Shortest Path Problem

Shortest path in a network.
- Directed graph $G = (V, E)$.
- Source $s$, destination $t$.
- Length $\ell_e = \text{length of edge } e$.

Shortest path problem: find shortest (directed) path from $s$ to $t$.
Single source shortest path problem: find shortest directed path from $s$ to every node in $V$

Cost of path $s$-2-3-5-$t$

\[
= 9 + 23 + 2 + 16
\]

\[
= 48.
\]
Dijkstra's algorithm.

- Maintain a set of explored nodes $S$ for which we have determined the shortest path distance $d(u)$ from $s$ to $u$.
- Initialize $S = \{s\}$, $d(s) = 0$.
- Repeatedly choose unexplored node $v$ which minimizes

$$
\pi(v) = \min_{e = (u,v): u \in S} d(u) + l_e,
$$

add $v$ to $S$, and set $d(v) = \pi(v)$.

![Diagram of Dijkstra's Algorithm](image)

shortest path to some $u$ in explored part, followed by a single edge $(u, v)$
Dijkstra's Algorithm

Dijkstra's algorithm.

- Maintain a set of explored nodes $S$ for which we have determined the shortest path distance $d(u)$ from $s$ to $u$.
- Initialize $S = \{s\}$, $d(s) = 0$.
- Repeatedly choose unexplored node $v$ which minimizes

$$
\pi(v) = \min_{e = (u,v): u \in S} d(u) + l_e,
$$

add $v$ to $S$, and set $d(v) = \pi(v)$.

shortest path to some $u$ in explored part, followed by a single edge $(u,v)$.
Dijkstra's Algorithm

Dijkstra's algorithm.

- Dijkstra’s algorithm is a greedy algorithm.
  - What defines a “step” towards our goal?
  - What is our optimization criteria at each step?

- The result is a globally optimal solution to the SSSP problem!

- How to implement?
Dijkstra's Algorithm: Implementation

For each unexplored node, explicitly maintain $\pi(v) = \min_{e = (u,v): u \in S} \{ d(u) + l_e \}$.

- Next node to explore = node with minimum $\pi(v)$.
- When exploring $v$, for each incident edge $e = (v, w)$, update
  $$\pi(w) = \min \{ \pi(w), \pi(v) + l_e \}.$$ 

**Efficient implementation.** Maintain a priority queue of unexplored nodes, prioritized by $\pi(v)$.

<table>
<thead>
<tr>
<th>Priority Queue Operation</th>
<th>Array Time Complexity</th>
<th>Binary Heap Time Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Insert</td>
<td>$n$</td>
<td>$\log n$</td>
</tr>
<tr>
<td>ExtractMin</td>
<td>$n$</td>
<td>$\log n$</td>
</tr>
<tr>
<td>ChangeKey</td>
<td>$1$</td>
<td>$\log n$</td>
</tr>
<tr>
<td>IsEmpty</td>
<td>$1$</td>
<td>$1$</td>
</tr>
<tr>
<td>Total</td>
<td>$n^2$</td>
<td>$m \log n$</td>
</tr>
</tbody>
</table>
Dijkstra's Algorithm Pseudocode

\textbf{Dijkstra}(G, s):

\begin{itemize}
\item \text{let } T \leftarrow (\{s\}, \emptyset)
\item \text{let } PQ \text{ be an empty priority queue}

\text{for each neighbor } v \text{ of } s, \text{ add edge } (s,v) \text{ to } PQ \text{ with priority } l(e)
\end{itemize}

\text{while } T \text{ doesn't have all vertices of } G \text{ and } PQ \text{ is non-empty:}

\begin{itemize}
\item \text{repeat } \{\n\item \hspace{1em} e \leftarrow PQ.\text{removeMin()} \text{ // skip edges with both ends in } T
\item \hspace{1em} \} \text{ until } PQ \text{ is empty or } e=(u,v) \text{ for } u \in T, \ v \notin T
\item \text{if } e=(u,v) \text{ for } u \in T, \ v \notin T
\item \hspace{1em} \text{add } e \text{ (and } v) \text{ to } T
\item \hspace{1em} \text{for each neighbor } w \text{ of } v
\item \hspace{2em} \text{add edge } (v,w) \text{ to } PQ \text{ with weight/key } d(s,v) + l(v,w)
\end{itemize}
Dijkstra's Algorithm: Proof of Correctness

Invariant. For each node \( u \in S \), \( d(u) \) is the length of the shortest \( s-u \) path.

Pf. (by induction on \( |S| \))

Base case: \( |S| = 1 \) and \( d(s)=0 \), which is true.

Inductive hypothesis: Assume true for \( |S| = k \leq n \). Consider \( |S|=k+1 \)

- Let \( v \) be last node added to \( S \), and let \( u-v \) be the chosen edge.
- By inductive hypothesis, all nodes in \( S-\{v\} \) have correct shortest path dis.
- \textbf{Claim:} the \( s-u \) path plus \( (u, v) \) is an \( s-v \) path of shortest length \( \pi(v) \).
  - Consider any \( s-v \) path \( P \). We'll see that it's no shorter than \( \pi(v) \).
  - Let \( x-y \) be the first edge in \( P \) that leaves \( S-\{v\} \), and let \( P' \) be the subpath to \( x \).

\[
\ell(P) \geq \ell(P') + \ell(x,y) \geq d(x) + \ell(x,y) \geq \pi(y) \geq \pi(v)
\]

- nonnegative weights
- inductive hypothesis
- defn of \( \pi(y) \)
- Dijkstra chose \( v \) instead of \( y \)