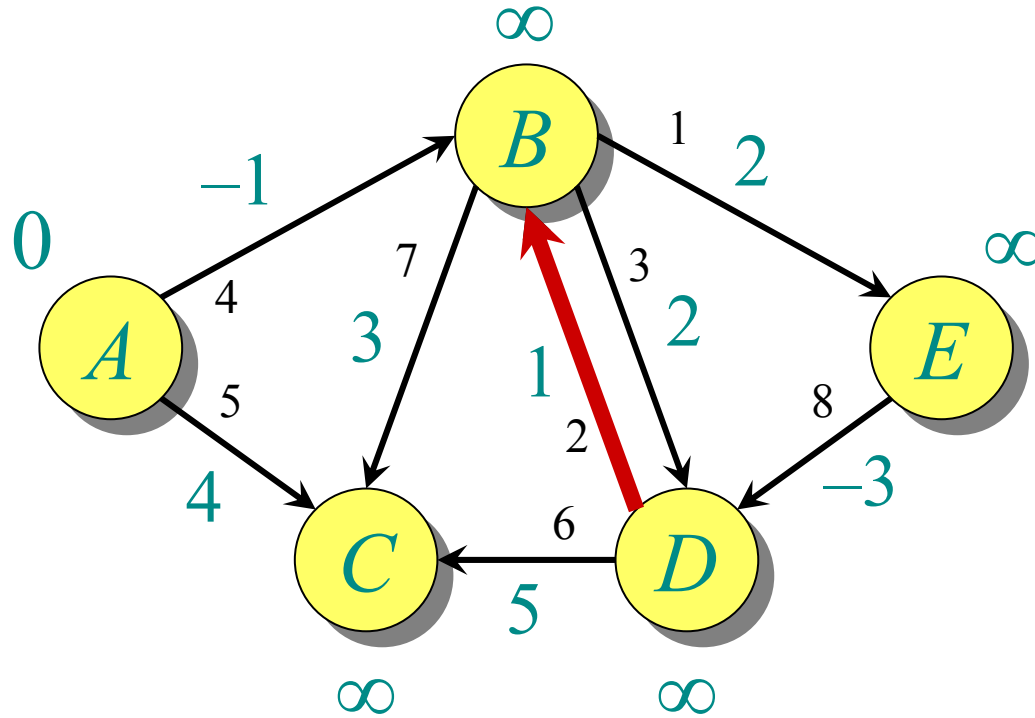
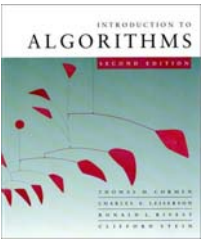
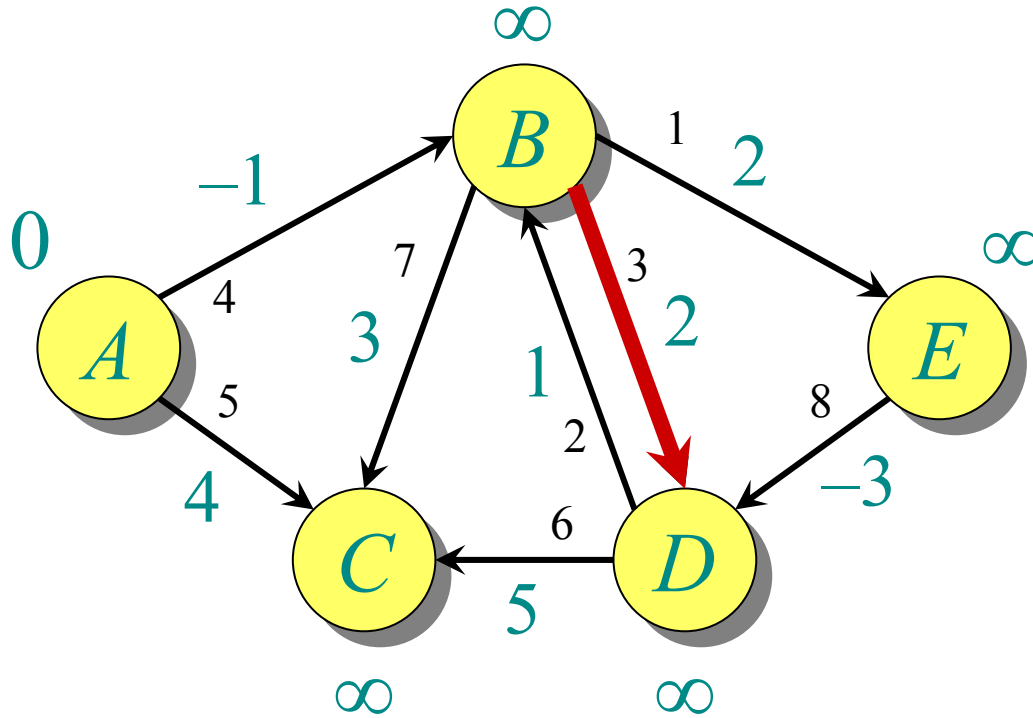


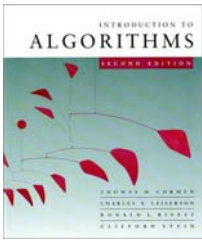
Bellman-Ford örneği



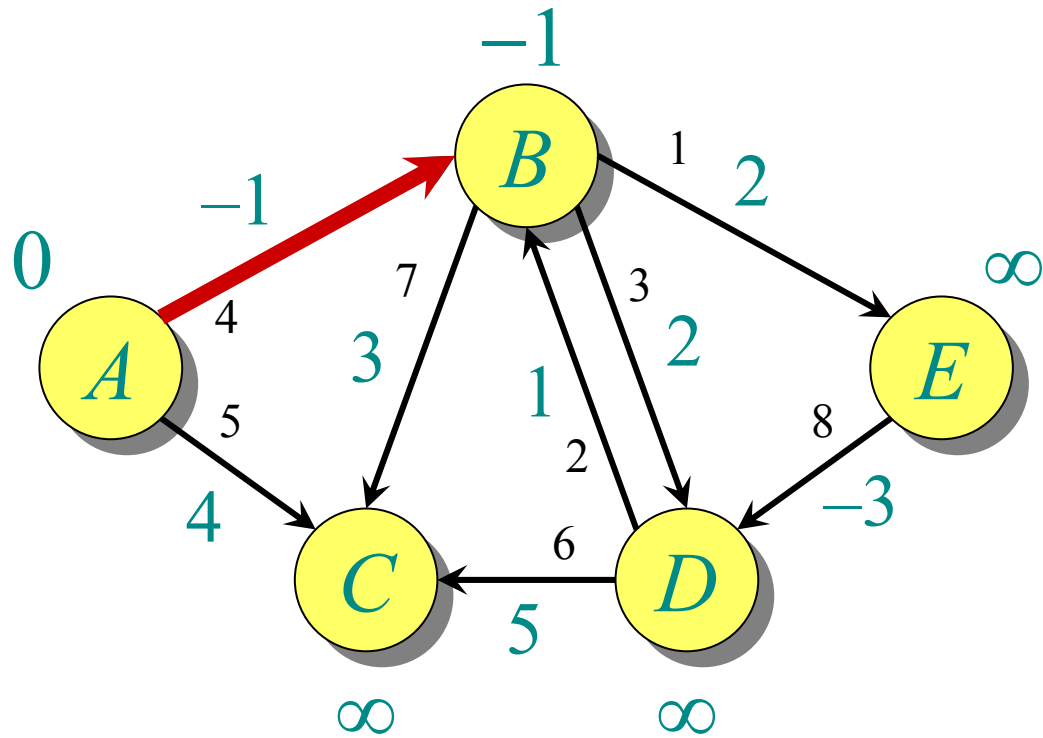


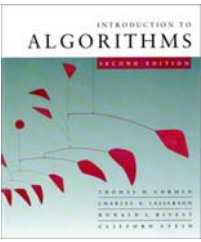
Bellman-Ford örneği



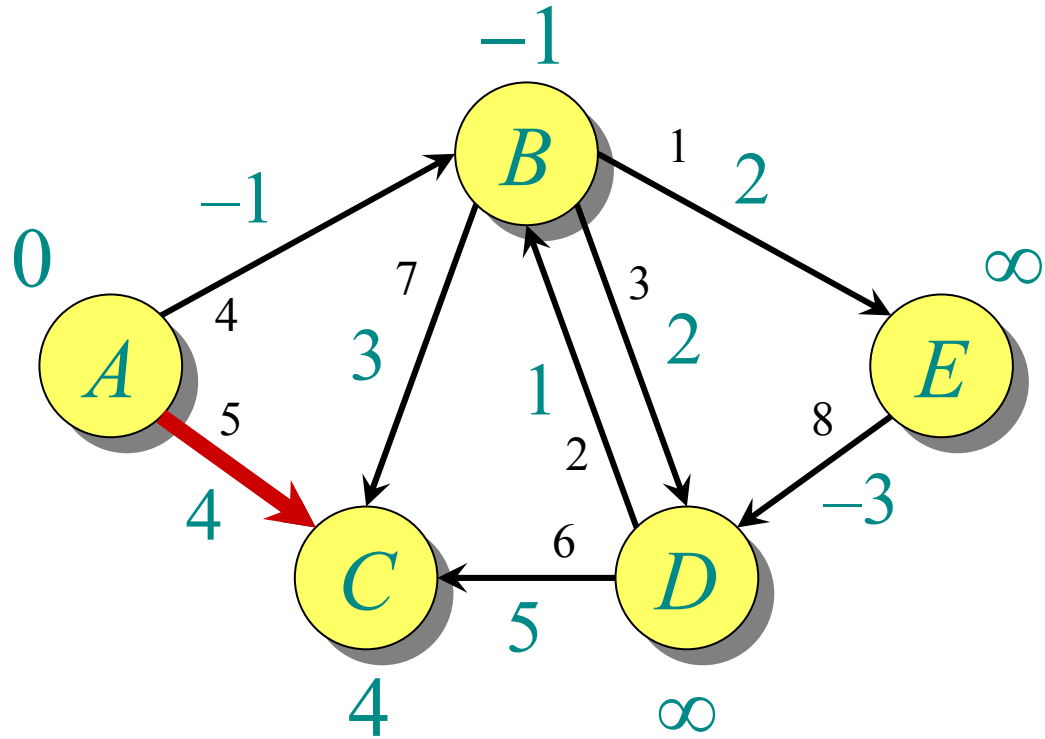


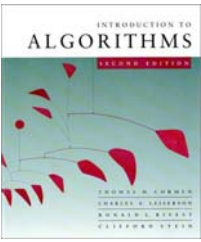
Bellman-Ford örneği



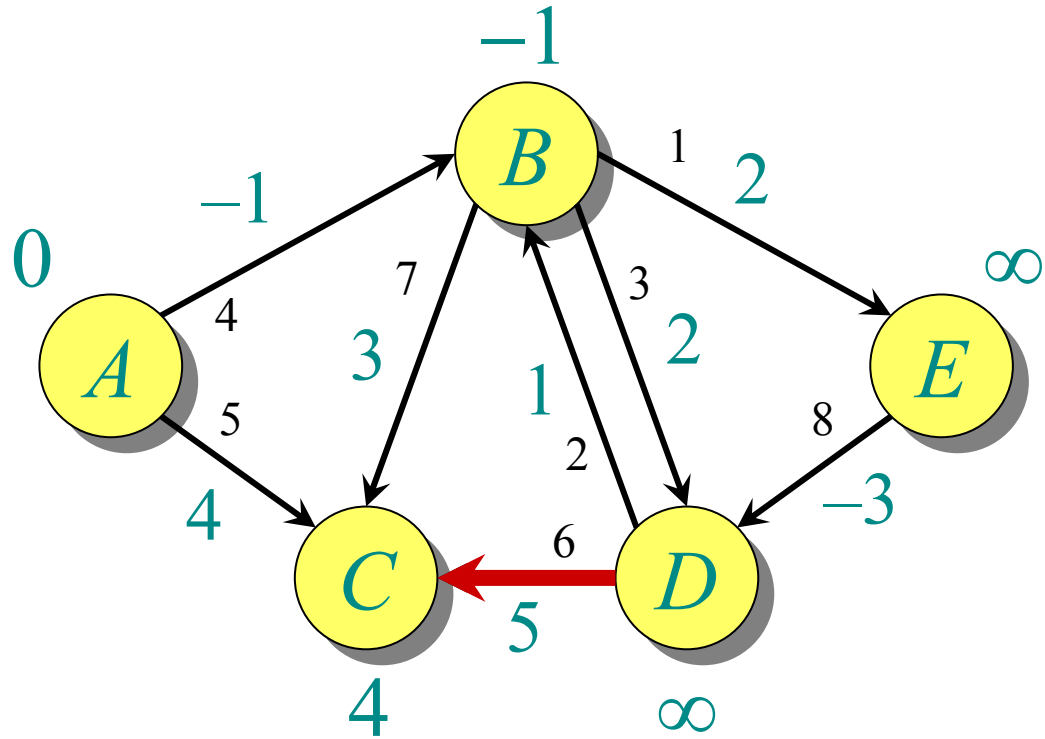


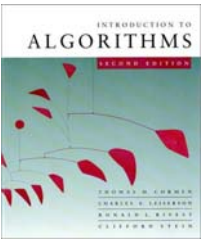
Bellman-Ford örneği



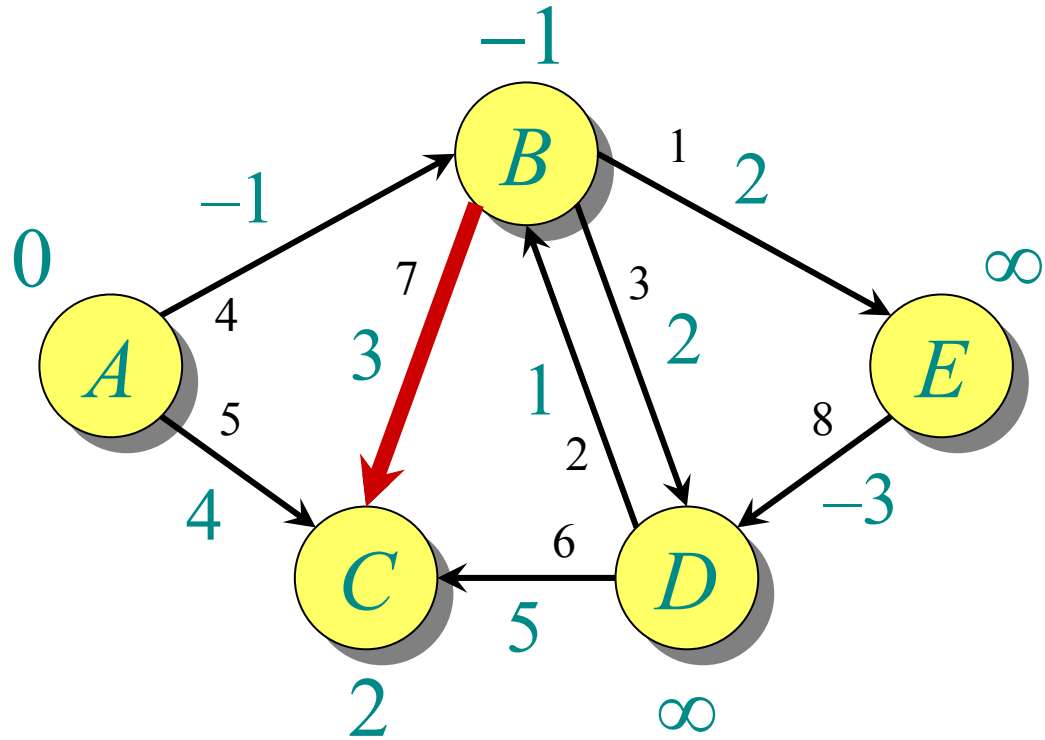


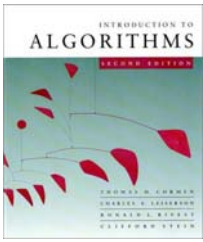
Bellman-Ford örneği



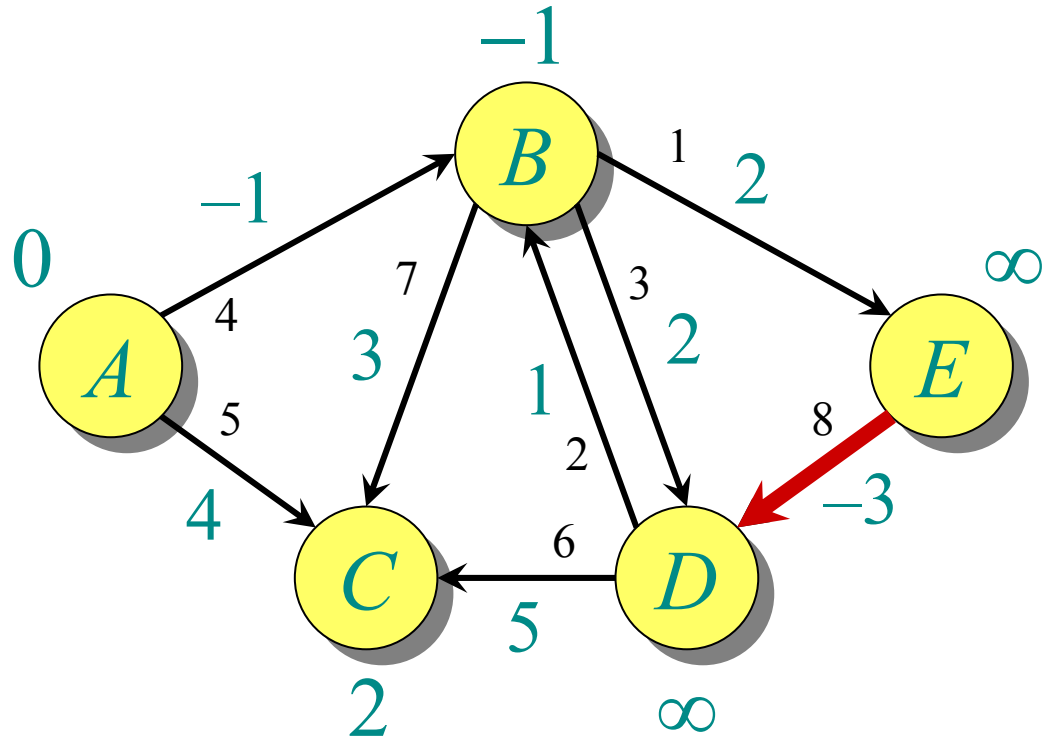


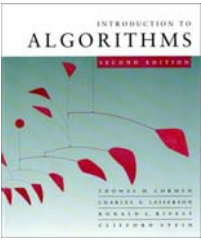
Bellman-Ford örneği



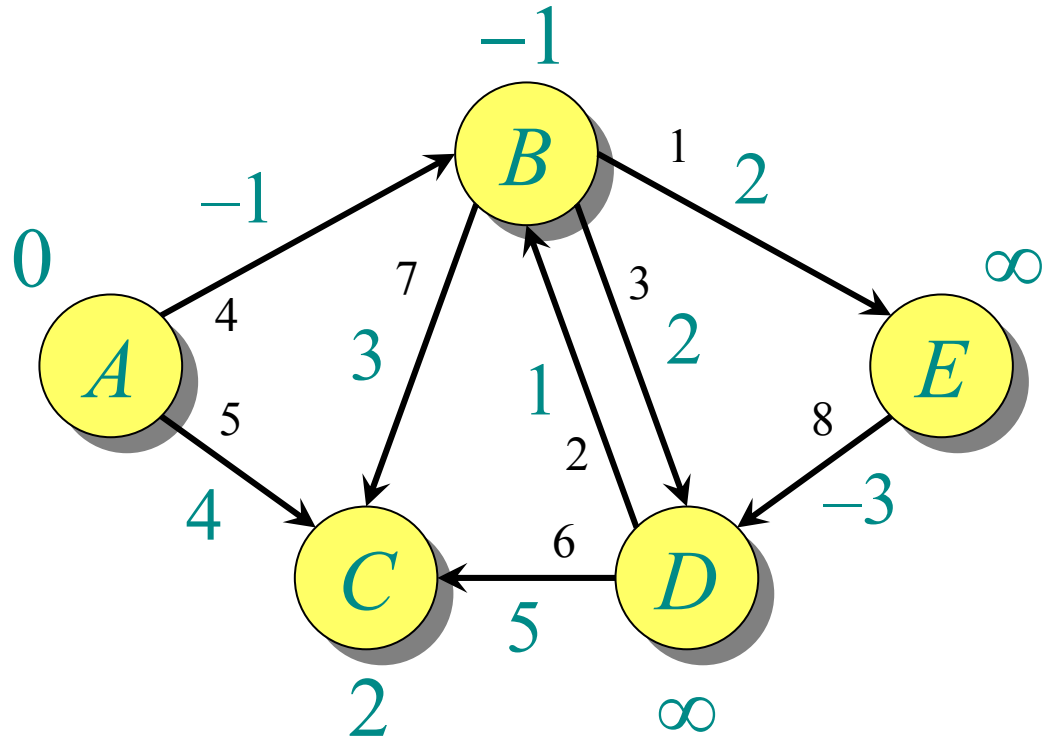


Bellman-Ford örneği

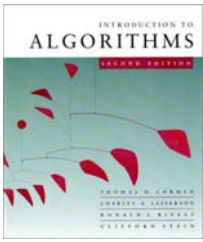




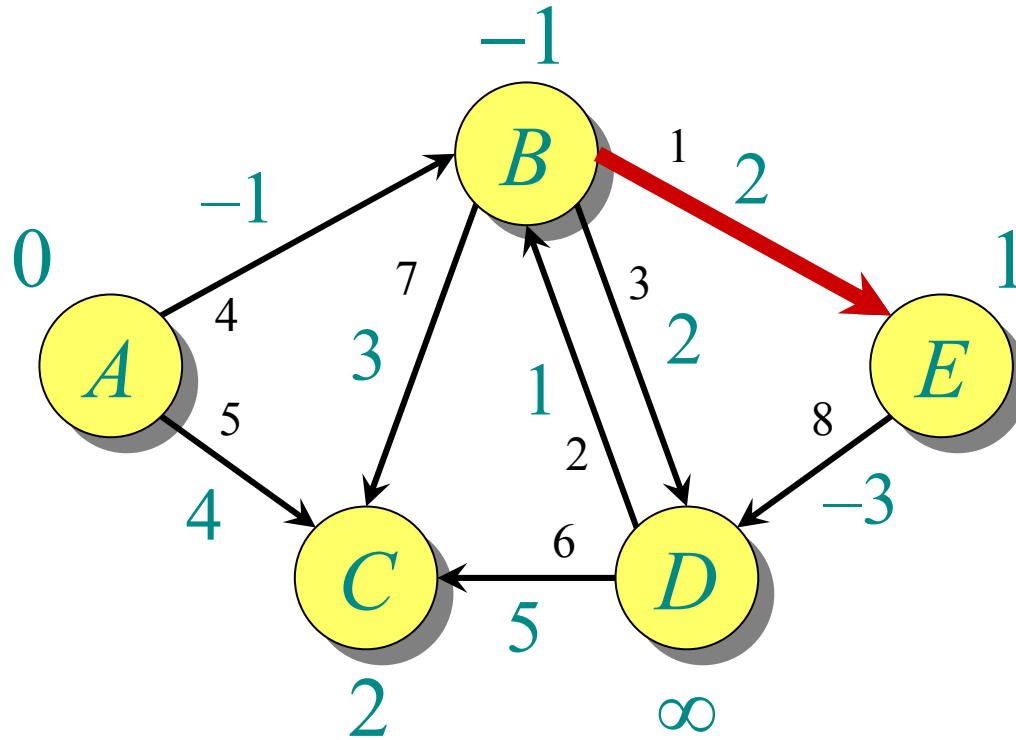
Bellman-Ford örneği

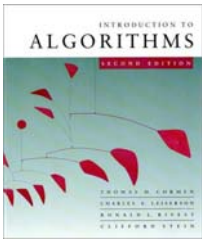


1. durumun sonunda

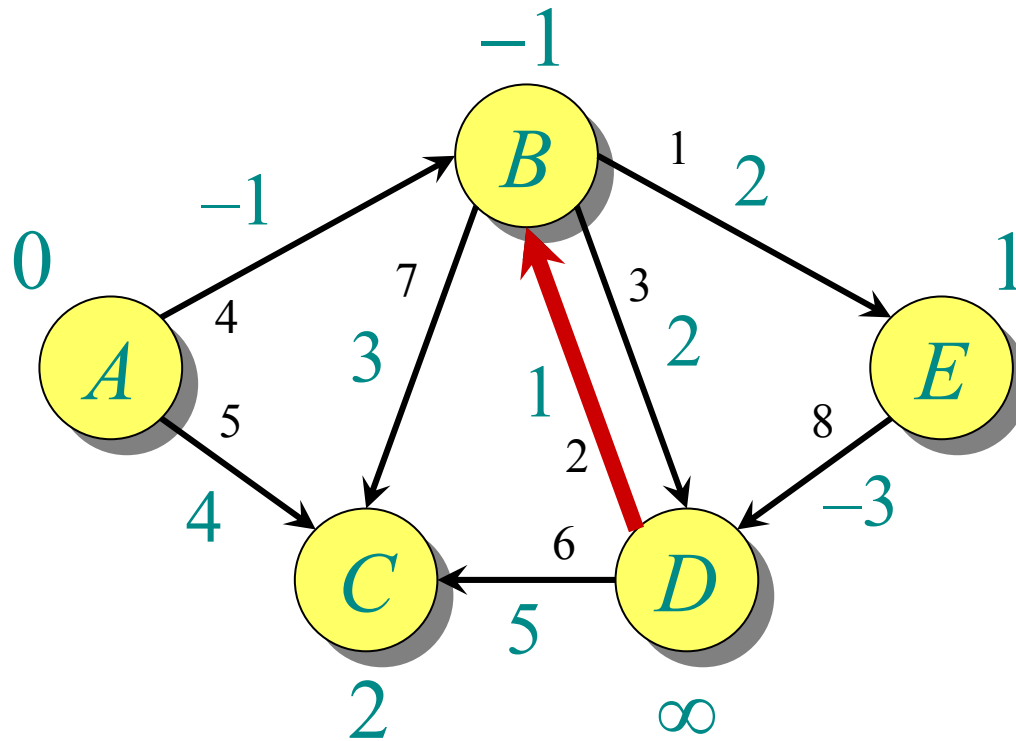


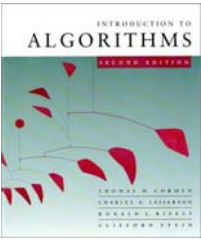
Bellman-Ford örneği



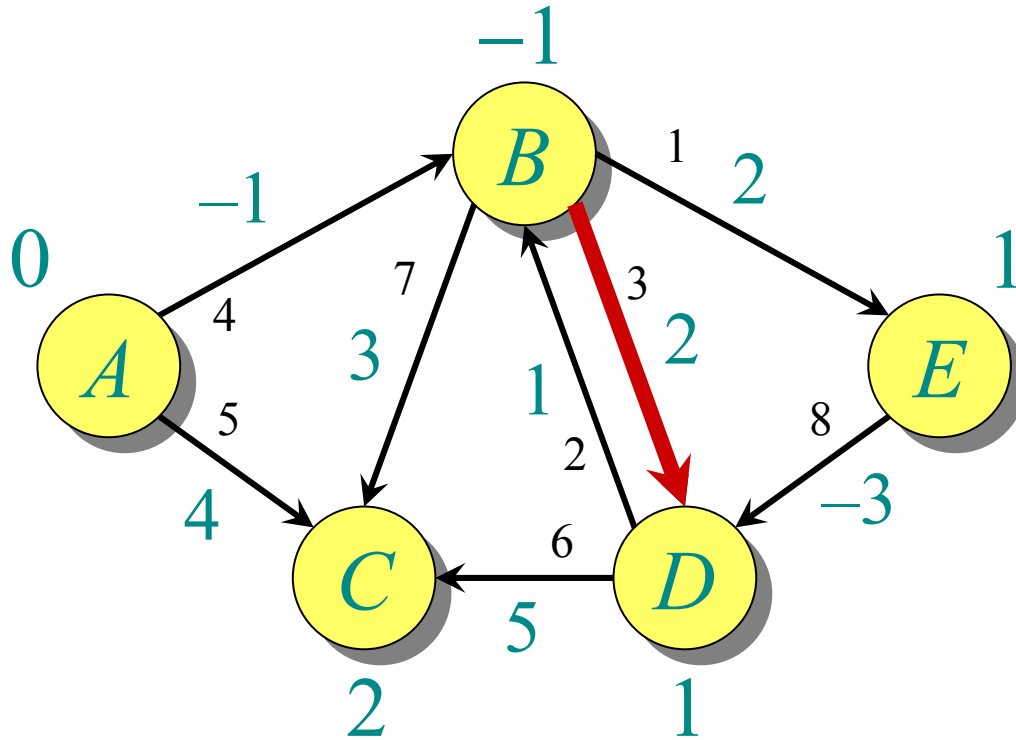


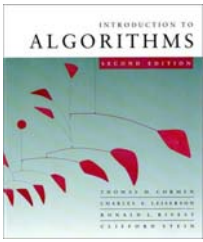
Bellman-Ford örneği



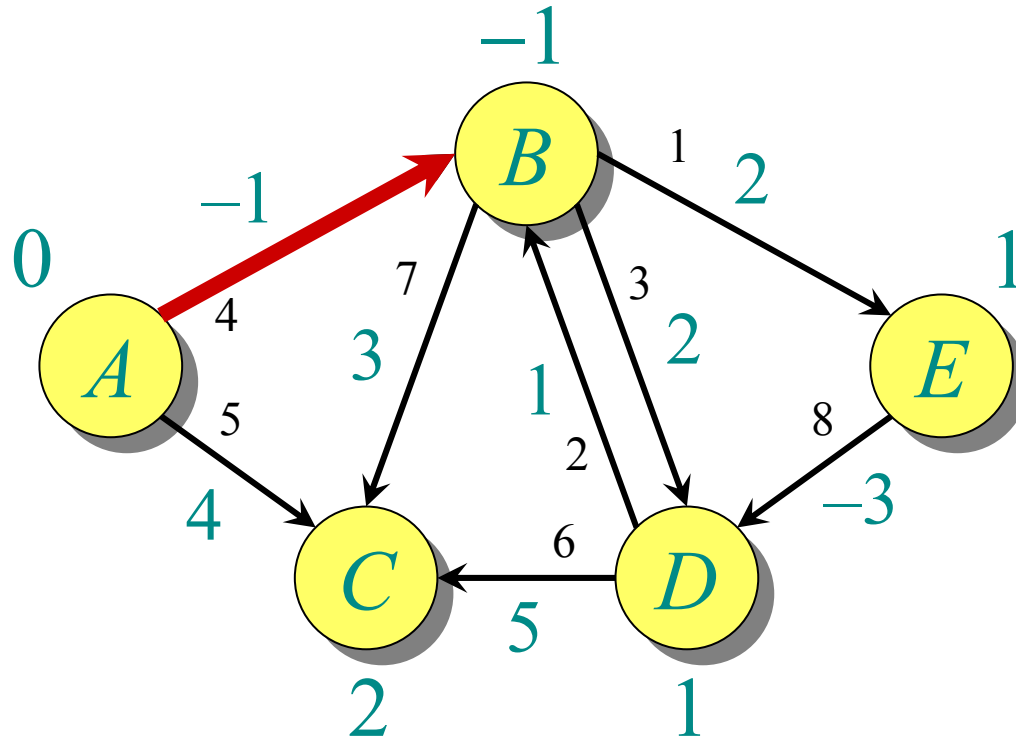


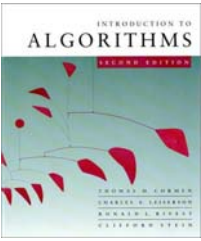
Bellman-Ford örneği



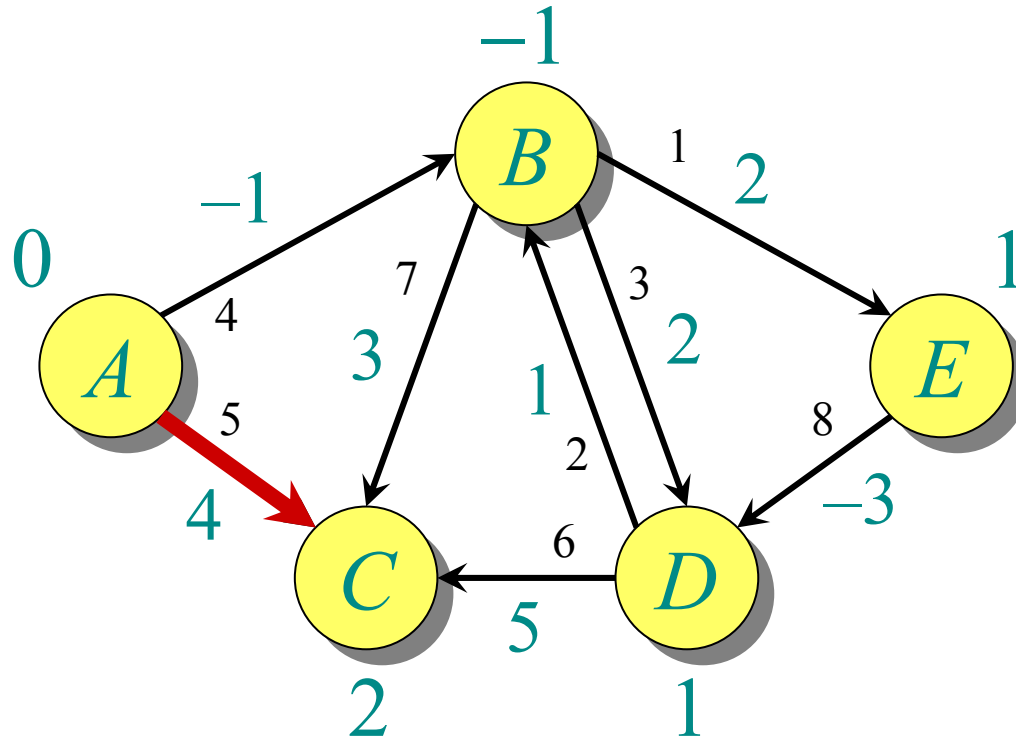


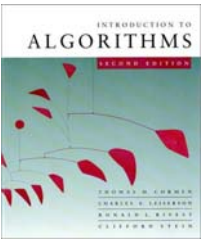
Bellman-Ford örneği



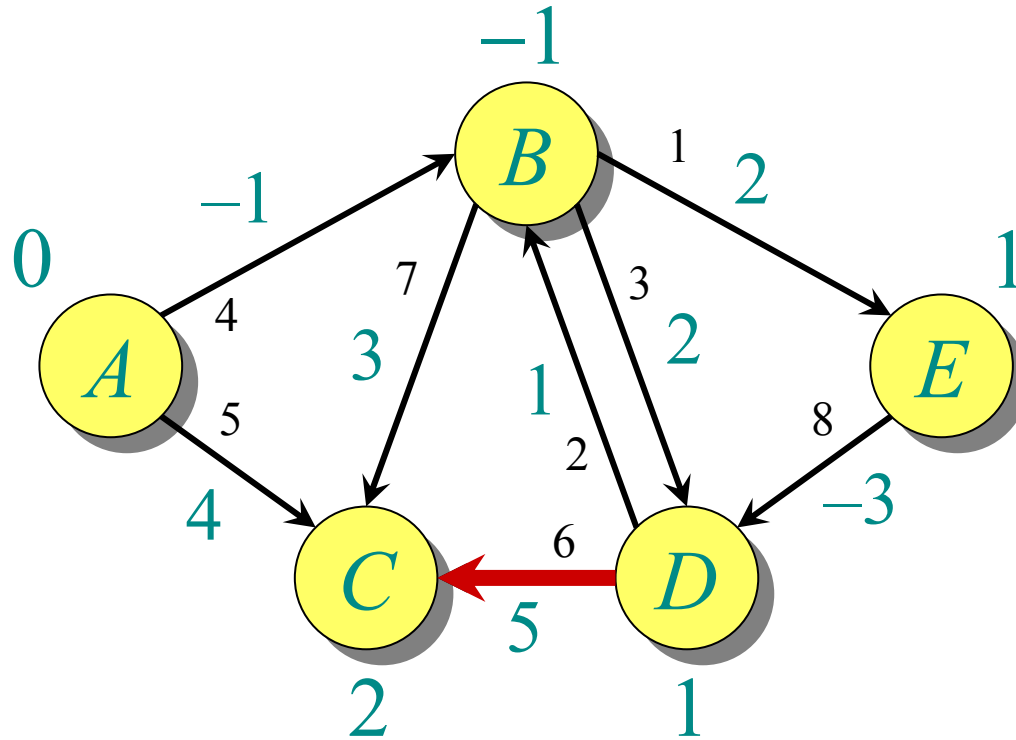


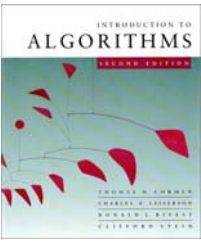
Bellman-Ford örneği



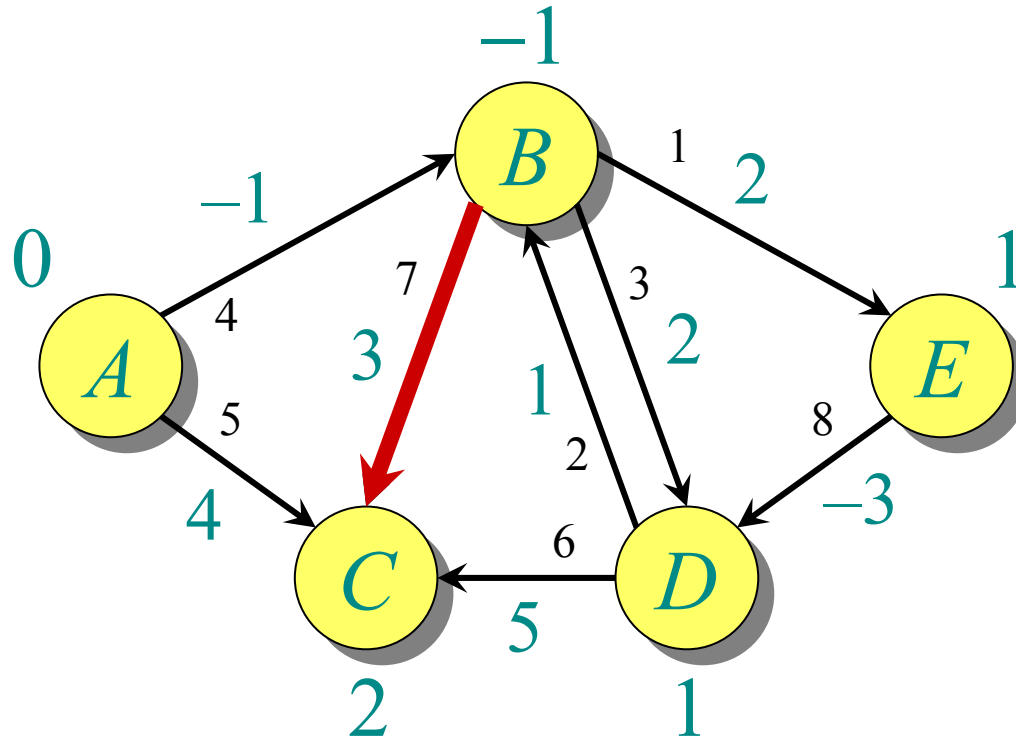


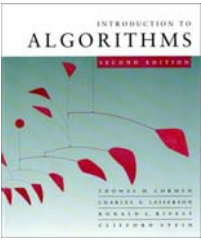
Bellman-Ford örneği



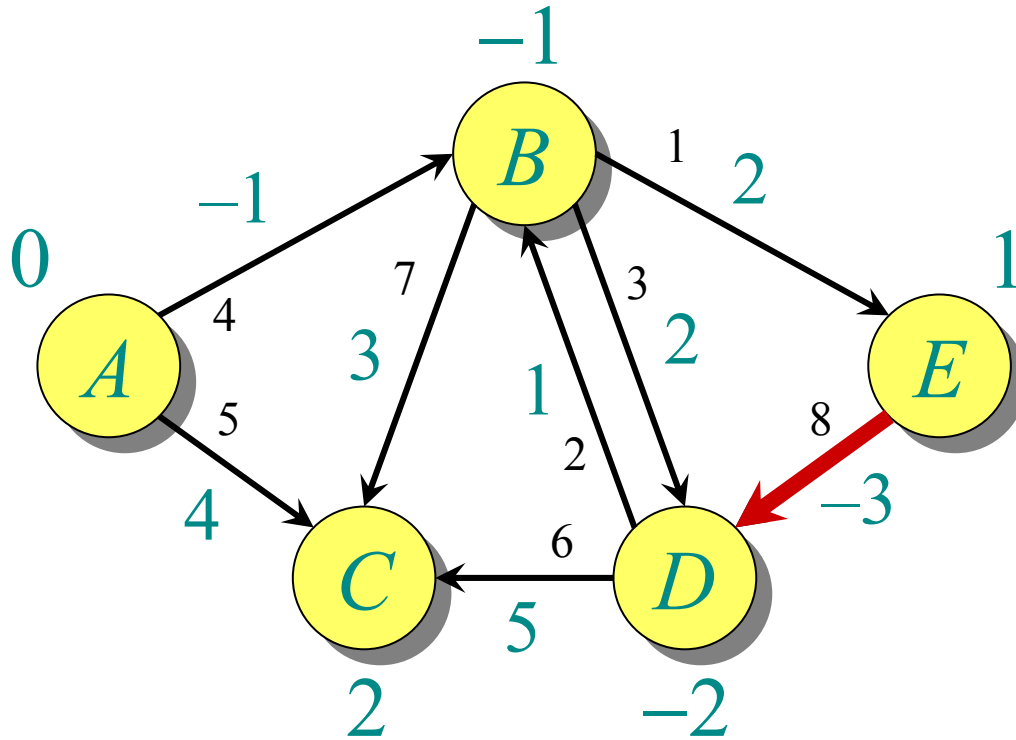


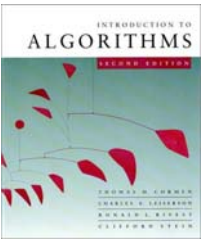
Bellman-Ford örneği



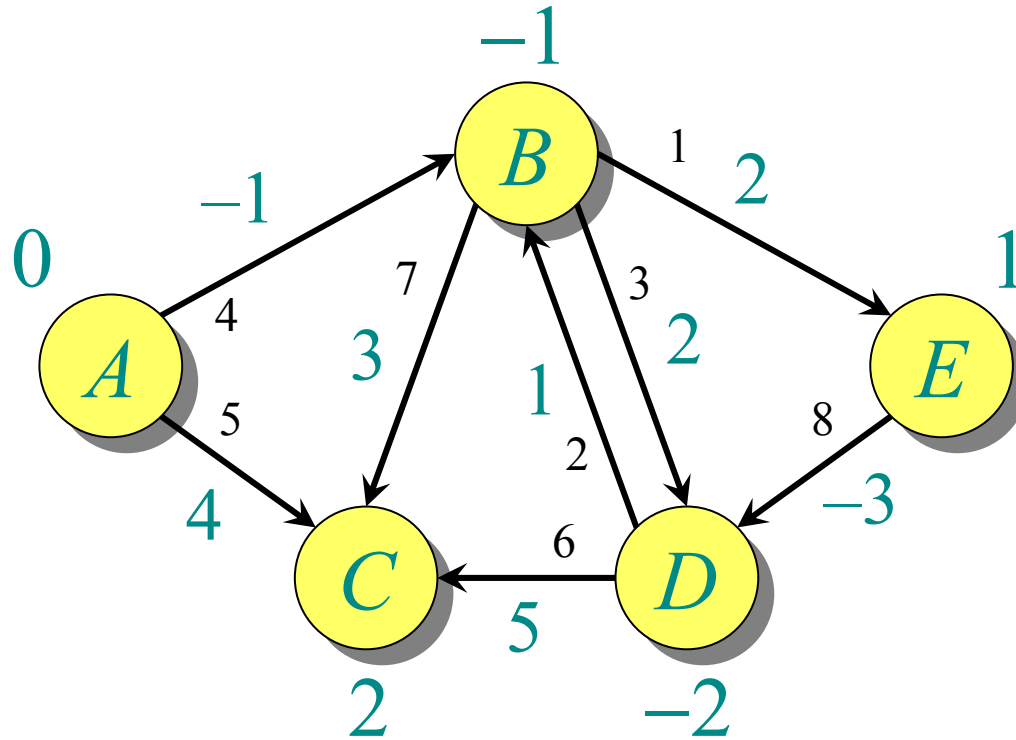


Bellman-Ford örneği

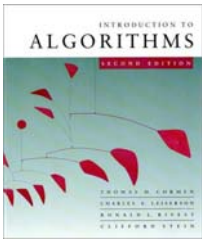




Bellman-Ford örneği

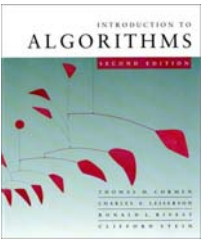


2. durumun sonunda (ve 3 ve 4).



Doğruluk

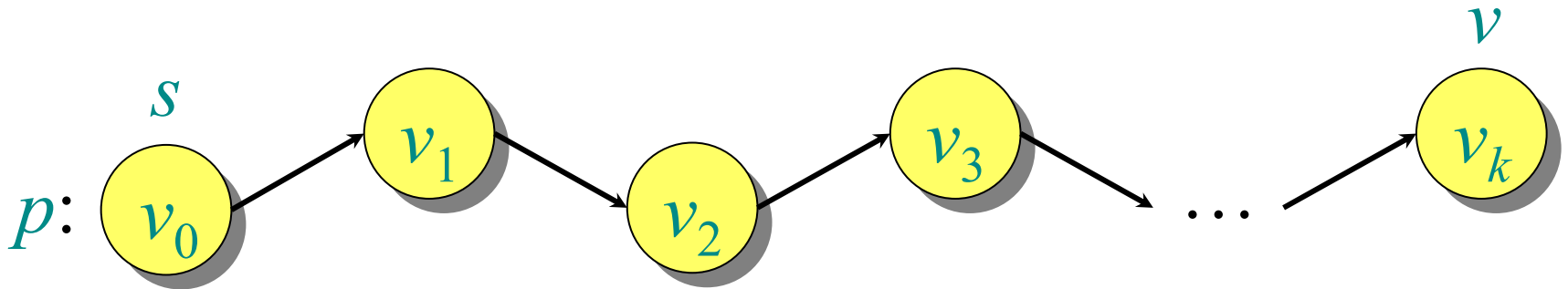
Teorem. Eğer $G = (V, E)$ hiç negatif ağırlık döngüsü içermiyorsa, sonrasında Bellman-Ford algoritması, bütün $v \in V$ ler için $d[v] = \delta(s, v)$ yi uygular.



Correctness

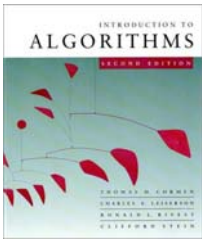
Theorem. If $G = (V, E)$ contains no negative-weight cycles, then after the Bellman-Ford algorithm executes, $d[v] = \delta(s, v)$ for all $v \in V$.

Proof. Let $v \in V$ be any vertex, and consider a shortest path p from s to v with the minimum number of edges.

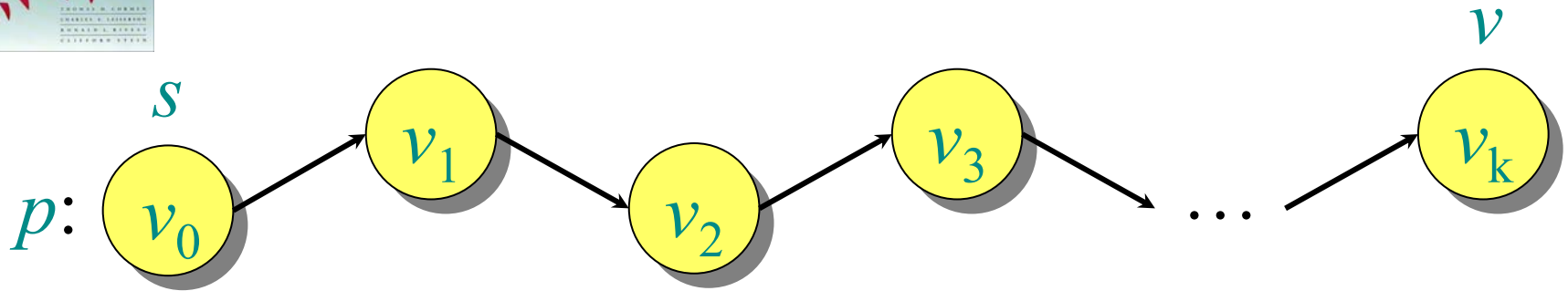


Since p is a shortest path, we have

$$\delta(s, v_i) = \delta(s, v_{i-1}) + w(v_{i-1}, v_i) .$$



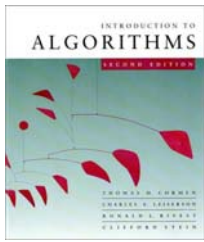
Correctness (continued)



Initially, $d[v_0] = 0 = \delta(s, v_0)$, and $d[v_0]$ is unchanged by subsequent relaxations (because of the lemma from Lecture 14 that $d[v] \geq \delta(s, v)$).

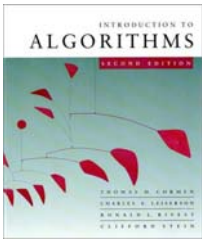
- After 1 pass through E , we have $d[v_1] = \delta(s, v_1)$.
- After 2 passes through E , we have $d[v_2] = \delta(s, v_2)$.
- \vdots
- After k passes through E , we have $d[v_k] = \delta(s, v_k)$.

Since G contains no negative-weight cycles, p is simple. Longest simple path has $\leq |V| - 1$ edges. \square



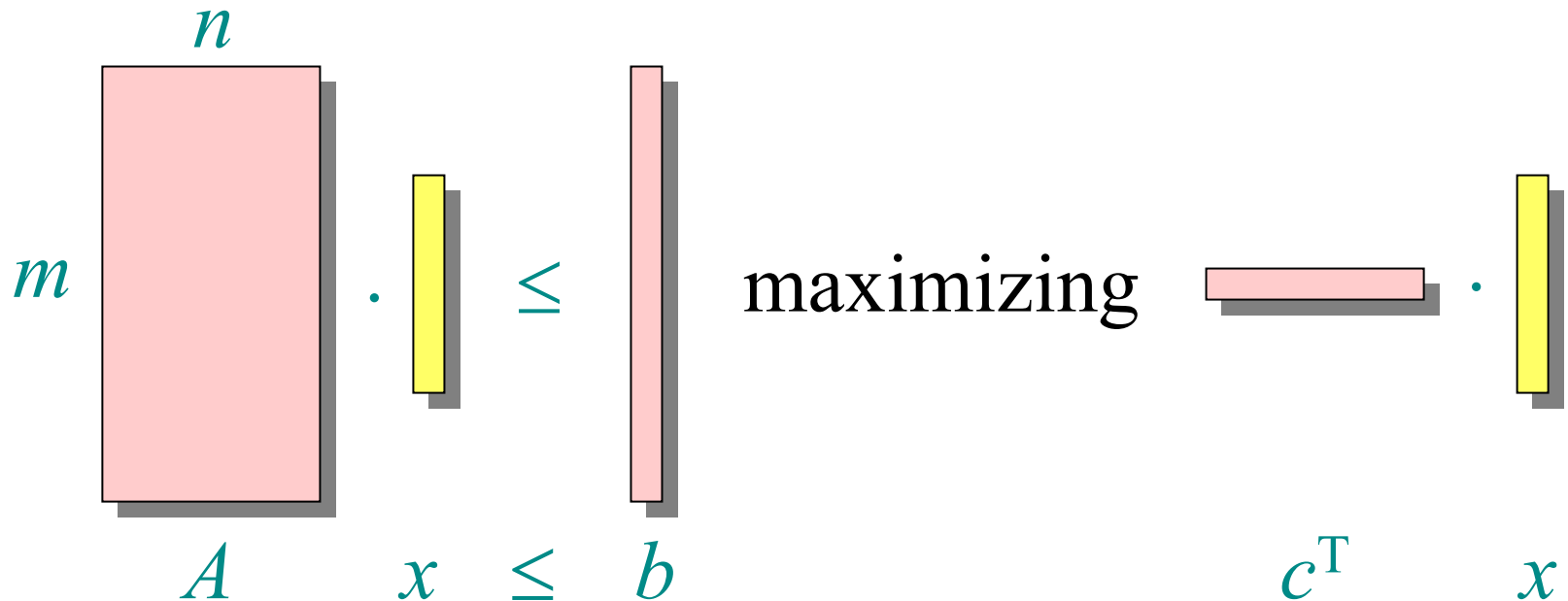
Detection of negative-weight cycles

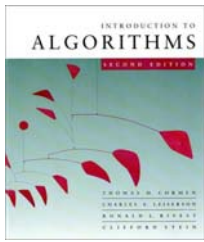
Corollary. If a value $d[v]$ fails to converge after $|V| - 1$ passes, there exists a negative-weight cycle in G reachable from s . \square



Linear programming

Let A be an $m \times n$ matrix, b be an m -vector, and c be an n -vector. Find an n -vector x that maximizes $c^T x$ subject to $Ax \leq b$, or determine that no such solution exists.

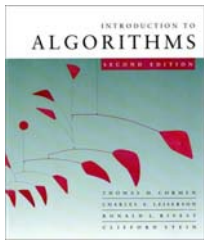




Linear-programming algorithms

Algorithms for the general problem

- Simplex methods — practical, but worst-case exponential time.
- Interior-point methods — polynomial time and competes with simplex.



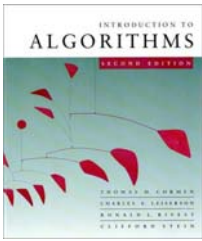
Linear-programming algorithms

Algorithms for the general problem

- Simplex methods — practical, but worst-case exponential time.
- Interior-point methods — polynomial time and competes with simplex.

Feasibility problem: No optimization criterion. Just find x such that $Ax \leq b$.

- In general, just as hard as ordinary LP.

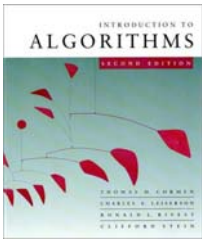


Solving a system of difference constraints

Linear programming where each row of A contains exactly one 1 , one -1 , and the rest 0 's.

Example:

$$\left. \begin{array}{l} x_1 - x_2 \leq 3 \\ x_2 - x_3 \leq -2 \\ x_1 - x_3 \leq 2 \end{array} \right\} x_j - x_i \leq w_{ij}$$



Solving a system of difference constraints

Linear programming where each row of A contains exactly one 1 , one -1 , and the rest 0 's.

Example:

$$x_1 - x_2 \leq 3$$

$$x_2 - x_3 \leq -2$$

$$x_1 - x_3 \leq 2$$

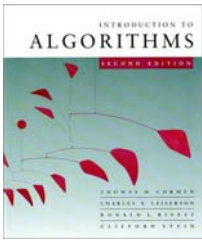
$$\left. \begin{array}{l} x_1 - x_2 \leq 3 \\ x_2 - x_3 \leq -2 \\ x_1 - x_3 \leq 2 \end{array} \right\} x_j - x_i \leq w_{ij}$$

Solution:

$$x_1 = 3$$

$$x_2 = 0$$

$$x_3 = 2$$



Solving a system of difference constraints

Linear programming where each row of A contains exactly one 1, one -1 , and the rest 0's.

Example:

$$x_1 - x_2 \leq 3$$

$$x_2 - x_3 \leq -2$$

$$x_1 - x_3 \leq 2$$

$$x_j - x_i \leq w_{ij}$$

Solution:

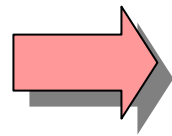
$$x_1 = 3$$

$$x_2 = 0$$

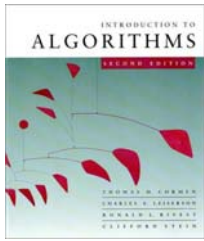
$$x_3 = 2$$

Constraint graph:

$$x_j - x_i \leq w_{ij}$$

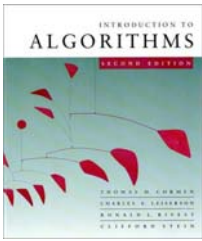


(The “ A ” matrix has dimensions $|E| \times |V|$.)



Unsatisfiable constraints

Theorem. If the constraint graph contains a negative-weight cycle, then the system of differences is unsatisfiable.

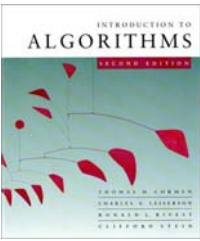


Unsatisfiable constraints

Theorem. If the constraint graph contains a negative-weight cycle, then the system of differences is unsatisfiable.

Proof. Suppose that the negative-weight cycle is $v_1 \rightarrow v_2 \rightarrow \cdots \rightarrow v_k \rightarrow v_1$. Then, we have

$$\begin{aligned}x_2 - x_1 &\leq w_{12} \\x_3 - x_2 &\leq w_{23} \\&\vdots \\x_k - x_{k-1} &\leq w_{k-1, k} \\x_1 - x_k &\leq w_{k1}\end{aligned}$$



Unsatisfiable constraints

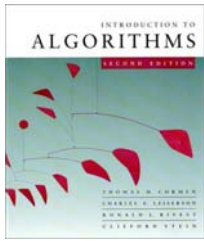
Theorem. If the constraint graph contains a negative-weight cycle, then the system of differences is unsatisfiable.

Proof. Suppose that the negative-weight cycle is $v_1 \rightarrow v_2 \rightarrow \cdots \rightarrow v_k \rightarrow v_1$. Then, we have

$$\begin{aligned}x_2 - x_1 &\leq w_{12} \\x_3 - x_2 &\leq w_{23} \\&\vdots \\x_k - x_{k-1} &\leq w_{k-1, k} \\x_1 - x_k &\leq w_{k1}\end{aligned}$$

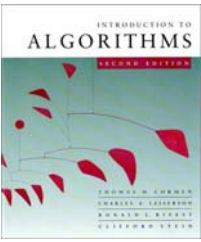
$$\begin{aligned}0 &\leq \text{weight of cycle} \\ &< 0\end{aligned}$$

Therefore, no values for the x_i can satisfy the constraints. □



Satisfying the constraints

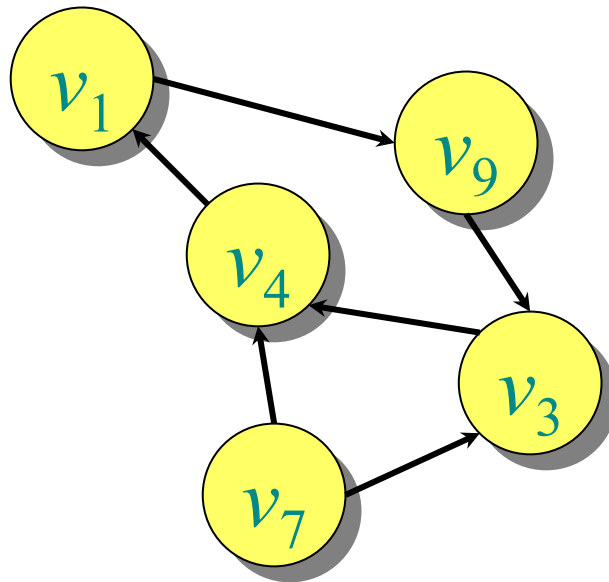
Theorem. Suppose no negative-weight cycle exists in the constraint graph. Then, the constraints are satisfiable.



Satisfying the constraints

Theorem. Suppose no negative-weight cycle exists in the constraint graph. Then, the constraints are satisfiable.

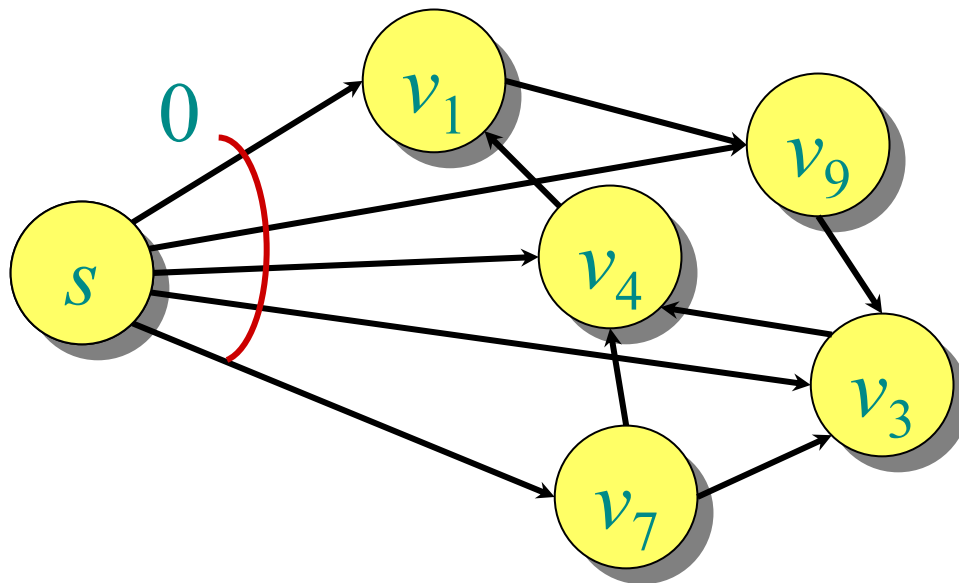
Proof. Add a new vertex s to V with a 0-weight edge to each vertex $v_i \in V$.



Satisfying the constraints

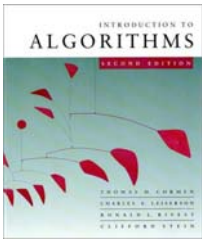
Theorem. Suppose no negative-weight cycle exists in the constraint graph. Then, the constraints are satisfiable.

Proof. Add a new vertex s to V with a 0-weight edge to each vertex $v_i \in V$.



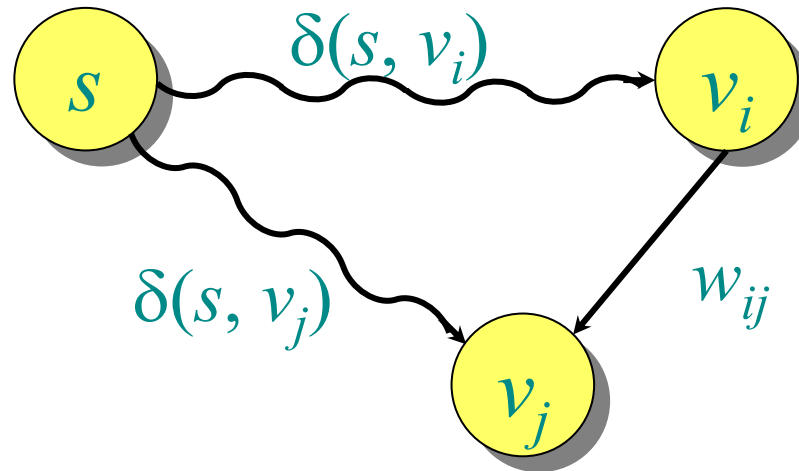
Note:

No negative-weight cycles introduced \Rightarrow shortest paths exist.

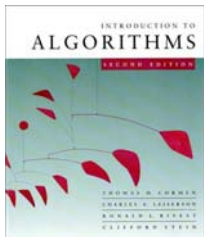


Proof (continued)

Claim: The assignment $x_i = \delta(s, v_i)$ solves the constraints. Consider any constraint $x_j - x_i \leq w_{ij}$, and consider the shortest paths from s to v_j and v_i :



The triangle inequality gives us $\delta(s, v_j) \leq \delta(s, v_i) + w_{ij}$. Since $x_i = \delta(s, v_i)$ and $x_j = \delta(s, v_j)$, the constraint $x_j - x_i \leq w_{ij}$ is satisfied. □



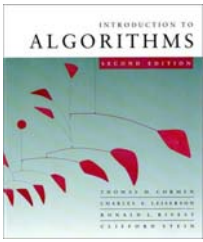
Bellman-Ford and linear programming

Corollary. The Bellman-Ford algorithm can solve a system of m difference constraints on n variables in $O(mn)$ time. \square

Single-source shortest paths is a simple LP problem.

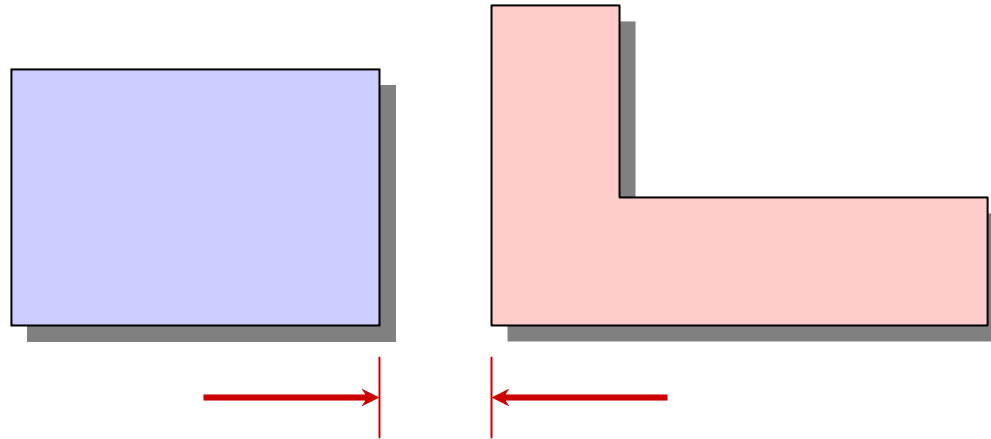
In fact, Bellman-Ford maximizes $x_1 + x_2 + \dots + x_n$ subject to the constraints $x_j - x_i \leq w_{ij}$ and $x_i \leq 0$ (exercise).

Bellman-Ford also minimizes $\max_i \{x_i\} - \min_i \{x_i\}$ (exercise).



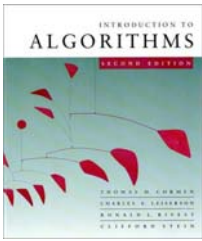
Application to VLSI layout compaction

*Integrated
-circuit
features:*

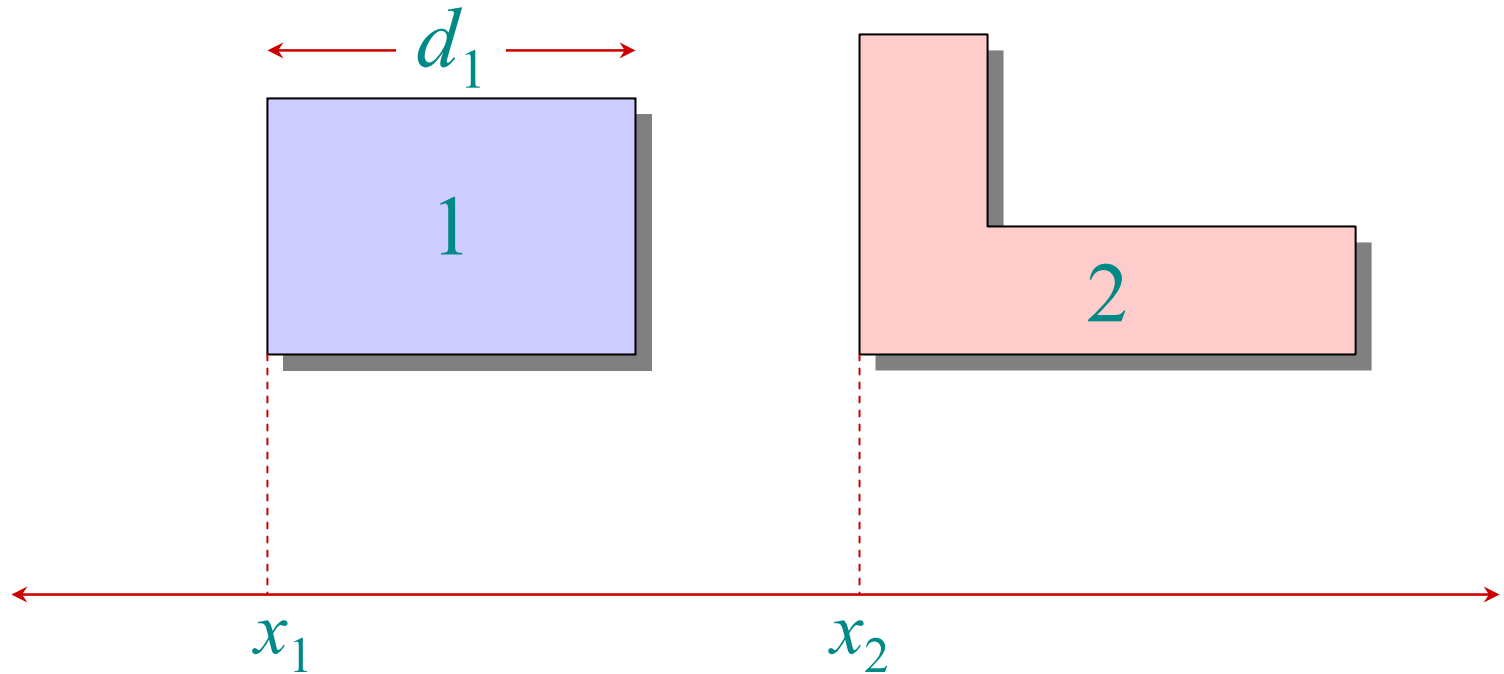


minimum separation λ

Problem: Compact (in one dimension) the space between the features of a VLSI layout without bringing any features too close together.



VLSI layout compaction



Constraint: $x_2 - x_1 \geq d_1 + \lambda$

Bellman-Ford minimizes $\max_i \{x_i\} - \min_i \{x_i\}$, which compacts the layout in the x -dimension.