CSE 421
Introduction to Algorithms

Richard Anderson

Lecture 16

Shortest Paths with Dynamic Programming

Announcements

- Dynamic Programming Reading:
  - 6.8 Shortest Paths (Bellman-Ford)
- Network Flow Reading
  - 7.1-7.3, Network Flow Problem and Algorithms
  - 7.5-7.12, Network Flow Applications

Shortest Path Problem

- Dijkstra’s Single Source Shortest Paths Algorithm
  - O(m log n) time, positive cost edges
- Directed Acyclic Graphs
  - O(n + m), Topological Sort + DP
- Bellman-Ford Algorithm
  - O(mn) time for graphs which can have negative cost edges

Lemma

- If a graph has no negative cost cycles, then the shortest paths are simple paths

- Shortest paths have at most n-1 edges

Shortest paths with a fixed number of edges

- Find the shortest path from s to w with exactly k edges

Express as a recurrence

- Compute distance from starting vertex s

- $Opt_k(w) = \min_x [Opt_{k-1}(x) + c_{xw}]$
- $Opt_0(w) = 0$ if $w = s$ and infinity otherwise

Algorithm, Version 1

```
for each w
    M[0, w] = infinity;
M[0, s] = 0;
for i = 1 to n-1
    for each w
        M[i, w] = min_x(M[i-1,x] + cost[x,w]);
```

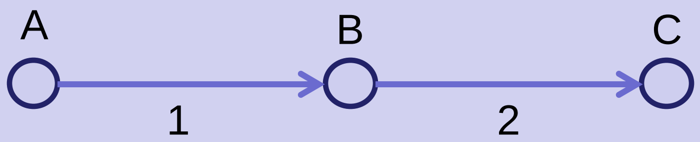
Algorithm, Version 2

```
for each w
    M[0, w] = infinity;
M[0, s] = 0;
for i = 1 to n-1
    for each w
        M[i, w] = min(M[i-1, w], min_x(M[i-1,x] + cost[x,w]));
```

Algorithm, Version 3

```
for each w
    M[w] = infinity;
M[s] = 0;
for i = 1 to n-1
    for each w
        M[w] = min(M[w], min_x(M[x] + cost[x,w]));
```

Example:



A	B	C

A	B	C

A	B	C

A	B	C

Correctness Proof for Algorithm 3

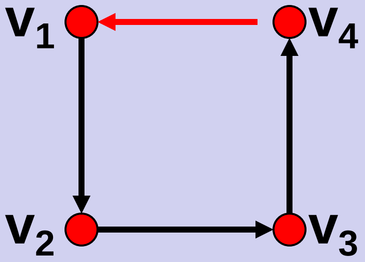
- Key lemma – at the end of iteration i, for all w, $M[w] \le M[i, w]$;

Algorithm, Version 4

```
for each w
    M[w] = infinity;
M[s] = 0;
for i = 1 to n-1
    for each w
        for each x
            if (M[w] > M[x] + cost[x,w])
                P[w] = x;
                M[w] = M[x] + cost[x,w] ;
```

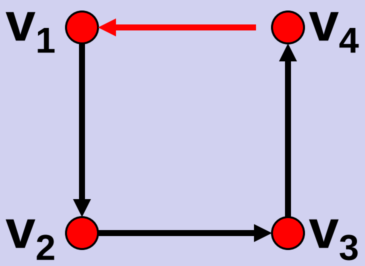
Theorem

If the pointer graph has a cycle, then the graph has a negative cost cycle



If the pointer graph has a cycle, then the graph has a negative cost cycle

- If P[w] = x then M[w] $\geq$ M[x] + cost(x,w)
  - Equal when w is updated
  - M[x] could be reduced after update
- Let v_1, v_2,…v_k be a cycle in the pointer graph with (v_k,v_1) the last edge added
  - Just before the update
    - M[v_j] $\geq$ M[v_{j+1}] + cost(v_{j+1}, v_j) for j < k
    - M[v_k] > M[v_1] + cost(v_1, v_k)
  - Adding everything up
    - 0 > cost(v_2,v_1) + cost(v_3,v_2) + … + cost(v_1, v_k)

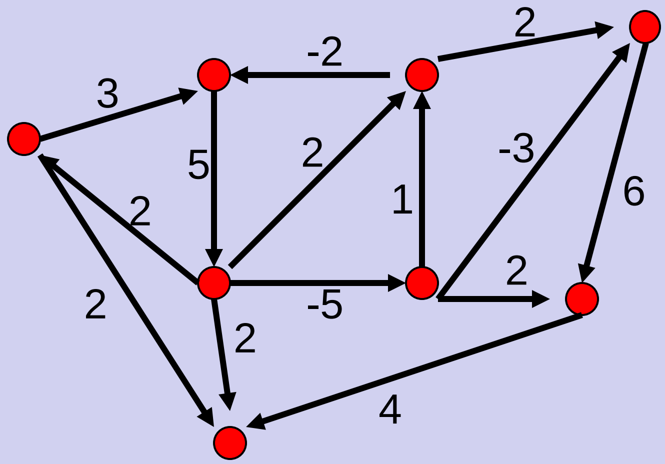


Negative Cycles

- If the pointer graph has a cycle, then the graph has a negative cycle
- Therefore: if the graph has no negative cycles, then the pointer graph has no negative cycles

Finding negative cost cycles

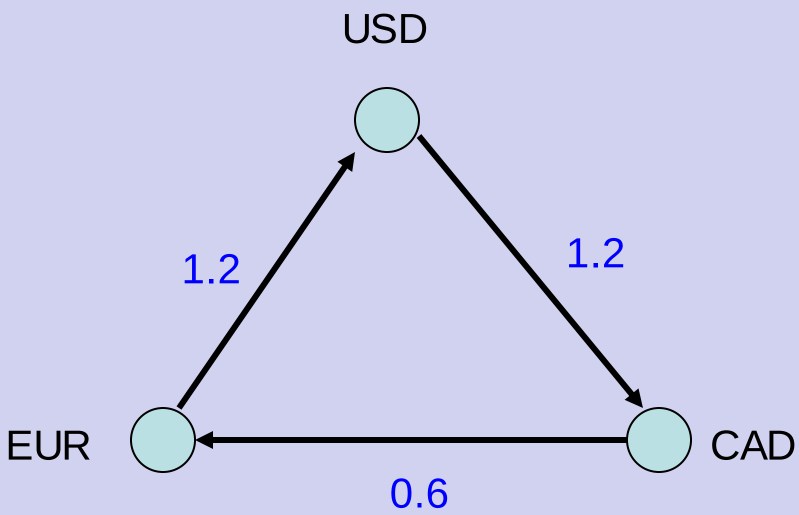
- What if you want to find negative cost cycles?

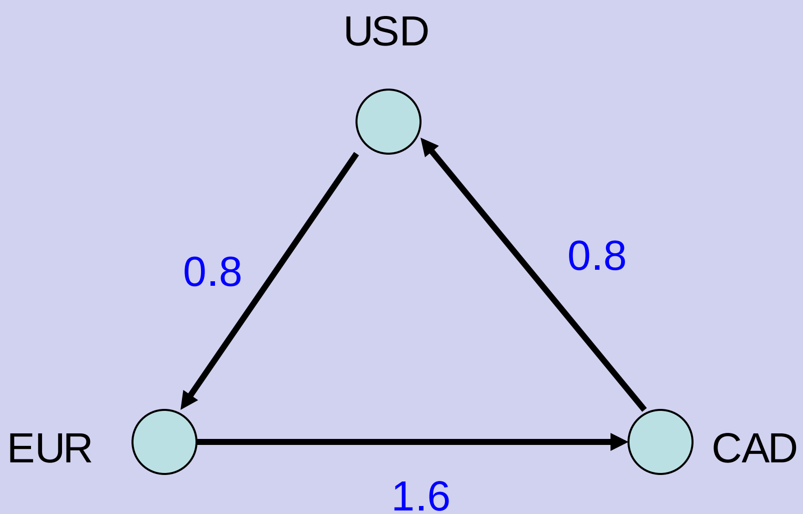


What about finding Longest Paths

- Can we just change Min to Max?

Foreign Exchange Arbitrage





	USD	EUR	CAD
USD	------	0.8	1.2
EUR	1.2	------	1.6
CAD	0.8	0.6	-----