Lecture 16: Shortest Paths II: Bellman-Ford

Lecture Overview

- Review: Notation
- Generic S.P. Algorithm
- Bellman Ford Algorithm
  - Analysis
  - Correctness

Recall:

\[ \text{path } p = \langle v_0, v_1, \ldots, v_k \rangle \]
\[ (v_i, v_{i+1}) \in E \quad 0 \leq i < k \]
\[ w(p) = \sum_{i=0}^{k-1} w(v_i, v_{i+1}) \]

Shortest path weight from \( u \) to \( v \) is \( \delta(u, v) \). \( \delta(u, v) \) is \( \infty \) if \( v \) is unreachable from \( u \), undefined if there is a negative cycle on some path from \( u \) to \( v \).

![Negative Cycle](image)

Figure 1: Negative Cycle

Generic S.P. Algorithm

Initialize:

- for \( v \in V \): \( d[v] \leftarrow \infty \)
- \( \Pi[v] \leftarrow \text{NIL} \)
- \( d[S] \leftarrow 0 \)

Main:

- repeat
  - select edge \((u, v)\) [somehow]
  - “Relax” edge \((u, v)\)
    \[
    \begin{cases}
    \text{if } d[v] > d[u] + w(u, v) : \\
    d[v] \leftarrow d[u] + w(u, v) \\
    \pi[v] \leftarrow u
    \end{cases}
    \]
- until you can’t relax any more edges or you’re tired or \ldots
Complexity:

Termination: Algorithm will continually relax edges when there are negative cycles present.

Figure 2: Algorithm may not terminate due to negative Cycles

Complexity could be exponential time with poor choice of edges.

Figure 3: Algorithm could take exponential time
5-Minute 6.006

Here’s what I want you to remember from 6.006 five years after you graduate

\[
T(n) = C_1 + C_2 T(n - C_3)
\]
\[
T(n) = C_1 + C_2 T(n / C_3)
\]

- **Exponential Bad** if \( C_2 > 1 \), trouble!
- **Polynomial Good**
  - if \( C_2 > 1 \) okay provided \( C_3 > 1 \)
  - if \( C_3 > 1 \)
  - **Divide & Conquer**

![Figure 4: Exponential vs. Polynomial](image)

**Bellman-Ford\((G, W, S)\)**

Initiate ()
for \( i = 1 \) to \(|v| - 1\)
  for each edge \((u, v) \in E\):
    Relax\((u, v)\)
  for each edge \((u, v) \in E\)
    do if \( d[v] > d[u] + w(u, v) \)
      then report a negative-weight cycle exists

At the end, \( d[v] = \delta(s, v) \), if no negative-weight cycles

![Figure 5: The numbers in circles indicate the order in which the \( \delta \) values are computed](image)
Theorem:
If $G = (V, E)$ contains no negative weight cycles, then after Bellman-Ford executes $d[v] = \delta(u, v)$ for all $v \in V$.

Proof:
$v \in V$ be any vertex. Consider path $p$ from $s$ to $v$ that is a shortest path with minimum number of edges.

\[
\delta(s, v) = \delta(s, v_1) + w(v_1, v)
\]

Figure 6: Illustration for proof

Initially $d[v_0] = 0 = \delta(S, V_0)$ and is unchanged since no negative cycles.
After 1 pass through $E$, we have $d[v_1] = \delta(s, v_1)$
After 2 passes through $E$, we have $d[v_2] = \delta(s, v_2)$
After $k$ passes through $E$, we have $d[v_k] = \delta(s, v_k)$
No negative weight cycles $\implies p$ is simple $\implies p$ has $\leq |V| - 1$ edges

Corollary
If a value $d[v]$ fails to converge after $|V| - 1$ passes, there exists a negative-weight cycle reachable from $s$. 