Johnson’s Algorithm for All-Paris Shortest Paths

• Input is Graph $G = (V, E)$ with arbitrary edge weights $c$.
• Assume strongly connected.
• Assume no negative cycle.
• Algorithm
  – Run single source shortest paths from one arbitrary node $s$. (Bellman Ford)
  – Use results of previous step to “reweight edges” so that all edges have non-negative weights
  – Run single source shortest paths from the other $n - 1$ vertices. (Dijkstra)
• Running Time is $O(nm + n(m + n \log n)) = O(nm + n^2 \log n)$, better than $O(n^3)$ for non-dense graphs.
How to Reweight

• Let $p(v)$ be some prices that we put on vertices.
• Consider reduced cost of edge $vw$, $c_p(vw) = c(vw) - p(v) + p(w)$.
• For a $P$, what is the relationship between $c(P)$ and $c_p(P)$?
• For a cycle $X$, what is the relationship between $c(X)$ and $c_p(X)$?
How to Reweight

- Let \( p(v) \) be some prices that we put on vertices.
- Consider reduced cost of edge \( vw \), \( c_p(vw) = c(vw) - p(v) + p(w) \).
- For a \( P \), what is the relationship between \( c(P) \) and \( c_p(P) \)?
- For a cycle \( X \), what is the relationship between \( c(X) \) and \( c_p(X) \)?

Path \( P = v_1v_2\ldots v_k \)

\[
\begin{align*}
c_p(P) &= c_p(v_1v_2) + c_p(v_2v_3) + \cdots + c_p(v_{k-1}v_k) \\
      &= c(v_1v_2) - p(v_1) + p(v_2) + c(v_2v_3) - p(v_2) + p(v_3) + \cdots + c(v_{k-1}v_k) - p(v_{k-1}) + p(v_k) \\
      &= c(P) - p(v_1) + p(v_k)
\end{align*}
\]

- The length of each path from \( v_1 \) to \( v_k \) is increased by the same amount, \( p(v_k) - p(v_1) \).
- Therefore, the shortest path is still the shortest path.
- For a cycle \( p(v_1) = p(v_k) \), so the distance does not change at all.
Reweighting for Shortest Paths

- We will set $p(v)$ to the negative of the shortest path length $d(v)$ from $s$ to $v$.
- We now have that $c_p(vw) = c(vw) - p(v) + p(w) = c(vw) + d(v) - d(w)$.
- But we know that, by the optimality condition for shortest paths:

$$d(w) \leq d(v) + c(vw) \Rightarrow c(vw) + d(v) - d(w) \geq 0 \Rightarrow c_p(vw) \geq 0$$

- So we have non-negative edge weights, still no negative cycles, and can use Dijkstra’s algorithm.