
Hence:

$$d_{ij}^{(k)} = \begin{cases} w_{ij} & k = 0 \\ \min\left(d_{ij}^{(k-1)}, d_{ik}^{(k-1)} + d_{kj}^{(k-1)}\right) & k > 0 \end{cases}$$

12.5

Floyd-Warshall (W)

1. $n = W.rows$
2. $D^{(0)} = W$
3. For $k = 1$ to n
4. Let $D^{(k)} = (d_{ij}^{(k)})$ be a new matrix
5. For $i = 1$ to n
6. For $j = 1$ to n
7. $d_{ij}^{(k)} = \min(d_{ij}^{(k-1)}, d_{ik}^{(k-1)} + d_{kj}^{(k-1)})$
8. Return $D^{(n)}$

Run time: $O(n^3) = O(|V|^3)$

How to compute Π?

Notice that in the k'th step we should take

$$\pi_{ij}^{(k)} = \begin{cases} \pi_{ij}^{(k-1)} & \text{if } \min \text{ is } d_{ij}^{(k-1)} \\ \pi_{kj}^{(k-1)} & \text{if } \min \text{ is } d_{ik}^{(k-1)} + d_{kj}^{(k-1)} \end{cases}$$

Simple to incorporate to alg.

~~Consider the following matrix~~