Shortest Path Algorithms

Single Source Shortest Path
Dijkstra’s Algorithm

Shortest path examples

- Highway system
  - Distance
  - Travel time
  - Number of stoplights
  - Krispy Kreme locations
- Network of airports
  - Travel time
  - Fares
  - Actual distance
Weighted path length

- Consider an edge-weighted graph $G = (V,E)$.
- Let $C(v_i, v_j)$ be the weight on the edge connecting $v_i$ to $v_j$.
- A path in $G$ is a non-empty sequence of vertices $P = \{v_{i}, v_2, v_3, \ldots, v_{k}\}$.
- The weighted path length is given by $\sum_{i=1}^{k-1} C(v_{i}, v_{i+1})$.

The general problem

- Given an edge-weighted graph $G = (V,E)$ and two vertices, $v_s \in V$ and $v_d \in V$, find the path that starts at $v_s$ and ends at $v_d$ that has the smallest weighted path length.

Single-source shortest path

- Given an edge-weighted graph $G = (V,E)$ and a vertex, $v_s \in V$, find the shortest path from $v_s$ to every other vertex in $V$.
- To find the shortest path from $v_s$ to $v_d$, we must find the shortest path from $v_s$ to every vertex in $G$. 
Shortest weighted path from b to f:
\{b, a, c, e, f\}

Shortest unweighted path from b to f:
\{b, c, e, f\}

Shortest path problem undefined for graphs with negative-cost cycles

\{d, a, c, e, f\} cost: 4
\{d, a, c, d, a, c, e, f\} cost: 2
\{d, a, c, d, a, c, d, a, c, e, f\} cost: 0
Dijkstra’s Algorithm

- Greedy algorithm for solving shortest path problem
- Assume non-negative weights
- Find shortest path from $v_s$ to each other vertex

Dijkstra’s Algorithm

- For each vertex $v$, need to know:
  - $k_v$: Is the shortest path from $v_s$ to $v$ known? (initially false for all $v \in V$)
  - $d_v$: What is the length of the shortest path from $v_s$ to $v$? (initially $\infty$ for all $v \in V$, except $v_s = 0$)
  - $p_v$: What vertex precedes (is parent of) $v$ on the shortest path from $v_s$ to $v$? (initially unknown for all $v \in V$)

<table>
<thead>
<tr>
<th>$v$</th>
<th>$k_v$</th>
<th>$d_v$</th>
<th>$p_v$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a$</td>
<td>$F$</td>
<td>$\infty$</td>
<td></td>
</tr>
<tr>
<td>$b$</td>
<td>$F$</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>$c$</td>
<td>$F$</td>
<td>$\infty$</td>
<td></td>
</tr>
<tr>
<td>$d$</td>
<td>$F$</td>
<td>$\infty$</td>
<td></td>
</tr>
<tr>
<td>$e$</td>
<td>$F$</td>
<td>$\infty$</td>
<td></td>
</tr>
<tr>
<td>$f$</td>
<td>$F$</td>
<td>$\infty$</td>
<td></td>
</tr>
</tbody>
</table>
\[ \begin{array}{|c|c|c|c|} \hline v & k_v & d_v & p_v \\ \hline a & F & 3 & b \\ b & T & 0 & -- \\ c & F & 5 & b \\ d & F & \infty & \\ e & F & \infty & \\ f & F & \infty & \\ \hline \end{array} \]
Dijsktra’s Algorithm

\[ \text{Dijsktra}(G, s_0) \]
\[
\begin{aligned}
// Initialize \\
n &= |V|; \\
\text{createtable}(n); // stores k, d, p \\
\text{createpq}(|E|); // empty pq \\
table[s_0].d &= 0; \\
\text{insertpq}(0, s_0); \\
\end{aligned}
\]

// con’t
\[
\begin{aligned}
\text{while (!pq.isempty)} \\
\{ \\
v_0 &= \text{getMin}(); \\
\text{if (!table[v_0].k)} // not known \\
\{ \\
table[v_0].k &= \text{true}; \\
\text{for each } v_i \in \text{Adj}[v_0] \\
\{ \\
\text{newd}=table[v_i].d + \text{weight}(v_i, v_0); \\
\end{aligned}
\]
Dijsktra’s Algorithm

// con’t
// newd = table[v_i].d + weight(v_i,v_j);  
if (table[v_i].d > newd) O( )
{
    table[v_i].d = newd; O( )
    table[v_i].p = v_0; O( )
    insertpq(newd,v_i); O( )
}

Dijsktra’s Algorithm

// con’t
for each v ∈ G(V,E) O( )
    // build vertex set in T
    v ∈ T(V,E’);
for each v ∈ G(V,E) O( )
    // build edge set in T
    (v,table[v].p) ∈ T(V,E’);

All-pairs shortest path problem

- Given an edge-weighted graph
  G = (V,E), for each pair of vertices in V
  find the length of the shortest weighted
  path between the two vertices

Solution: Run Dijkstra V times