All-pairs shortest paths

- If no negative-weight edges and sparse graphs run Dijkstra from each vertex $O(nm \log n)$
- What about other cases?

Floyd’s Algorithm Idea

- Interior vertices in a path
  - Dijkstra’s Algorithm
    - at each step always computed shortest paths that only had interior vertices from a set $S$ at each step
  - Floyd’s Algorithm
    - slowly add interior vertices on fixed schedule rather than depending on the graph

Floyd’s Algorithm

- Vertices $V = \{v_1, \ldots, v_n\}$
- Let $D_k[i,j] = \text{length of shortest path from } v_i \text{ to } v_j \text{ that only allows } \{v_1, \ldots, v_k\} \text{ as interior vertices}$
- Note:
  - $D_0[i,j] = \begin{cases} \text{weight of edge } (v_i,v_j) & \text{if } (v_i,v_j) \in E \\ \infty & \text{if } (v_i,v_j) \not\in E \end{cases}$
  - $D_n[i,j] = \text{length of shortest path from } v_i \text{ to } v_j$

Floyd’s Algorithm

- Computing $D_k[i,j]$
  - Case 1: $v_k$ not used
    - $D_k[i,j] = D_{k-1}[i,j]$


Floyd’s Algorithm

Case 2: $v_k$ used

vertices from $v_1,\ldots,v_{k-1}$

not on shortest path since no negative cycles

vertices from $v_1,\ldots,v_{k-1}$

D_{k}[i,j] = D_{k-1}[i,k] + D_{k-1}[k,j]

Multiplying Faster

On the first HW you analyzed our usual algorithm for multiplying numbers

Θ(n²) time

We can do better!

We’ll describe the basic ideas by multiplying polynomials rather than integers

Advantage is we don’t get confused by worrying about carries at first

Polynomial Multiplication

Given:

- Degree n-1 polynomials P and Q
  - $P = a_n x^{n-1} + a_{n-2} x^{n-2} + \ldots + a_2 x^2 + a_1 x + a_0$
  - $Q = b_n x^{n-1} + b_{n-2} x^{n-2} + \ldots + b_2 x^2 + b_1 x + b_0$

Compute:

- Degree 2n-2 Polynomial $P \cdot Q$
  - $P \cdot Q = a_n b_n x^{2n-2} + (a_n b_{n-1} + a_{n-2} b_{n+1}) x^{2n-3} + \ldots + (a_n b_1 + a_{n-2} b_{n+2} + \ldots + a_1 b_{n+1}) x^{n-1} + \ldots + a_0 b_0$

Obvious Algorithm:

- Compute all $a_i b_j$ and collect terms

O(n²) time

Note on Polynomials

These are just formal sequences of coefficients so when we show something multiplied by $x^k$ it just means shifted $k$ places to the left
**Naive Divide and Conquer**

- Assume $n$ is a power of 2
- $P = (a_{n-1}x^{n-1} + a_{n-2}x^{n-2} + \ldots + a_0) x^{n/2} + (a_{n/2-1} x^{n/2-1} + \ldots + a_2 x^2 + a_1 x + a_0)$
- $Q = Q_1 x^{n/2} + Q_0$
- $P \cdot Q = (P_1 x^{n/2} + P_0)(Q_1 x^{n/2} + Q_0)$
- $4$ sub-problems of size $n/2$ plus linear combining
  - $T(n) = 4T(n/2) + cn$
  - Solution $T(n) = O(n^2)$

**Karatsuba’s Algorithm**

- A better way to compute the terms
- Compute
  - $P_0Q_0$
  - $P_1Q_1$
  - $(P_1 + P_0)(Q_1 + Q_0)$ which is $P_1Q_1 + P_1Q_0 + P_0Q_1 + P_0Q_0$
- Then
  - $P_0Q_1 + P_1Q_0 = (P_1 + P_0)(Q_1 + Q_0) - P_0Q_0 - P_1Q_1$
  - $3$ sub-problems of size $n/2$ plus $O(n)$ work
  - $T(n) = 3 T(n/2) + cn$
  - $T(n) = O(n^\alpha)$ where $\alpha = \log_2 3 = 1.59...$