Review: Dijkstra: Single-Source Shortest Path (SSSP)

Use a priority queue ("abstract data structure" that allows Insert, ReduceKey, and DeleteMin operations)

\[ Q: \quad 5 \quad a \quad b \quad c \quad d \]

\[ 0 \quad \infty \quad \infty \quad \infty \quad 0 \]

\[ \text{DeleteMin} \]

\[ \Rightarrow \text{ReduceKey for neighbors if found shorter path.} \]

\[ - \quad 2 \quad 10 \quad \infty \quad 0 \]

\[ \Rightarrow \text{Does not update because } 11 > 10 \]

\[ 6 \]

\[ 7 \]

\[ Q \text{ empty } \Rightarrow \text{done!} \]
global priority queue Q

Dijkstra (V, E, EC, S) // s = "Source"

for all v in V do
    SP[v] ← ∞
end for

for all v in V do
    insert v into Q with key SP[v]
end for

while Q ≠ ∅ and not all keys are ∞ do
    v ← DeleteMin(Q)
    for each out-neighbor w of v do
        if SP[w] > SP[v] + EC(v, w) then
            SP[w] ← SP[v] + EC(v, w)
        endif
    end for
end while

Running time: depends on implementation of abstract priority Q

<table>
<thead>
<tr>
<th>VI Insertions</th>
<th>N DeleteMin</th>
<th>E Reduce key</th>
</tr>
</thead>
<tbody>
<tr>
<td>O(V log V)</td>
<td>O(E)</td>
<td>O(E)</td>
</tr>
</tbody>
</table>

Heap: $O((|V|+|E|) \log |V|)$

Fibonacci: $O(|V| \log |V| + |E|)$

Heap: will use this one because we know how heaps work (but prob not how Fibonacci work)
Variations (pg. 467-8)

#3 All points to single destination t

For shortest path from i to t:

Construct G' with all edges reversed & same cost

and run "regular" Dijkstra with source t.

#5 Given G = (V, E) with special vertex be V

"the bakery", find \( \delta(i, j) \) \( \in V \), the shortest
path from i to j and \( \delta(j, i) \) back to j.

Solution:

\[ \delta(i, j) = \text{length of SP } i \rightarrow j \text{ in } G \]

Running Time: \( O(V(V + E) \log V) \)
Solution 2: 0 Use Dijkstra on reversed graph with source b.

This gives path $i \rightarrow b \forall i \in V$ in $\mathcal{G}$.

2 Use Dijkstra on $\mathcal{G}$ with source $b$.

This gives path $b \rightarrow j \forall i \in V$ in $\mathcal{G}$.

3 $\forall (ij)$, add lengths (concat paths).

$\text{SP } i \rightarrow b \text{ and } b \rightarrow j$.

Running Time: $O((|V|+|E|) \log (|V|) + N^2)$

$\Rightarrow$

SOLUTION 2 gives faster algorithm.

Problem: Dijkstra's algorithm need not work correctly if edges have negative cost. Show example when it fails.

```
\begin{array}{c|ccc}
 & A & B & C \\
\hline
S & 0 & \infty & \infty \\
\hline
- & 1 & 0 & 99 \\
\hline
\end{array}
```

$Q$ empty $\Rightarrow$ done! But SP $s \rightarrow A$ has cost $-20$ NOT 0
Problem: Let $G = (V,E)$ be a directed graph with exactly one negative edge. Show how to compute SSSP on $G$.

Hint: Use Dijkstra on modified graph.

Let $G'$ be $G$ but with the negative edge removed.

Run Dijkstra on $G'$.

Suppose negative cost edge is $(u,v)$. Check if $SP_s[u] + EC(u,v) < SP_s[v]$.

Yes $\Rightarrow$ done.

Otherwise, run Dijkstra with source $v$ in $G$.

For $t \in V$, length of $SP$ $s \rightarrow t$ in $G$ is $min(SP_t + (u,v) + SP_v)$.