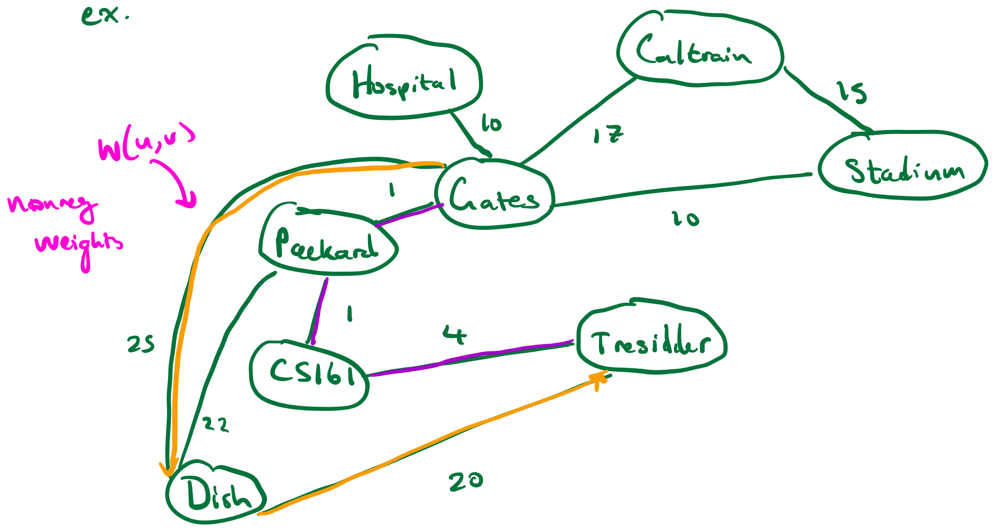
Weighted Graphs

ex.



Shortest path on weighted graphs

Def. Cost of a path is sum of weights on that path
Shortest path is path w/ minimum cost

Distance $d(u,v)$ between vertices u, v is cost of shortest path between u, v

BFS no longer works on weighted graphs to find shortest paths?

(e.g. BFS shortest path from Gates → Tresidder involves Dish, cost 45)

True shortest path: Gates → Packard → CS161 → Tresidder
cost 6

Shortest sub-path theorem

Thm. A sub-path of a shortest path is also a shortest path

(i.e., if x is on the shortest path from s to t, then the given path from s to x is the shortest path from s to x)

Proof: Contradiction (if there was a shorter path from x to s, then there would be a shorter path from s to t via x)

Dijkstra's Algorithm for Single-Source Shortest Paths

Problem statement. Given weighted graph $G = (V, E)$ and vertex $u \in V$,
find $d(u, w)$, $\forall w \in V$.

Procedure

Algorithm Dijkstra (s, V, E)

Set all vertices to not-sure

$d[v] = 0 \quad \forall v \in V$

$d[s] = 0$

While there are not-sure nodes:

Pick the not-sure node w/ smallest estimate $d[u]$ — use priority queue (min-heap)

For v in u.neighbors:

$d[v] = \min(d[v],\ d[u] + \text{edgeWeight}(u, v))$

Mark u as not-sure

return d

Proof of correctness

Thm. Suppose we run Dijkstra on $G = (V, E)$ starting from s

At the end of the algorithm, $d[v] = d(s, v)$, $\forall v \in V$

Claim 1. $\forall v$, $d[v] \geq d(s, v)$

Intuition: on update of $d[v]$, we have path $s \to u \to v$ in mind

$d[v] = \min(\underset{\text{prev path}}{d[v]},\ \underset{\text{new path}}{d[u] + E(u, v)})$

$\Rightarrow$ $d[v] \geq$ length of intended path
$\geq$ length of shortest path
$= d(s, v)$

Formally: use induction.

Claim 2. When a vertex is marked sure, $d[v] = d(s,v)$

Proof.

IH: When we mark the t'th vertex v as sure, $d[v] = d(s,v)$

BC ($t=1$): First vertex marked sure is s, and $d[s] = d(s,s) = 0$

IS: Assume by induction that every v already marked sure has $d[v] = d(s,v)$

Suppose adding u to sure list

(picked u w/ smallest $d[u]$ that is not sure)

WTS: $d[u] = d(s,u)$

Define v as good if $d[v] = d[s,v]$, suppose u not good.

Suppose z is last good vertex on $s \to u$ shortest path before u, z' is next after z.

$$d[z] \underset{\substack{\uparrow \\ z \text{ good}}}{=} d(s,z) \underset{\substack{\uparrow \\ \text{subpath thm}}}{\leq} d(s,u) \leq d[u]$$

contradiction assumption

u not good $\Rightarrow d[z] \neq d[u] \Rightarrow d[z] < d[u]$, so z is sure

$\Rightarrow$ already updated z':

$$d[z'] \underset{\substack{\uparrow \\ \text{update}}}{\leq} d[z] + w(z,z') \underset{\substack{\uparrow \\ \text{induction assumption}}}{=} d(s,z) + w(z,z') \underset{\substack{\uparrow \\ \text{subpath thm}}}{=} d(s,z') \underset{\substack{\uparrow \\ \text{Claim 1}}}{\leq} d[z']$$

$\Rightarrow d[z'] \leq d[z'] \to d[z'] = d(s,z')$, z' good

$\Rightarrow u$ good! so contradiction

Claim 1 + Claim 2: Imply thm.

$\to$ v marked sure $\to d[v] = d(s,v)$

$\to$ $d[v] \geq d(s,v)$ and never increases, so after v marked sure, $d[v]$ stops changing

$\to$ Any time after v marked sure, $d[v] = d(s,v)$

$\to$ All vertices are sure at the end $\to d[v] = d(s,v)$ $\forall v \in V$

Runtime

Data structure dependent

Operations:

Stores unseen vertices $d[v]$

Can find u w/ minimum $d[u]$

Can remove that u

Can update (decrease) $d[v]$

Total time is $O(|V|(T(\text{findmin}) + T(\text{remove min})) + |E|(\text{update key}))$

Using array:

findMin $O(|V|)$

removeMin $O(|V|)$

updateKey $O(1)$

$\Rightarrow$ total runtime $O(|V|^2 + |E|) = O(|V|^2)$

Using RB-tree:

findMin $O(\log|V|)$

removeMin $O(\log|V|)$

updateKey $O(\log|V|)$

$\Rightarrow$ total runtime $O((|E| + |V|)\log|V|)$

Using min-heap:

findMin $O(1)$ amortized

removeMin $O(\log|V|)$ amortized

updateKey $O(1)$ amortized

$\Rightarrow$ total runtime $O(|V|\log|V| + |E|)$

estimate $O(|E|\log|V|)$