All Pairs Shortest Paths

• Find the shortest path between all pairs of vertices in a graph
• Allow negative edge weights (but not negative cycles!)
  – So you cannot use Dijkstra’s algorithm
Warm-up questions

• True or false?
  – A subpath of a shortest path is still a shortest path
    
    (A subpath of path \( p = \langle v_0, v_1, ..., v_k \rangle \) is a contiguous subsequence of its vertices. That is, for any \( 0 \leq i \leq j \leq k \), the subsequence of vertices \( v_i, v_{i+1}, ..., v_j \) is a subpath of \( p \)
    
    \[ \text{Proof. Cut and paste:} \]

  – Suppose \( p_1 \) is the shortest path between \( u \) and \( x \); \( p_2 \) is the shortest path between \( x \) and \( v \);
    
    \[ |p_1| + |p_2| \text{ is the shortest weight for a path which uses} \]
    
    \( x \) as the intermediate vertex between \( u \) and \( v \)
Warshall’s algorithm: definitions

• Define $c_{ij}^{(k)} =$ weight of a shortest path from $i$ to $j$ with intermediate vertices belonging to the set \{1, 2, ..., $k$\}. (note: $k$ is not the number of intermediate vertices! And $k$ may not be in the shortest path from $i$ to $j$!)

Thus, $\delta(i, j) = c_{ij}^{(n)}$. Also, $c_{ij}^{(0)} = W_{ij}$ (edge weight from $i$ to $j$)
**Example**

$c_{v1,v2}^{(0)}$ = Shortest edge weight between $v1,v2 = 3$

$c_{v1,v2}^{(1)}$ = Shortest path weight by using a subset of $\{v1\}$ as the intermediate vertices = 3. (the chosen subset could be empty!)

$c_{v1,v2}^{(2)}$ = Shortest path weight by using only a subset of $\{v1,v2\}$ as the intermediate vertices = 3

$c_{v1,v2}^{(3)}$ = Shortest path weight by using only a subset of $\{v1,v2,v3\}$ as the intermediate vertices = -2

$c_{v1,v2}^{(4)}$ = ? (-2)

$c_{v1,v2}^{(5)}$ = ? (-13)

---

all the paths between v1 and v2, allowing intermediate vertices with index up to k:

$p(v1,v2,0)=\{v1\rightarrow v2\}$
$p(v1,v2,1)=\{v1\rightarrow v2\}$
$p(v1,v2,2)=\{v1\rightarrow v2\}$
$p(v1,v2,3)=\{v1\rightarrow v2, v1\rightarrow v3\rightarrow v2\}$
$p(v1,v2,4)=\{v1\rightarrow v2, v1\rightarrow v3\rightarrow v2\}$
$p(v1,v2,5)=\{v1\rightarrow v2, v1\rightarrow v3\rightarrow v2, v1\rightarrow v5\rightarrow v4\rightarrow v3\rightarrow v2\}$
Recursive structure in the algorithm

\[ c_{ij}^{(k)} = \min \{ c_{ij}^{(k-1)}, c_{ik}^{(k-1)} + c_{kj}^{(k-1)} \} \]

Two possibilities for the shortest path between i and j:
1. the shortest path does not use k as an intermediate vertex
2. the shortest path uses k as an intermediate vertex
Pseudocode

\[
\text{for } k \leftarrow 1 \text{ to } n \quad \text{// } n = |V|, \text{ (number of vertices)}
\]
\[
\qquad \text{do for } i \leftarrow 1 \text{ to } n
\]
\[
\qquad \qquad \text{do for } j \leftarrow 1 \text{ to } n
\]
\[
\qquad \qquad \qquad \text{do if } c_{ij} > c_{ik} + c_{kj}
\]
\[
\qquad \qquad \qquad \qquad \text{then } c_{ij} \leftarrow c_{ik} + c_{kj}
\]
\[
\quad \text{relaxation}
\]

Notes:

• Okay to omit superscripts, since extra relaxations can’t hurt.
• Runs in \( \Theta(n^3) \) time.
• Simple to code.
• Efficient in practice.
Another version of pseudocode for Warshall’s algorithm

Warshall(G)

D ← 2D Array |V| x |V| ; Will be the distances
P ← 2D Array |V| x |V| ; Will be the paths, the last “helper”. Finally, P[u,v] contains the last intermediate vertex in the shortest path from u to v.

for i ← 1 to |V|
    for j ← 1 to |V|
        D[i, j] ← w(i, j)
        P[i, j] ← i

for k ← 1 to |V|
    for i ← 1 to |V|
        for j ← 1 to |V|
            if D[i, k] + D[k, j] < D[i, j] then
                D[i, j] ← D[i, k] + D[k, j]
                P[i, j] ← P[k, j]

D[i,j] = shortest path weight from i to j by using a subset of \{v_1,v_2,...,v_k\} as the intermediate vertices
Example

Initial Configuration
\( D_{i,j}^{(1)} = \min(D_{i,j}^{(0)}, D_{i,1}^{(0)} + D_{1,j}^{(0)}) \)

Allowed intermediate vertices: subset of \{V1\}
$D_{i,j}^{(2)} = \min(D_{i,j}^{(1)}, D_{i,2}^{(1)} + D_{2,j}^{(1)})$

Allowed intermediate vertices: subset of \{V1, V2\}
$D_{i,j}^{(3)} = \min(D_{i,j}^{(2)}, D_{i,3}^{(2)} + D_{3,j}^{(2)})$

Allowed intermediate vertices: subset of \{V1,V2,V3\}
**K=4**

\[ D_{i,j}^{(4)} = \min(D_{i,j}^{(3)}, D_{i,4}^{(3)} + D_{4,j}^{(3)}) \]

Allowed intermediate vertices: subset of \{V1,V2,V3,V4\}

<table>
<thead>
<tr>
<th></th>
<th>V1</th>
<th>V2</th>
<th>V3</th>
<th>V4</th>
<th>V5</th>
</tr>
</thead>
<tbody>
<tr>
<td>V1</td>
<td>0</td>
<td>3</td>
<td>-1</td>
<td>4</td>
<td>-4</td>
</tr>
<tr>
<td>V2</td>
<td>3</td>
<td>0</td>
<td>-4</td>
<td>1</td>
<td>-1</td>
</tr>
<tr>
<td>V3</td>
<td>7</td>
<td>4</td>
<td>0</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>V4</td>
<td>2</td>
<td>-1</td>
<td>-5</td>
<td>0</td>
<td>-2</td>
</tr>
<tr>
<td>V5</td>
<td>8</td>
<td>5</td>
<td>1</td>
<td>6</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>V1</th>
<th>V2</th>
<th>V3</th>
<th>V4</th>
<th>V5</th>
</tr>
</thead>
<tbody>
<tr>
<td>V1</td>
<td>-</td>
<td>1</td>
<td>4</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>V2</td>
<td>4</td>
<td>-</td>
<td>4</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>V3</td>
<td>4</td>
<td>3</td>
<td>-</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>V4</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>-</td>
<td>1</td>
</tr>
<tr>
<td>V5</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>-</td>
</tr>
</tbody>
</table>
Done!  

\[ K=5 \]

\[ D_{i,j}^{(5)} = \min(D_{i,j}^{(4)}, D_{i,5}^{(4)} + D_{5,j}^{(4)}) \]

Allowed intermediate vertices: subset of \{V1, V2, V3, V4, V5\}

Trace back the shortest path between \( v3, v5 \): V3 \( \rightarrow \) v2 \( \rightarrow \) v4 \( \rightarrow \) v1 \( \rightarrow \) v5

v3 \( \rightarrow \) v2 \( \rightarrow \) v4 \( \rightarrow \) v1 \( \rightarrow \) v5
Warshall's algorithm

Why is this correct?

Claim: $D[i, j]$ is the length of the shortest path in $v_1, v_2, ..., v_k$

Proof: by induction on $k$
Basis: $k=0$, trivially true

Inductive hypothesis: Assume that the claim is true for $k-1$.
Inductive step: We will prove the claim is true for $k$.

Consider the shortest path from $i$ to $j$ using nodes 1 to $k$.
two cases:
    case 1: The path from $i$ to $j$ goes through $k$
    case 2: It does not go through $k$.
In either case, we can prove the shortest special path will be $D[i, j]$