Lecture 14: Johnson's Algorithm

Previously

Restrictions		SSSP Algorithm	
Graph	Weights	Name	Running Time $O(\cdot)$
General	Unweighted	BFS	$\|V\| + \|E\|$
DAG	Any	DAG Relaxation	$\|V\| + \|E\|$
General	Non-negative	Dijkstra	$\|V\| \log \|V\| + \|E\|$
General	Any	Bellman-Ford	$\|V\| \cdot \|E\|$

All-Pairs Shortest Paths (APSP)

- **Input:** directed graph $G = (V, E)$ with weights $w : E \to \mathbb{Z}$
- **Output:** $\delta(u, v)$ for all $u, v \in V$, or abort if G contains negative-weight cycle
- Useful when understanding whole network, e.g., transportation, circuit layout, supply chains...
- Just doing a SSSP algorithm $|V|$ times is actually pretty good, since output has size $O(|V|^2)$
  - $|V| \cdot O(|V| + |E|)$ with BFS if weights positive and bounded by $O(|V| + |E|)$
  - $|V| \cdot O(|V| + |E|)$ with DAG Relaxation if acyclic
  - $|V| \cdot O(|V| \log |V| + |E|)$ with Dijkstra if weights non-negative or graph undirected
  - $|V| \cdot O(|V| \cdot |E|)$ with Bellman-Ford (general)
- **Today:** Solve APSP in any weighted graph in $|V| \cdot O(|V| \log |V| + |E|)$ time

Approach

- **Idea:** Make all edge weights non-negative while **preserving shortest paths**!
- i.e., reweight G to G' with no negative weights, where a shortest path in G is shortest in G'
- If non-negative, then just run Dijkstra $|V|$ times to solve APSP
- **Claim:** Can compute distances in G from distances in G' in $O(|V|(|V| + |E|))$ time
  - Compute shortest-path tree from distances, for each $s \in V'$ in $O(|V| + |E|)$ time (L11)
  - Also shortest-paths tree in G, so traverse tree with DFS while also computing distances
  - Takes $O(|V| \cdot (|V| + |E|))$ time (which is less time than $|V|$ times Dijkstra)
- But how to make G' with non-negative edge weights? Is this even possible??
- **Claim:** Not possible if G contains a negative-weight cycle
- **Proof:** Shortest paths are simple if no negative weights, but not if negative-weight cycle $\square$
- Given graph G with negative weights but no negative-weight cycles, can we make edge weights non-negative while preserving shortest paths?

Making Weights Non-negative

- **Idea!** Add negative of smallest weight in G to every edge! All weights non-negative! :)
- **FAIL:** Does not preserve shortest paths! Biases toward paths traversing fewer edges :(
- **Idea!** Given vertex v, add h to all **outgoing edges** and subtract h from all **incoming edges**
- **Claim:** Shortest paths are preserved under the above reweighting
- **Proof:**
  - Weight of every path starting at v changes by h
  - Weight of every path ending at v changes by $-h$
  - Weight of a path passing through v **does not change** (locally) $\square$
- This is a very general and useful trick to transform a graph while preserving shortest paths!

- Even works with multiple vertices!
- Define a **potential function** $h : V \to \mathbb{Z}$ mapping each vertex $v \in V$ to a potential $h(v)$
- Make graph G': same as G but edge $(u, v) \in E$ has weight $w'(u, v) = w(u, v) + h(u) - h(v)$
- **Claim:** Shortest paths in G are also shortest paths in G'
- **Proof:**
  - Weight of path $\pi = (v_0, \dots, v_k)$ in G is $w(\pi) = \sum_{i=1}^{k} w(v_{i-1}, v_i)$
  - Weight of π in G' is: $\sum_{i=1}^{k} w(v_{i-1}, v_i) + h(v_{i-1}) - h(v_i) = w(\pi) + h(v_0) - h(v_k)$
  - (Sum of h's telescope, since there is a positive and negative $h(v_i)$ for each interior i)
  - Every path from v_0 to v_k changes by the same amount
  - So any shortest path will still be shortest $\square$

Making Weights Non-negative

- Can we find a potential function such that G' has no negative edge weights?
- i.e., is there an h such that $w(u, v) + h(u) - h(v) \geq 0$ for every $(u, v) \in E$?
- Re-arrange this condition to $h(v) \leq h(u) + w(u, v)$, looks like **triangle inequality**!
- **Idea!** Condition would be satisfied if $h(v) = \delta(s, v)$ and $\delta(s, v)$ is finite for some s
- But graph may be disconnected, so may not exist any such vertex s... :(
- **Idea!** Add a new vertex s with a directed 0-weight edge to every $v \in V$! :)
- $\delta(s, v) \leq 0$ for all $v \in V$, since path exists a path of weight 0
- **Claim:** If $\delta(s, v) = -\infty$ for any $v \in V$, then the original graph has a negative-weight cycle
- **Proof:**
  - Adding s does not introduce new cycles (s has no incoming edges)
  - So if reweighted graph has a negative-weight cycle, so does the original graph $\square$
- Alternatively, if $\delta(s, v)$ is finite for all $v \in V$:
  - $w'(u, v) = w(u, v) + h(u) - h(v) \geq 0$ for every $(u, v) \in E$ by triangle inequality!
  - New weights in G' are non-negative while preserving shortest paths!

Johnson's Algorithm

- Construct G_x from G by adding vertex x connected to each vertex $v \in V$ with 0-weight edge
- Compute $\delta_x(x, v)$ for every $v \in V$ (using Bellman-Ford)
- If $\delta_x(x, v) = -\infty$ for any $v \in V$:
  - Abort (since there is a negative-weight cycle in G)
- Else:
  - Reweight each edge $w'(u, v) = w(u, v) + \delta_x(x, u) - \delta_x(x, v)$ to form graph G'
  - For each $u \in V$:
    * Compute shortest-path distances $\delta'(u, v)$ to all v in G' (using Dijkstra)
    * Compute $\delta(u, v) = \delta'(u, v) - \delta_x(x, u) + \delta_x(x, v)$ for all $v \in V$

Correctness

- Already proved that transformation from G to G' preserves shortest paths
- Rest reduces to correctness of Bellman-Ford and Dijkstra
- Reducing from **Signed APSP** to **Non-negative APSP**
- Reductions save time! No induction today! :)

Running Time

- $O(|V| + |E|)$ time to construct G_x
- $O(|V||E|)$ time for Bellman-Ford
- $O(|V| + |E|)$ time to construct G'
- $O(|V| \cdot (|V| \log |V| + |E|))$ time for $|V|$ runs of Dijkstra
- $O(|V|^2)$ time to compute distances in G from distances in G'
- $O(|V|^2 \log |V| + |V||E|)$ time in total