CMPSCI 611: Advanced Algorithms
Lecture 9: Dijkstra’s Algorithm

Andrew McGregor
Shortest Paths

Let $G = (V, E)$ be a directed graph with weights $w : E \rightarrow \mathbb{R}^+$. 

**Definition**
For path $p = (v_1, \ldots, v_k)$ be a path, define 

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

The *shortest path* between $u$ and $v$ is 

$$\delta(u, v) = \min \{ w(p) : p \text{ is a path from } u \text{ to } v \}$$

if there is a path from $u$ to $v$ and $\infty$ otherwise.
Dijkstra’s Warm-Up

Single-Source Problem: Given \( s \in V \), find \( \delta(s, v) \) for all \( v \in V \).

Dijkstra’s algorithm solves problem if all edges are non-negative:

- Maintains array \( (d[v] : v \in V) \) where \( d[v] \) will always be \( \infty \) or the length of some path from \( s \) to \( v \), not necessarily the shortest. Hence,

\[
d[v] \geq \delta(s, v)
\]

- Maintains a set of processed vertices \( R \). We’ll prove that for all \( v \in R \):

\[
d[v] = \delta(s, v)
\]
Dijkstra's Algorithm

Algorithm

1. \( d[s] = 0 \) and for \( s \neq v \):
   \[
   d[v] = w(s, v) \text{ if } (s, v) \in E \text{ and } \infty \text{ otherwise}
   \]

2. \( R \leftarrow \{s\} \)

3. While \( |R| < |V| \):
   3.1 \( u \leftarrow \arg\min_{v \notin R} d[v] \)
   3.2 \( R \leftarrow R + u \)
   3.3 For each \( v \notin R \) that is a neighbor of \( u \):
      \[
      d[v] = \min(d[u] + w(u, v), d[v])
      \]

Running Time: \( O(|V|^2) \) for simple implementation but can be improved.
Correctness of Algorithm

The correctness of the algorithm follows because a) $d[v]$ never increases, b) $d[v] \geq \delta(s, u)$ at all times, and c) appealing to the following lemma:

**Lemma**

*When $u$ is added to $R$, $d[u] = \delta(s, u)$*
When \( u \) gets added to \( R \), \( d[u] \) is correct

Let \( d_u[v] \) be value of \( d[v] \) just before \( u \) is chosen as minimum.

**Lemma**

*For all \( u \), \( d_u[u] = \delta(s, u) \)*

- By contradiction: Let \( u \) be first vertex put in \( R \) with \( d_u[u] > \delta(s, u) \)
- Consider a shortest path from \( s \) to \( u \). Let \( y \) be first vertex not in \( R \).
  - Note that \( y \) may or may not be \( u \).
- Claim: \( d_u[y] = \delta(s, y) \)
  - Let \( x \) be the predecessor of \( y \) on the path. Note that \( x \in R \).
  - \( d_x[x] = \delta(s, x) \) by assumption that \( u \) is first bad vertex.
  - After iteration where \( x \) is added to \( R \): \( d[y] \leq \delta(s, x) + w(x, y) \)
  - \( \delta(s, x) + w(x, y) = \delta(s, y) \) since path included shortest path to \( y \)
- Since \( y \) lies on shortest path to \( u \): \( \delta(s, y) \leq \delta(s, u) \)
- Putting above two lines together:
  \[
  d_u[y] = \delta(s, y) \leq \delta(s, u) < d_u[u]
  \]
- If \( y \neq u \): Contradiction because \( u \) was the next minimum and so
  \[
  d_u[u] \leq d_u[y]
  \]
- If \( y = u \): Contradiction because we deduced above \( d_u[y] = \delta(s, y) \)