Graph algorithms

EECS 214

November 25, 2015

Take-aways

- How can we find shortest paths in a weighted digraph (directed graph)?
- How might we choose between SSSP algorithms?

Single-source shortest path

Problem: Given a starting vertex, find the shortest path to every other vertex.

Relaxation for SSSP

Main algorithmic idea: *relaxation*:

1. Start with over-approximations of the distances
2. Use known distances and edge weights to iteratively improve distances

One relaxation step

Suppose:

- The shortest *known* path to vertex v has length d_v
- The shortest *known* path to vertex u has length d_u
- There is an edge (v, u) with weight w

One relaxation step

Suppose:

- The shortest *known* path to vertex v has length d_v
- The shortest *known* path to vertex u has length d_u
- There is an edge (v, u) with weight w

If $d_v + w < d_u$, then we've found a shorter path to d_u: Use our path to v and then follow edge (v, u).

Relaxation, concretely

Suppose:

- `dist[v]` = `dv`
- `dist[u]` = `du`
- `get_edge(v, u)` = `w`

If `dv + w < du` then change `dist[u]` to `dv + w` and `pred[u]` to `v`.

Two SSSP Algorithms

Bellman–Ford

Relax every edge enough times that the information fully propagates ($|V| - 1$ passes).

Dijkstra

Relax every edge once, greedily choosing the lightest edge, to find paths in one pass.

Two SSSP Algorithms

Bellman–Ford

Relax every edge enough times that the information fully propagates ($|V| - 1$ passes). Handles negative edge weights.

Dijkstra

Relax every edge once, greedily choosing the lightest edge, to find paths in one pass. Fails on negative edge weights.

Take-aways

- How can we find shortest paths in a weighted digraph (directed graph)?
- How might we choose between SSSP algorithms?