The Bellman-Ford Algorithm

Single-Source Shortest Paths Revisited

Algorithms: Design and Analysis, Part II

The Single-Source Shortest Path Problem

Input: Directed graph $G = (V, E)$, edge lengths c_e for each $e \in E$, source vertex $s \in V$. [Can assume no parallel edges.]

Goal: For every destination $v \in V$, compute the length (sum of edge costs) of a shortest s-v path.

On Dijkstra's Algorithm

Good news: $O(m \log n)$ running time using heaps
(n = number of vertices, m = number of edges)

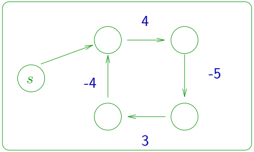
Bad news:

(1) Not always correct with negative edge lengths
[e.g. if edges $\mapsto$ financial transactions]

(2) Not very distributed (relevant for Internet routing)

Solution: The Bellman-Ford algorithm

On Negative Cycles



Question: How to define shortest path when G has a negative cycle?

Solution #1: Compute the shortest s-v path, with cycles allowed.

Problem: Undefined or $-\infty$. [will keep traversing negative cycle]

Solution #2: Compute shortest cycle-free s-v path.

Problem: NP-hard (no polynomial algorithm, unless P=NP)

Solution #3: (For now) Assume input graph has no negative cycles.

Later: Will show how to quickly check this condition.

Quiz

Quiz: Suppose the input graph G has no negative cycles. Which of the following is true? [Pick the strongest true statement.] [$n = \#$ of vertices, $m = \#$ of edges]

A) For every v, there is a shortest s-v path with $\leq n-1$ edges.

B) For every v, there is a shortest s-v path with $\leq n$ edges.

C) For every v, there is a shortest s-v path with $\leq m$ edges.

D) A shortest path can have an arbitrarily large number of edges in it.