Data Structures and Algorithms

Lecture 33

Aniket Basu Roy

2026-04-17 Fri

Contents

1 Agenda

1.1 All-Pairs Shortest Paths

- Problem definition
- Dynamic programming via edge-count subproblems
- Matrix multiplication analogy: the $(min, +)$ semiring
- Slow APSP: $O(n^4)$
- Fast APSP via repeated squaring: $O(n^3 \log n)$

2 Problem Definition

Input: Weighted directed graph $G = (V, E, w)$, $|V| = n$; assume no negative-weight cycles.

Output: $\delta(i, j)$ for all pairs $i, j \in V$; represented as an $n \times n$ matrix Δ.

2.1 Why not simply run SSSP n times?

- $n\times$ Bellman-Ford: $O(n \cdot VE) = O(n^4)$ for dense graphs.
- $n\times$ Dijkstra (non-negative weights): $O(n \cdot m \log n) = O(n^3 \log n)$ for dense graphs.

Today's approach also achieves $O(n^3 \log n)$ but works with negative weights (no negative cycles).

3 Weight Matrix

Represent the graph as an $n \times n$ matrix $W = (w_{ij})$:

$$w_{ij} = \begin{cases} 0 & i = j, \\ w(i, j) & (i, j) \in E, \\ \infty & (i, j) \notin E. \end{cases}$$

4 Dynamic Programming Formulation

4.1 Subproblem

Let $\ell_{ij}^{(m)}$ = minimum weight of any path from i to j using **at most m edges**.

Denote the matrix $L^{(m)} = \left(\ell_{ij}^{(m)}\right)$.

4.2 Base Case

A path using ≤ 1 edge:

$$\ell_{ij}^{(1)} = w_{ij} \quad \Rightarrow \quad L^{(1)} = W.$$

(Recall $w_{ii} = 0$, so trivial zero-length paths are captured.)

4.3 Recurrence

Extend paths of $\leq m-1$ edges by one more edge $(k \to j)$:

$$\ell_{ij}^{(m)} = \min_{1 \leq k \leq n} \left\{ \ell_{ik}^{(m-1)} + w_{kj} \right\}.$$

Proof: Any path $i \rightsquigarrow j$ using $\leq m$ edges either uses $\leq m-1$ edges (captured by $k = j$, $w_{jj} = 0$), or has last edge (k, j) with prefix using $\leq m-1$ edges. By optimal substructure, the prefix weight is $\ell_{ik}^{(m-1)}$. ■

4.4 Stopping Condition

Since there are no negative-weight cycles, all shortest paths are simple. A simple path in a graph with n vertices uses at most $n-1$ edges, so:

$$L^{(n-1)} = \Delta \quad \text{(the true all-pairs shortest-path matrix).}$$

5 The $(min, +)$ Matrix Product

5.1 Definition

For two $n \times n$ matrices A and B, define their $(min, +)$ **product** $A \otimes B$ (pronounced as A op B):

$$(A \otimes B)_{ij} = \min_{1 \leq k \leq n} (a_{ik} + b_{kj}).$$

Replace the standard (sum, $\times$) semiring with (min, +).

5.2 Observation

$$L^{(m)} = L^{(m-1)} \otimes W = \underbrace{W \otimes W \otimes \cdots \otimes W}_{m \text{ times}}.$$

Computing $L^{(n-1)}$ is equivalent to raising W to the $(n-1)$-st power in the $(min, +)$ semiring.

6 EXTEND: One $(min, +)$ Matrix Multiplication

```
EXTEND(A, B, n)              // computes C = A op B
1.  let C be a new n×n matrix
2.  for i = 1 to n
3.      for j = 1 to n
4.          c[i][j] = infinity
5.          for k = 1 to n
6.              c[i][j] = min(c[i][j], a[i][k] + b[k][j])
7.  return C
```

Time: $\Theta(n^3)$ — identical structure to ordinary matrix multiplication.

7 Slow APSP

Compute $L^{(1)}, L^{(2)}, \ldots, L^{(n-1)}$ iteratively:

```
SLOW-APSP(W, n)
1.  L = W                          // L = L^(1)
2.  for m = 2 to n-1
3.      L = EXTEND(L, W, n)        // L <- L op W
4.  return L                       // L = L^(n-1) = Delta
```

Each of the $n-2$ calls to EXTEND costs $\Theta(n^3)$:

$$\boxed{T_{\text{Slow-APSP}} = \Theta(n^4).}$$

8 Fast APSP: Repeated Squaring

Key observation: $(min, +)$ multiplication is associative, so:

$$L^{(2m)} = L^{(m)} \otimes L^{(m)}.$$

We do not need every intermediate matrix. Compute only $L^{(1)}, L^{(2)}, L^{(4)}, \ldots, L^{(2^{\lceil \lg(n-1) \rceil})}$.

```
FASTER-APSP(W, n)
1.  L = W                          // L = L^(1) = L^(2^0)
2.  m = 1
3.  while m < n-1
4.      L = EXTEND(L, L, n)        // L <- L op L  (doubles m)
5.      m = 2*m
6.  return L
```

8.1 Correctness

The while-loop runs $\lceil \lg(n-1) \rceil$ times. After termination, $m \geq n-1$, so $L = L^{(m)}$. Since $L^{(m)} = L^{(n-1)} = \Delta$ for all $m \geq n-1$ (shortest paths are simple), the result is correct. ✓

8.2 Time Complexity

$O(\log n)$ calls to EXTEND, each $\Theta(n^3)$:

$$\boxed{T_{\text{Faster-APSP}} = O(n^3 \log n).}$$

9 Summary

Algorithm	Handles Neg. Weights?	Time Complexity
$n\times$ Dijkstra	No (non-negative weights only)	$O(n^3 \log n)$
Slow APSP (DP)	Yes (no neg. cycles)	$\Theta(n^4)$
Faster APSP (repeated sq.)	Yes (no neg. cycles)	$O(n^3 \log n)$

10 Questions

- Verify $L^{(2)} = W \otimes W$ gives correct shortest paths using ≤ 2 edges on a small example.
- Why is it safe to stop at $m \geq n-1$ even if $m > n-1$? What property of $L^{(m)}$ ensures this?