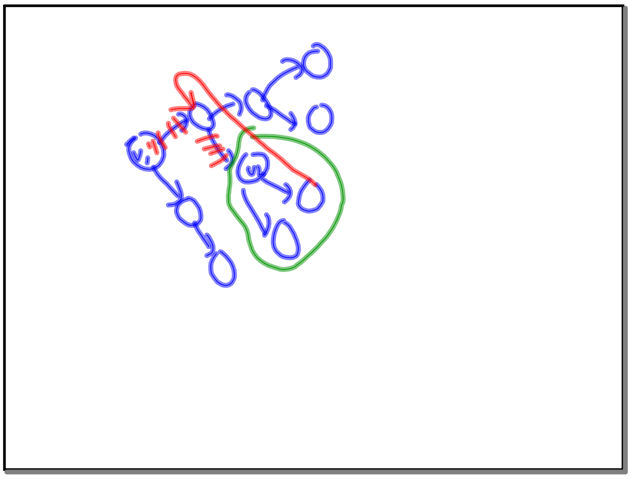
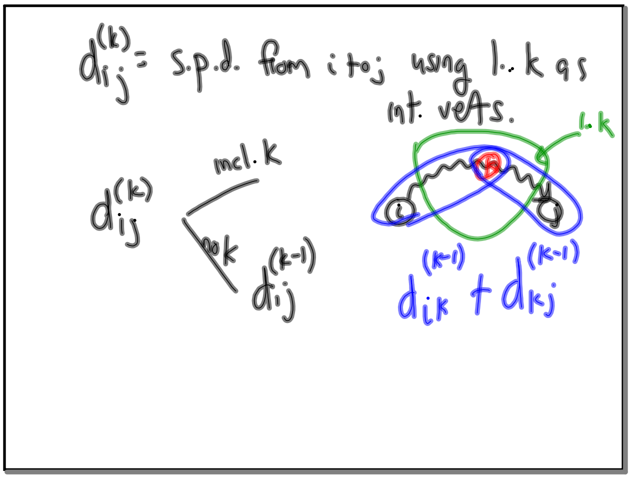
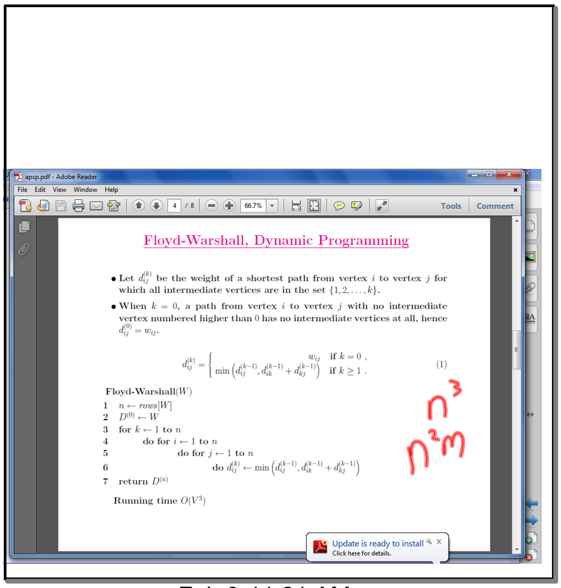


Feb 9-11:08 AM



Feb 9-11:08 AM



Floyd-Warshall, Dynamic Programming

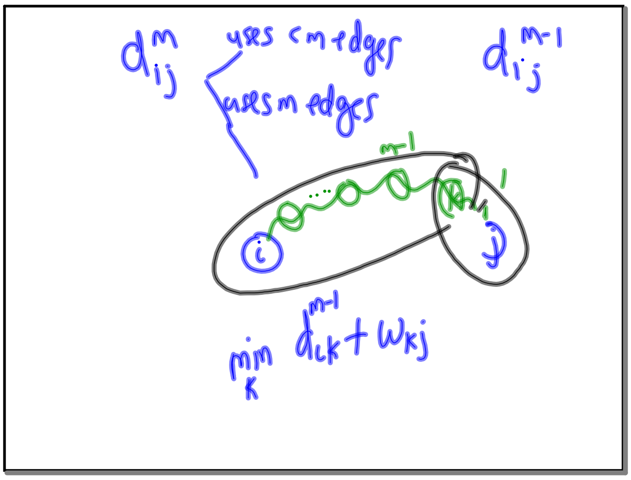
- Let $d_{ij}^{(k)}$ be the weight of a shortest path from vertex i to vertex j for which all intermediate vertices are in the set $\{1, 2, \ldots, k\}$.
- When $k = 0$, a path from vertex i to vertex j with no intermediate vertex numbered higher than 0 has no intermediate vertices at all, hence $d_{ij}^{(0)} = w_{ij}$.

$$d_{ij}^{(k)} = \begin{cases} w_{ij} & \text{if } k = 0 \, , \\ \min\left(d_{ij}^{(k-1)}, d_{ik}^{(k-1)} + d_{kj}^{(k-1)}\right) & \text{if } k \ge 1 \, . \end{cases} \quad (1)$$

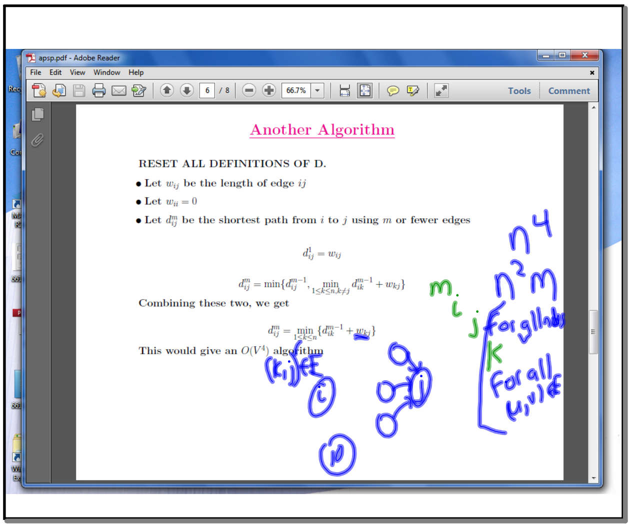
```
Floyd-Warshall(W)
n ← rows[W]
D^(0) ← W
for k ← 1 to n
    do for i ← 1 to n
        do for j ← 1 to n
            do d_ij^(k) ← min(d_ij^(k-1), d_ik^(k-1) + d_kj^(k-1))
return D^(n)
```

Running time $O(V^3)$

Feb 9-11:21 AM



Feb 9-11:28 AM



Another Algorithm

RESET ALL DEFINITIONS OF D.

- Let w_{ij} be the length of edge ij
- Let $w_{ii} = 0$
- Let d_{ij}^m be the shortest path from i to j using m or fewer edges

$$d_{ij}^1 = w_{ij}$$

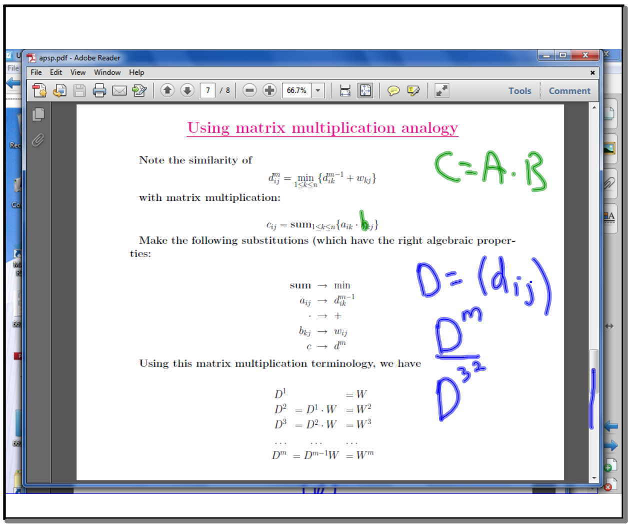
$$d_{ij}^m = \min\{d_{ij}^{m-1}, \min_{1 \le k \le n, k \ne j} d_{ik}^{m-1} + w_{kj}\}$$

Combining these two, we get

$$d_{ij}^m = \min_{1 \le k \le n}\{d_{ik}^{m-1} + w_{kj}\}$$

This would give an $O(V^4)$ algorithm

Feb 9-11:39 AM



Using matrix multiplication analogy

Note the similarity of

$$d_{ij}^m = \min_{1 \le k \le n}\{d_{ik}^{m-1} + w_{kj}\}$$

with matrix multiplication:

$$c_{ij} = \text{sum}_{1 \le k \le n}\{a_{ik} \cdot b_{kj}\}$$

Make the following substitutions (which have the right algebraic properties:

$$\begin{aligned} \text{sum} &\to \min \\ a_{ij} &\to d_{ik}^{m-1} \\ \cdot &\to + \\ b_{kj} &\to w_{ij} \\ c &\to d^m \end{aligned}$$

Using this matrix multiplication terminology, we have

$$\begin{aligned} D^1 & & &= W \\ D^2 &= D^1 \cdot W &&= W^2 \\ D^3 &= D^2 \cdot W &&= W^3 \\ \ldots & \quad \ldots && \quad \ldots \\ D^m &= D^{m-1} W &&= W^m \end{aligned}$$

Feb 9-11:44 AM

$D^{21} = D^{16} \cdot D^{4} \cdot D^{1}$

D^m

$2 \lg m$

$d_{ij}^m =$

D_2, D_4, D_8, D_{16}, D

$D^{16} \cdot D^{4} \cdot D^{1}$

6 MM

$O(n^3 \lg n)$

Feb 9-11:44 AM

Johnson's Algorithm

Arbitrary wts.

BF once from v_1

use $d(v_1, v)$ to help

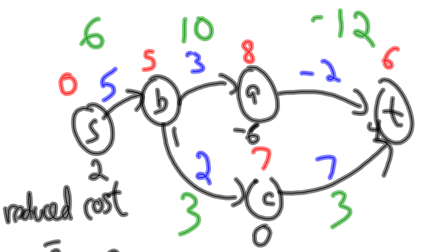
- reweight the graph to make all edge wts. ≥ 0

Run Dijkstra n-1 times.

$nm + n(m + n \lg n)$

$O(nm + n^2 \lg n)$

Feb 9-11:53 AM



reduced cost

c_{ij} = cost (ij)

P_i = price

$\bar{c}_{ij} = c_{ij} + p_i - p_j$

c^p_{ij}

$i \; k_1 \; k_2 \; j$

Q path from i to j

$c(Q)$

$\bar{c}(Q) = c(Q) + p(i) - p(j)$

$\bar{c}(Q) = c(i,k_1) + p(i) - p(k_1) + c(k_1,k_2) + p(k_1) - p(k_2) + c(k_2,j) + p(k_2) - p(j)$

Feb 9-11:57 AM

- If Q is a shortest i, j path w.r.t. c, then Q is a shortest i, j path w.r.t. $\bar{c}$ (for any p)
- For any cycle X, any P,

$c(X) = \bar{c}(X)$.

Feb 9-12:10 PM

Run B.F. w/ v_1 as source (assume path from v_1 to all v) to get $d(v_1, v)$.

- Let $p(v) = d(v_1, v)$.
- Compute $\bar{c}_{ij} = c_{ij} + p_i - p_j$
- Run n-1 remaining s.p. w costs $\bar{c}$

Feb 9-12:14 PM