Dijkstra's shortest path

\[ G = (V, E) \text{ a weighted graph} \]

\[ \omega(u, v) \]

Always pick the node with the shortest distance from the source.

For further expansion.

Negative edge causes violation of the above principle.

\[ \Rightarrow \text{Dijkstra's fails!} \]

**I)** larger path from \( s \) to any other vertex has at most \(|V| - 1\) edges.

**II)** update the \( \text{dist}(x) \) at each step one complete traversal over the edge.

Bellman-Ford \((G, s)\):

for all \( u \in V \):

\[ \text{dist}(u) = \infty \] \[ \text{prev}(u) = E \]

\( \text{list} \leftarrow \text{keep track of possible shortest path} \)

\[ \text{dist}(s) = 0 \]

for \( \text{iter} = 1 \ldots |V| - 1 \)

for all \( e = (u, v) \in E \)

\[ \text{update} (e) \]
for all \( e = (u, v) \in E \) do one more round over \( e \in E \) if any dist decreases return \( \text{FAIL/WN6 cycle} \)

\[
\begin{align*}
O(1v1) &= O(1v1) \text{ spare} \Rightarrow O(1v1^2) \text{ due}\n
\text{Negative cycle: a cycle whose sum of edge weights is negative}
\end{align*}
\]

\[
\begin{align*}
\text{DAGs: Shorter paths} \Rightarrow \text{topological sort} \Rightarrow \text{then update the distance in that order}
\end{align*}
\]

C → dist = 6

\( \text{prove: } C \in \{A, B, E, F, G\} \) 

\( \text{route: } S, G, F, A, E, G, C \).
Greedy Algorithms

Pick the "best" option at each step

Local best

Minimum Spanning Tree (MST) \( G = (V, E) \)

Given a weighted undirected graph, find a tree with the least
sum of edge weights

Spanning \( \equiv \) must include all the vertices

Spanning Tree: \( T = (V', E') \)

\( V' = V \) (Span all vertices)
\( E' \subseteq E \) (\( E' \) is a subset)

Tree \( \equiv \) \( |V| - 1 \) edge and connected

![Graph diagram]

Spanning tree
\( T_1: \{ ab, bc, bd, df, fe \} \)
\( \text{Cost}(T_1): 23 \)
\( = 1 + 2 + 4 + 5 \)

\( T_2: \{ ab, ac, cd, cf, fe \} \)
\( \text{Cost}(T_2): 4 + 1 + 2 + 4 + 5 \)
\( = 16 \)

Initial idea: (greedy)

1) sort the edge in increasing order
Prim's Algorithm (G):

for \( u \in V \):

\[ \text{Cut}(u) = \infty \]
\[ \text{prev}(u) = \phi \]

Pick any source node \( s \in V \)
\[ \text{Cut}(s) = 0 \]
\( PQ = [s] \) \( \leftarrow \) priority queue

while \( PQ \) not empty:

1. \( \text{sort} \) \( PQ \)
2. \( \text{pick} \) the next best edge \( \text{for} \; T \) (exclude an edge of cycle)
3. \( \text{keep} \; T \) connected (grow \( T \) by adding one more edge from the existing node in \( T \))

\( T = \phi \) works!

When can this fail?

\[ \begin{array}{c}
ac \rightarrow 1 \leftarrow \\
cd \rightarrow 2 \leftarrow \times
\end{array} \]
\[ PQ = [s] \leftarrow \text{priority queue} \]

while \( PQ \) not empty:

\[
\begin{align*}
    u &= \text{deleteMin}(PQ) \leftarrow \text{picks the lowest cost vertex (greedy)} \\
    \text{for all } (y, v) \in E \\
    \quad &\text{if } c_{uv} > \frac{w(y, v)}{\text{prv}(v)} \\
    \quad &\text{consider current edge weight} \\
    \quad &c_{uv} = w(y, v) \\
    \quad &\text{prv}(v) = u \\
    \quad &\text{decreaseKey}(PQ, v, c_{uv})
\end{align*}
\]

\[ O(|E|) \text{ decrease key} \]
\[ O(|V|) \text{ deleteMin} \]

\[ O\left(\frac{|E| + |V|}{\log |V|}\right) \text{ via binary heap} \]

\[ \downarrow \]

\[ O\left(|E| \log |V|\right) \]

\[ T = \{ \text{red edge} \} \]

- \{\text{look at prv for each node}\}
- \{\text{e -> f 1}\}
- \{\text{f -> c 2}\}
- \{\text{c -> b 1}\}
- \{\text{d -> b 2}\}
- \{\text{c -> b 3}\}

Correctness?

- **Cut property**
  - cut in a graph is simply a partition of the vertex set \( V \) into two parts
  - \( S \subseteq V \)
  - \( S \cup \bar{S} = V \)
  - \( S, \bar{S} \)
  - \( S \subseteq V \)
  - \( S \cup \bar{S} = V \)
  - Complement of \( S \)
  - \( \bar{S} = V - S \)
1) If the only one edge between $S \cup V - S$, then we may include it.

2) There is more than 1 edge between $S$ and $V - S$.

Let $e$ be the min cost/worst edge.

$w(e) \leq w(e')$

Show that changing the lower cost edge is guaranteed to give us an MST $T'$

Assume some MST $T$ that does not include $e$, then it must include $e'$ (i.e., some other edge that crosses the cut).

Cost of other MST $T'$:

$\text{cost}(T') = w(S) + w(V - S) + w(e') \geq w(S) + w(V - S) + w(e)$

we can construct a new MST $T''$ that includes $e$ instead of $e'$

true for any cut in $G$ that respects $T$

$\exists$ no vertex that crosses from $S$ to $V - S$.

$\therefore$ great choice.
\[ \Rightarrow \text{ greedy choice (edge with the least weight) must be part of some MST} \]

\[ \text{guarantees correctness} \]

\[ \text{Kruskal's method: drop the connectivity constraint} \]

\[ S = \{a, c, e, f\} \]

\[ V - S = \{b, d\} \]

\[ S = \{a, c, d, e, f\} \]

\[ V - S = \{b\} \]

Maintain a forest

? \[ \text{Check for cycle} \]

? \[ \text{Check for disconnection} \]