Warm up:
Modify Dijkstra’s Algorithm to find the shortest paths by product of edge weights (assume all weights are at least 1)
Dijkstra’s Algorithm

Initialize $d_v = \infty$ for each node $v$

Keep a priority queue $PQ$ of nodes, using $d_v$ as key

Pick a start node $s$, set $d_s = 0$

While $PQ$ is not empty:

$v = PQ.extractMin()$

for each $u \in V$ s.t. $(v, u) \in E$:

$PQ.decreaseKey(u, \min(d_u, d_v + w(v, u)))$

Modify Dijkstra’s Algorithm to find the shortest paths by *product* of edge weights
(assume all weights are at least 1)
Dijkstra’s Algorithm (for min product)

Initialize $d_v = \infty$ for each node $v$
Keep a priority queue $PQ$ of nodes, using $d_v$ as key
Pick a start node $s$, set $d_s = 1$
While $PQ$ is not empty:

$v = PQ.\text{extractmin}()$

for each $u \in V$ s.t. $(v, u) \in E$:

$PQ.\text{decreaseKey}(u, \min(d_u, d_v \cdot w(v, u)))$

How do we know this works?
Shortest path by product

Goal: find the path \((s = v_1, v_2, \ldots, v_{k-1}, v_k)\)
which minimizes:
\[
w(v_1, v_2) \cdot w(v_2, v_3) \cdot \ldots \cdot w(v_{k-1}, v_k)
\]

Observation: \(\log(x \cdot y) = \log x + \log y\)
\[
\log(w(v_1, v_2) \cdot w(v_2, v_3) \cdot \ldots \cdot w(v_{k-1}, v_k)) \\
= \log w(v_1, v_2) + \log w(v_2, v_3) + \ldots + \log w(v_{k-1}, v_k)
\]

New Goal: find the path \((s = v_1, v_2, \ldots, v_{k-1}, v_k)\) which
minimizes:
\[
\log(w(v_1, v_2)) + \log(w(v_2, v_3)) + \ldots \\
+ \log(w(v_{k-1}, v_k))
\]
Dijkstra’s Algorithm (for min product)

Initialize $d_v = \infty$ for each node $v$

Keep a priority queue $PQ$ of nodes, using $d_v$ as key

Pick a start node $s$, set $d_s = 0$

While $PQ$ is not empty:

$v = PQ.ex\text{tract}\text{min}()$

for each $u \in V$ s.t. $(v, u) \in E$:

$PQ.de\text{creaseKey}(u, \min(d_u, d_v \cdot w(v, u)))$

$d_v + \log(w(v, u))$
Today’s Keywords

- Graphs
- Shortest path
- Bellman-Ford
  - OG DP
- Floyd-Warshall
CLRS Readings

• Chapter 22
• Chapter 23
• Chapter 24
Homeworks

• HW7 Due **Tomorrow** April 16 @11pm
  – Written (use latex)
  – Graphs

• HW8 Released Tomorrow April 16 @11:30pm
  – Due Tuesday April 23 @11pm
  – Programming (Python or Java)
  – Graphs/Flow (hint, see Wednesday’s class)
# Currency Exchange

1 Dollar = 3.87 Ringgit

<table>
<thead>
<tr>
<th>Currency code</th>
<th>Currency name</th>
<th>Units per USD</th>
<th>USD per Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>USD</td>
<td>US Dollar</td>
<td>1.0000000000</td>
<td>1.0000000000</td>
</tr>
<tr>
<td>EUR</td>
<td>Euro</td>
<td>0.8783121137</td>
<td>1.1385474303</td>
</tr>
<tr>
<td>GBP</td>
<td>British Pound</td>
<td>0.6956087704</td>
<td>1.4375896950</td>
</tr>
<tr>
<td>INR</td>
<td>Indian Rupee</td>
<td>66.1909310706</td>
<td>0.0151078098</td>
</tr>
<tr>
<td>AUD</td>
<td>Australian Dollar</td>
<td>1.3050318080</td>
<td>0.7662648480</td>
</tr>
<tr>
<td>CAD</td>
<td>Canadian Dollar</td>
<td>1.2997506294</td>
<td>0.7693783541</td>
</tr>
<tr>
<td>SGD</td>
<td>Singapore Dollar</td>
<td>1.3478961522</td>
<td>0.7418969172</td>
</tr>
<tr>
<td>CHF</td>
<td>Swiss Franc</td>
<td>0.9590451582</td>
<td>1.0427037678</td>
</tr>
<tr>
<td>MYR</td>
<td>Malaysian Ringgit</td>
<td>3.8700000000</td>
<td>0.2583979328</td>
</tr>
<tr>
<td>JPY</td>
<td>Japanese Yen</td>
<td>112.5375383115</td>
<td>0.0088859239</td>
</tr>
<tr>
<td>CNY</td>
<td>Chinese Yuan Renminbi</td>
<td>6.4492409303</td>
<td>0.1550570076</td>
</tr>
<tr>
<td>NZD</td>
<td>New Zealand Dollar</td>
<td>1.4480018872</td>
<td>0.6906068347</td>
</tr>
<tr>
<td>THB</td>
<td>Thai Baht</td>
<td>35.1005319022</td>
<td>0.0284895968</td>
</tr>
<tr>
<td>HUF</td>
<td>Hungarian Forint</td>
<td>275.7012427385</td>
<td>0.0036271146</td>
</tr>
<tr>
<td>AED</td>
<td>Emirati Dirham</td>
<td>3.6730000000</td>
<td>0.2722570106</td>
</tr>
<tr>
<td>HKD</td>
<td>Hong Kong Dollar</td>
<td>7.7563973683</td>
<td>0.1289258341</td>
</tr>
<tr>
<td>MXN</td>
<td>Mexican Peso</td>
<td>17.3168505322</td>
<td>0.0577472213</td>
</tr>
<tr>
<td>ZAR</td>
<td>South African Rand</td>
<td>14.7201431400</td>
<td>0.0679341220</td>
</tr>
</tbody>
</table>
1 Dollar = 0.8783121137 Euro

1 Euro = 4.1823100458 Dirham

1 Dirham = 1.0548325619 Ringgit

1 Dollar = 0.8783121137 * 4.1823100458 * 1.0548325619 Ringgit

= 3.87479406049 Ringgit

Directly: 1 Dollar = 3.87 Ringgit
Currency Exchange

1 Dollar = 3.87479406049 Ringgit

<table>
<thead>
<tr>
<th>Currency code</th>
<th>Currency name</th>
<th>Units per USD</th>
<th>USD per Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>USD</td>
<td>US Dollar</td>
<td>1.0000000000</td>
<td>1.0000000000</td>
</tr>
<tr>
<td>EUR</td>
<td>Euro</td>
<td>0.8783121137</td>
<td>1.1385474303</td>
</tr>
<tr>
<td>GBP</td>
<td>British Pound</td>
<td>0.6956087704</td>
<td>1.4375896950</td>
</tr>
<tr>
<td>INR</td>
<td>Indian Rupee</td>
<td>66.1909310706</td>
<td>0.0151078098</td>
</tr>
<tr>
<td>AUD</td>
<td>Australian Dollar</td>
<td>1.3050318080</td>
<td>0.7662648480</td>
</tr>
<tr>
<td>CAD</td>
<td>Canadian Dollar</td>
<td>1.2997506294</td>
<td>0.7693783541</td>
</tr>
<tr>
<td>SGD</td>
<td>Singapore Dollar</td>
<td>1.4173961522</td>
<td>0.7418969172</td>
</tr>
<tr>
<td>CHF</td>
<td>Swiss Franc</td>
<td>451582</td>
<td>1.0427037678</td>
</tr>
<tr>
<td>MYR</td>
<td>Malaysian Ringgit</td>
<td>3.8700000000</td>
<td>0.2583979328</td>
</tr>
<tr>
<td>JPY</td>
<td>Japanese Yen</td>
<td>112.5375383115</td>
<td>0.0088859239</td>
</tr>
<tr>
<td>CNY</td>
<td>Chinese Yuan</td>
<td>6.449240933</td>
<td>0.1550570076</td>
</tr>
<tr>
<td>NZD</td>
<td></td>
<td></td>
<td>0.6906068347</td>
</tr>
<tr>
<td>THB</td>
<td></td>
<td></td>
<td>0.0284895968</td>
</tr>
<tr>
<td>HUF</td>
<td>Hungarian Forint</td>
<td>275.7012427385</td>
<td>0.0036271146</td>
</tr>
<tr>
<td>AED</td>
<td>Emirati Dirham</td>
<td>3.6730000000</td>
<td>0.2722570106</td>
</tr>
<tr>
<td>HKD</td>
<td>Hong Kong Dollar</td>
<td>7.7563973683</td>
<td>0.1289258341</td>
</tr>
<tr>
<td>MXN</td>
<td>Mexican Peso</td>
<td>17.3168505322</td>
<td>0.0577472213</td>
</tr>
<tr>
<td>ZAR</td>
<td>South African Rand</td>
<td>14.7201431400</td>
<td>0.0679341220</td>
</tr>
</tbody>
</table>

1 Ringgit = 0.2583979328 Dollar

1 Dollar = 3.87479406049 * 0.2583979328 Dollar
= 1.00123877526 Dollar

Free Money!
Best Currency Exchange

Best way to transfer USD to MYR:
Given a graph of currencies (edges are exchange rates)
find the shortest path by product of edge weights

$$\max_p \prod_{e \in p} w(e)$$

Invert edge weights to make it a minimization problem
Best Currency Exchange

Best way to transfer USD to MYR:
Given a graph of currencies (edges are exchange rates)
find the shortest path by product of edge weights

\[
\min_p \prod_{e \in p} \frac{1}{w(e)}
\]

Take log of edge weights to make summation

Diagram:

- USD to Euro: 1.14
- Euro to AED: 3.87
- AED to MYR: 0.95
- USD to MYR: 0.26
Best Currency Exchange

Best way to transfer USD to MYR:

Given a graph of currencies (edges are exchange rates)
find the shortest path by product of edge weights

$$\min_p \sum_{e \in p} \log \frac{1}{w(e)}$$

Now a shortest path problem!

Negative Edge Weights!
Problem with negative edges

There is no shortest path from A to I!

What we need: an algorithm that finds the shortest path in graphs with negative edge weights (if one exists)

$$w(C, F, D, C) = -1$$

Weight if we take the cycle 0 times: 31
Weight if we take the cycle 1 time: 30
Weight if we take the cycle 2 times: 29
...
Note

Any simple path has at most $V - 1$ edges

Pigeonhole Principle!

More than $V - 1$ edges means some node appears twice (i.e., there is a cycle)

If there is a shortest path of more than $V - 1$ edges, there is a negative weight cycle
Bellman-Ford

Idea: Use Dynamic Programming!

Short\( (i, v) \) = weight of the shortest path from \( s \) to \( v \) using at most \( i \) edges

Two options:

1. A path of \( i - 1 \) edges from \( s \) to some node \( x \), then edge \( (x, v) \)

2. OR

3. A path from \( s \) to \( v \) of at most \( i - 1 \) edges

\[
\text{Short}(i, v) = \min\left\{ \min_x (\text{Short}(i - 1, x) + w(x, v)) \right. \\
\left. \text{Short}(i - 1, v) \right\}
\]
Bellman Ford

Start node is E
Initialize all others to $\infty$

$Short(i, v) = \text{weight of the shortest path from } s \text{ to } v \text{ using at most } i \text{ edges}$

$Short(i, v) = \min_x \left\{ \text{Short}(i - 1, x) + w(x, v) \right\}
\begin{array}{c|cccccccc}
  i  \quad v & A & B & C & D & E & F & G & H & I \\
  \hline
  0 & \infty & \infty & \infty & \infty & 0 & \infty & \infty & \infty & \infty \\
  1 & & & & \infty & \infty & \infty & \infty & \infty & \infty \\
  2 & & & & & \infty & \infty & \infty & \infty & \infty \\
  3 & & & & & & \infty & \infty & \infty & \infty \\
  4 & & & & & & & \infty & \infty & \infty \\
  5 & & & & & & & & \infty & \infty \\
  6 & & & & & & & & & \infty \\
  7 & & & & & & & & & \\
\end{array}$
Start node is E
Initialize all others to $\infty$

$$Short(i, v) = \min_x \left( \min(Short(i - 1, x) + w(x, v)) \right)$$

$$Short(i, v) = \min\left\{ \begin{align*}
\min_x (Short(i - 1, x) + w(x, v)) \\
Short(i - 1, v)
\end{align*} \right\}$$

<table>
<thead>
<tr>
<th>$i$</th>
<th>$v = A$</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
<th>I</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>0</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
</tr>
<tr>
<td>1</td>
<td>$\infty$</td>
<td>8</td>
<td>$\infty$</td>
<td>7</td>
<td>0</td>
<td>$\infty$</td>
<td>5</td>
<td>5</td>
<td>$\infty$</td>
</tr>
</tbody>
</table>
Bellman Ford

Start node is E
Initialize all others to $\infty$

$$\text{Short}(i, v) = \begin{cases} \min \limits_x (\text{Short}(i - 1, x) + w(x, v)) \\ \text{Short}(i - 1, v) \end{cases}$$
Bellman Ford

Start node is E
Initialize all others to $\infty$

$Short(i, v) = \text{weight of the shortest path from } s \text{ to } v \text{ using at most } i \text{ edges}$

$Short(i, v) = \min_x \left\{ \min(Short(i - 1, x) + w(x, v)) \right\}$

$Short(i - 1, v)$

<table>
<thead>
<tr>
<th>$i$</th>
<th>$v$</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
<th>I</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>0</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>$\infty$</td>
<td>8</td>
<td>$\infty$</td>
<td>7</td>
<td>0</td>
<td>$\infty$</td>
<td>5</td>
<td>5</td>
<td>$\infty$</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>18</td>
<td>8</td>
<td>4</td>
<td>7</td>
<td>0</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>-8</td>
<td>8</td>
<td>4</td>
<td>7</td>
<td>0</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Bellman Ford

Start node is E
Initialize all others to $\infty$

$$\text{Short}(i, v) = \begin{cases} \text{weight of the shortest path from } s \text{ to } v \text{ using at most } i \text{ edges} \\ \min_x \text{Short}(i-1, x) + w(x, v) \\ \text{Short}(i-1, v) \end{cases}$$

<table>
<thead>
<tr>
<th>$i$</th>
<th>$v =$</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
<th>I</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>0</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
</tr>
<tr>
<td>1</td>
<td>$\infty$</td>
<td>8</td>
<td>$\infty$</td>
<td>7</td>
<td>0</td>
<td>$\infty$</td>
<td>5</td>
<td>5</td>
<td>$\infty$</td>
<td>$\infty$</td>
</tr>
<tr>
<td>2</td>
<td>18</td>
<td>8</td>
<td>4</td>
<td>7</td>
<td>0</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>7</td>
<td>$\infty$</td>
</tr>
<tr>
<td>3</td>
<td>-8</td>
<td>8</td>
<td>4</td>
<td>7</td>
<td>0</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>7</td>
<td>$\infty$</td>
</tr>
<tr>
<td>4</td>
<td>-8</td>
<td>8</td>
<td>4</td>
<td>7</td>
<td>0</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>7</td>
<td>$\infty$</td>
</tr>
<tr>
<td>5</td>
<td>-8</td>
<td>8</td>
<td>4</td>
<td>7</td>
<td>0</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>7</td>
<td>$\infty$</td>
</tr>
<tr>
<td>6</td>
<td>-8</td>
<td>8</td>
<td>4</td>
<td>7</td>
<td>0</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>7</td>
<td>$\infty$</td>
</tr>
<tr>
<td>7</td>
<td>-8</td>
<td>8</td>
<td>4</td>
<td>7</td>
<td>0</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>7</td>
<td>$\infty$</td>
</tr>
</tbody>
</table>
Bellman Ford: Negative cycles

Start node is E
Initialize all others to $\infty$

$$Short(i, v) = \text{weight of the shortest path from } s \text{ to } v \text{ using at most } i \text{ edges}$$

$$Short(i, v) = \min_{x} \left\{ \min(Short(i - 1, x) + w(x, v)), \text{Short}(i - 1, v) \right\}$$

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
<th>I</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>0</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
</tr>
<tr>
<td>1</td>
<td>$\infty$</td>
<td>8</td>
<td>7</td>
<td>0</td>
<td>$\infty$</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>7</td>
<td>0</td>
<td>4</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>5</td>
<td>0</td>
<td>4</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>5</td>
<td>0</td>
<td>2</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>5</td>
<td>0</td>
<td>2</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>3</td>
<td>4</td>
<td>0</td>
<td>2</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>3</td>
<td>4</td>
<td>0</td>
<td>1</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

If we computed row V, values change
There is a negative weight cycle!
Bellman Ford Run Time

Initialize array $Short[V][V]$ \[ V^2 \]
Initialize $Short[0][v] = \infty$ for each vertex \[ V \]
Initialize $Short[0][s] = 0$ \[ 1 \]
For $i = 1, \ldots, V - 1$: \[ V \text{ times} \]
   for each $e = (x, y) \in E$: \[ E \text{ times} \]
   $Short[i][y] = \min\{$
   $\begin{align*}
   \text{Short}[i - 1][x] + w(x, y), \quad &\text{1} \\
   \text{Short}[i - 1][y] \}
   \end{align*}$

\[ \Theta(V^2 + EV) \]
\[ \Theta(EV) \]
Why Use Bellman-Ford?

• Dijkstra’s:
  – only works for positive edge weights
  – Run Time: $\Theta(E \log V)$
  – Not good for dynamic graphs (where edge weights are variable)
    • Must recalculate “from scratch”

• Bellman-Ford:
  – Works for negative edge weights
  – Run Time: $\Theta(E \cdot V)$
  – More efficient for dynamic graphs
    • $\Theta(E)$ time to recalculate
Bellman Ford: Dynamic

Each node will update its neighbors if edge weight changes.

\[
Short(i, v) = \min_{x} \left\{ \min(Short(i - 1, x) + w(x, v)) \right\}
\]

<table>
<thead>
<tr>
<th>(i)</th>
<th>(v) = A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
<th>I</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>0</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>(\infty)</td>
</tr>
<tr>
<td>1</td>
<td>(\infty)</td>
<td>5</td>
<td>(\infty)</td>
<td>7</td>
<td>0</td>
<td>(\infty)</td>
<td>5</td>
<td>5</td>
<td>(\infty)</td>
</tr>
<tr>
<td>2</td>
<td>18</td>
<td>8</td>
<td>4</td>
<td>7</td>
<td>0</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td>3</td>
<td>-8</td>
<td>8</td>
<td>4</td>
<td>7</td>
<td>0</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td>4</td>
<td>-8</td>
<td>8</td>
<td>4</td>
<td>7</td>
<td>0</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td>5</td>
<td>-8</td>
<td>8</td>
<td>4</td>
<td>7</td>
<td>0</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td>6</td>
<td>-8</td>
<td>8</td>
<td>4</td>
<td>7</td>
<td>0</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td>7</td>
<td>-8</td>
<td>8</td>
<td>4</td>
<td>7</td>
<td>0</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>7</td>
</tr>
</tbody>
</table>
Bellman Ford: Dynamic

Each node will update its neighbors if edge weight changes.

\[
\text{Short}(i, v) = \min \begin{cases} \\
\text{weight of the shortest path from } s \text{ to } v \text{ using at most } i \text{ edges} \\
\text{Short}(i - 1, x) + w(x, v) \\
\text{Short}(i - 1, v)
\end{cases}
\]

| \( i \) | \( v = \) | A | B | C | D | E | F | G | H | I |
|---|---|---|---|---|---|---|---|---|---|
| 0 | \( \infty \) | \( \infty \) | \( \infty \) | \( \infty \) | 0 | \( \infty \) | \( \infty \) | \( \infty \) | \( \infty \) |
| 1 | \( \infty \) | 5 | \( \infty \) | 7 | 0 | \( \infty \) | 5 | 5 | \( \infty \) |
| 2 | 15 | 5 | 1 | 7 | 0 | 4 | 5 | 5 | 7 |
| 3 | -8 | 5 | 1 | 7 | 0 | 4 | 3 | 5 | 7 |
| 4 | -8 | 5 | 1 | 7 | 0 | 4 | 3 | 5 | 7 |
| 5 | -8 | 5 | 1 | 7 | 0 | 4 | 3 | 5 | 7 |
| 6 | -8 | 5 | 1 | 7 | 0 | 4 | 3 | 5 | 7 |
| 7 | -8 | 5 | 1 | 7 | 0 | 4 | 3 | 5 | 7 |
Each node will update its neighbors if edge weight changes

$$Short(i, v) = \begin{cases} \text{weight of the shortest path from } s \text{ to } v \text{ using at most } i \text{ edges} \\ \min \left( \min_x (Short(i - 1, x) + w(x, v)) \right) \\ Short(i - 1, v) \end{cases}$$

<table>
<thead>
<tr>
<th>i</th>
<th>v = A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
<th>I</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>0</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
</tr>
<tr>
<td>1</td>
<td>$\infty$</td>
<td>5</td>
<td>$\infty$</td>
<td>7</td>
<td>0</td>
<td>$\infty$</td>
<td>5</td>
<td>5</td>
<td>$\infty$</td>
</tr>
<tr>
<td>2</td>
<td>15</td>
<td>5</td>
<td>1</td>
<td>7</td>
<td>0</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td>3</td>
<td>-11</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>0</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td>4</td>
<td>-11</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>0</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td>5</td>
<td>-11</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>0</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td>6</td>
<td>-11</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>0</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td>7</td>
<td>-11</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>0</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>7</td>
</tr>
</tbody>
</table>
Find the quickest way to get from each place to every other place

Given a graph $G = (V, E)$ for each start node $s \in V$ and destination node $v \in V$ find the least-weight path from $s \rightarrow v$
All-Pairs Shortest Path

- Can clearly be found in $O(V^2 \cdot E)$
  - Run Bellman-Ford with each node being the start

\[
\text{for each } s \in V: \quad V \text{ times} \\
\text{BellmanFord}(s) \quad O(V \cdot E)
\]
Floyd-Warshall

- Finds all-pairs shortest paths in $\Theta(V^3)$
- Uses Dynamic Programming

$$\text{Short}(i, j, k) = \text{the length of the shortest path from node } i \text{ to node } j \text{ using only intermediate nodes } 1, \ldots, k$$

Two options:

1. Shortest path from $i$ to $j$ includes $k$
   $$\text{OR}$$
   Shortest path from $i$ to $j$ excludes $k$

$$\text{Short}(i, j, k) = \min \left[ \text{Short}(i, k, k - 1) + \text{Short}(k, j, k - 1), \text{Short}(i, j, k - 1) \right]$$
Shortest Paths Review

• Single Source Shortest Paths
  – Dijkstra’s Algorithm $\Theta(E \log V)$
    • No negative edge weights
  – Bellman-Ford $\Theta(EV)$
    • First Dynamic Programming Algorithm
    • Allows negative edge weights (finds negative weight cycles)
    • Update memory in $\Theta(E)$ time on edge weight updates

• All Pairs Shortest Paths
  – Floyd-Warshall $\Theta(V^3)$
    • Allows negative edge weights