1 Introduction

In this lecture we will look at Bellman-Ford algorithm for Single-Source Shortest Paths problem and Floyd-Warshall algorithm for All-Pairs Shortest Paths problem.

2 Bellman-Ford algorithm for Single-Source Shortest Paths

Dijkstra’s algorithm is good if there is no edge of negative length (check this as an exercise!), but the Bellman-Ford works as long as there is no “negative cycle”, i.e., a cycle whose edges sum to a negative value and it can even detect such cycles.

Algorithm 1 Bellman-Ford(G,v)

1: $SP[v] = 0$;
2: $SP[w] = \infty$ for $w \neq v$;
3: for $i = 1$ to $|V| - 1$ do
4:   for each $(u, w) \in E$ do
5:       if $SP[u] + \text{length}(u, w) < SP[w]$ then
6:           $SP[w] := SP[u] + \text{length}(u, w)$;

Time Complexity: Clearly it is $O(|V| \cdot |E|)$ time which is worse than Dijkstra’s $O\left((|V| + |E|) \log |V|\right)$.

Proof of Correctness: This can be shown using induction.
Lecture 20

3 FLOYD-WARSHALL ALGORITHM FOR ALL-PAIRS SHORTEST PATHS

Induction Hypothesis

If there is a path from v to u with at most i edges, then SP[u] is at most the length of the shortest path from v to u with at most i edges, where i is the number of repetitions.

Proof: Consider the shortest path P from v to u with at most i edges. Let w be the last vertex before u on this path. Then the part of P from v to w is a shortest path from v to w with at most i-1 edges and by induction SP[w] after i iterations is at most the length of this path. This SP[w] + length(w,u) is at most the length of P and we find it in the i-th iteration.

Now if there is no negative cycle, the length of any shortest path in terms of its edges is at most |V| - 1 and thus we find it by Induction Hypothesis for i = |V| - 1. The Induction Hypothesis itself is a good property of this algorithm, which has applications in routing protocols as well.

3 Floyd-Warshall Algorithm for All-Pairs Shortest Paths

The Problem: Given a weighted graph G = (V, E) with non-negative edge lengths, find the minimum length paths between all pairs of vertices. Of course we can run Djikstra’s algorithm for |V| times to get a total time of \( O(|V|^2(|V| + |E|) \log |V|) = O(|V|^3 \cdot |E| \log |V|) \) which is good for sparse graphs but not the best for dense graphs.

Algorithm 2 Floyd-Warshall(G)

1: for m = 1 to n do
2:   for x = 1 to n do
3:     for y = 1 to n do
4:       if weight[x, m] + weight[m, y] < weight[x, y] then
5:         weight[x, y] := weight[x, m] + weight[m, y];

Time Complexity: Very simple algorithm to implement in \( O(|V|^3) \) time.

Proof of Correctness: This can be shown using induction.

Induction Hypothesis

We know the lengths of the shortest paths between all pairs of vertices such that only k-paths, i.e., except endpoints, the highest-labeled vertex on the path is labeled k, for some k ≤ m, are considered. In i-th iteration of the loop we computed all these.

Proof: For m=1, the basis is correct since we have only direct edges as paths. For general m, the shortest path between any pair can have \( v_m \) at most one and then the paths to and from \( v_m \) are k-paths, for k ≤ m − 1. Since we sum the
paths to and from $v_m$ and compare it with the best path found so far in the algorithm, we are done.

As you saw in Bellman-Ford and Floyd-Warshall, the whole idea is to get the Induction Hypothesis correct and the rest is then trivial.

References