CS 312: Algorithms

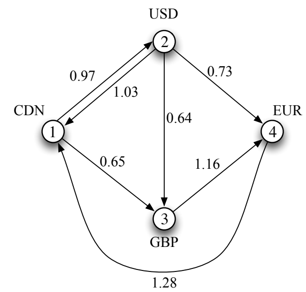
Shortest Paths with Negative Edge Lengths: Bellman-Ford

Dan Sheldon

Mount Holyoke College

Last Compiled: November 12, 2018

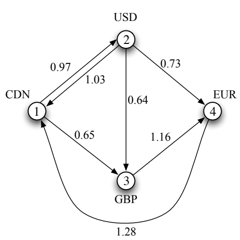
Currency Trading



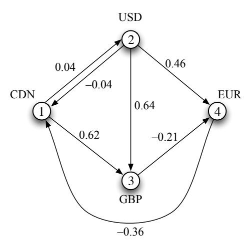
- **Problem**: given directed graph with exchange rate r_e on edge e, find $s \to t$ path P to maximize overall exchange rate $\prod_{e \in P} r_e$
- **Assumption** (no arbitrage): no cycles C such that $\prod_{e \in C} r_e > 1$.

From Rates to Costs

- This problem is similar, but not the same as finding a shortest path. But let's change from rates to costs by transforming the problem.
- Let $c_e = -\log r_e$ be the *cost* of edge e



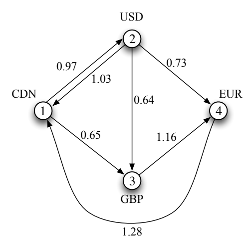
Rates



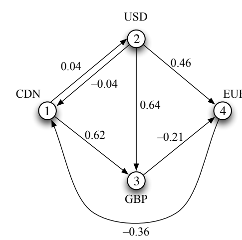
Costs

From Rates to Costs

- After this tranformation, the *highest rate* path becomes the *shortest path* (the one with the smallest sum of edge costs)



Rates



Costs

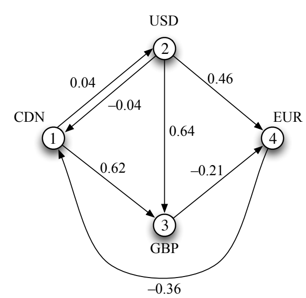
Maximum-Rate Path $\to$ Minimum-Cost Path

- Define $\text{cost}(P)$ to be the negative log of its exchange rate. Then the highest rate path is now the lowest cost path.
- But $\text{cost}(P)$ is also the sum of its edge costs:

$$\begin{aligned}\text{cost}(P) &= -\log \prod_{e \in P} r_e \\ &= \sum_{e \in P} (-\log r_e) \\ &= \sum_{e \in P} c_e\end{aligned}$$

- **New problem**: find the $s \to t$ path of minimum cost

Currency Trading as Shortest Path Problem



- Negative edge weights!
- **Problem**: given a graph with edge weights that may be negative, find shortest $s \to t$ path
- **Assumption**: no cycle C such that $\sum_{e \in C} c_e < 0$. Why?

Dynamic Programming Approach (False Start)

- Let $\mathrm{OPT}(v)$ be the cost of the shortest $v \to t$ path
- What goes wrong with this?

Bellman-Ford Algorithm

With negative edge lengths, paths can get *shorter* as we include more edges. What is the largest number of edges we need to worry about? **Fact.** If no negative cycles, shortest path has at most $n-1$ edges.

Recursive principle:

- Let $\mathrm{OPT}(i, v)$ be cost of shortest $v \to t$ path with at most i edges, and let P be the optimal $v \to t$ path using at most i edges.
- If P uses *exactly* i edges, then $P = v \to w \leadsto t$ where $w \leadsto t$ path uses $i-1$ edges.

$$\mathrm{OPT}(i, v) = \min_{w}\{c_{v,w} + \mathrm{OPT}(i-1, w)\}$$

- Else P uses at most $i-1$, so: $\mathrm{OPT}(i, v) = \mathrm{OPT}(i-1, v)$.

Bellman-Ford

$$\mathrm{OPT}(i, v) = \min\left\{\mathrm{OPT}(i-1, v),\ \ \min_{w \in V}\{c_{v,w} + \mathrm{OPT}(i-1, w)\}\right\}$$

Shortest-Path(G, s, t)
  n = number of nodes in G
  Create array M of size $n \times n$
  Set $M[0, t] = 0$ and $M[0, v] = \infty$ for all other v
  for $i = 1$ to $n-1$ **do**
    for all nodes v in any order **do**
      Compute $M[i, v]$ using the recurrence above
    end for
  end for

Running time? $O(n^3)$. Better analysis: $O(mn)$.