18.4
All Pairs Shortest Paths

18.4.1
Problem definition and what we can already do

Shortest Path Problems

Shortest Path Problems

Input A (undirected or directed) graph $G = (V, E)$ with edge lengths (or costs). For edge $e = (u, v)$, $\ell(e) = \ell(u, v)$ is its length.

1. Given nodes s, t find shortest path from s to t.
2. Given node s find shortest path from s to all other nodes.
3. Find shortest paths for all pairs of nodes.

SSSP: Single-Source Shortest Paths

Single-Source Shortest Path Problems

Input A (undirected or directed) graph $G = (V, E)$ with edge lengths. For edge $e = (u, v)$, $\ell(e) = \ell(u, v)$ is its length.

1. Given nodes s, t find shortest path from s to t.
2. Given node s find shortest path from s to all other nodes.

Dijkstra's algorithm for non-negative edge lengths. Running time: $O((m + n) \log n)$ with heaps and $O(m + n \log n)$ with advanced priority queues.

Bellman-Ford algorithm for arbitrary edge lengths. Running time: $O(nm)$.

SSSP: Single-Source Shortest Paths

Single-Source Shortest Path Problems

Input A (undirected or directed) graph $G = (V, E)$ with edge lengths. For edge $e = (u, v)$, $\ell(e) = \ell(u, v)$ is its length.

1. Given nodes s, t find shortest path from s to t.
2. Given node s find shortest path from s to all other nodes.

Dijkstra's algorithm for non-negative edge lengths. Running time: $O((m + n) \log n)$ with heaps and $O(m + n \log n)$ with advanced priority queues.

Bellman-Ford algorithm for arbitrary edge lengths. Running time: $O(nm)$.

All-Pairs Shortest Paths

Using the shortest paths algorithms we already have...

All-Pairs Shortest Path Problem

Input A (undirected or directed) graph $G = (V, E)$ with edge lengths. For edge $e = (u, v)$, $\ell(e) = \ell(u, v)$ is its length.

1. Find shortest paths for all pairs of nodes.

Apply single-source algorithms n times, once for each vertex.

1. Non-negative lengths. $O(nm \log n)$ with heaps and $O(nm + n^2 \log n)$ using advanced priority queues.
2. Arbitrary edge lengths: $O(n^2 m)$.
   $\Theta(n^4)$ if $m = \Omega(n^2)$.

Can we do better?

All-Pairs Shortest Paths

Using the shortest paths algorithms we already have...

All-Pairs Shortest Path Problem

Input A (undirected or directed) graph $G = (V, E)$ with edge lengths. For edge $e = (u, v)$, $\ell(e) = \ell(u, v)$ is its length.

1. Find shortest paths for all pairs of nodes.

Apply single-source algorithms n times, once for each vertex.

1. Non-negative lengths. $O(nm \log n)$ with heaps and $O(nm + n^2 \log n)$ using advanced priority queues.
2. Arbitrary edge lengths: $O(n^2 m)$.
   $\Theta(n^4)$ if $m = \Omega(n^2)$.

Can we do better?

All-Pairs Shortest Paths

Using the shortest paths algorithms we already have...

All-Pairs Shortest Path Problem

Input A (undirected or directed) graph $G = (V, E)$ with edge lengths. For edge $e = (u, v)$, $\ell(e) = \ell(u, v)$ is its length.

1. Find shortest paths for all pairs of nodes.

Apply single-source algorithms n times, once for each vertex.

1. Non-negative lengths. $O(nm \log n)$ with heaps and $O(nm + n^2 \log n)$ using advanced priority queues.
2. Arbitrary edge lengths: $O(n^2 m)$.
   $\Theta(n^4)$ if $m = \Omega(n^2)$.

Can we do better?

THE END

...

(for now)