Lecture 18: Shortest Paths IV - Speeding up Dijkstra

Lecture Overview

- Single-source single-target Dijkstra
- Bidirectional search

**DIJKSTRA** single-source, single-target

```
Initialize()
Q ← V[G]
while Q ≠ φ
    do u ← EXTRACT_MIN(Q) (stop if u = t!)
    for each vertex v ∈ Adj[u]
        do RELAX(u, v, w)
```

Observation: If only shortest path from s to t is required, stop when t is removed from Q, i.e., when u = t
Bi-Directional Search

Note: Speedup techniques covered here do not change worst-case behavior, but reduce the number of visited vertices in practice.

Bi-D Search

Alternate forward search from \( s \)

backward search from \( t \)

(follow edges backward)

\( d_f(u) \) distances for forward search

\( d_b(u) \) distances for backward search

Algorithm terminates when some vertex \( w \) has been processed, i.e., deleted from the queue of both searches, \( Q_f \) and \( Q_b \)

Subtlety: After search terminates, find node \( x \) with minimum value of \( d_f(x) + d_b(x) \).

\( x \) may not be the vertex \( w \) that caused termination as in example to the left!

Find shortest path from \( s \) to \( x \) using \( \Pi_f \) and shortest path backwards from \( t \) to \( x \) using \( \Pi_b \). Note: \( x \) will have been deleted from either \( Q_f \) or \( Q_b \) or both.

Figure 1: Bi-directional Search Idea.
Minimum value for $d_f(x) + d_b(x)$ over all vertices that have been processed in at least one search (see Figure 3):

\begin{align*}
    d_f(u) + d_b(u) &= 3 + 6 = 9 \\
    d_f(u') + d_b(u') &= 6 + 3 = 9 \\
    d_f(w) + d_b(w) &= 5 + 5 = 10
\end{align*}
Figure 3: Forward and Backward Search and Termination.