Shortest Paths: Intro

Homework preview
Weighted graphs
General Approach
Negative edges
Optimal substructure

Motivation
Shortest way to drive from A to B
Google maps "get directions"

Formulation: Problem on a weighted graph $G(V,E)$

$W: E \rightarrow \mathbb{R}$

Two algorithms: Dijkstra $O(V \log V + E)$
assumes non-negative edge weights
Bellman Ford $O(VE)$
general algorithm
Problem Set 5

- Use Dijkstra to find shortest path from CalTech to MIT
  - See "CalTech Cannon Hack" photos @ web.mit.edu
  - See Google maps from CalTech to MIT

- Model as a weighted graph \( G(V, E), w: E \rightarrow \mathbb{R} \)
  - \( V \) = vertices (street intersections)
  - \( E \) = edges (streets, roads); directed edges (one way roads)
  - \( w(u, v) = \text{weight of edge from } u \text{ to } v \) (distance, toll)

---

path \( p = \langle v_0, v_1, \ldots, v_k \rangle \)

\((v_i, v_{i+1}) \in E \text{ for } 0 \leq i < k\)

\( w(p) = \sum_{i=0}^{k-1} w(v_i, v_{i+1}) \)

Notation: \( v_0 \xrightarrow{p} v_k \) means \( p \) is a path
from \( v_0 \) to \( v_k \)

\((v_0)\) is a path from \( v_0 \) to \( v_0 \) of weight 0

Define: Shortest path weight from \( u \) to \( v \) as

\( \delta(u, v) = \begin{cases} 
\min \{ w(p): u \xrightarrow{p} v \} & \text{if any such path exists} \\
\infty & \text{otherwise (v unreachable from u)}
\end{cases} \)
Single Source Shortest Paths

Given $G = (V, E)$, $w$ and a source vertex $s$, find $\delta(s, v)$ [and the best path] from $s$ to each $v \in V$.

Data structures: 

- $d[v]$: value inside circle
  - $0$ if $v = s$
  - $\infty$ otherwise

- $\delta(s, v)$: at end

- $d[v] \geq \delta(s, v)$ at all times,
- $d[v]$ decreases as we find better paths to $v$
- $\Pi[v]$: predecessor on best path to $v$, $\Pi[s] = \text{NIL}$

**EXAMPLE**

- Edges give predecessor $\Pi$ relationships.
NEGATIVE-WEIGHT EDGES

- Natural in some applications (e.g., logarithms used for weights)
- Some algorithms disallow negative weight edges (e.g., Dijkstra)
- If you have negative weight edges, you might also have negative weight cycles
  \[ \checkmark \]
  may make certain shortest paths undefined!

EXAMPLE

![Graph diagram]

\[ B \rightarrow D \rightarrow C \] has weight \(-6 + 2 + 3 = -1 < 0\)!

Shortest path \( S \rightarrow C \) (or \( B, D, E \)) is undefined can go around \( B \rightarrow D \rightarrow C \) as many times as you like.

Shortest path \( S \rightarrow A \) is defined and has weight 2.

If negative wt edges are present, S.P. algorithm should find neg wt cycles (e.g., Bellman Ford)
**General Structure of S.P. Algs (no neg cycles)**

Initialize: \( \text{for } v \in V: \quad \text{d}[v] \leftarrow 0 \)

\( \text{d}[s] \leftarrow 0 \)

\( \text{\Pi}[v] \leftarrow \text{NIL} \)

Main: \( \text{repeat:} \)

Select edge \((u, v)\) \(\text{[somehow]}\)

"Relax" edge \((u, v)\)

\[ \begin{align*}
\text{if } \text{d}[v] &> \text{d}[u] + w(u, v) : \\
\text{d}[v] &\leftarrow \text{d}[u] + w(u, v) \\
\text{\Pi}[v] &\leftarrow u \\
\text{until all edges have } \text{d}[v] &\leq \text{d}[u] + w(u, v)
\end{align*} \]

**Complexity**

Termination? (needs to be shown even w/o negative cycles)

Could be exponential time with poor choice of edges.

\[
\begin{align*}
T(0) &= 0 \\
T(n+2) &= 3 + 2T(n) \\
T(n) &= \Theta(2^{n/2})
\end{align*}
\]
**Optimal Substructure**

**Theorem:** Subpaths of shortest paths are shortest paths.

Let \( p = <v_0, v_1, \ldots, v_k> \) be a shortest path.

Let \( p_{ij} = <v_i, v_{i+1}, \ldots, v_j> \) for \( 0 \leq i \leq j \leq k \).

Then \( p_{ij} \) is a shortest path.

**Proof:**

\[ p = v_0 \xrightarrow{P_{0j}} v_i \xrightarrow{P_{ij}} v_j \xrightarrow{P_{jk}} v_k \]

If \( p_{ij} \) is shorter than \( p_{ij'} \), cut out \( p_{ij} \) and replace with \( p_{ij'} \). Result is shorter than.

Contradiction.

---

**Triangle Inequality**

**Theorem:** For all \( u, v, x \in X \), we have \( d(u, v) \leq d(u, x) + d(x, v) \).

**Proof:**

\[ d(u, v) \leq d(u, x) + d(x, v) \]