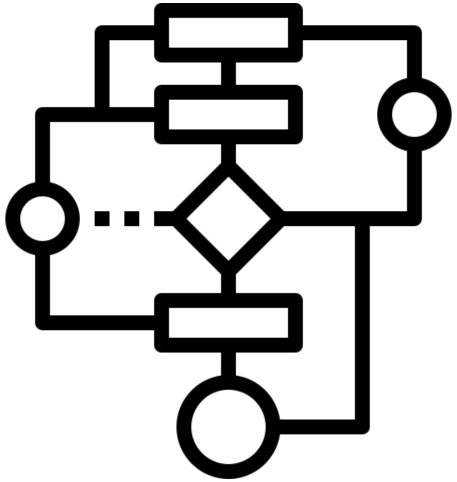
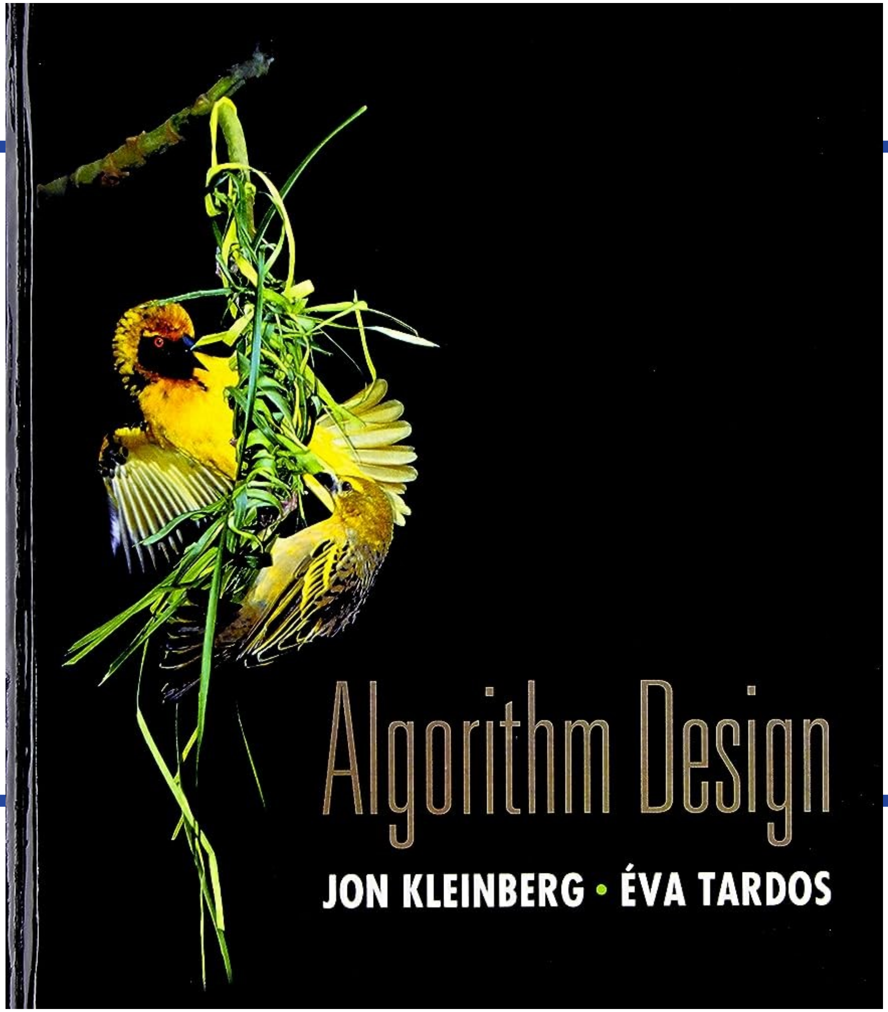


CS 310: Algorithms

Lecture 20

Instructor: Naveed Anwar Bhatti



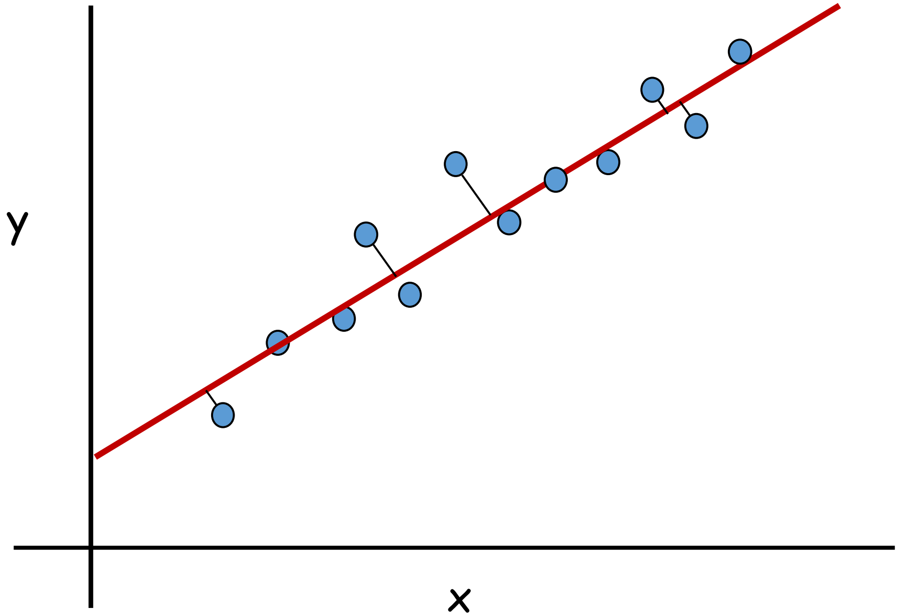
Chapter 6:
Dynamic Programming

Section :
Segmented Least Squares

Segmented Least Squares

- **Least squares:**
  - Foundational problem in statistic and numerical analysis.
  - Given n points in the plane: $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$.
  - Find a line y = ax + b that minimizes the sum of the squared error:

$$E = \sum_{i=1}^{n} (y_i - a x_i - b)^2$$



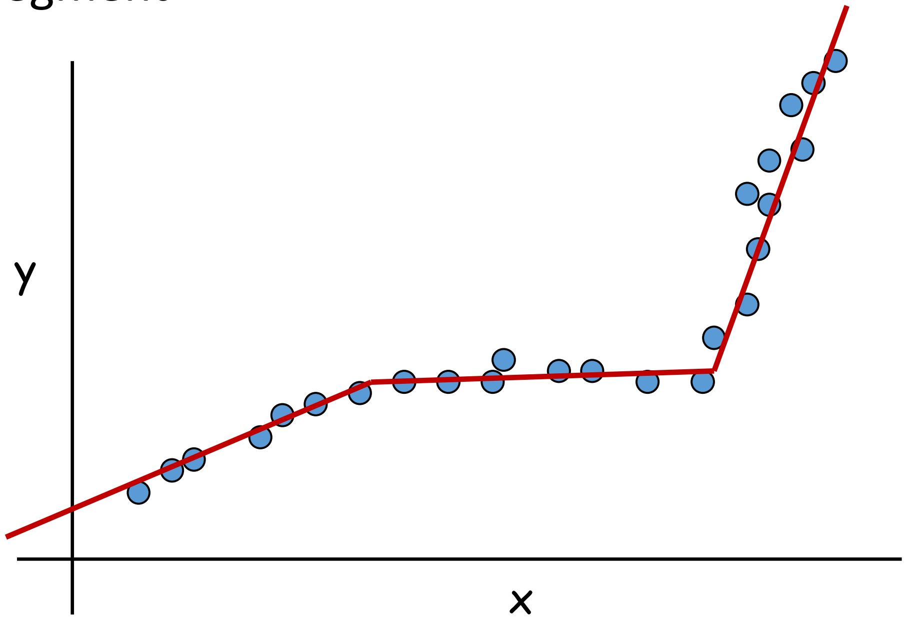
- **Solution:** Calculus $\Rightarrow$ min error is achieved when

$$a = \frac{n \sum_i x_i y_i - (\sum_i x_i)(\sum_i y_i)}{n \sum_i x_i^2 - \left(\sum_i x_i\right)^2}$$

$$b = \frac{\sum_i y_i - a \sum_i x_i}{n}$$

Segmented Least Squares

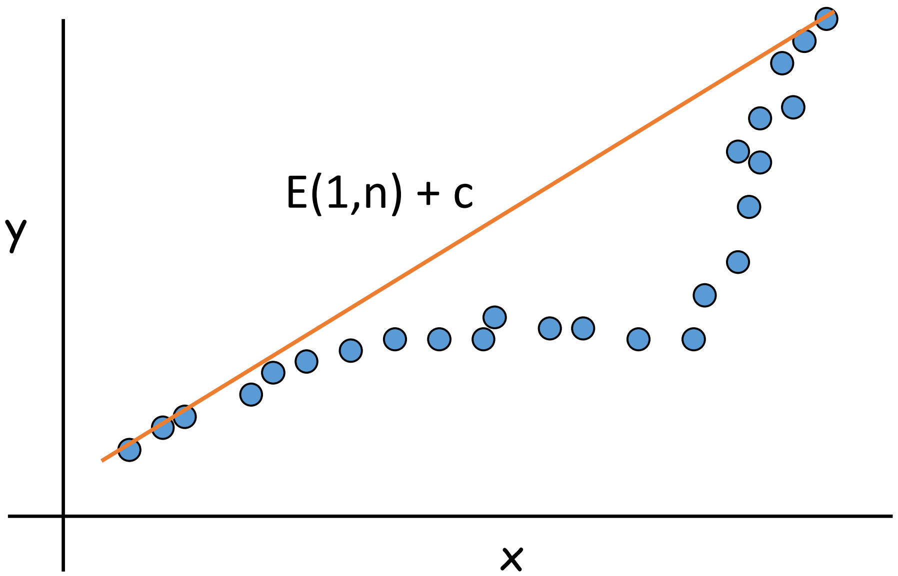
- **Segmented least squares:**
  - Points lie roughly on a sequence of **several line segments**
  - Given ***n*** points in the plane $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$
  - find a sequence of lines that minimizes ***f***(x):
    - the sum of the sums of the squared errors **E** in each segment
    - the number of lines L



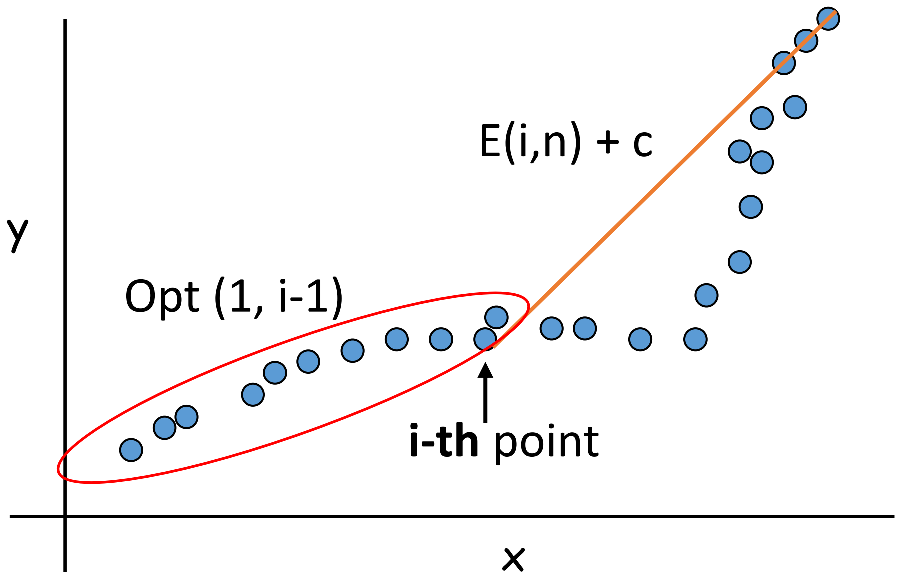
Q: What's a reasonable choice for ***f***(x) to balance accuracy and parsimony?

Tradeoff function: **E + c L**, for some constant **c > 0**

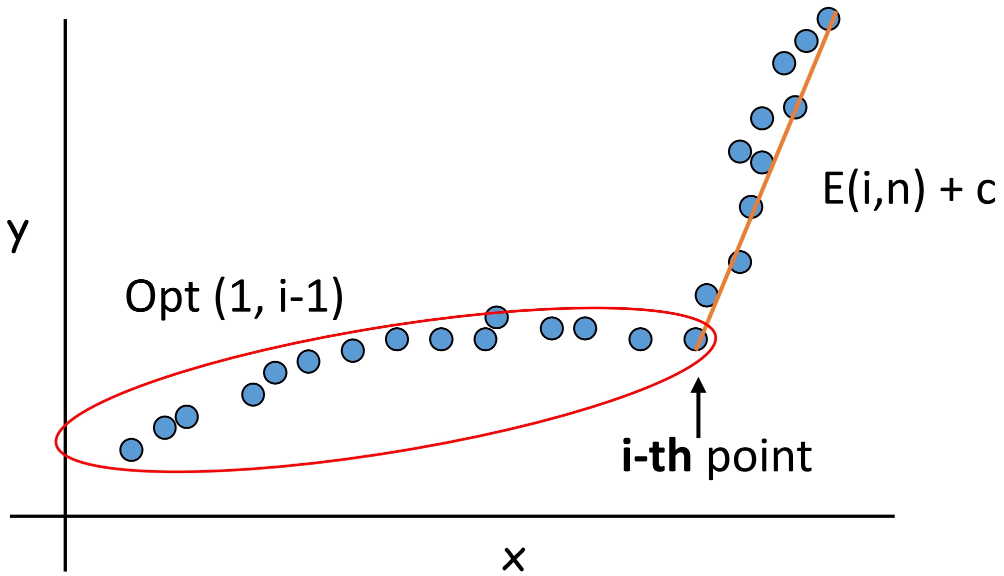
Segmented Least Squares



Segmented Least Squares



Segmented Least Squares



Dynamic Programming: Multiway Choice

- Notation.
  - OPT(n) = minimum cost for points p_1, p_{i+1} , . . . , p_n.

- To compute OPT(n):
  - Last segment uses points p_i, p_{i+1} , . . . , p_n for some **i**.
  - e(i, n) = minimum sum of squares for points p_i, p_{i+1} , . . . , p_n.
  - Cost = e(i, n) + c + OPT(i-1).

$$OPT(n) = \begin{cases} 0 & \text{if } n = 0 \\ \min\limits_{1 \le i \le n} \{e(i,n) + c + OPT(i-1)\} & \text{otherwise} \end{cases}$$

Dynamic Programming: Multiway Choice

$$OPT(n) = \begin{cases} 0 & \text{if } n = 0 \\ \min\limits_{1 \le i \le n} \{e(i,n) + c + OPT(i-1)\} & \text{otherwise} \end{cases}$$

Let say we have 4 points i.e., **n=4**

OPT(4) = min{ e(1, 4) + c + OPT(0) , e(2, 4) + c + OPT(1) , e(3, 4) + c + OPT(2) , e(4, 4) + c + OPT(3)}

OPT(3) = min{ e(1, 3) + c + OPT(0), e(2, 3) + c + OPT(1), e(3, 3) + c + OPT(2)}

OPT(2) = min{ e(1, 2) + c + OPT(0), e(2, 2) + c + OPT(1)}

OPT(1) = min{ e(1, 1) + c + OPT(0)}

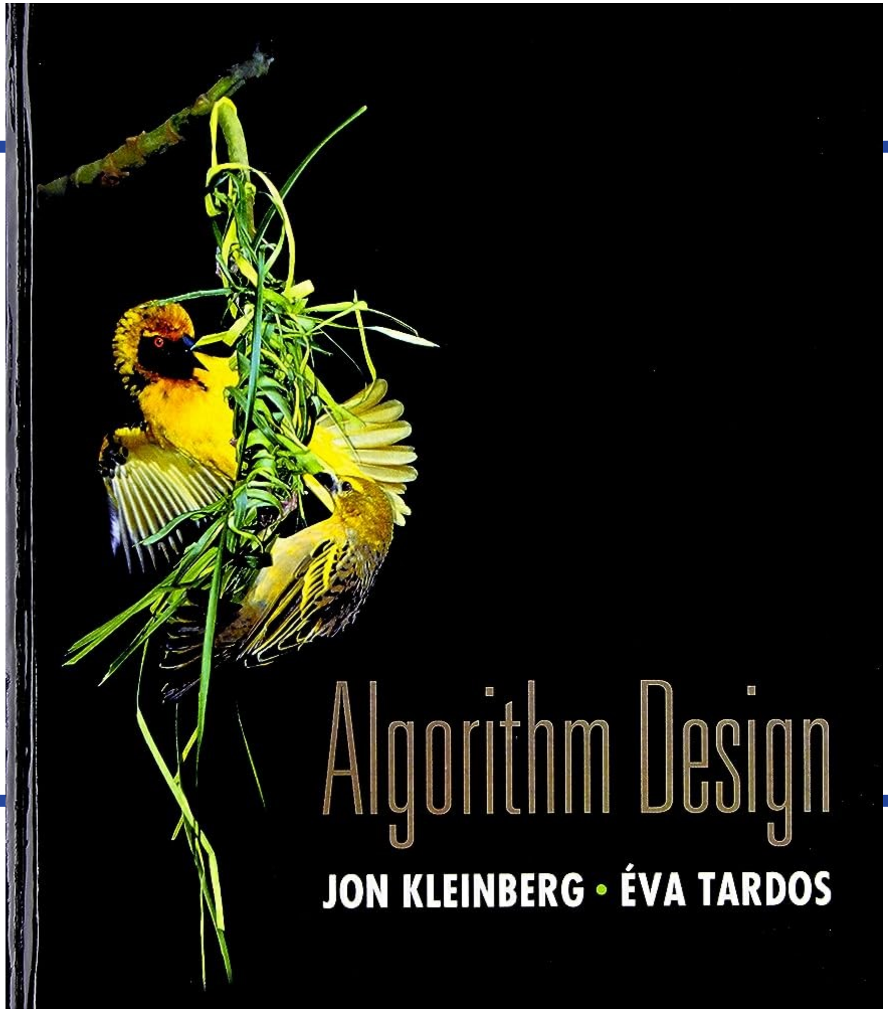
OPT(0) = 0

Segmented Least Squares: Algorithm

```
INPUT: n, p1,…,pN , c

for i=0 to n:
     M[n]= -1
M[0]= 0

OPT(){
    if n == 0:
        return 0

    if M[0]!=-1:
        return M[n]

    min_cost = 0;                                          O(n)

    for i=1 to n:   O(n)
        cost = e(i,n) + c + OPT(i - 1)    O(n)
        if cost < min_cost:
            min_cost = cost

    M[n] = min_cost
    return min_cost
}
```

Running time: $O(n^3)$.

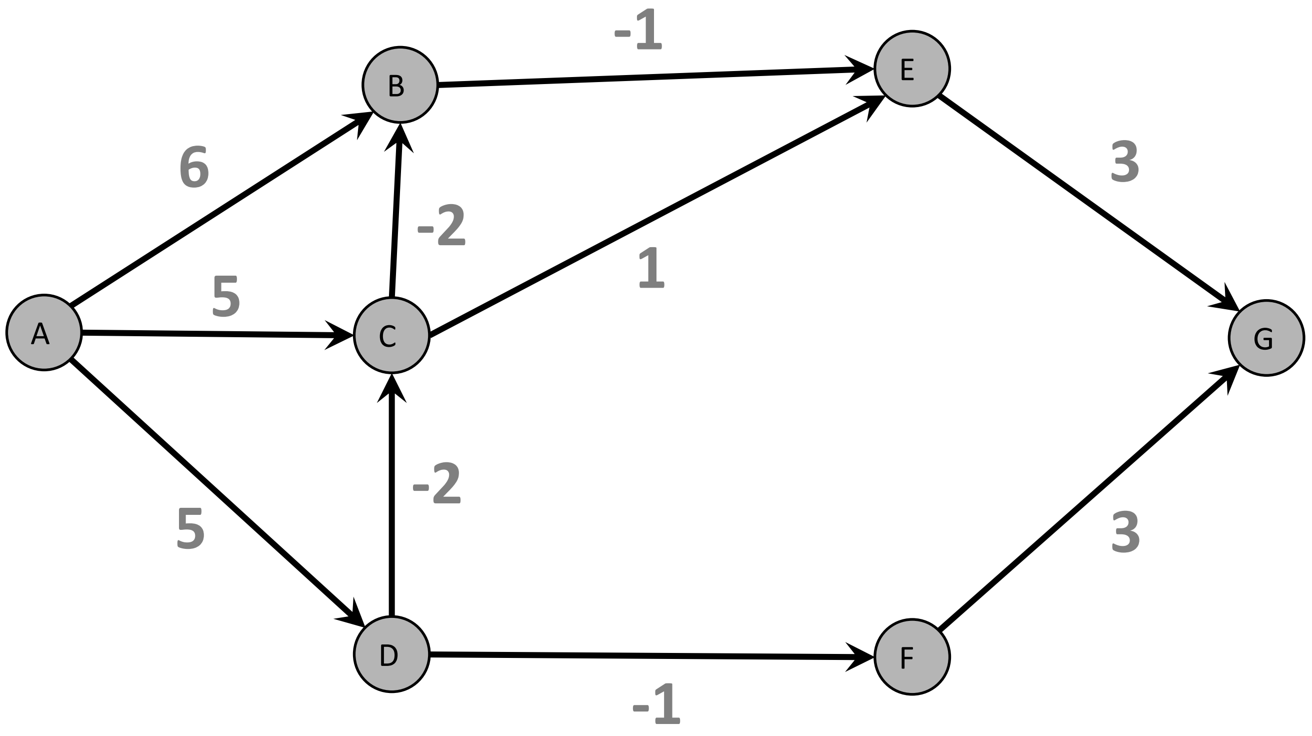
- Bottleneck = computing e(i, j) for $O(n^2)$ pairs, $O(n)$ per pair using previous formula.



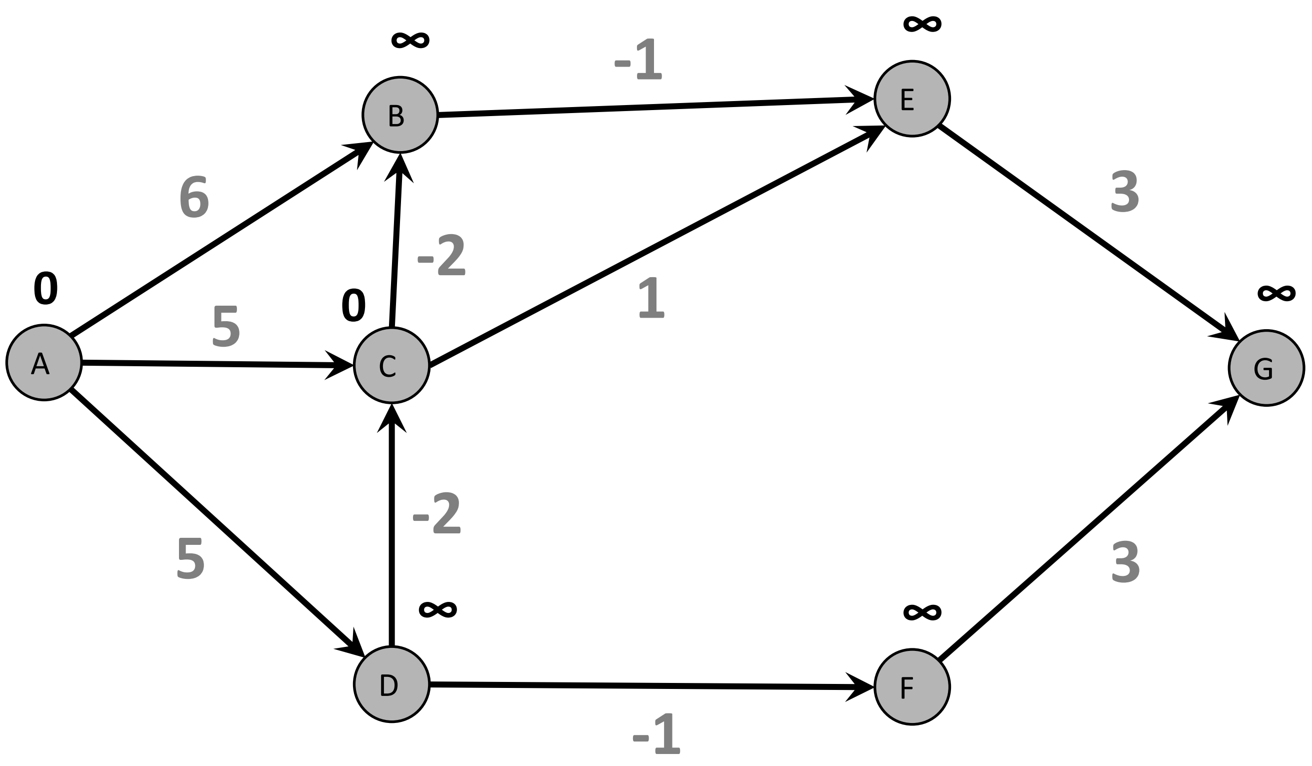
Chapter 6:
Dynamic Programming

Section : **Bonus Topic**
Bellman-ford Algorithm

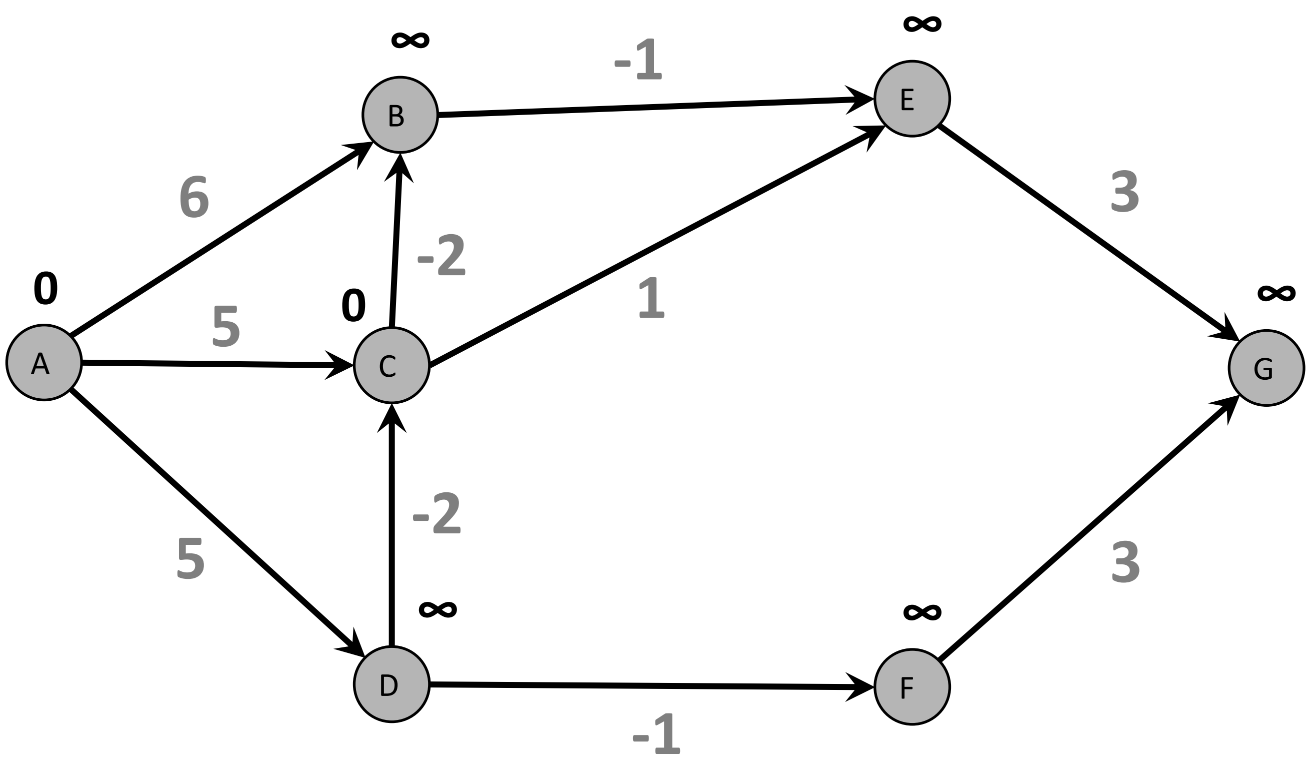
Bellman-ford: Single Source Shortest Path



Bellman-ford: Single Source Shortest Path



Bellman-ford: Single Source Shortest Path

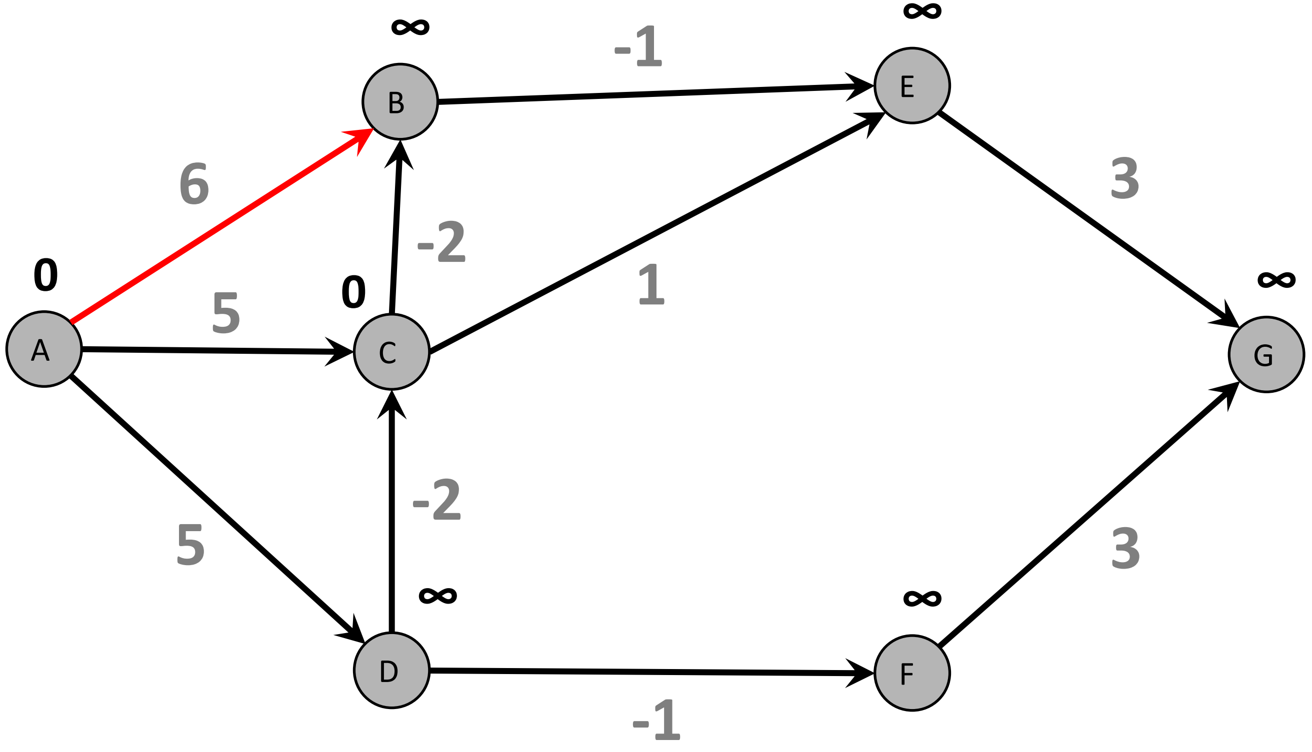


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (A,D) (C,B) (D,C) (D,F) (C,E) (B,E) (F,G) (E,G)

Bellman-ford: Single Source Shortest Path

1st Iteration

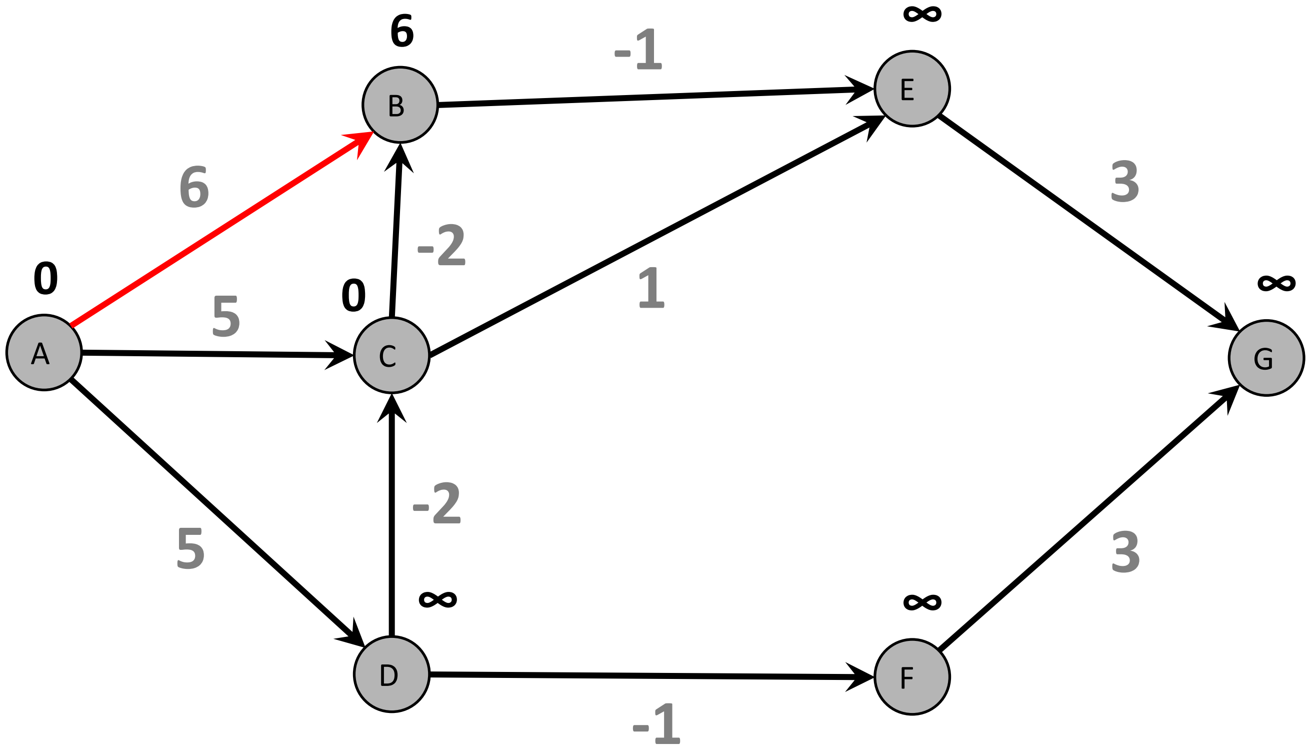


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (A,D) (C,B) (D,C) (D,F) (C,E) (B,E) (F,G) (E,G)

Bellman-ford: Single Source Shortest Path

1st Iteration

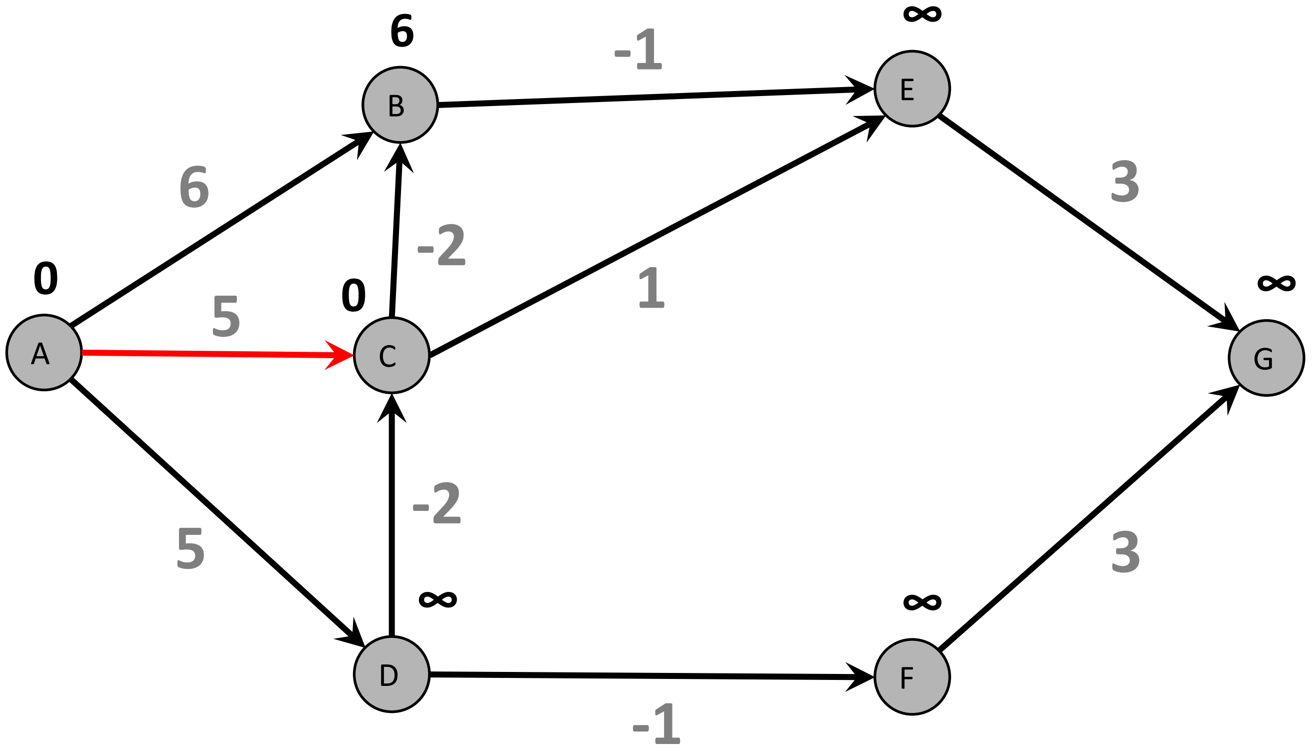


If $(d[u]+ e(u,v) < d[v])$
$d[v]=d[u]+ e(u,v)$

(A,B) (A,C) (A,D) (C,B) (D,C) (D,F) (C,E) (B,E) (F,G) (E,G)

Bellman-ford: Single Source Shortest Path

1st Iteration

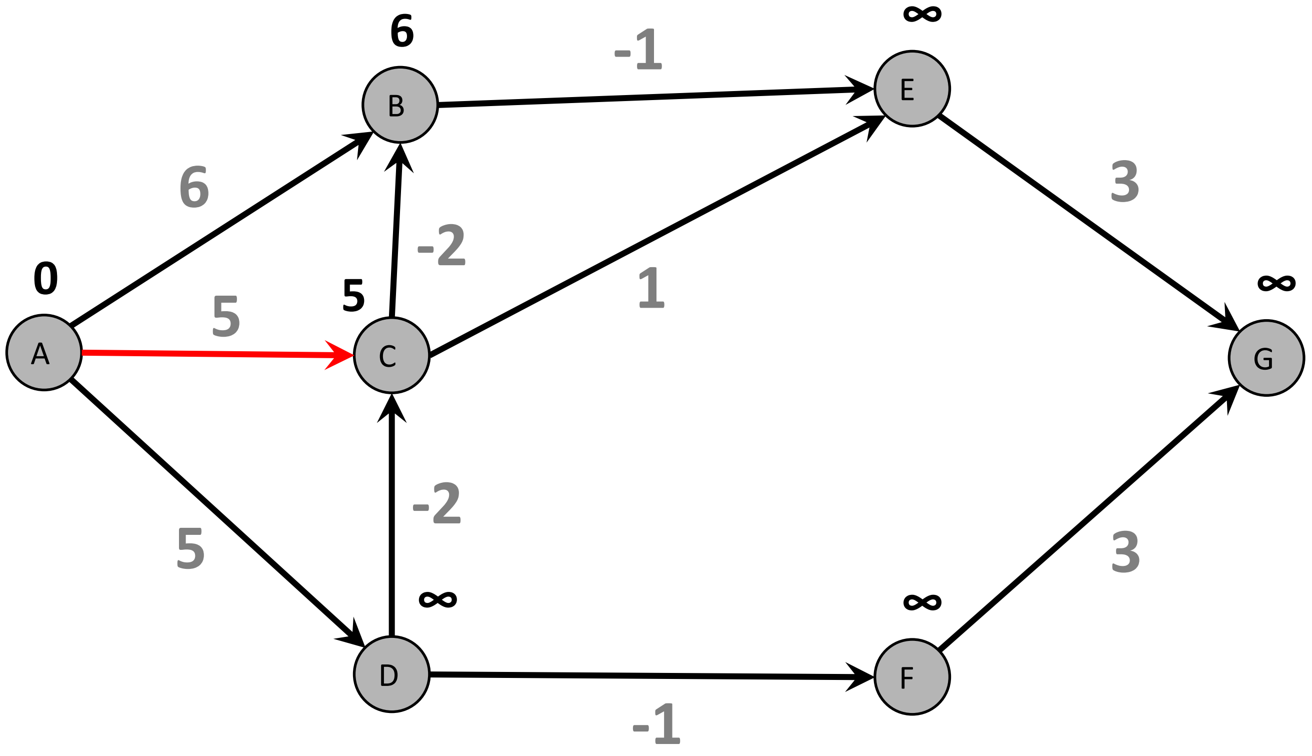


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) **(A,C)** (A,D) (C,B) (D,C) (D,F) (C,E) (B,E) (F,G) (E,G)

Bellman-ford: Single Source Shortest Path

1st Iteration

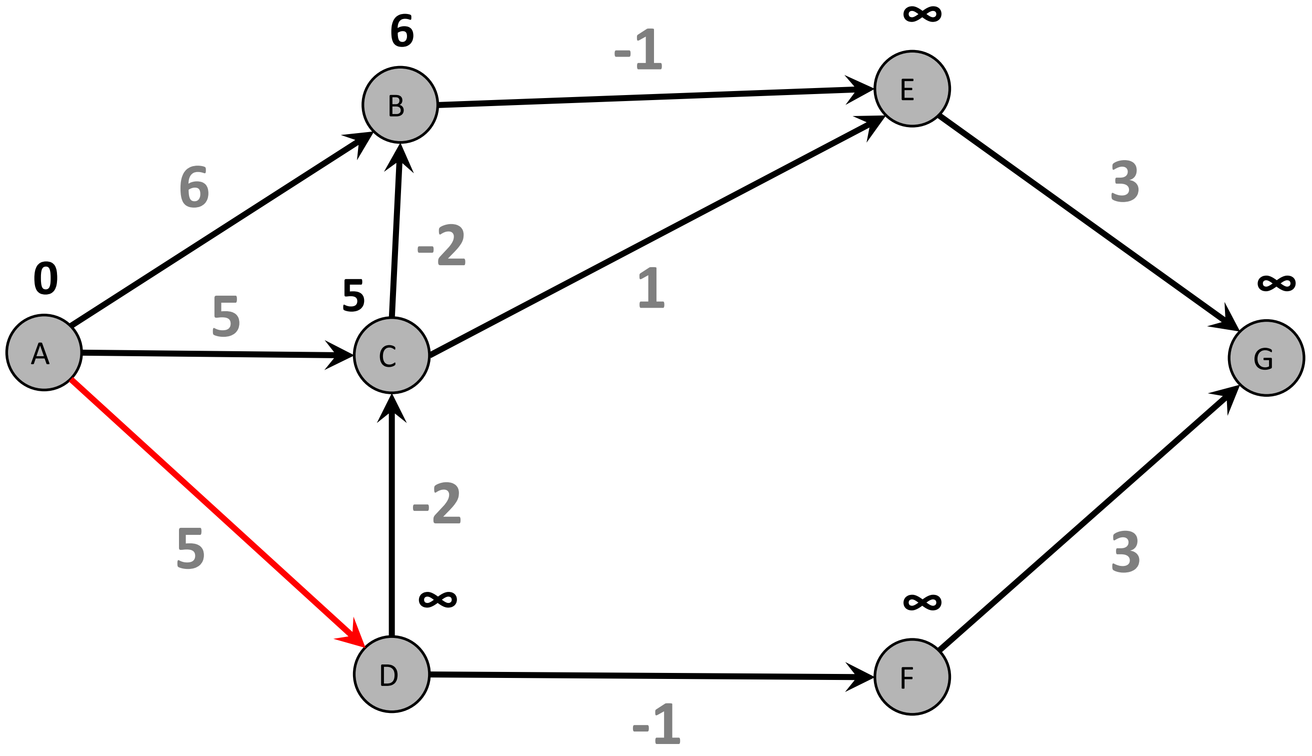


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) **(A,C)** (A,D) (C,B) (D,C) (D,F) (C,E) (B,E) (F,G) (E,G)

Bellman-ford: Single Source Shortest Path

1st Iteration

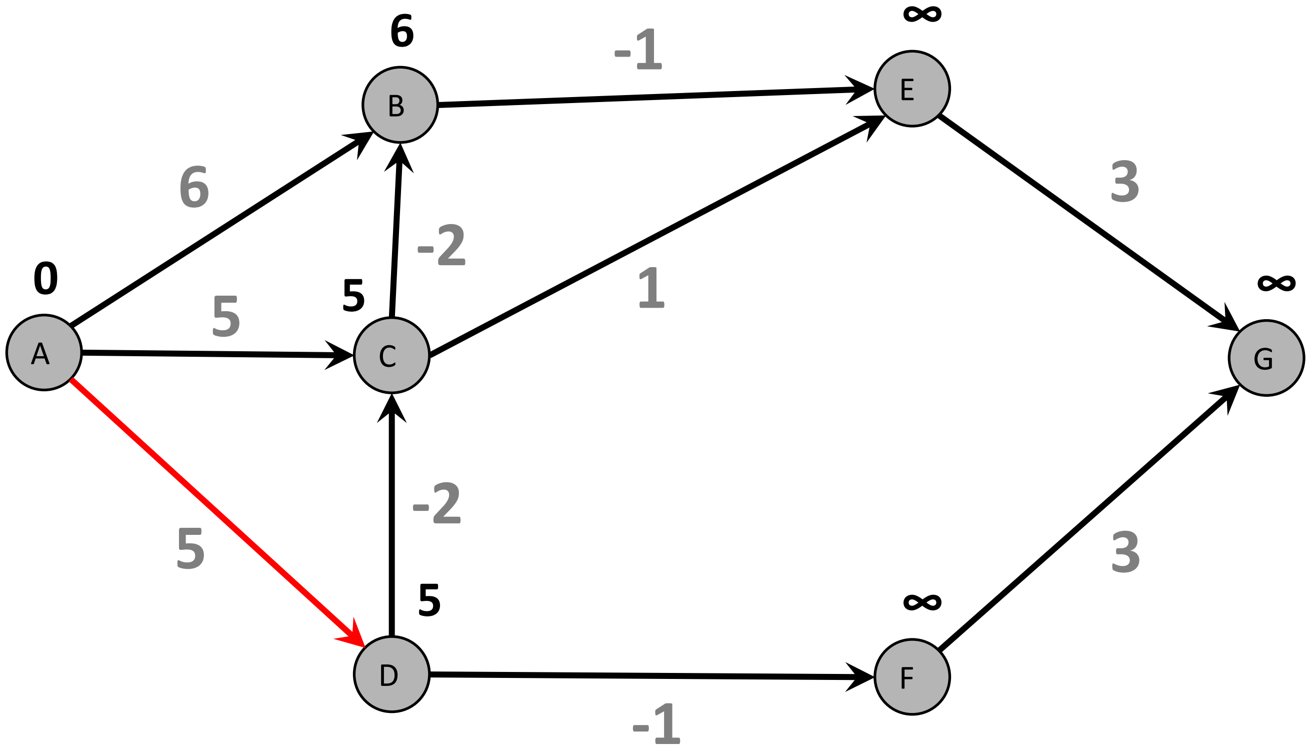


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) **(A,D)** (C,B) (D,C) (D,F) (C,E) (B,E) (F,G) (E,G)

Bellman-ford: Single Source Shortest Path

1st Iteration

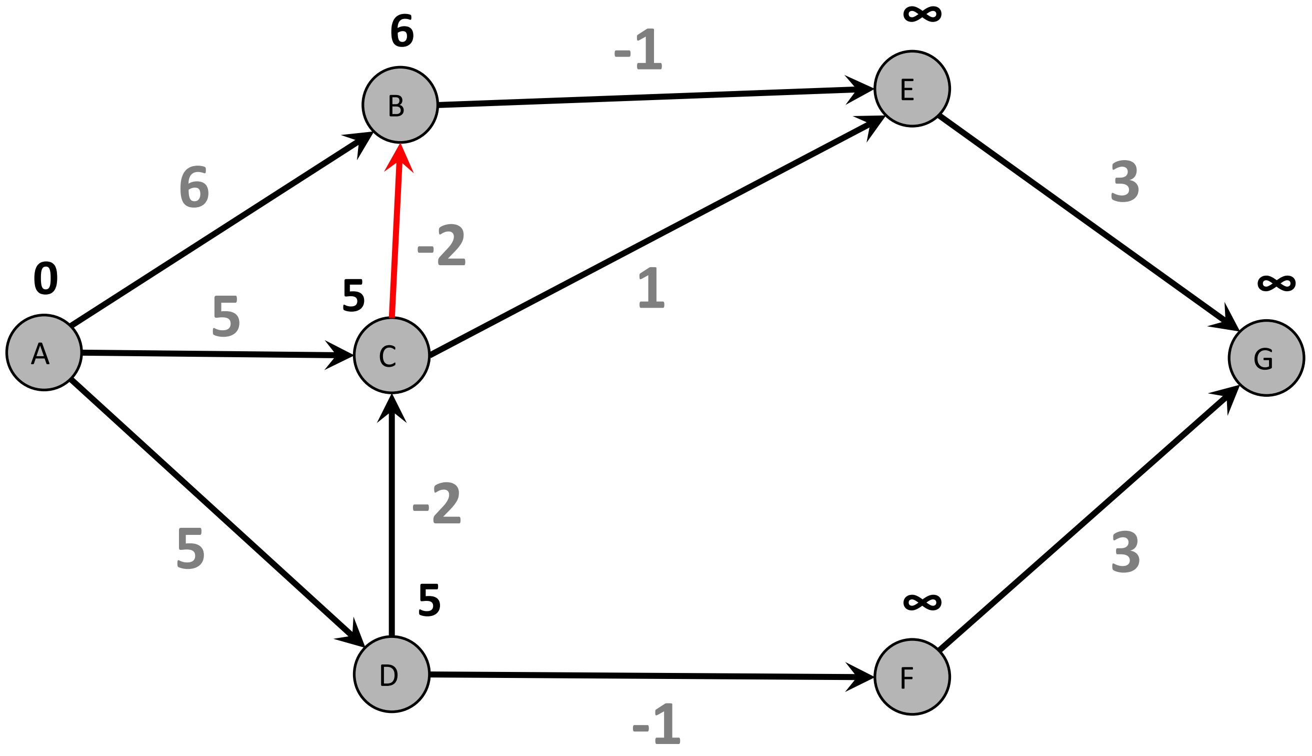


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) **(A,D)** (C,B) (D,C) (D,F) (C,E) (B,E) (F,G) (E,G)

Bellman-ford: Single Source Shortest Path

1st Iteration

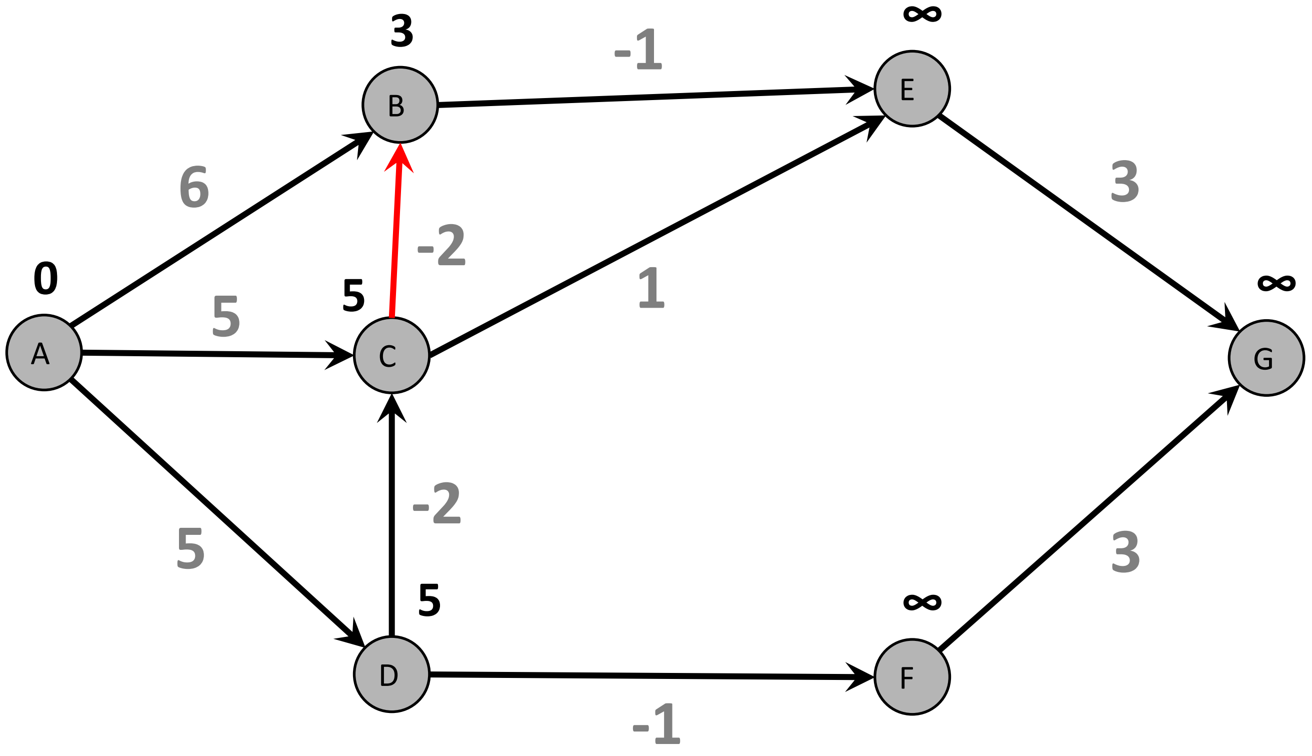


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (A,D) **(C,B)** (D,C) (D,F) (C,E) (B,E) (F,G) (E,G)

Bellman-ford: Single Source Shortest Path

1st Iteration

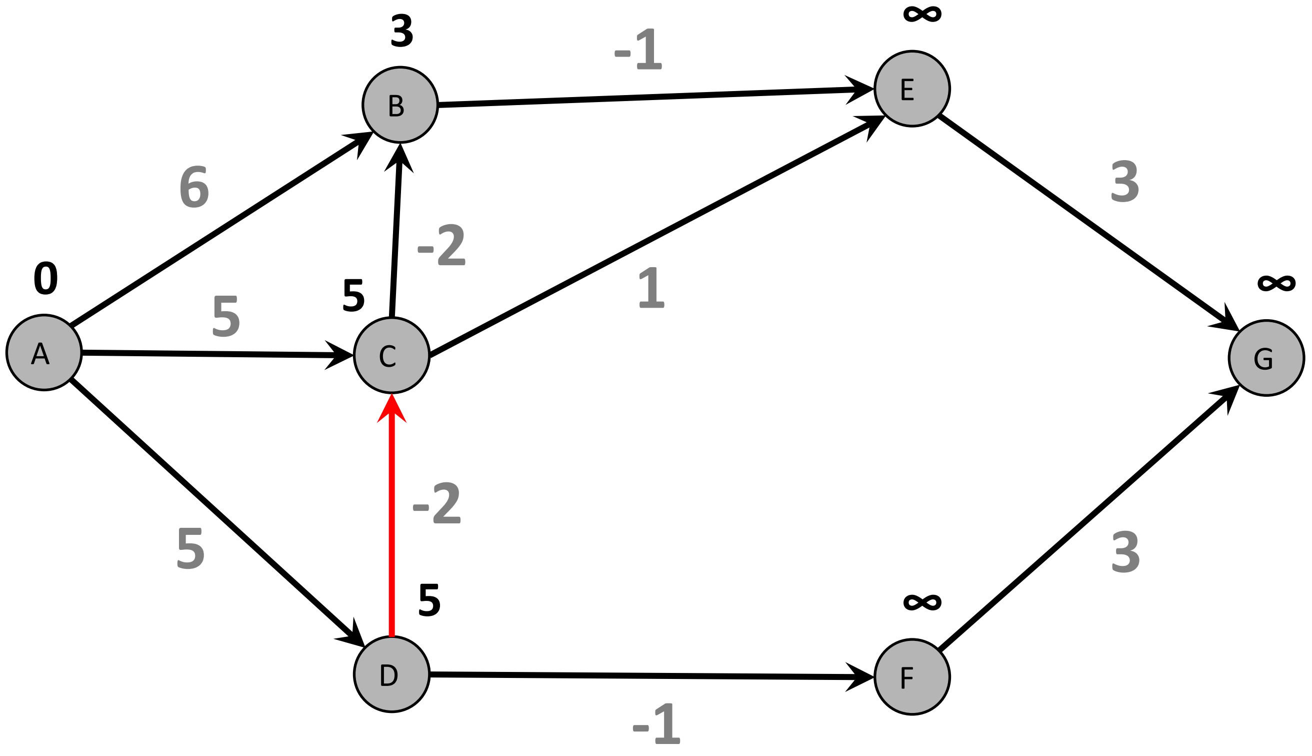


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (A,D) **(C,B)** (D,C) (D,F) (C,E) (B,E) (F,G) (E,G)

Bellman-ford: Single Source Shortest Path

1st Iteration

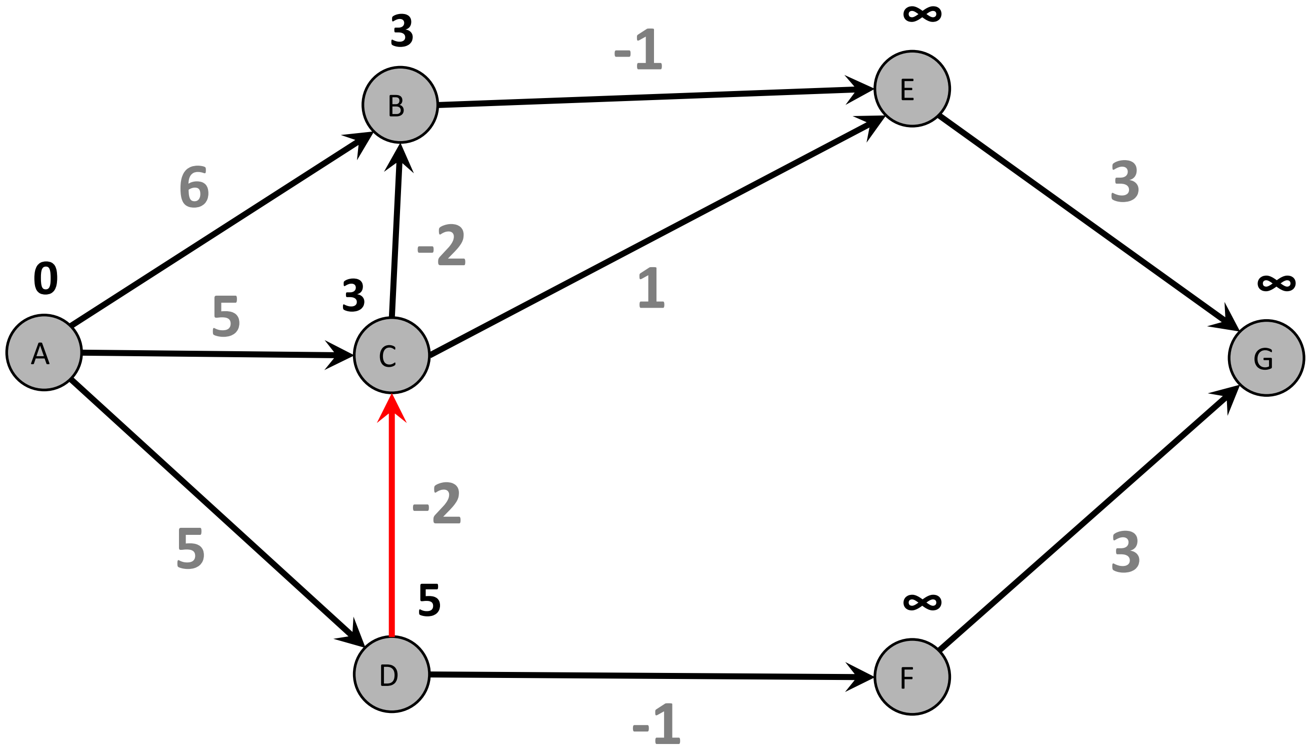


If $(d[u]+e(u,v) < d[v])$
$d[v]=d[u]+e(u,v)$

(A,B) (A,C) (A,D) (C,B) **(D,C)** (D,F) (C,E) (B,E) (F,G) (E,G)

Bellman-ford: Single Source Shortest Path

1st Iteration

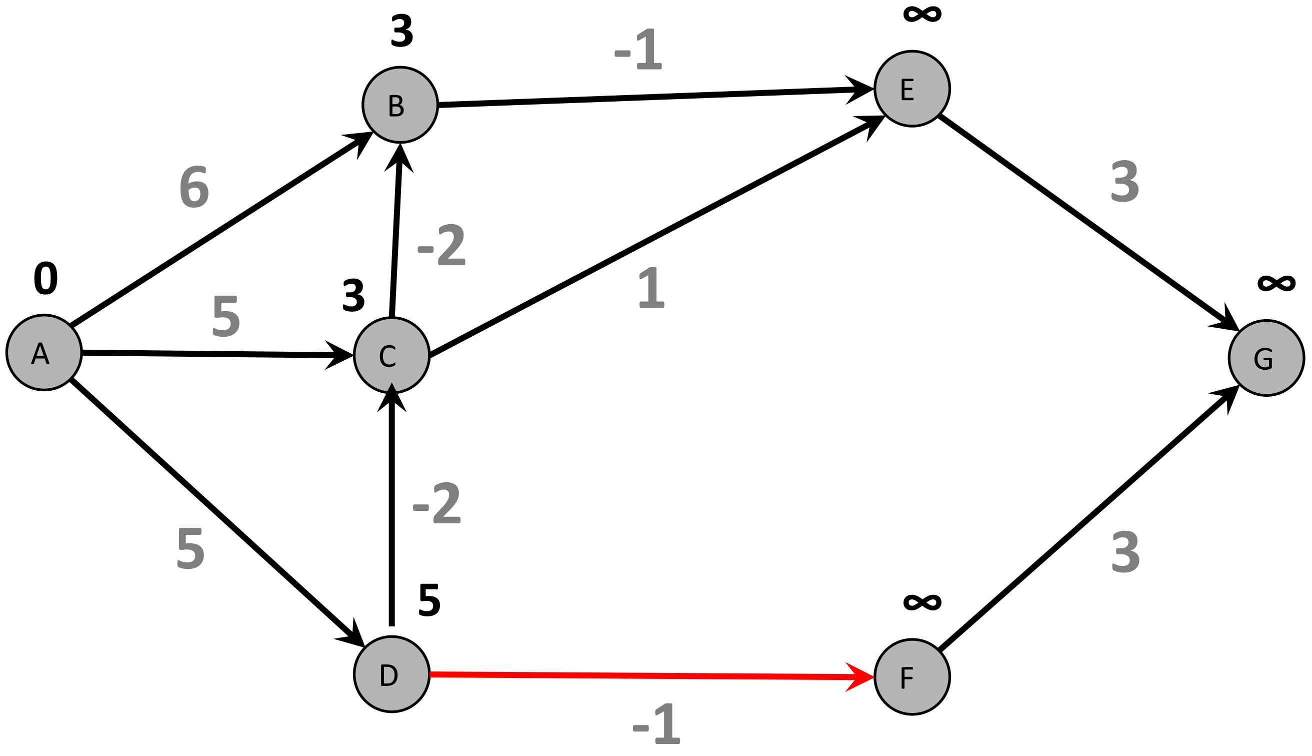


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (A,D) (C,B) **(D,C)** (D,F) (C,E) (B,E) (F,G) (E,G)

Bellman-ford: Single Source Shortest Path

1st Iteration

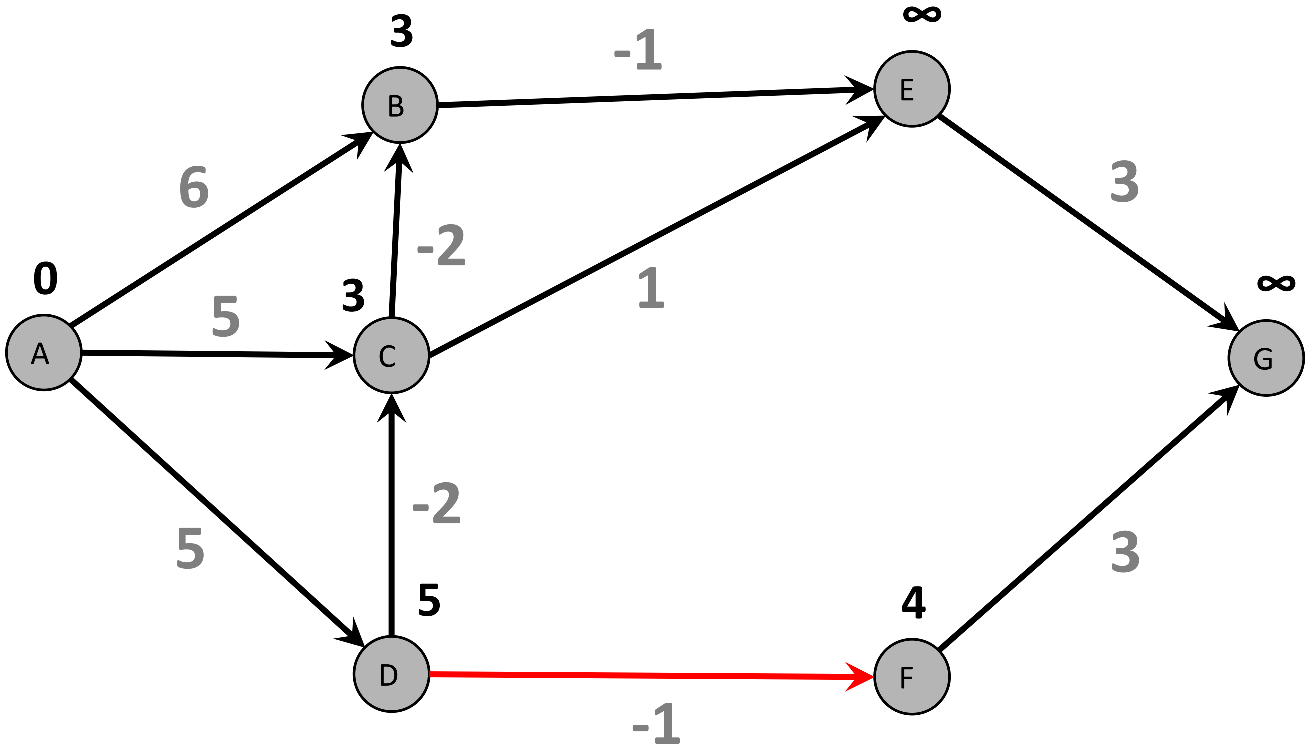


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (A,D) (C,B) (D,C) **(D,F)** (C,E) (B,E) (F,G) (E,G)

Bellman-ford: Single Source Shortest Path

1st Iteration

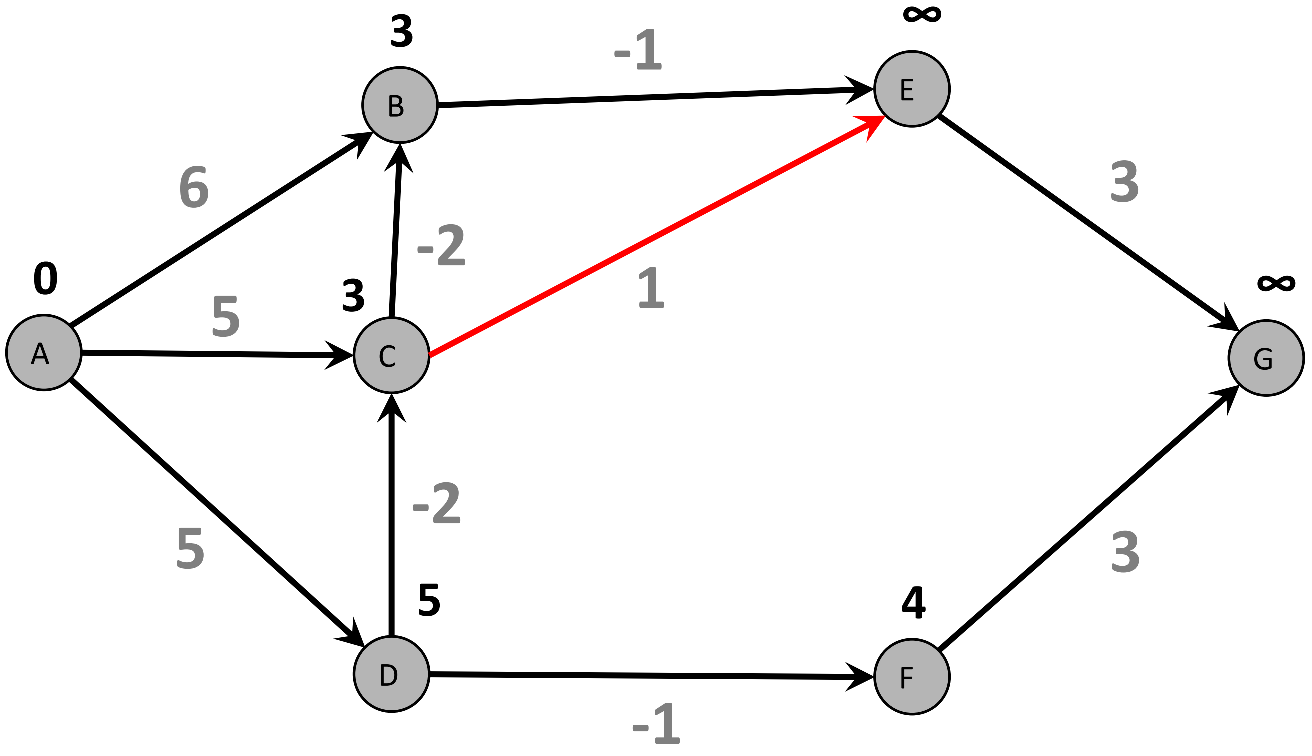


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (A,D) (C,B) (D,C) **(D,F)** (C,E) (B,E) (F,G) (E,G)

Bellman-ford: Single Source Shortest Path

1st Iteration

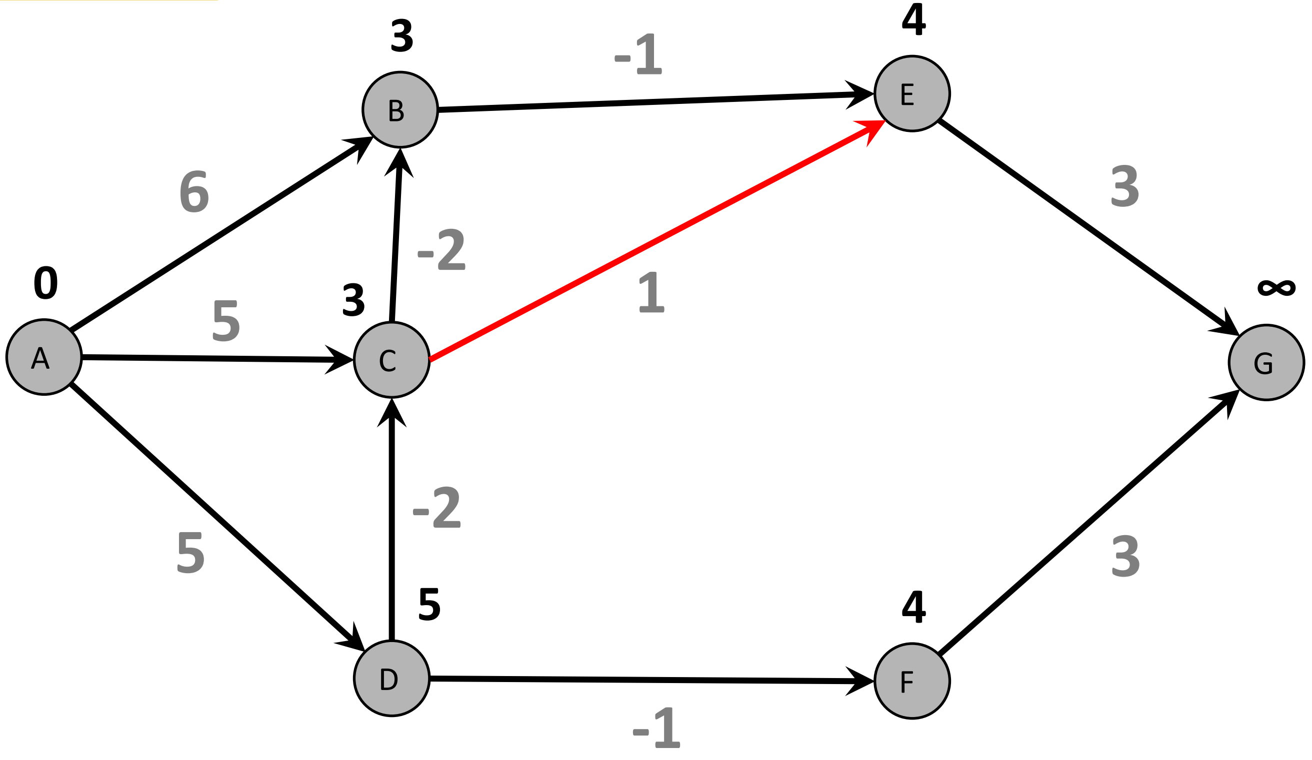


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (A,D) (C,B) (D,C) (D,F) **(C,E)** (B,E) (F,G) (E,G)

Bellman-ford: Single Source Shortest Path

1st Iteration

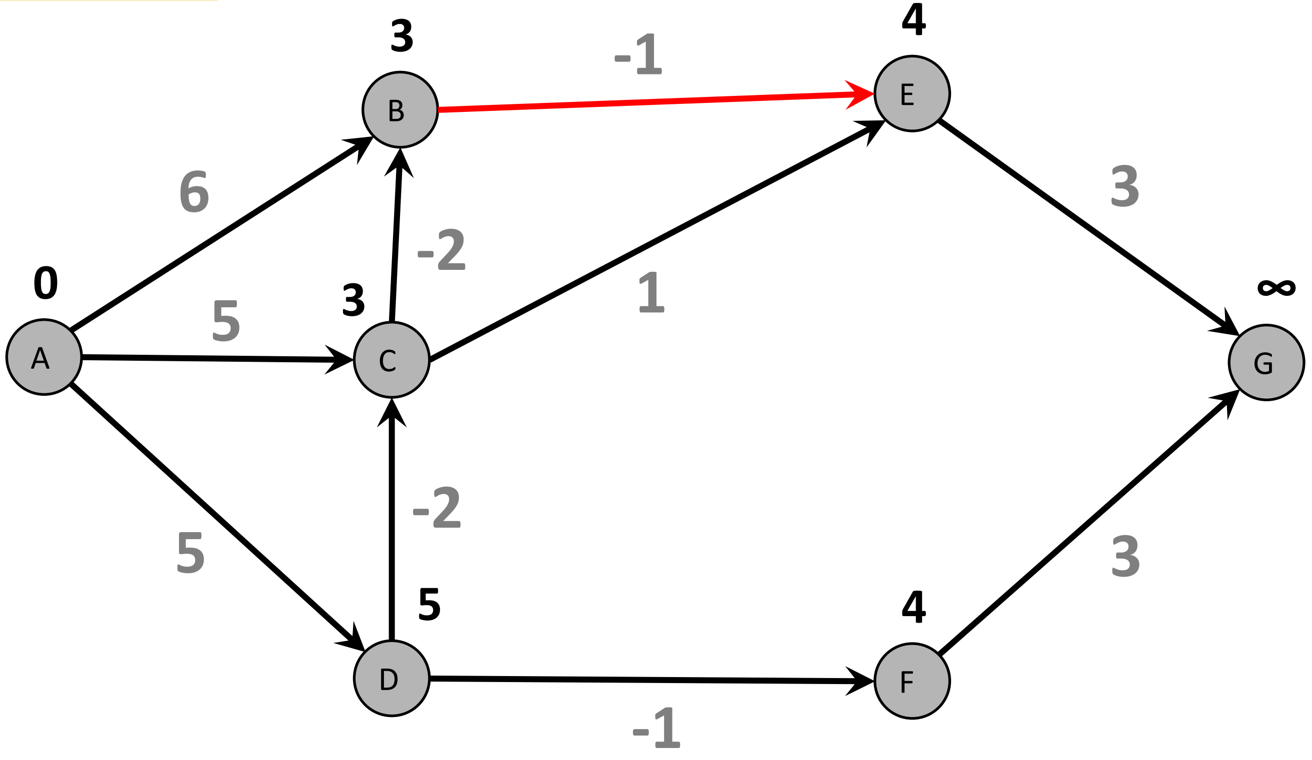


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (A,D) (C,B) (D,C) (D,F) **(C,E)** (B,E) (F,G) (E,G)

Bellman-ford: Single Source Shortest Path

1st Iteration

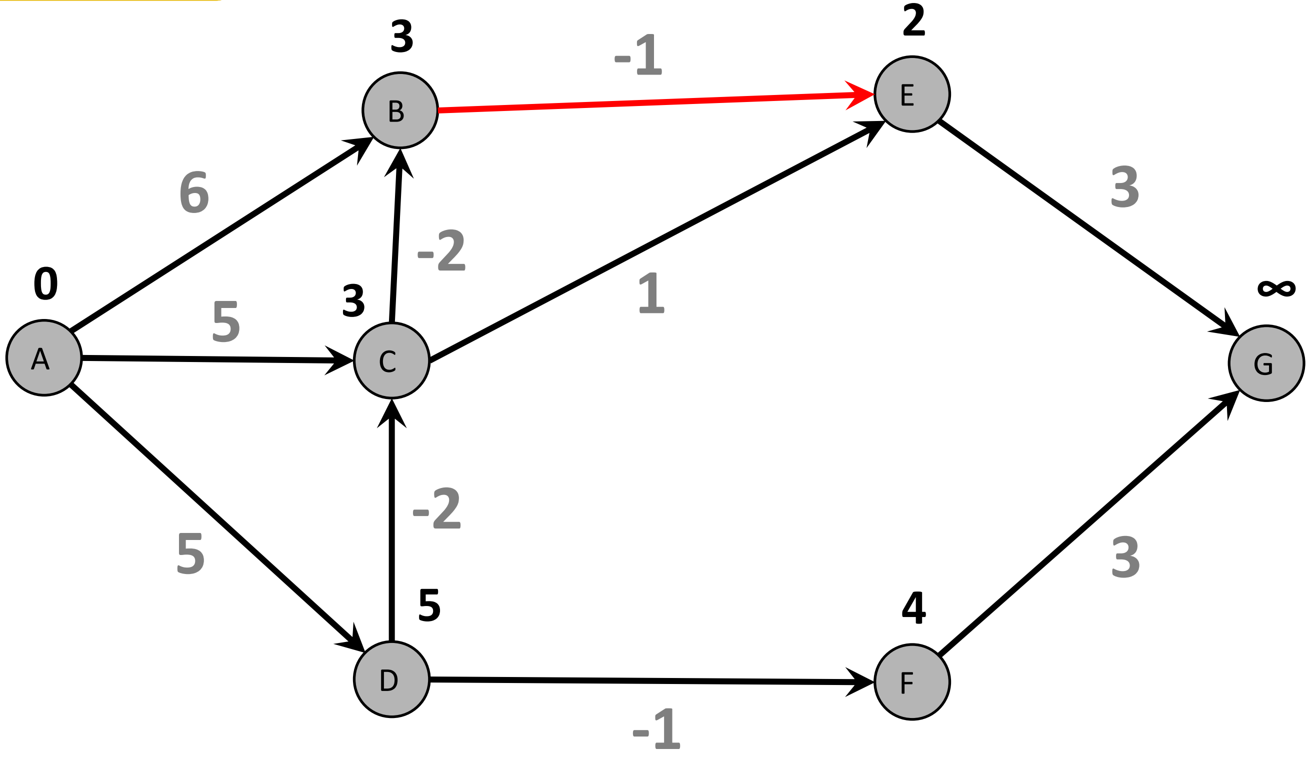


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (A,D) (C,B) (D,C) (D,F) (C,E) **(B,E)** (F,G) (E,G)

Bellman-ford: Single Source Shortest Path

1st Iteration

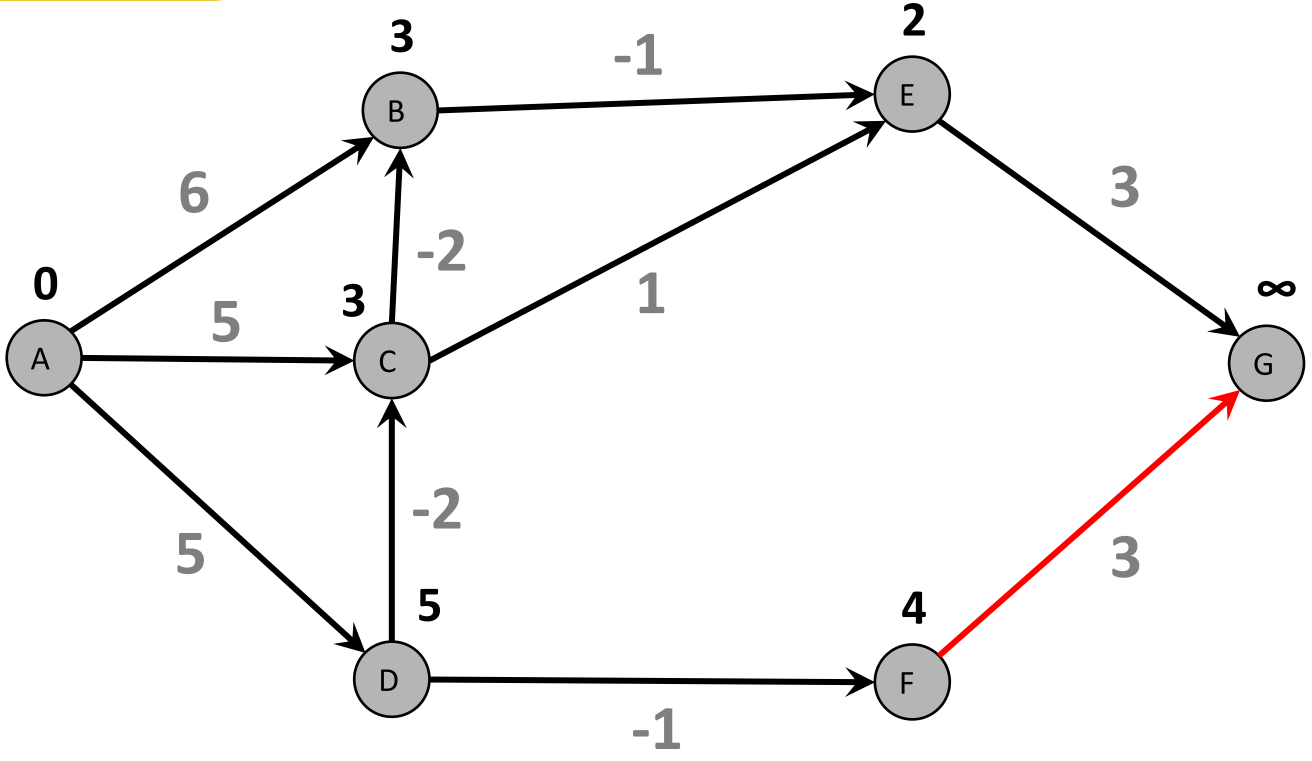


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (A,D) (C,B) (D,C) (D,F) (C,E) **(B,E)** (F,G) (E,G)

Bellman-ford: Single Source Shortest Path

1st Iteration

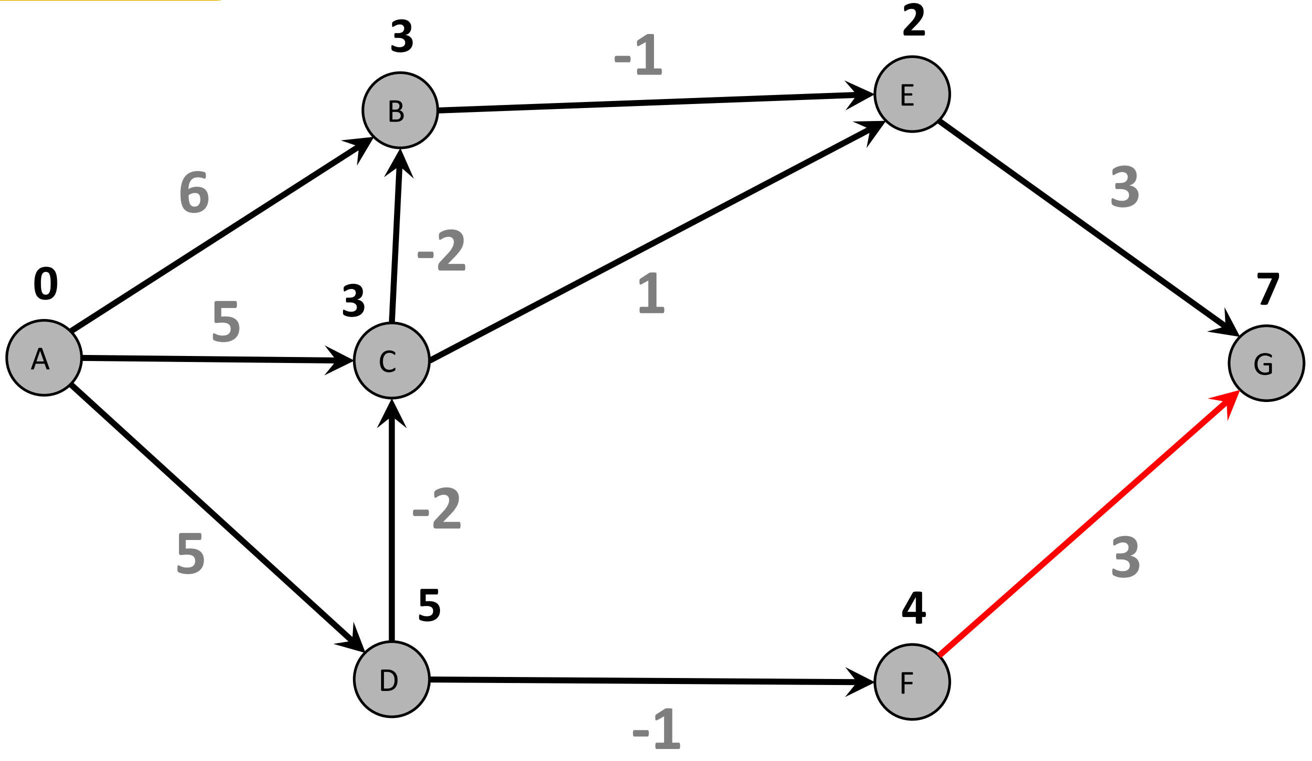


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (A,D) (C,B) (D,C) (D,F) (C,E) (B,E) **(F,G)** (E,G)

Bellman-ford: Single Source Shortest Path

1^{st} Iteration

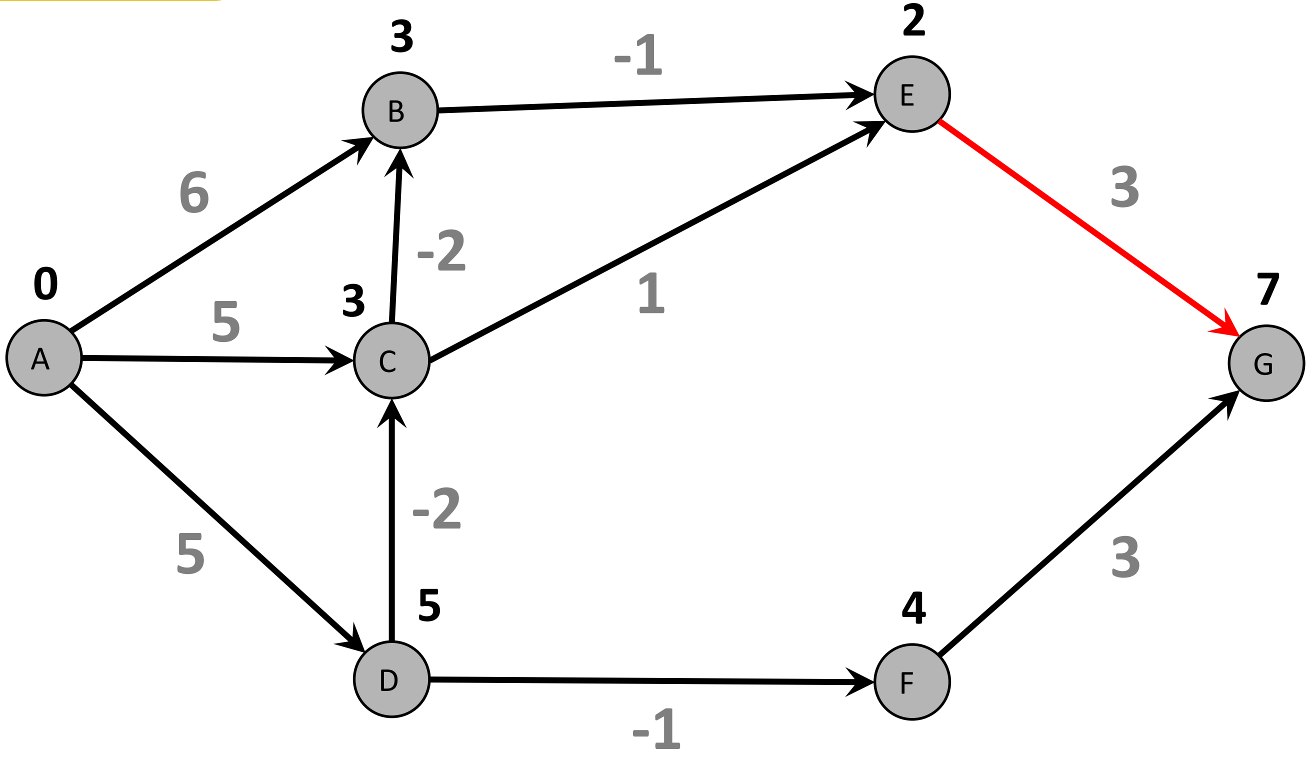


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (A,D) (C,B) (D,C) (D,F) (C,E) (B,E) **(F,G)** (E,G)

Bellman-ford: Single Source Shortest Path

1^{st} Iteration

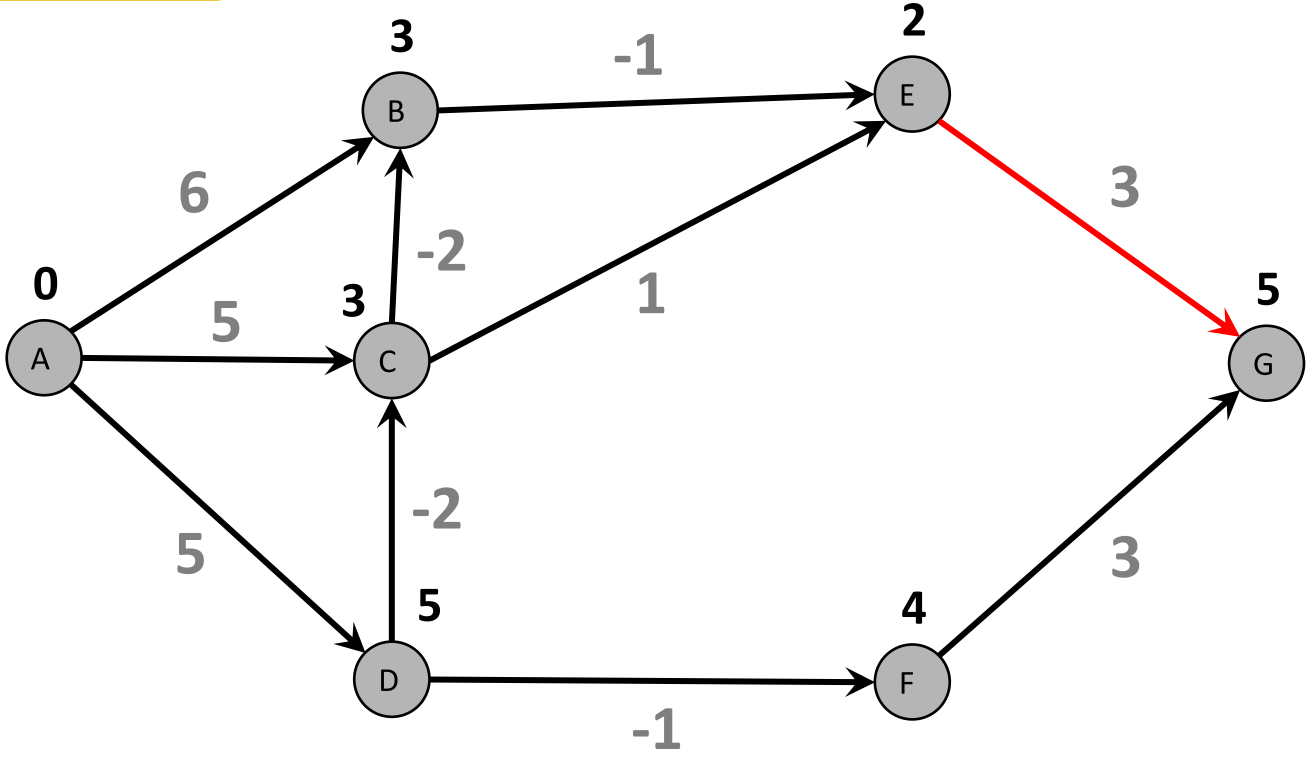


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (A,D) (C,B) (D,C) (D,F) (C,E) (B,E) (F,G) **(E,G)**

Bellman-ford: Single Source Shortest Path

1st Iteration

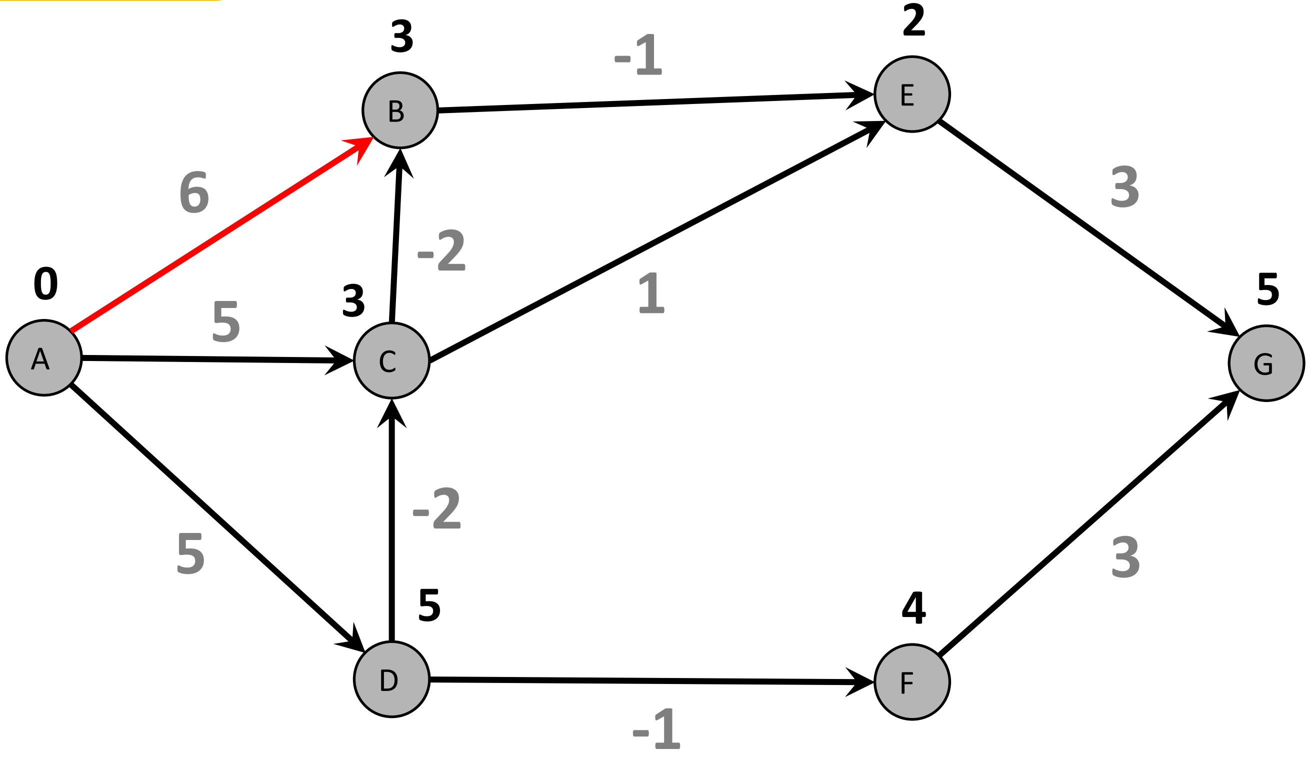


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (A,D) (C,B) (D,C) (D,F) (C,E) (B,E) (F,G) **(E,G)**

Bellman-ford: Single Source Shortest Path

2nd Iteration

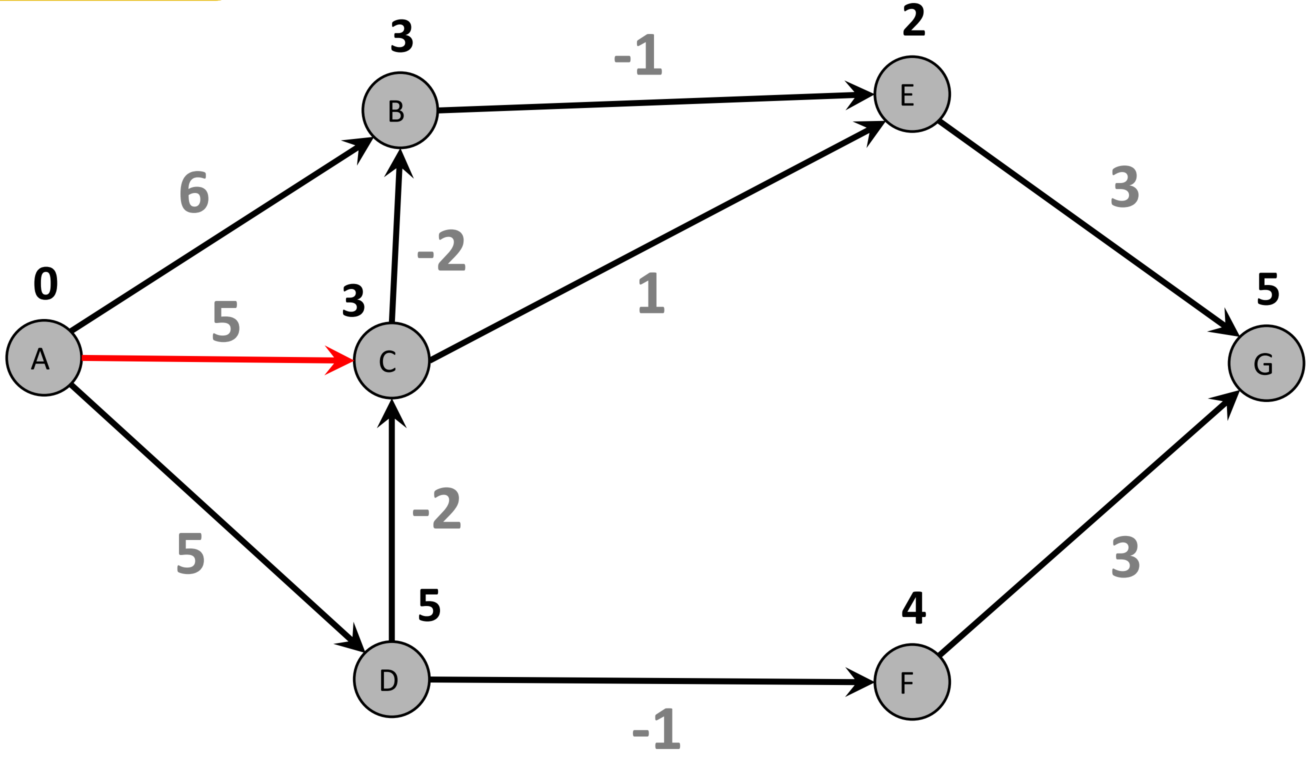


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (A,D) (C,B) (D,C) (D,F) (C,E) (B,E) (F,G) (E,G)

Bellman-ford: Single Source Shortest Path

2nd Iteration

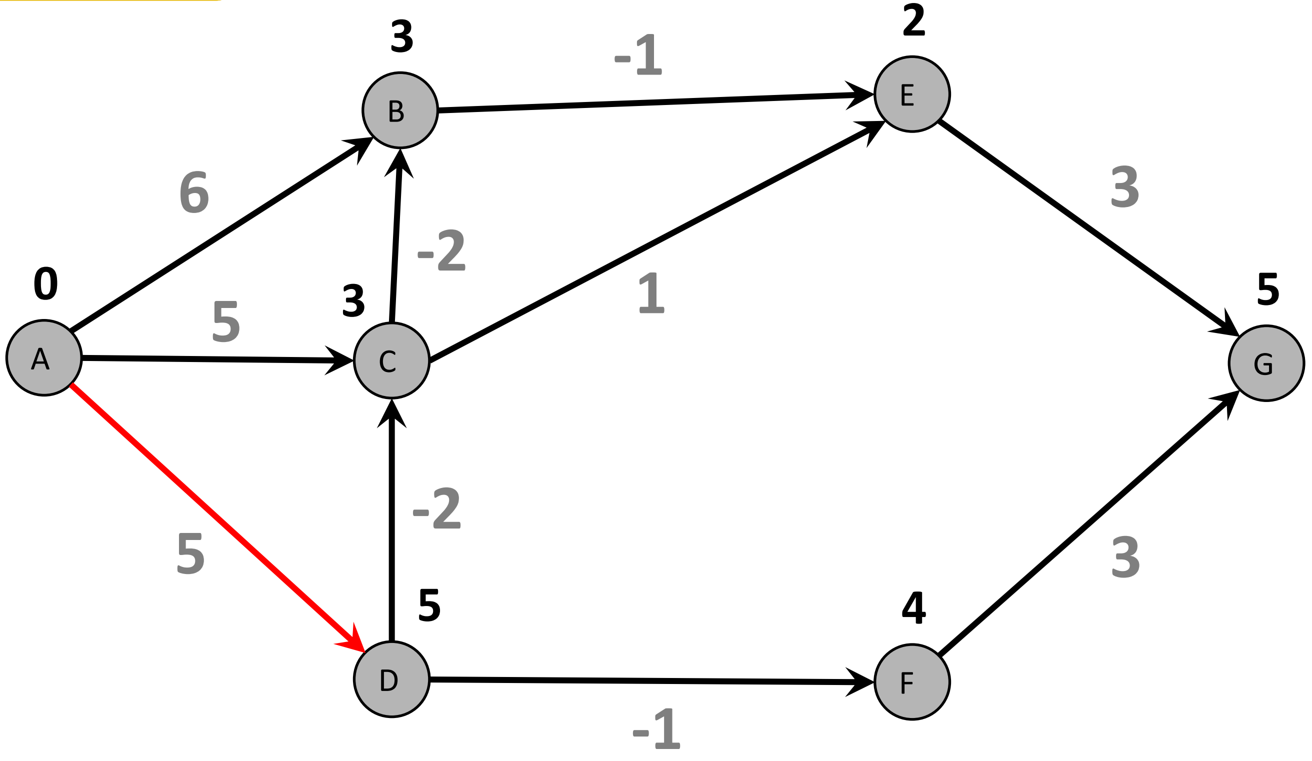


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) **(A,C)** (A,D) (C,B) (D,C) (D,F) (C,E) (B,E) (F,G) (E,G)

Bellman-ford: Single Source Shortest Path

2nd Iteration

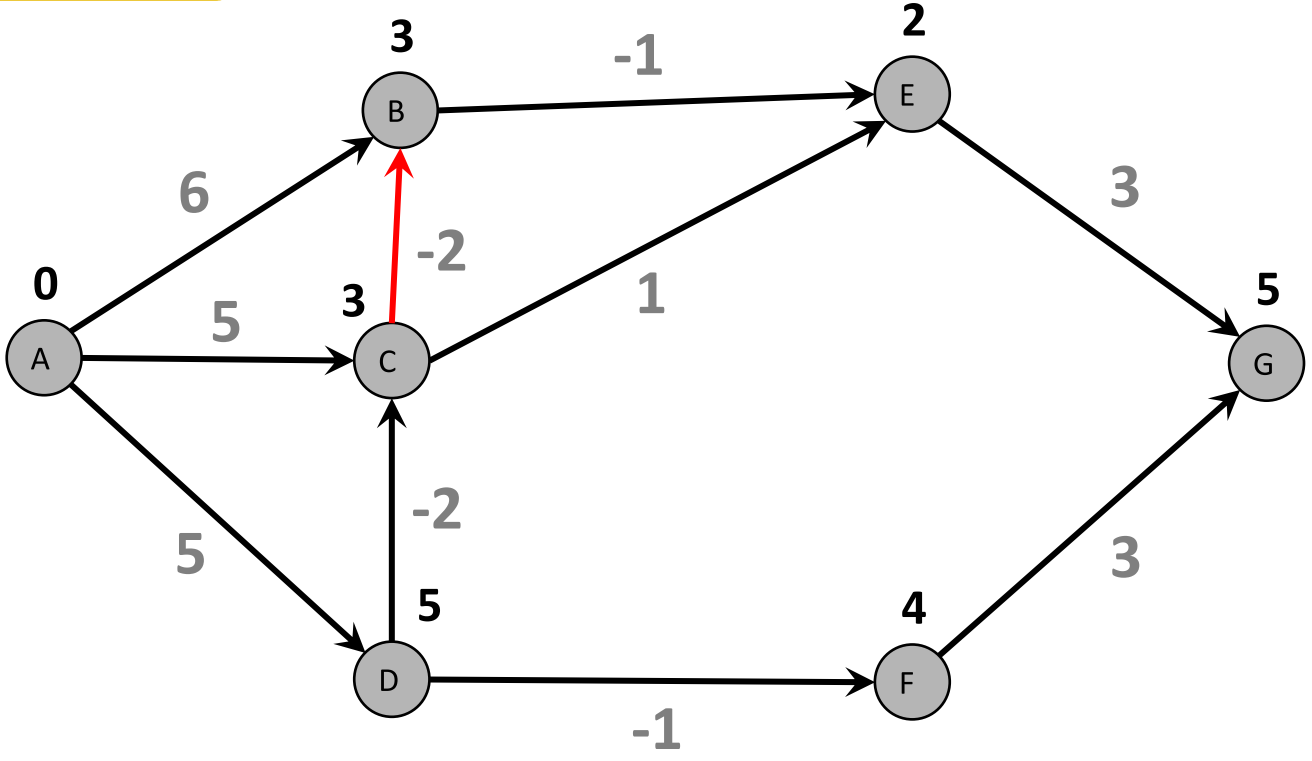


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) **(A,D)** (C,B) (D,C) (D,F) (C,E) (B,E) (F,G) (E,G)

Bellman-ford: Single Source Shortest Path

2nd Iteration

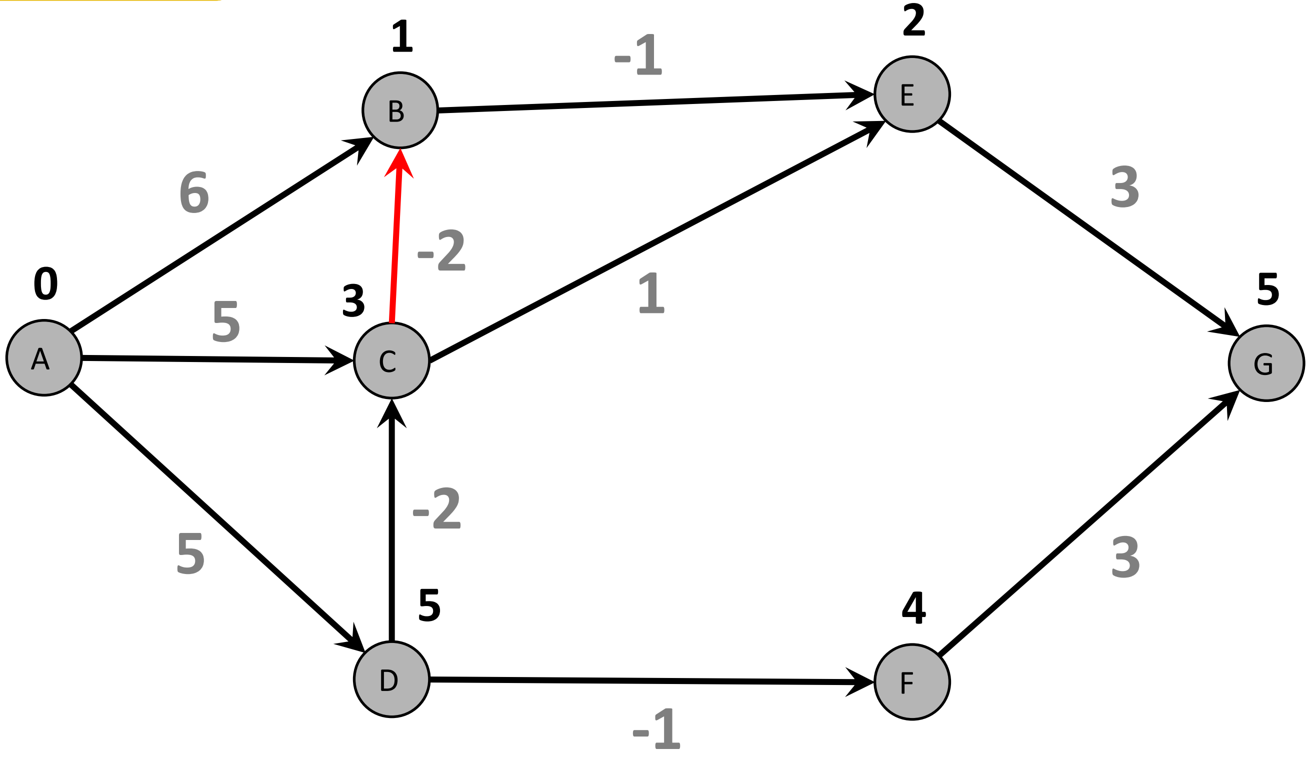


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (A,D) **(C,B)** (D,C) (D,F) (C,E) (B,E) (F,G) (E,G)

Bellman-ford: Single Source Shortest Path

2nd Iteration

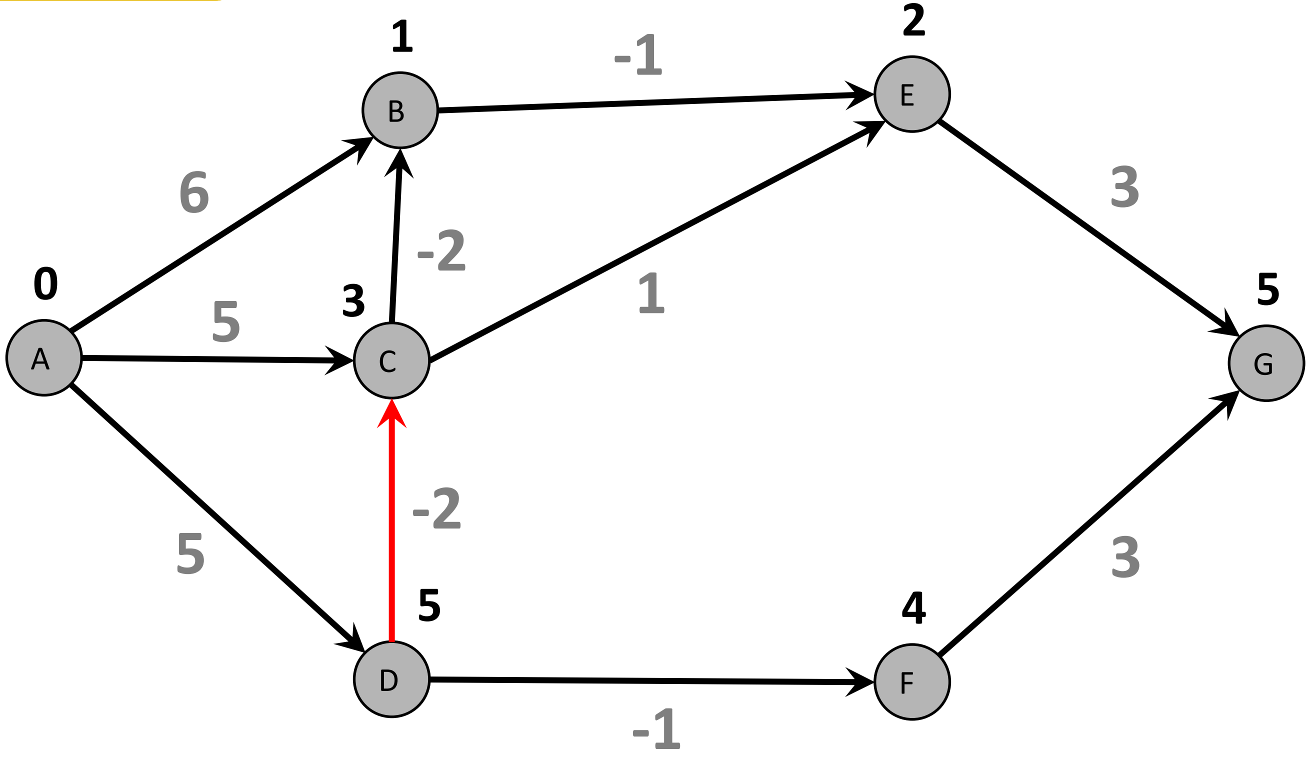


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (A,D) **(C,B)** (D,C) (D,F) (C,E) (B,E) (F,G) (E,G)

Bellman-ford: Single Source Shortest Path

2nd Iteration

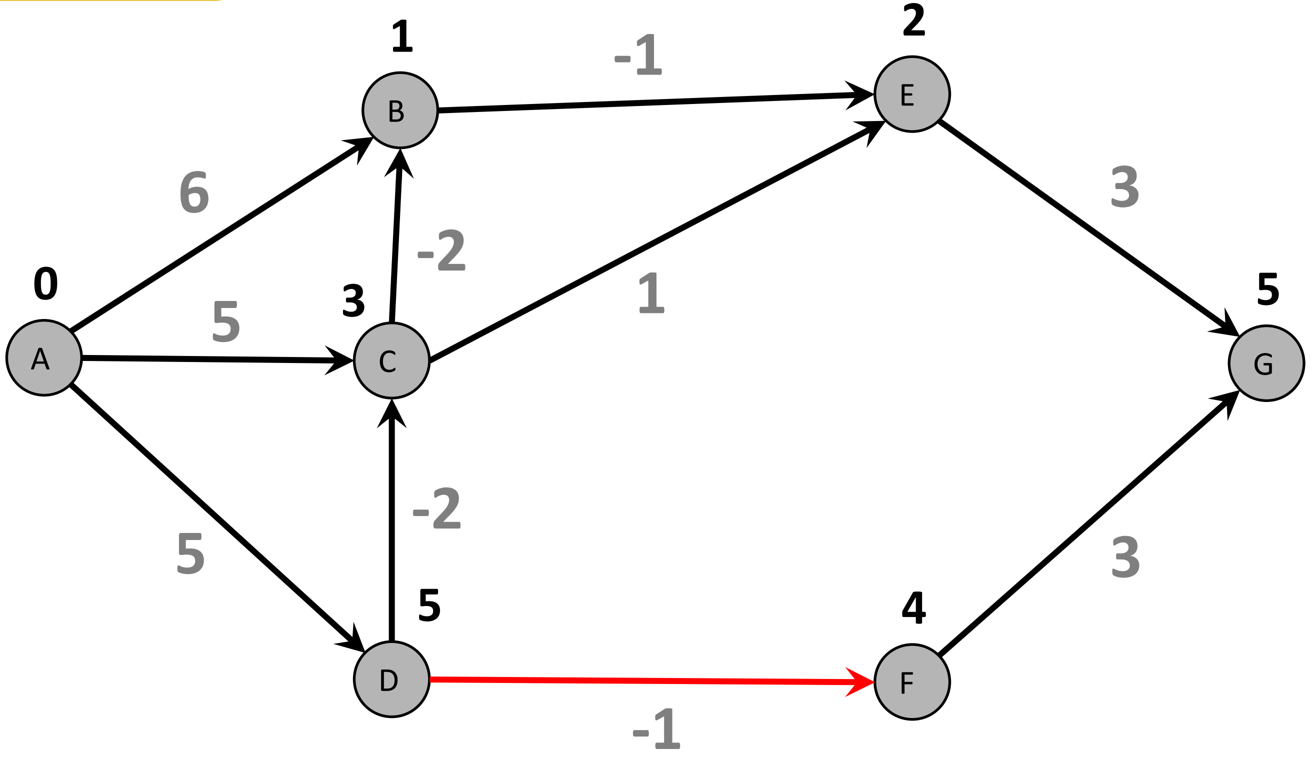


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (A,D) (C,B) **(D,C)** (D,F) (C,E) (B,E) (F,G) (E,G)

Bellman-ford: Single Source Shortest Path

2^{nd} Iteration

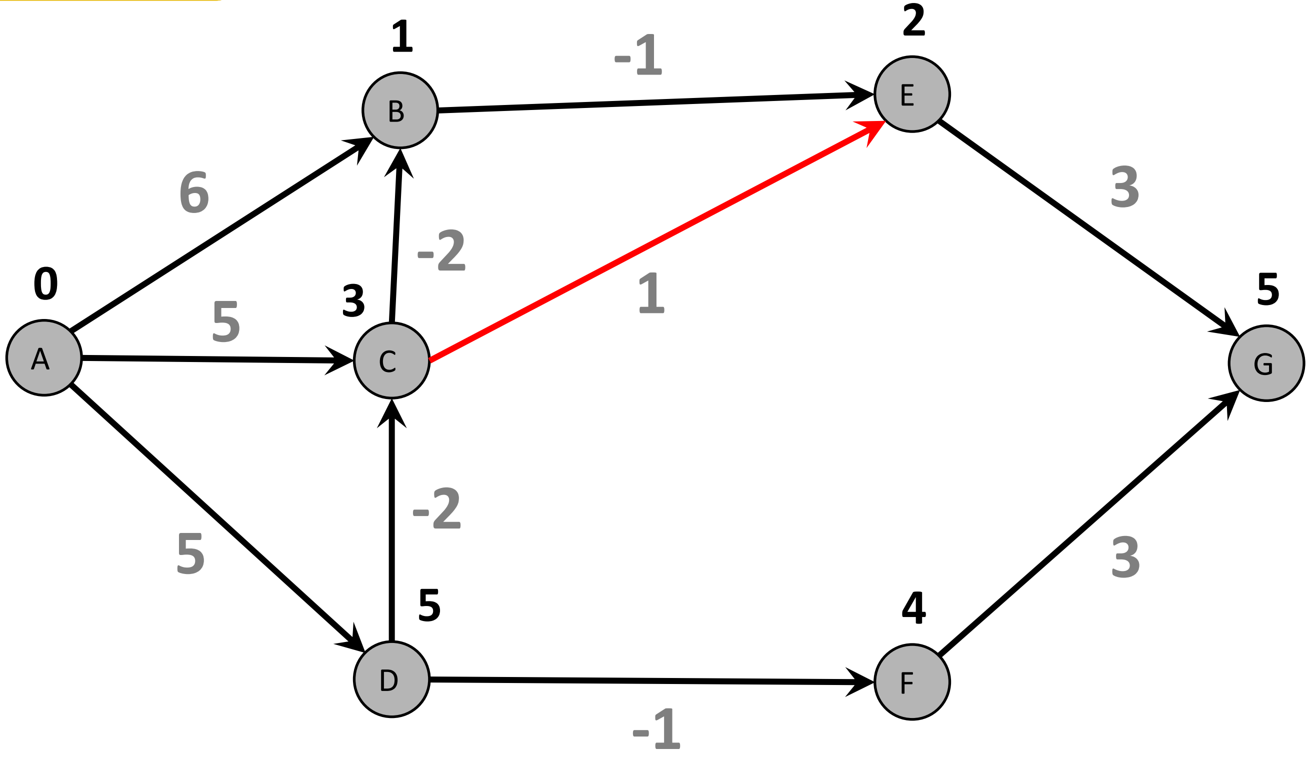


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (A,D) (C,B) (D,C) **(D,F)** (C,E) (B,E) (F,G) (E,G)

Bellman-ford: Single Source Shortest Path

2nd Iteration

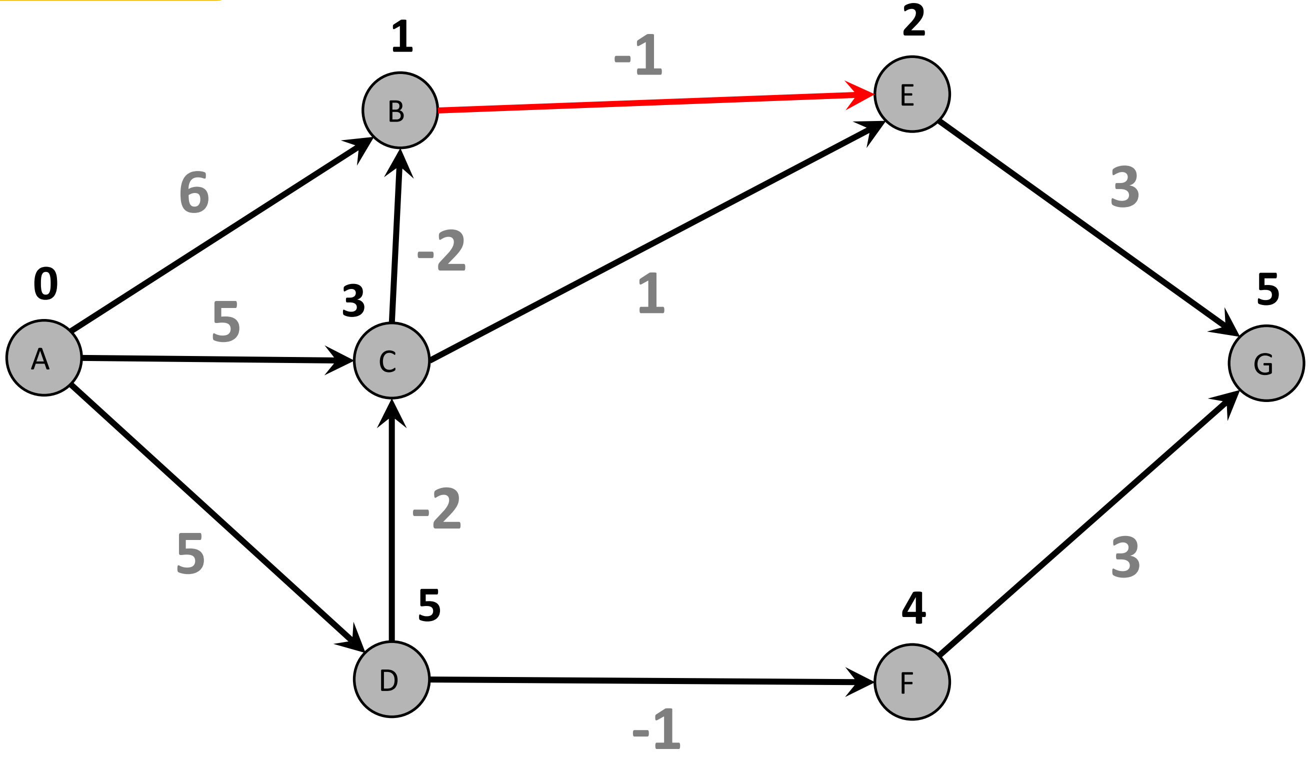


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (A,D) (C,B) (D,C) (D,F) **(C,E)** (B,E) (F,G) (E,G)

Bellman-ford: Single Source Shortest Path

2^{nd} Iteration

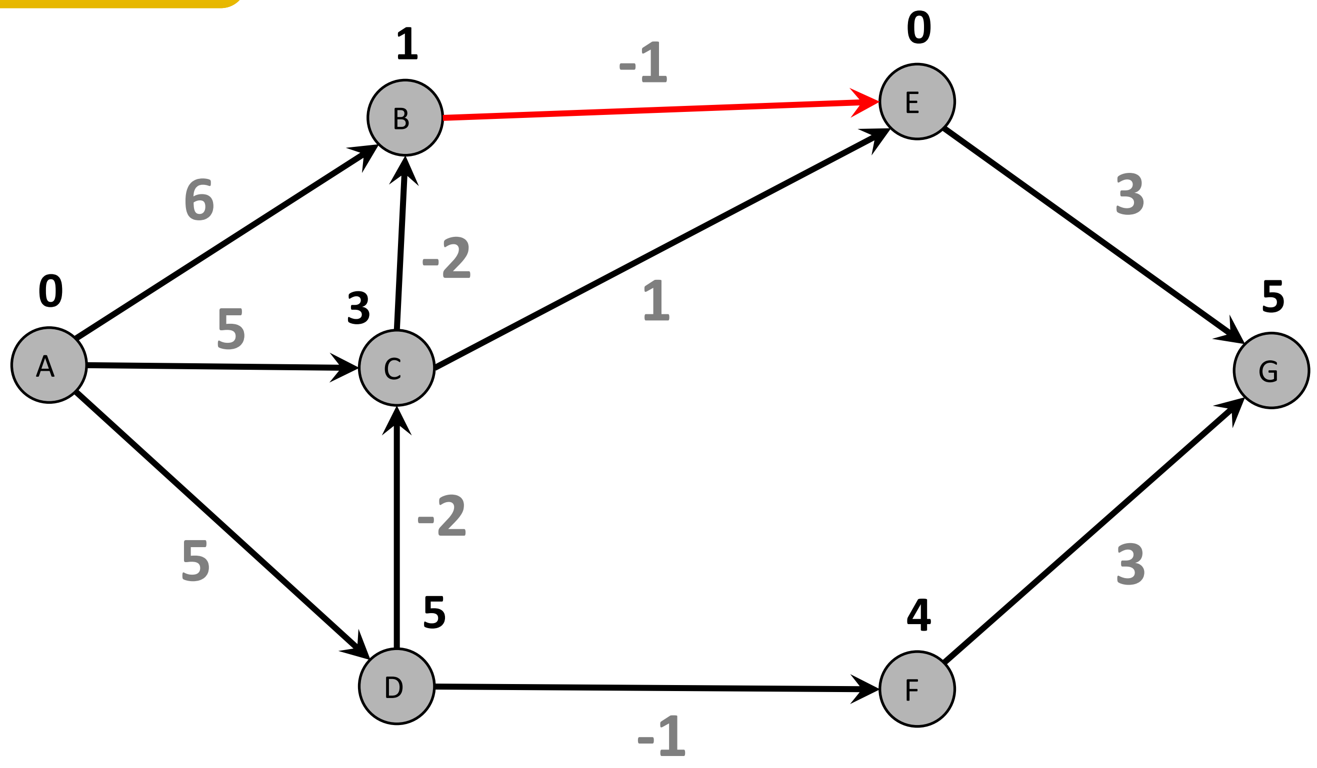


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (A,D) (C,B) (D,C) (D,F) (C,E) **(B,E)** (F,G) (E,G)

Bellman-ford: Single Source Shortest Path

2nd Iteration

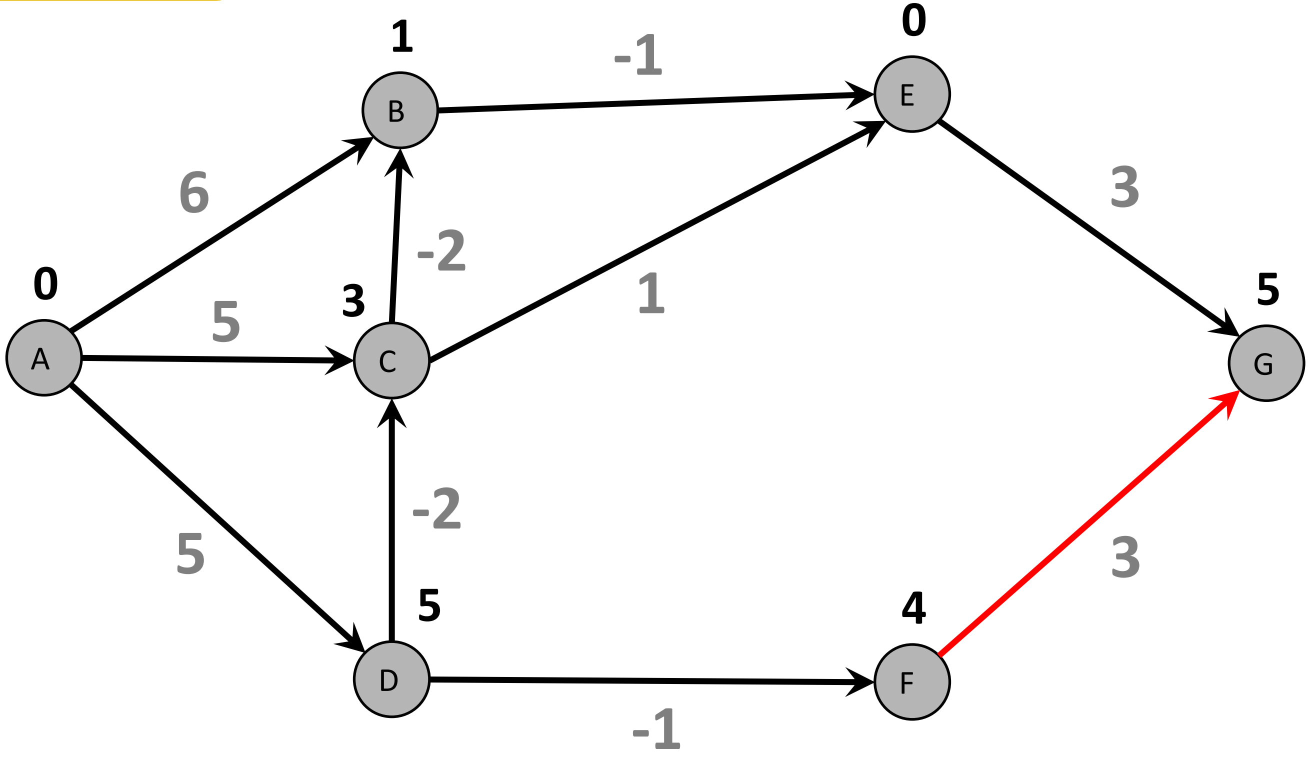


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (A,D) (C,B) (D,C) (D,F) (C,E) **(B,E)** (F,G) (E,G)

Bellman-ford: Single Source Shortest Path

2^{nd} Iteration

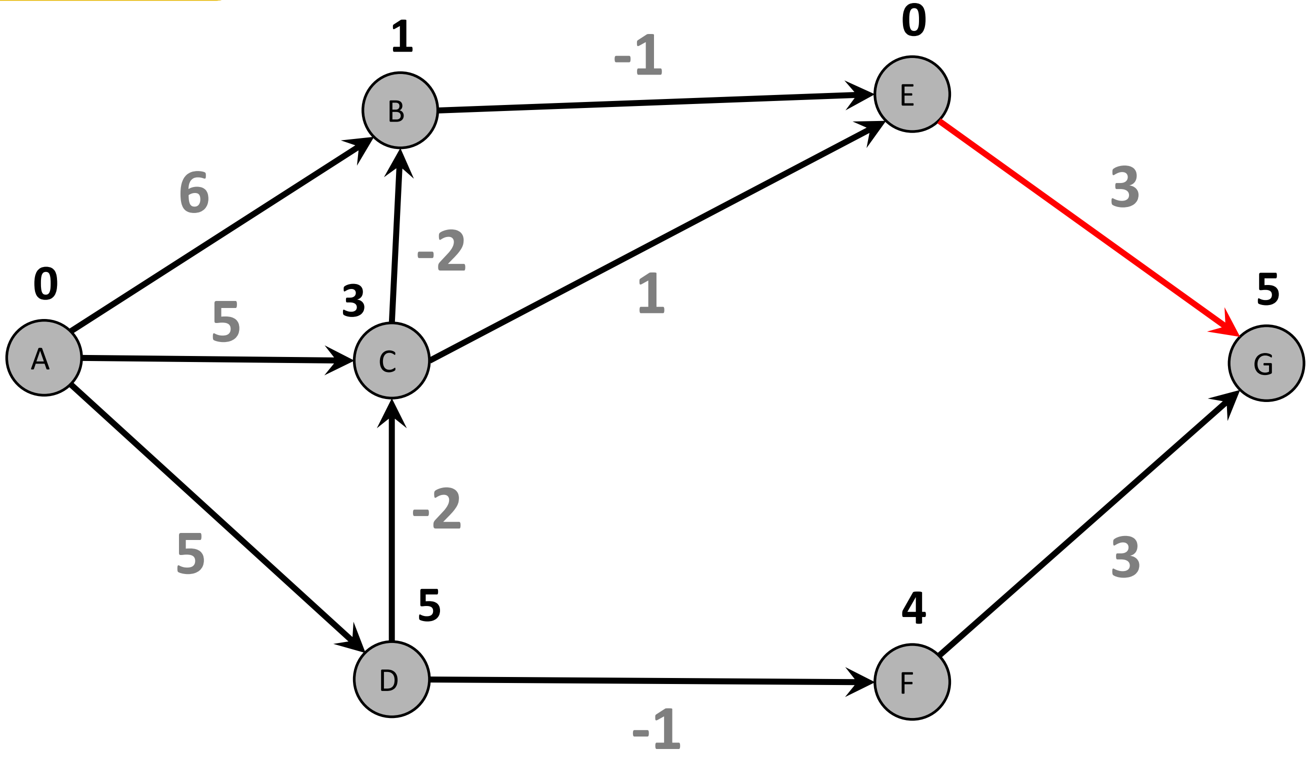


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (A,D) (C,B) (D,C) (D,F) (C,E) (B,E) **(F,G)** (E,G)

Bellman-ford: Single Source Shortest Path

2nd Iteration

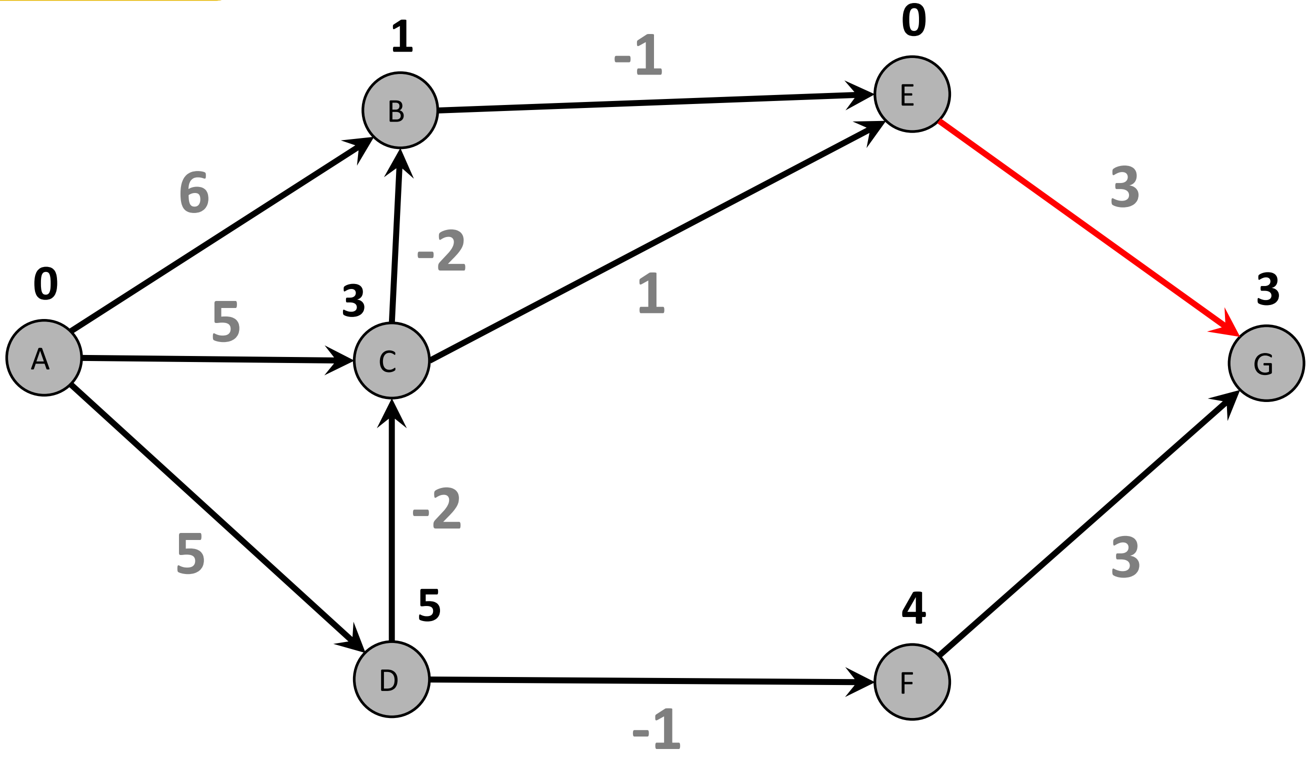


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (A,D) (C,B) (D,C) (D,F) (C,E) (B,E) (F,G) **(E,G)**

Bellman-ford: Single Source Shortest Path

2^{nd} Iteration

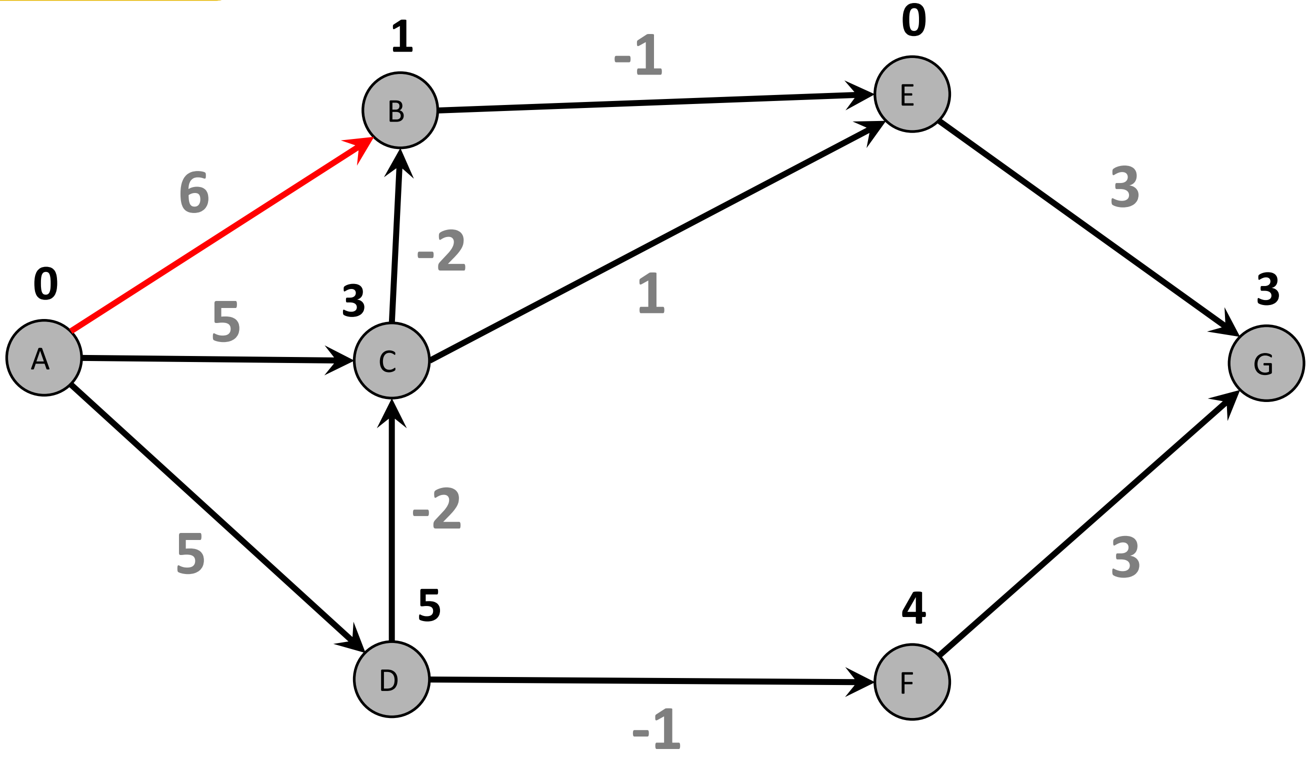


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (A,D) (C,B) (D,C) (D,F) (C,E) (B,E) (F,G) **(E,G)**

Bellman-ford: Single Source Shortest Path

3rd Iteration

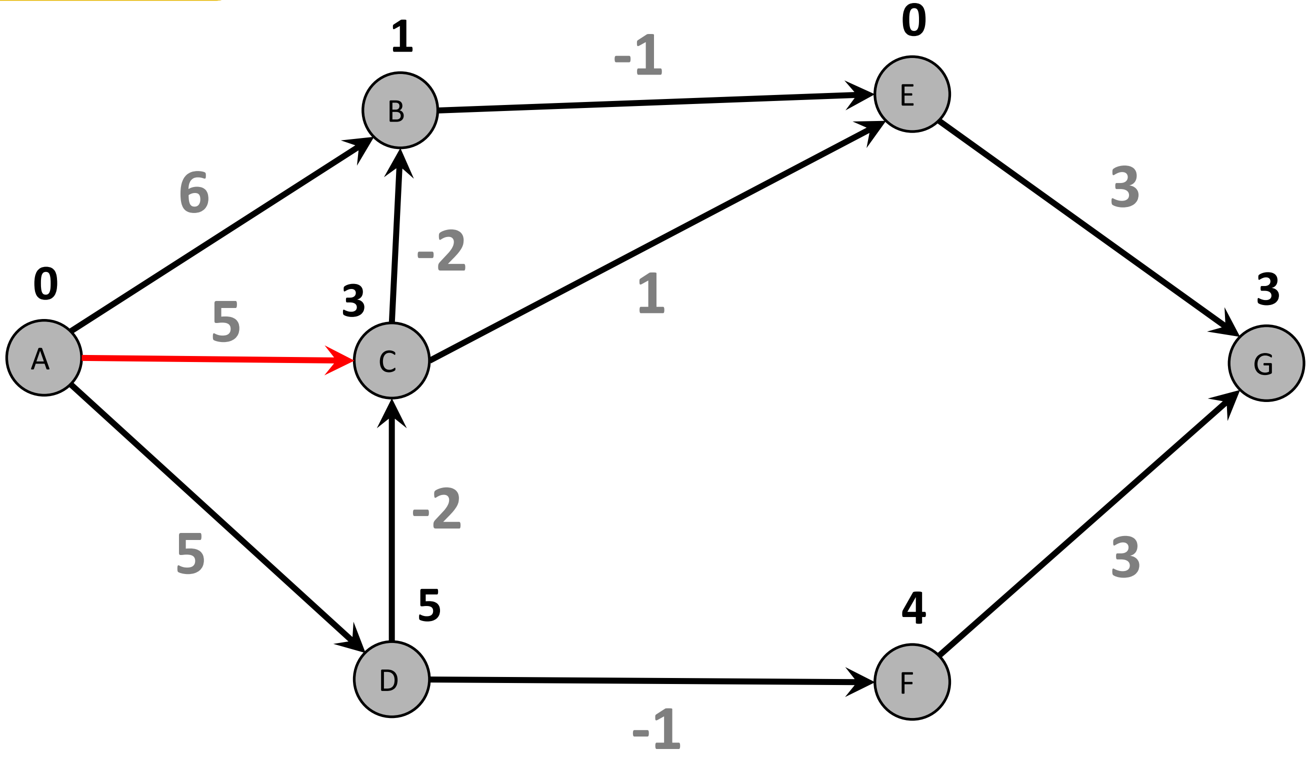


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (A,D) (C,B) (D,C) (D,F) (C,E) (B,E) (F,G) (E,G)

Bellman-ford: Single Source Shortest Path

3rd Iteration

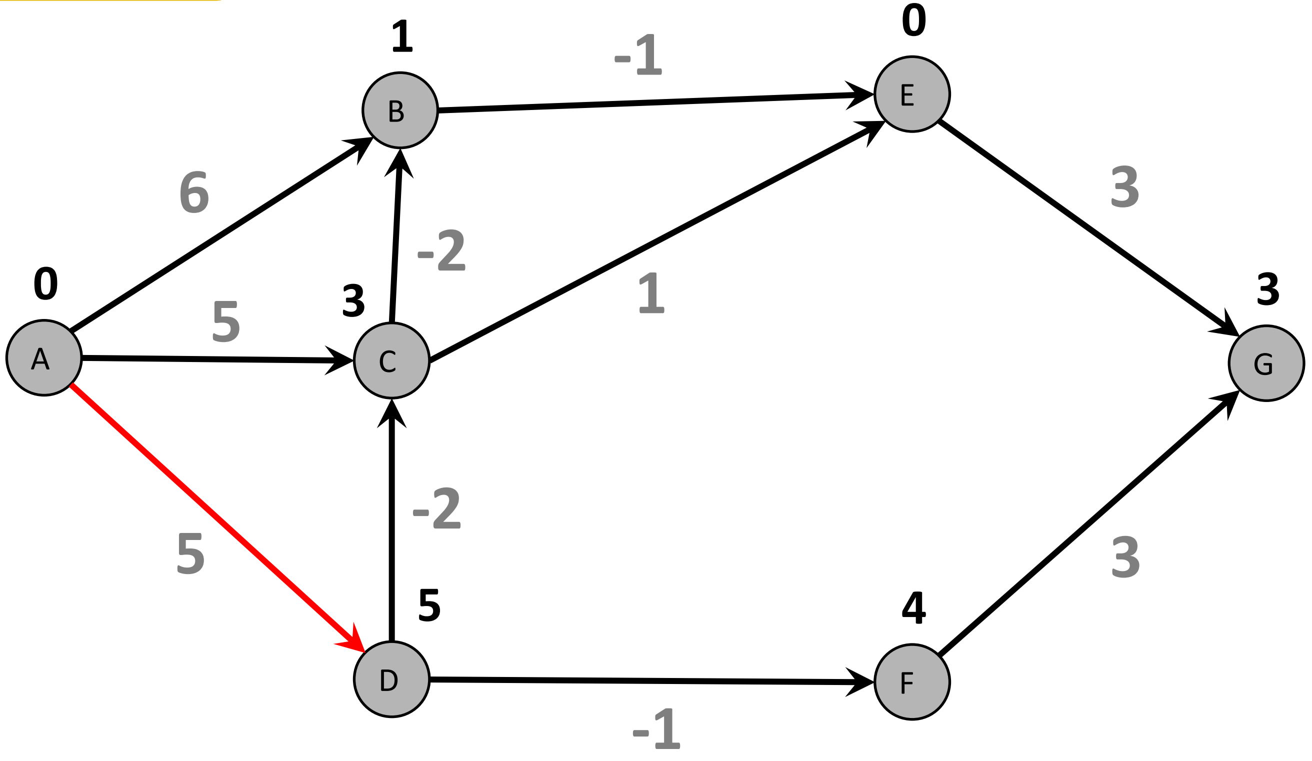


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) **(A,C)** (A,D) (C,B) (D,C) (D,F) (C,E) (B,E) (F,G) (E,G)

Bellman-ford: Single Source Shortest Path

3rd Iteration

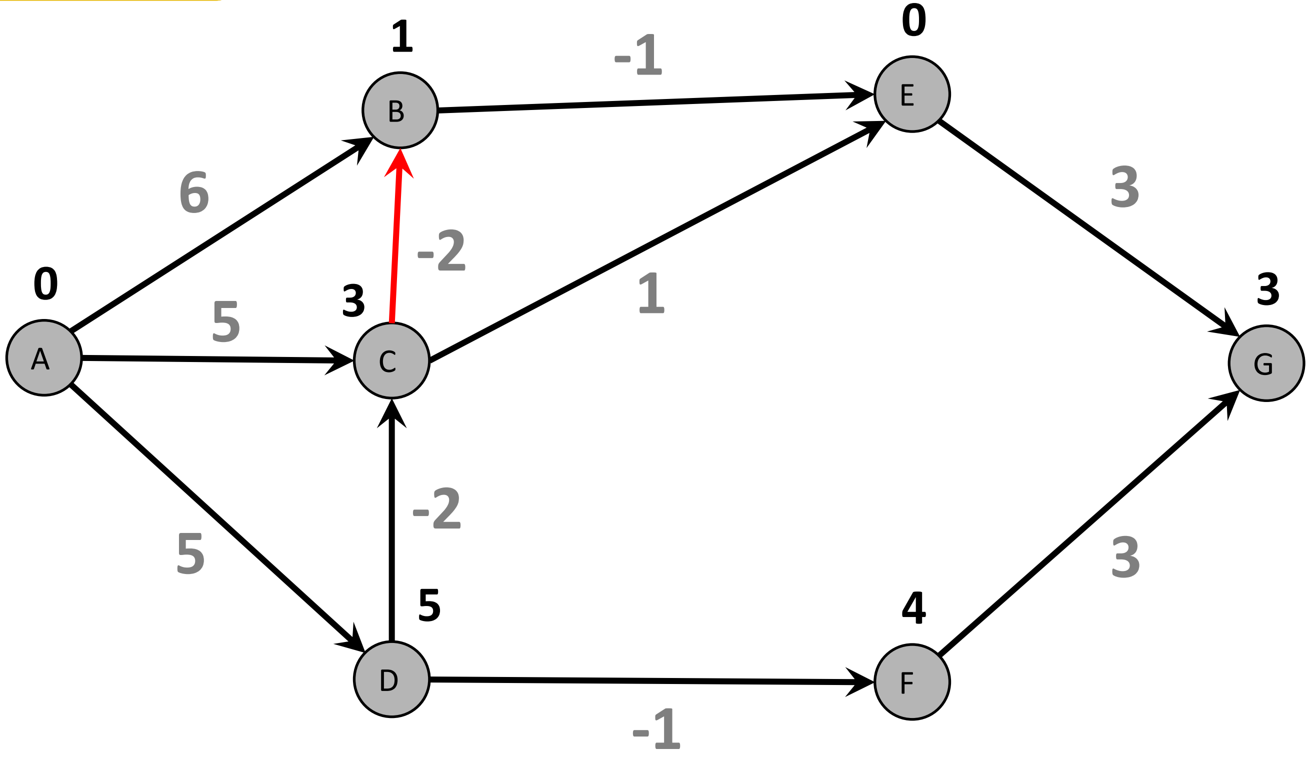


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) **(A,D)** (C,B) (D,C) (D,F) (C,E) (B,E) (F,G) (E,G)

Bellman-ford: Single Source Shortest Path

3^{rd} Iteration

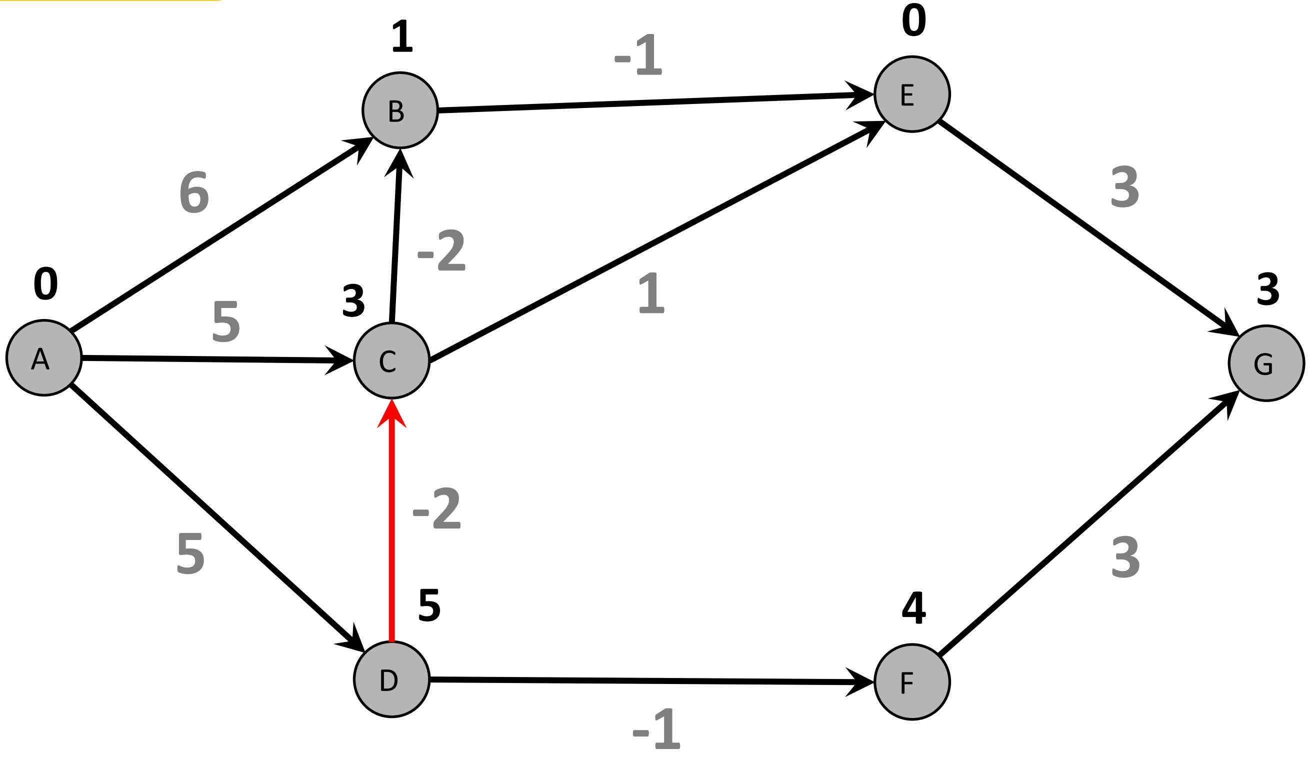


If $(d[u] + e(u,v) < d[v])$
$d[v] = d[u] + e(u,v)$

(A,B) (A,C) (A,D) **(C,B)** (D,C) (D,F) (C,E) (B,E) (F,G) (E,G)

Bellman-ford: Single Source Shortest Path

3rd Iteration

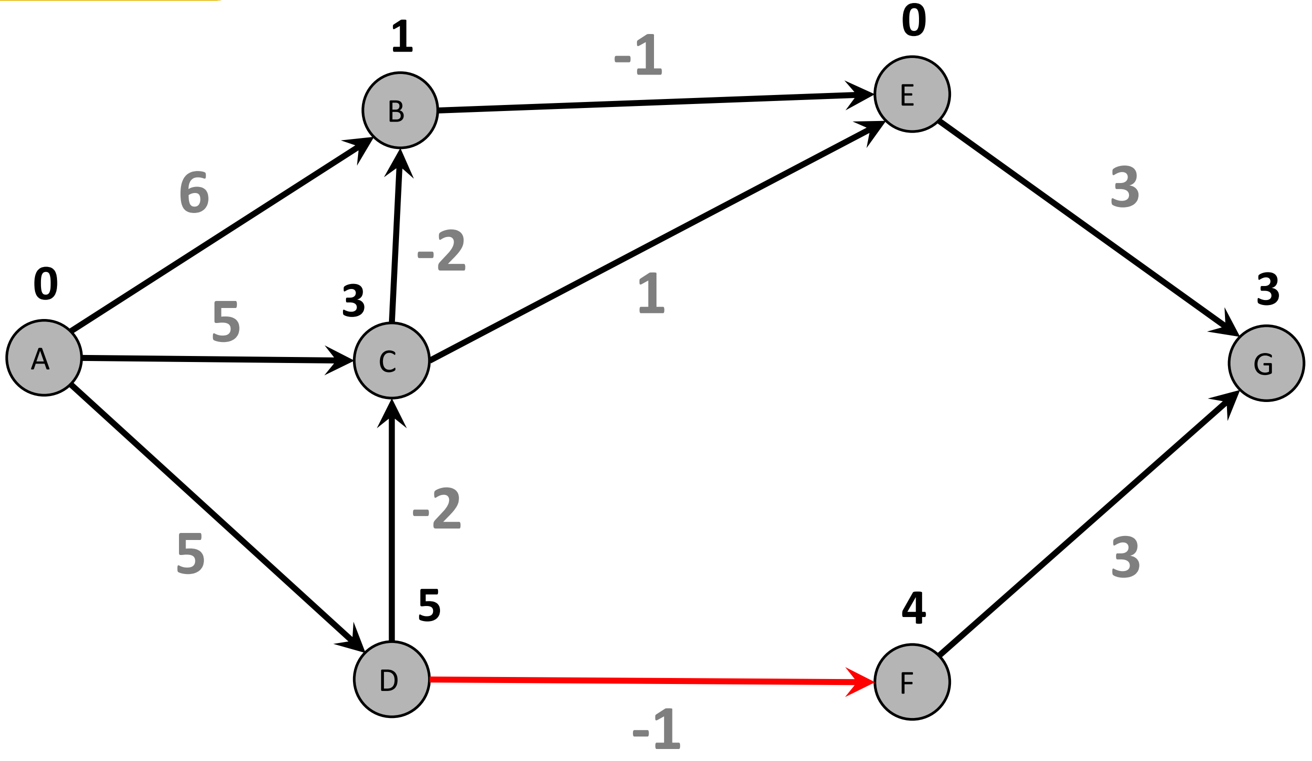


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (A,D) (C,B) **(D,C)** (D,F) (C,E) (B,E) (F,G) (E,G)

Bellman-ford: Single Source Shortest Path

3rd Iteration

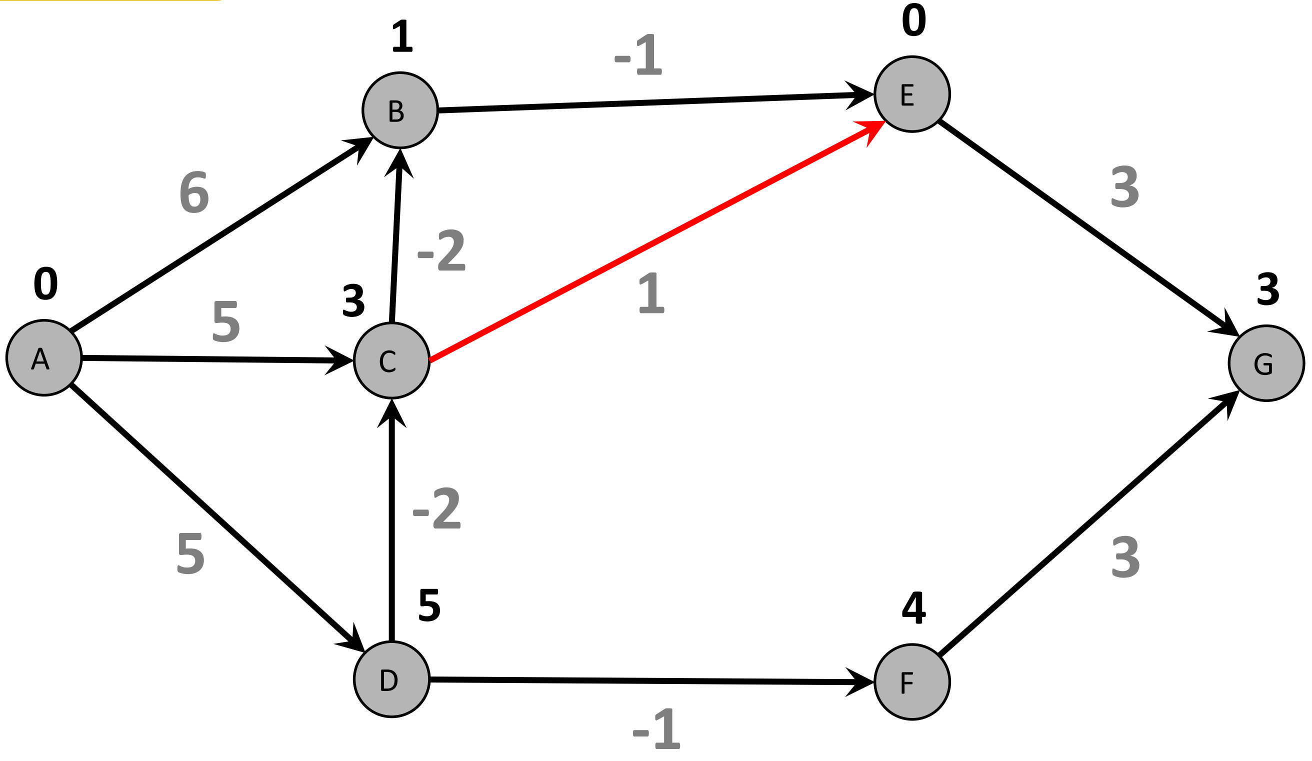


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (A,D) (C,B) (D,C) **(D,F)** (C,E) (B,E) (F,G) (E,G)

Bellman-ford: Single Source Shortest Path

3rd Iteration

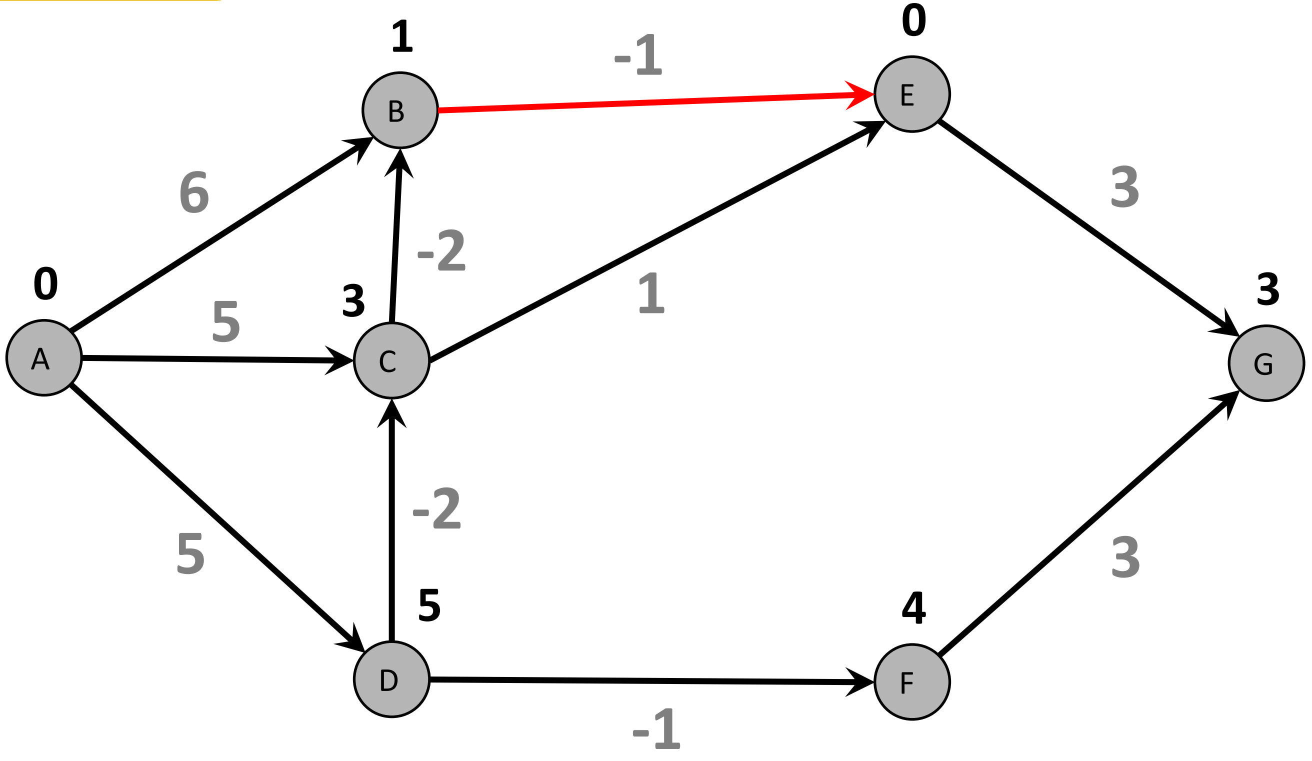


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (A,D) (C,B) (D,C) (D,F) **(C,E)** (B,E) (F,G) (E,G)

Bellman-ford: Single Source Shortest Path

3rd Iteration

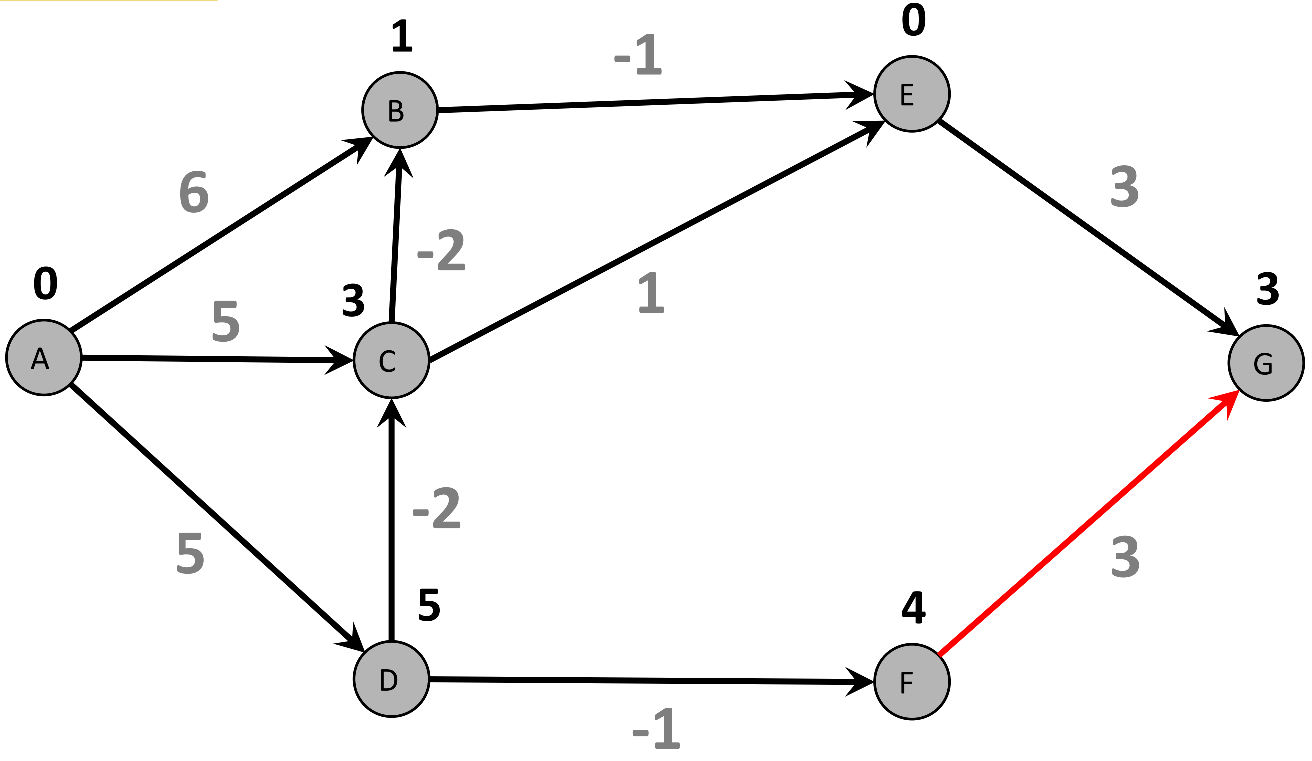


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (A,D) (C,B) (D,C) (D,F) (C,E) **(B,E)** (F,G) (E,G)

Bellman-ford: Single Source Shortest Path

3rd Iteration

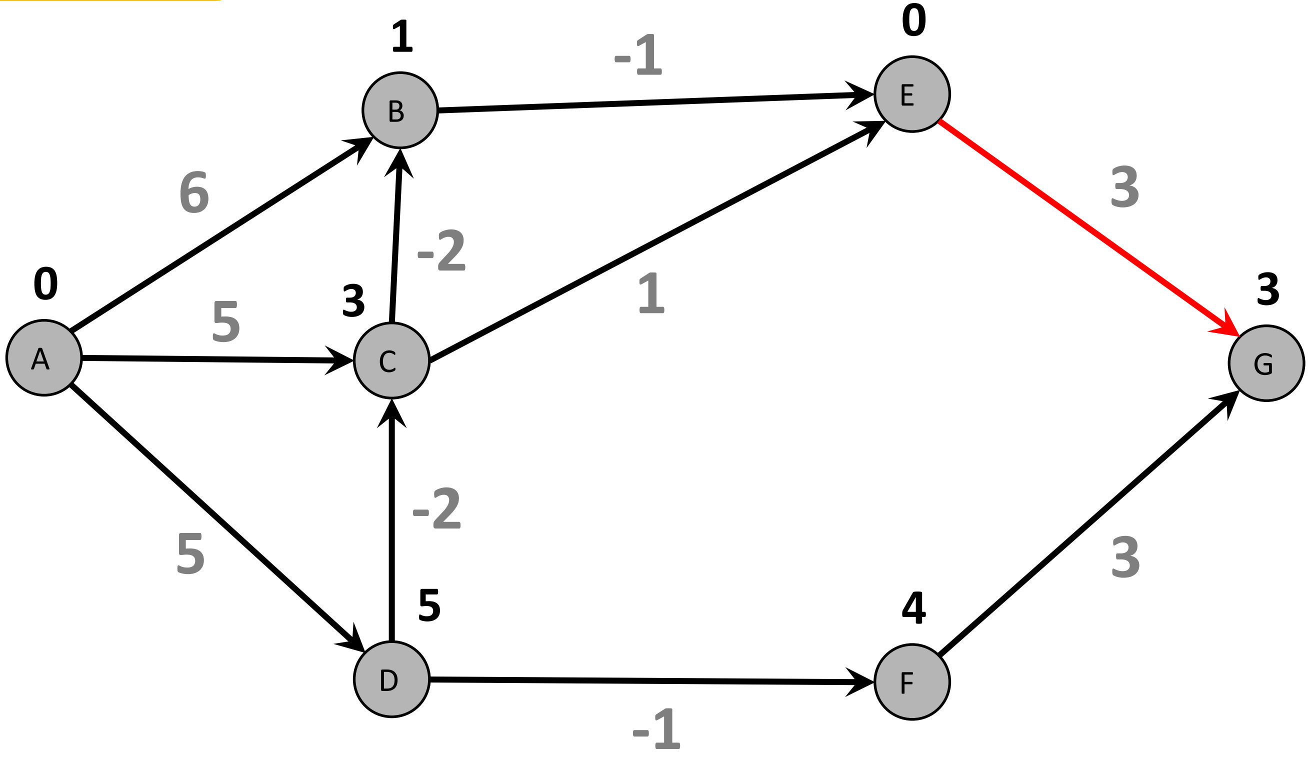


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (A,D) (C,B) (D,C) (D,F) (C,E) (B,E) **(F,G)** (E,G)

Bellman-ford: Single Source Shortest Path

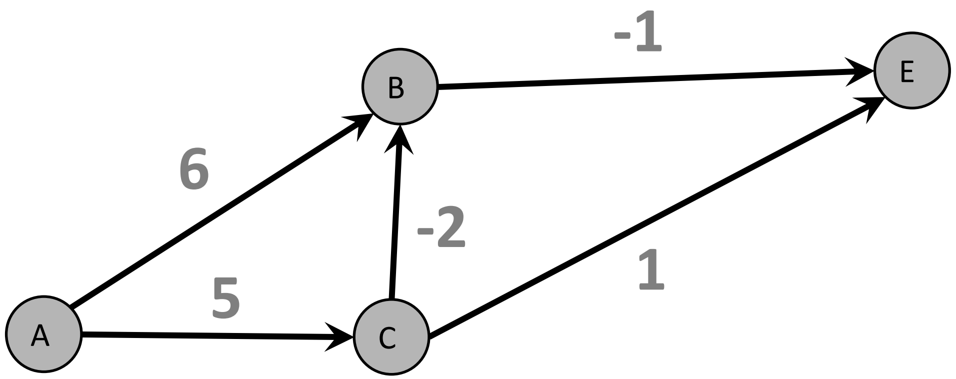
3rd Iteration



If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

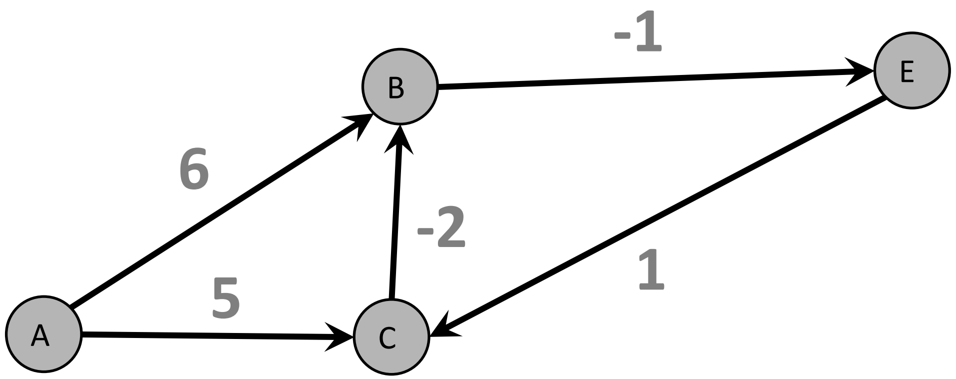
(A,B) (A,C) (A,D) (C,B) (D,C) (D,F) (C,E) (B,E) (F,G) **(E,G)**

Bellman-ford: Single Source Shortest Path



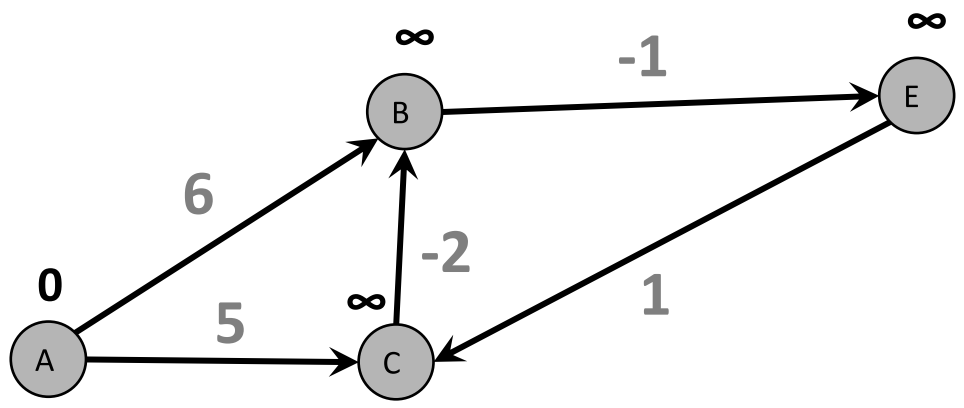
If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

Bellman-ford: Single Source Shortest Path



If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

Bellman-ford: Single Source Shortest Path

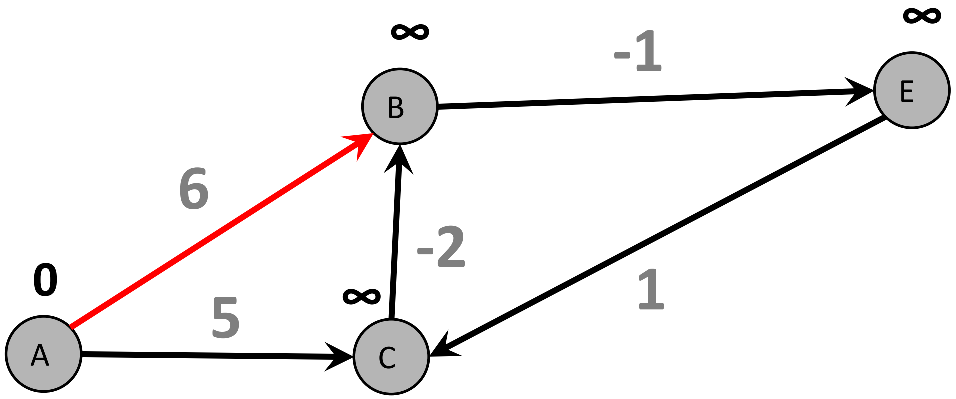


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (C,B) (B,E) (E,C)

Bellman-ford: Single Source Shortest Path

1st Iteration

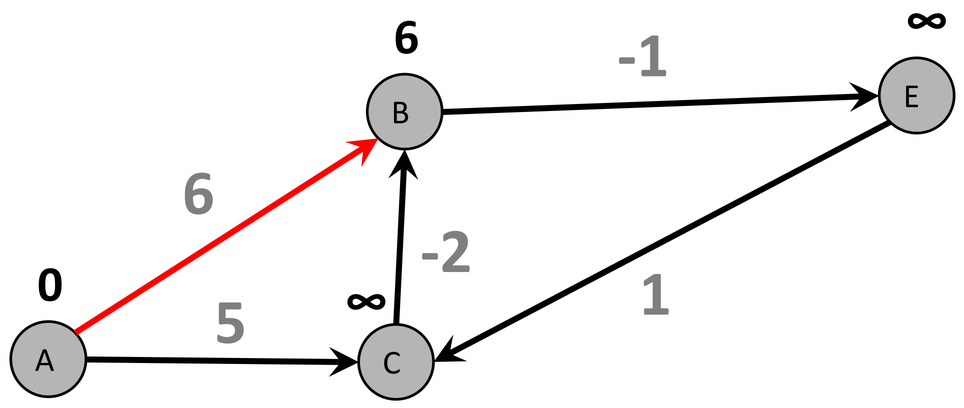


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (C,B) (B,E) (E,C)

Bellman-ford: Single Source Shortest Path

1st Iteration

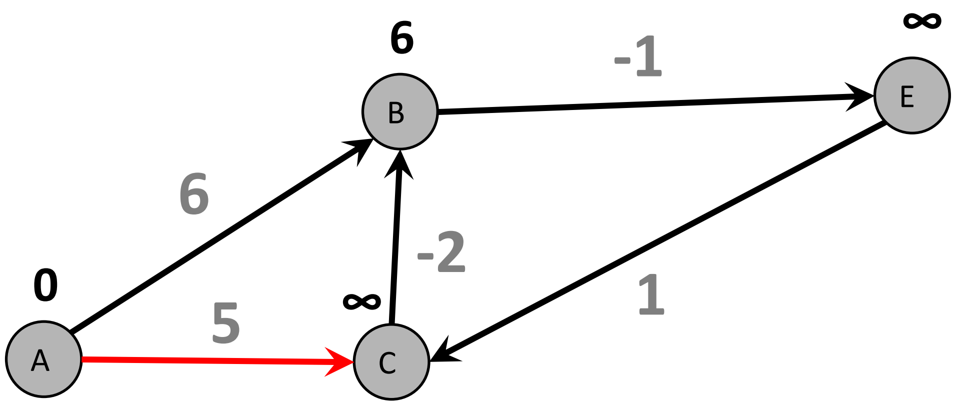


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (C,B) (B,E) (E,C)

Bellman-ford: Single Source Shortest Path

1st Iteration

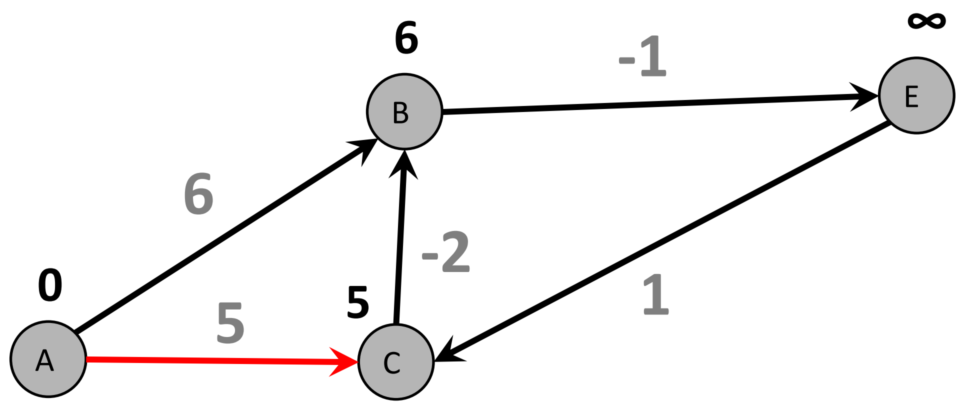


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) **(A,C)** (C,B) (B,E) (E,C)

Bellman-ford: Single Source Shortest Path

1st Iteration

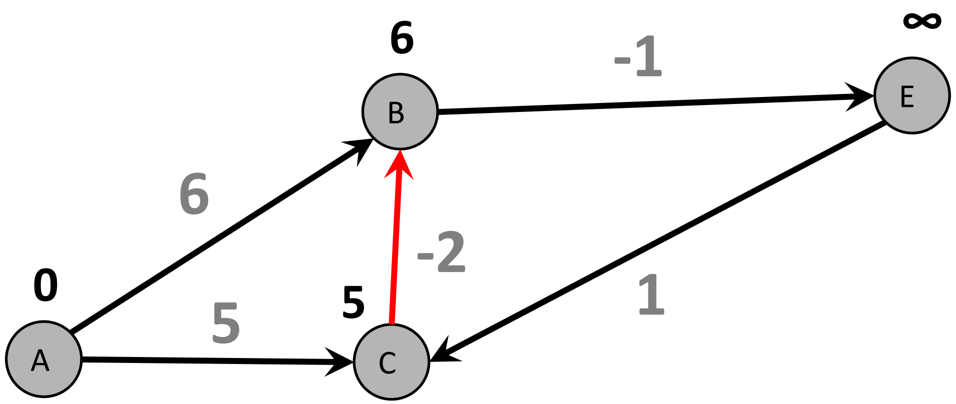


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) **(A,C)** (C,B) (B,E) (E,C)

Bellman-ford: Single Source Shortest Path

1st Iteration

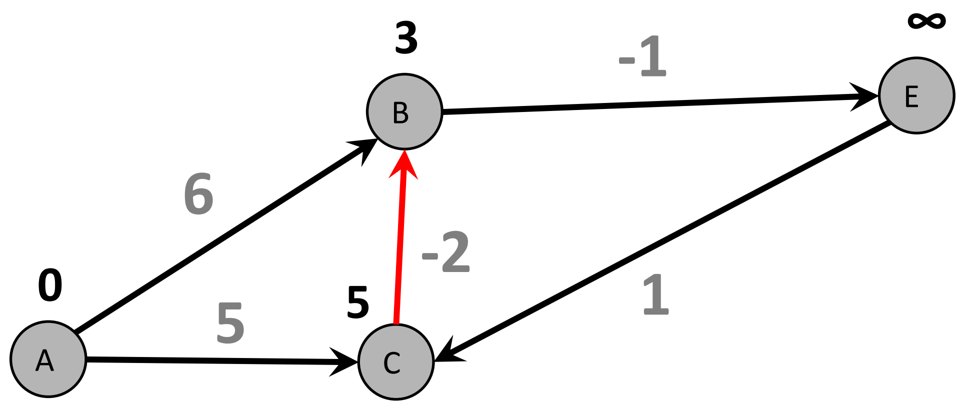


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) **(C,B)** (B,E) (E,C)

Bellman-ford: Single Source Shortest Path

1st Iteration

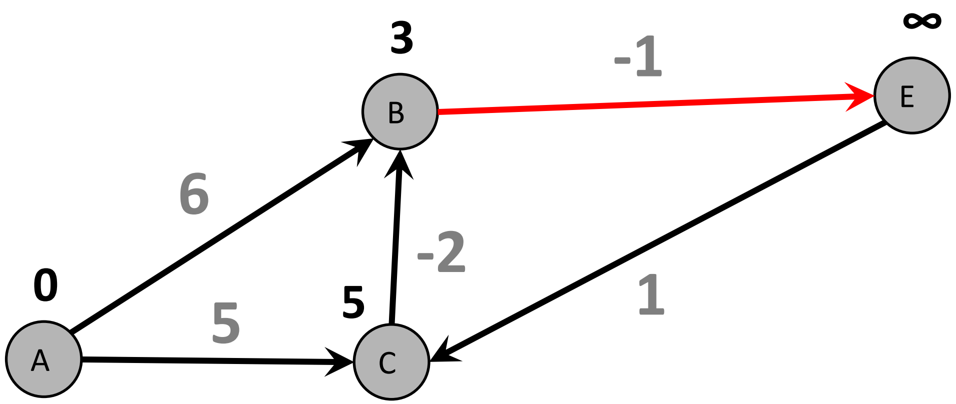


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) **(C,B)** (B,E) (E,C)

Bellman-ford: Single Source Shortest Path

1st Iteration

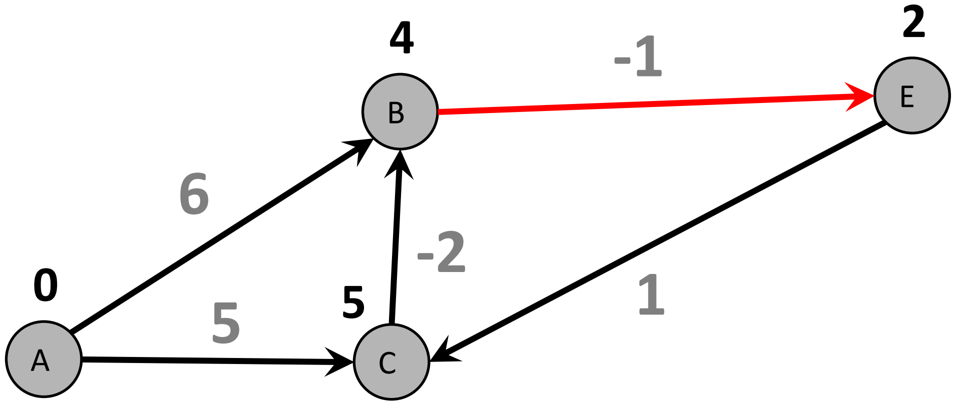


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (C,B) **(B,E)** (E,C)

Bellman-ford: Single Source Shortest Path

1st Iteration

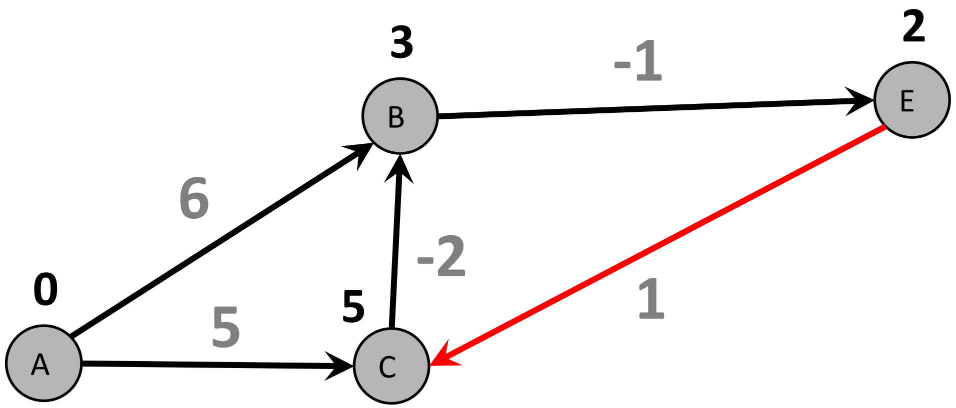


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (C,B) **(B,E)** (E,C)

Bellman-ford: Single Source Shortest Path

1st Iteration

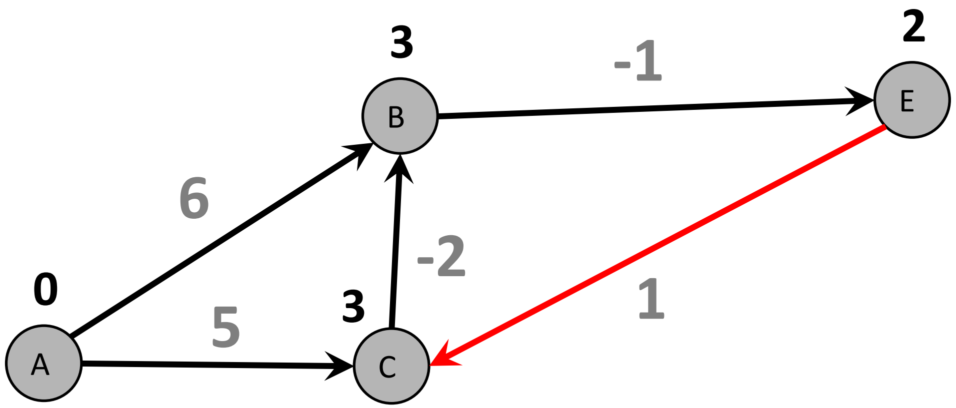


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (C,B) (B,E) **(E,C)**

Bellman-ford: Single Source Shortest Path

1st Iteration

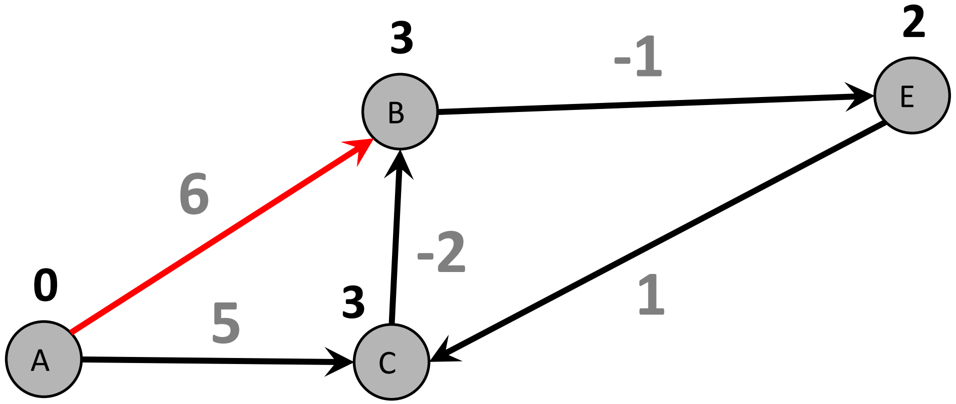


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (C,B) (B,E) **(E,C)**

Bellman-ford: Single Source Shortest Path

2^{nd} Iteration

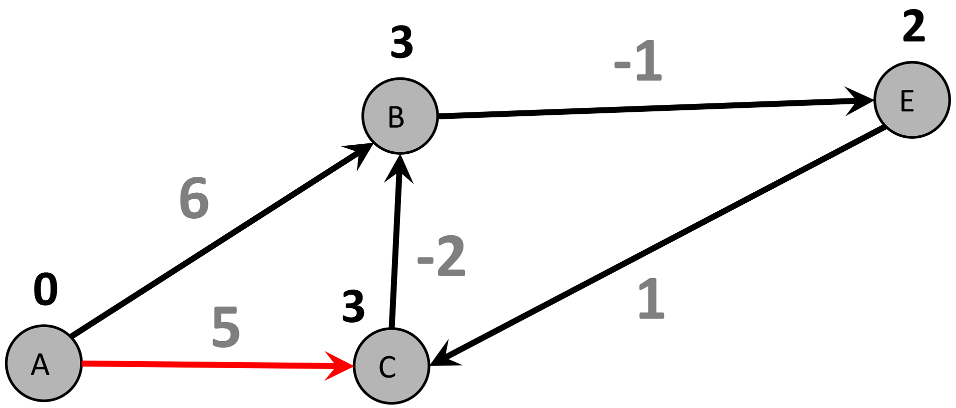


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (C,B) (B,E) (E,C)

Bellman-ford: Single Source Shortest Path

2nd Iteration

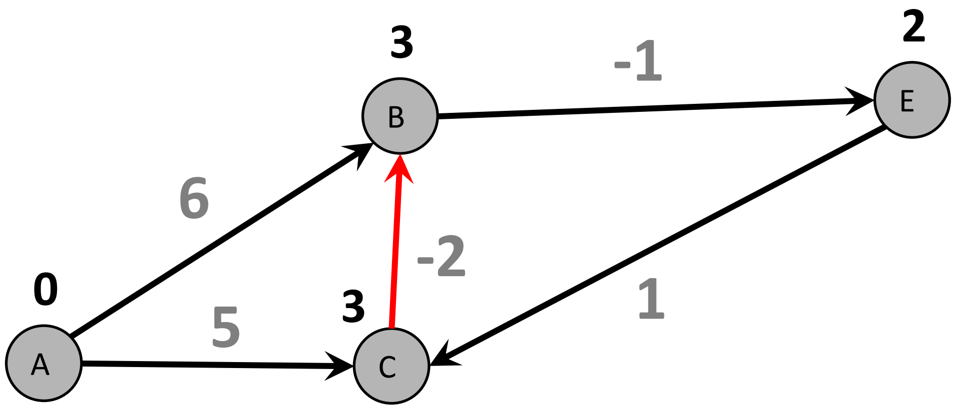


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) **(A,C)** (C,B) (B,E) (E,C)

Bellman-ford: Single Source Shortest Path

2nd Iteration

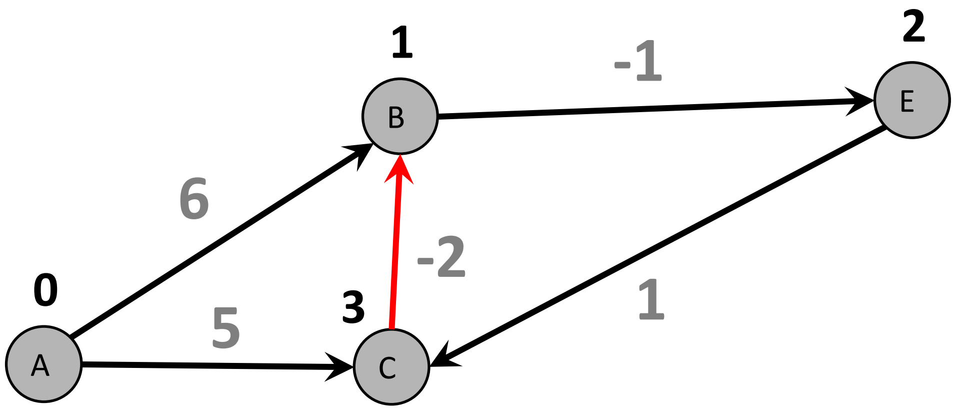


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) **(C,B)** (B,E) (E,C)

Bellman-ford: Single Source Shortest Path

2nd Iteration

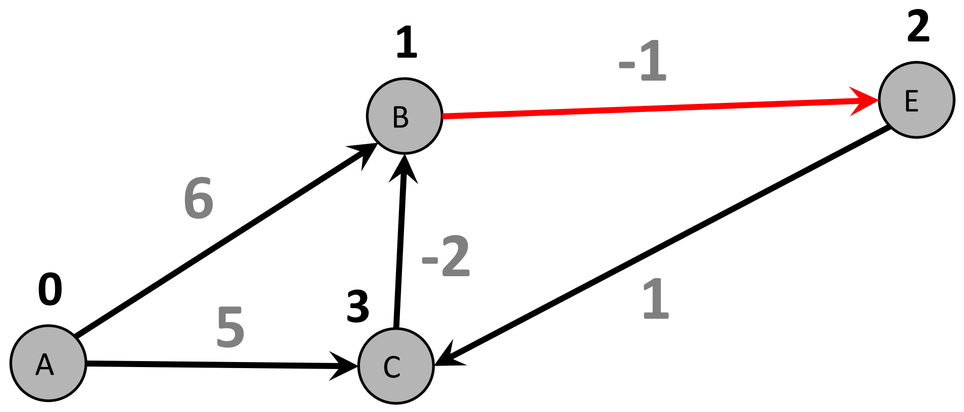


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) **(C,B)** (B,E) (E,C)

Bellman-ford: Single Source Shortest Path

2nd Iteration

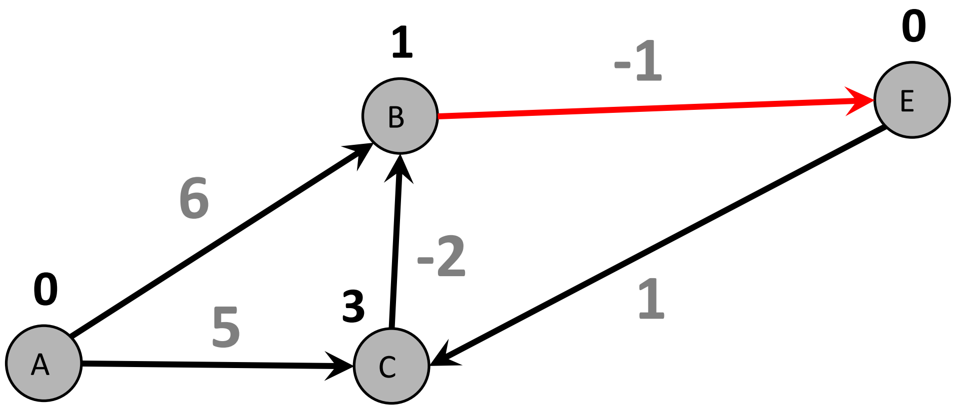


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (C,B) **(B,E)** (E,C)

Bellman-ford: Single Source Shortest Path

2nd Iteration

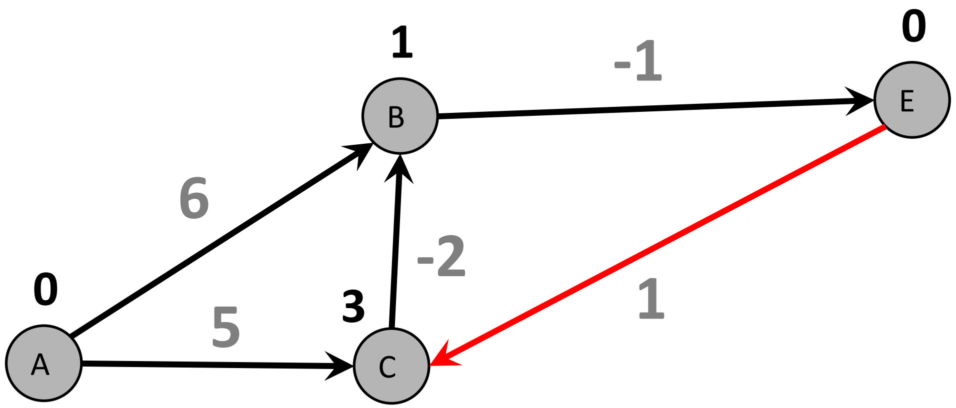


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (C,B) **(B,E)** (E,C)

Bellman-ford: Single Source Shortest Path

2^{nd} Iteration

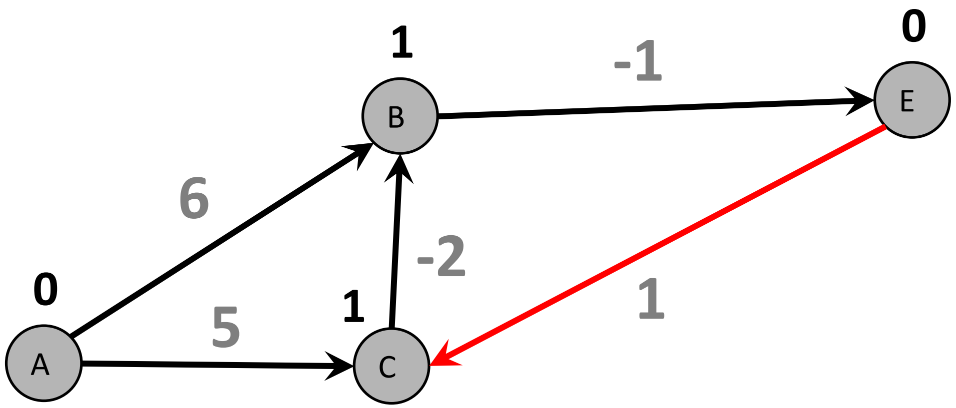


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (C,B) (B,E) **(E,C)**

Bellman-ford: Single Source Shortest Path

2nd Iteration

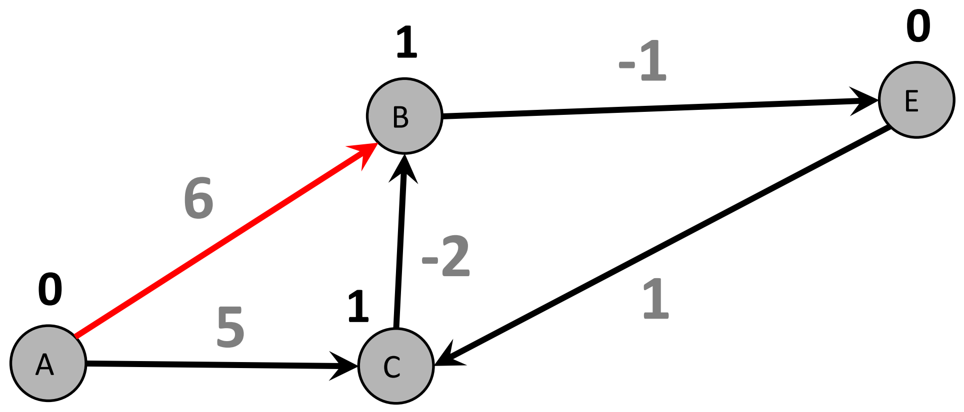


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (C,B) (B,E) **(E,C)**

Bellman-ford: Single Source Shortest Path

3rd Iteration

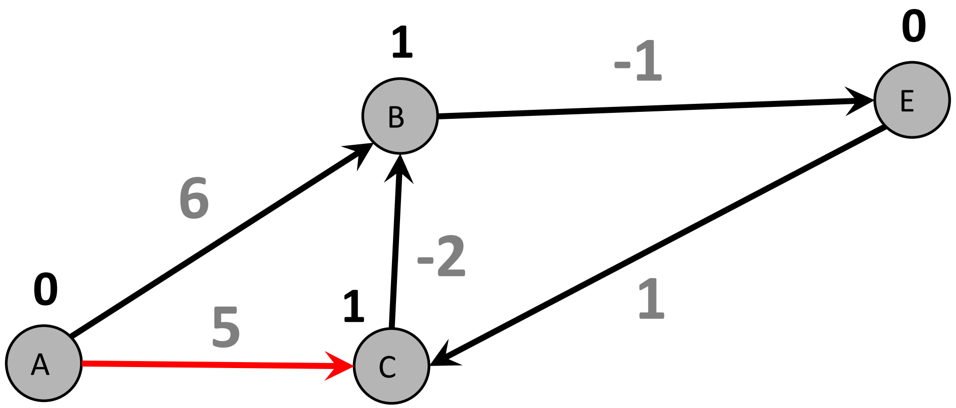


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (C,B) (B,E) (E,C)

Bellman-ford: Single Source Shortest Path

3rd Iteration

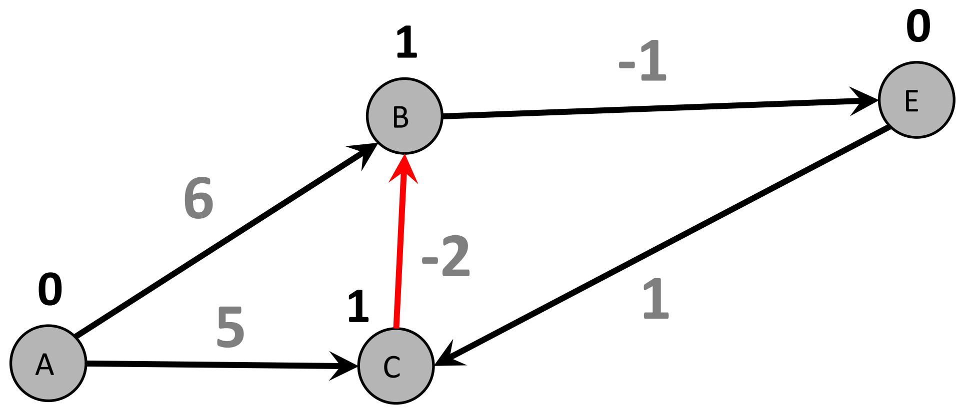


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) **(A,C)** (C,B) (B,E) (E,C)

Bellman-ford: Single Source Shortest Path

3rd Iteration

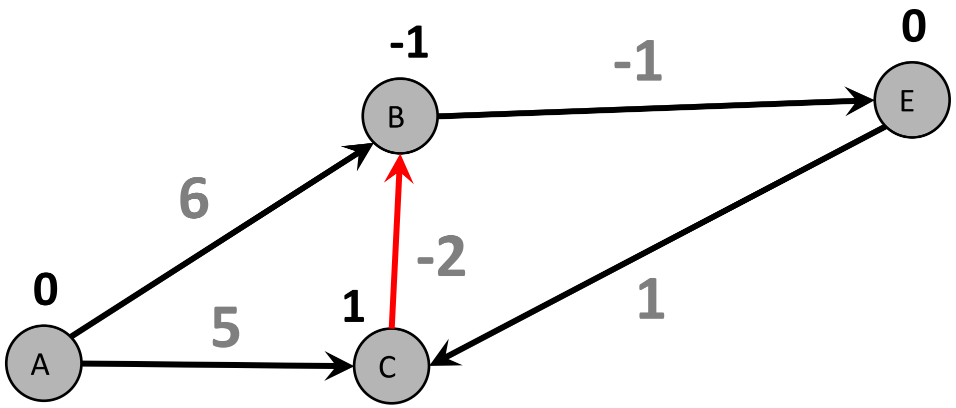


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) **(C,B)** (B,E) (E,C)

Bellman-ford: Single Source Shortest Path

3rd Iteration

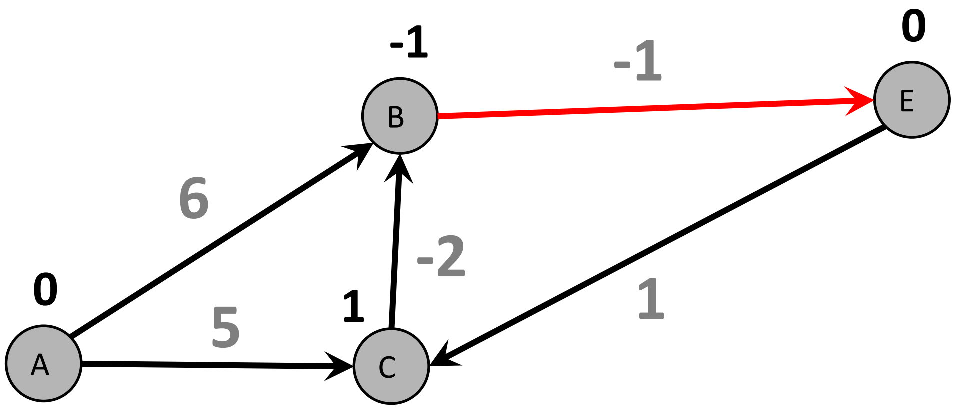


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) **(C,B)** (B,E) (E,C)

Bellman-ford: Single Source Shortest Path

3rd Iteration

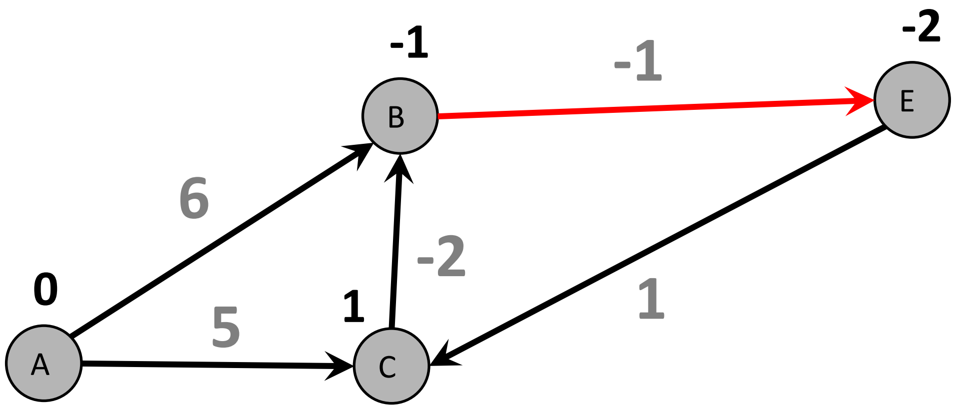


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (C,B) **(B,E)** (E,C)

Bellman-ford: Single Source Shortest Path

3rd Iteration

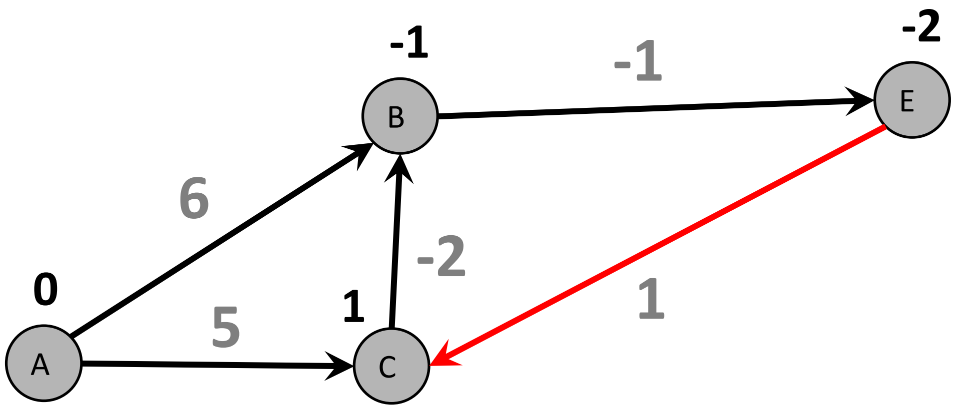


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (C,B) **(B,E)** (E,C)

Bellman-ford: Single Source Shortest Path

3rd Iteration

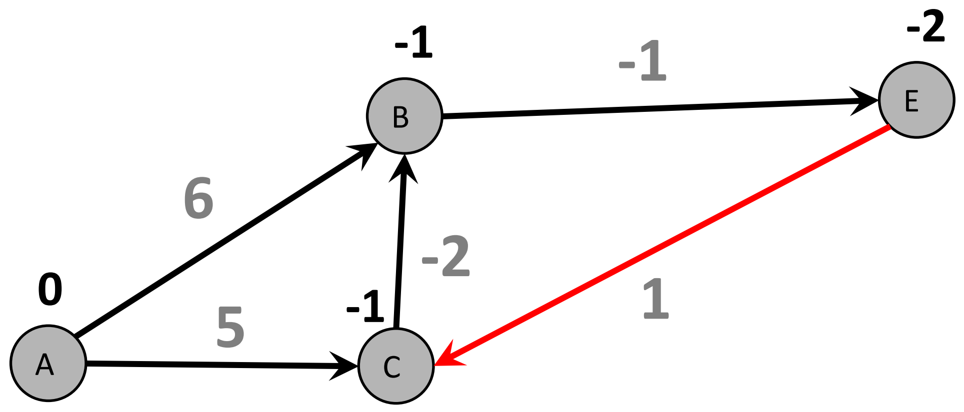


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (C,B) (B,E) **(E,C)**

Bellman-ford: Single Source Shortest Path

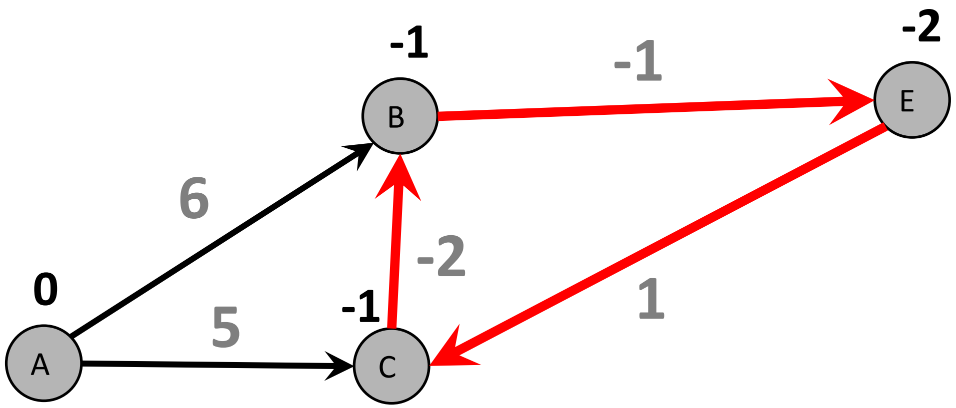
3rd Iteration



If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

(A,B) (A,C) (C,B) (B,E) **(E,C)**

Bellman-ford: Single Source Shortest Path

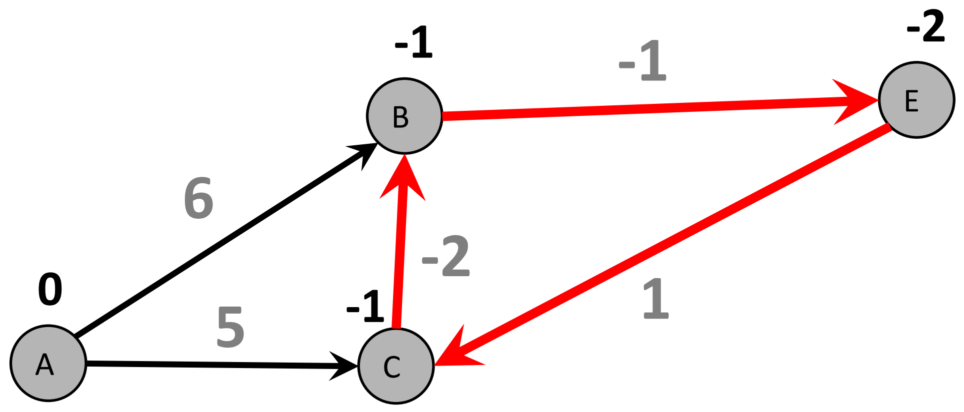


Negative Cycle

If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

Bellman-ford: Single Source Shortest Path

$(-2) + (-1) + 1 = \mathbf{-2}$

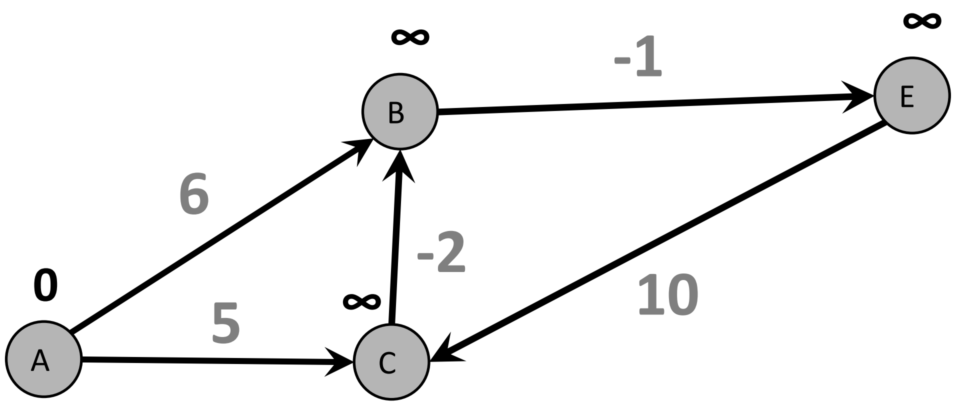


If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

Negative Cycle

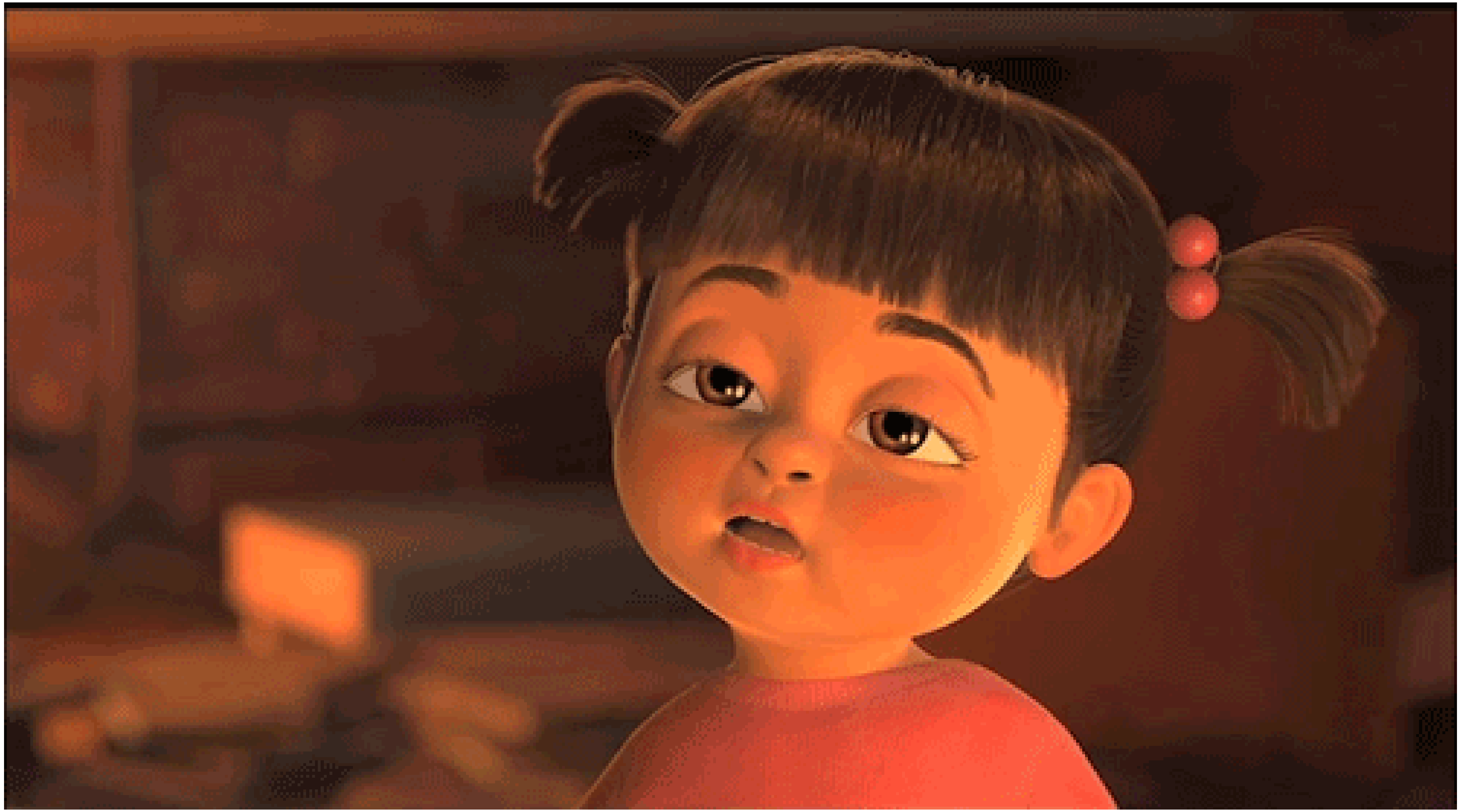
Bellman-ford: Single Source Shortest Path

$(-2) + (-1) + 10 = \mathbf{7}$



If (d[u]+ e(u,v) < d[v])
d[v]=d[u]+ e(u,v)

Thanks a lot



If you are taking a Nap, **wake up**……..***Lecture OVER***