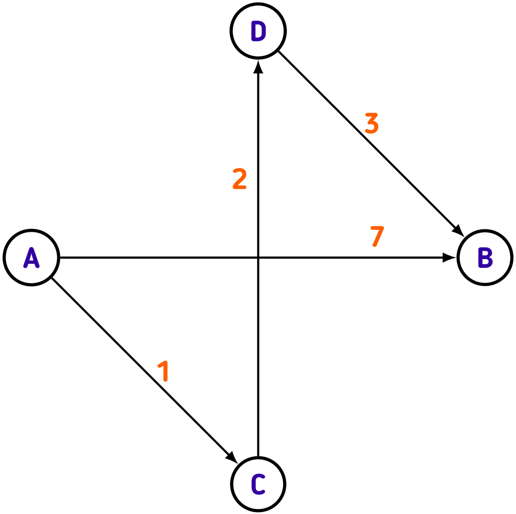
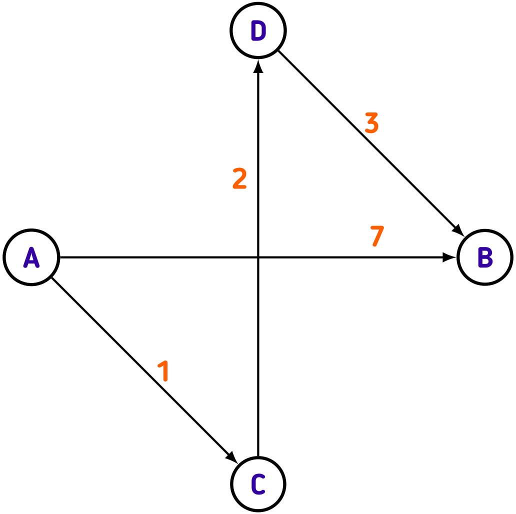
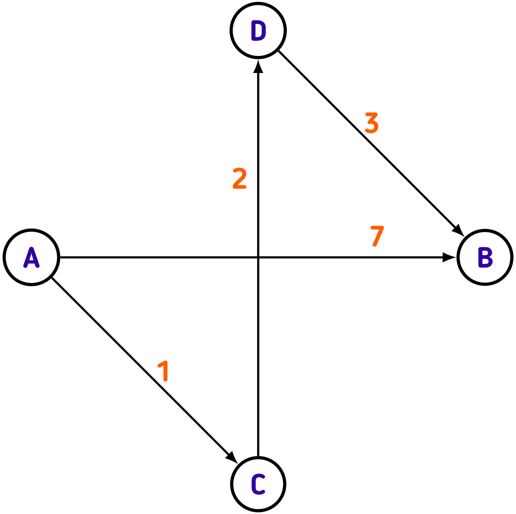

D
3
2
A
7
B
1
C

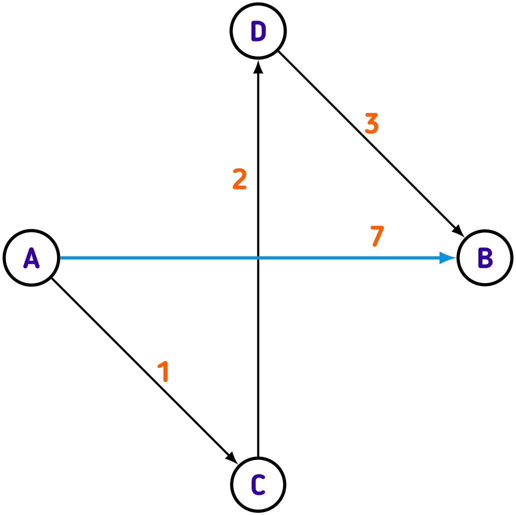


	A	B	C	D
dist$(u, \mathbf{A})$	0	∞	∞	∞



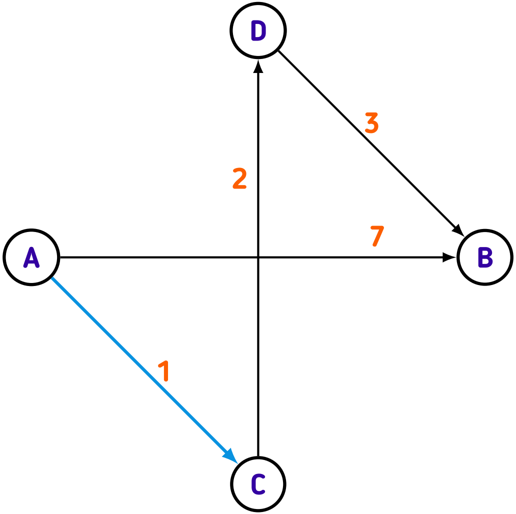
	A	B	C	D
dist$(u, \textbf{A})$	0	∞	∞	∞

	A	B	C	D
pred(u)	A	-	-	-



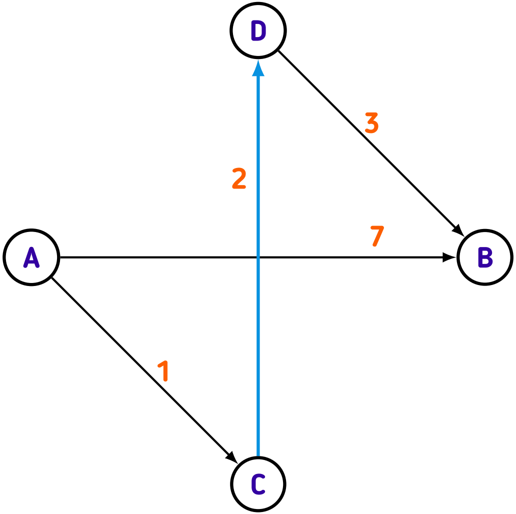
	A	B	C	D
$\mathrm{dist}(u, \mathbf{A})$	0	7	∞	∞

	A	B	C	D
$\mathrm{pred}(u)$	A	A	-	-



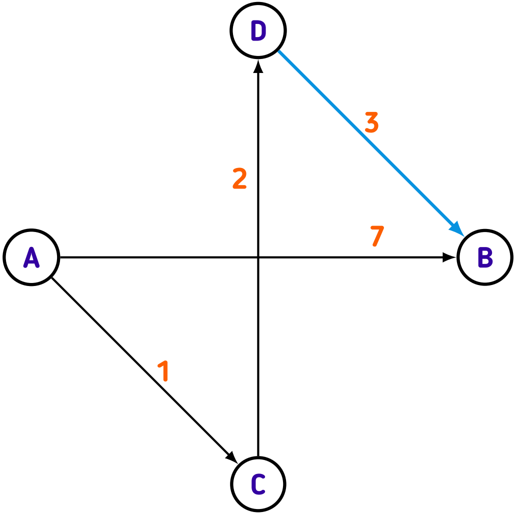
	A	B	C	D
dist(u, **A**)	0	7	**1**	∞

	A	B	C	D
pred(u)	A	A	A	-



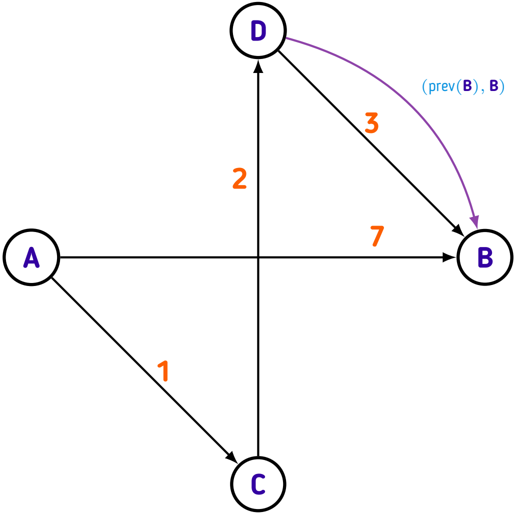
	A	B	C	D
dist$(u, \mathbf{A})$	0	7	1	**3**

	A	B	C	D
pred(u)	A	A	A	C



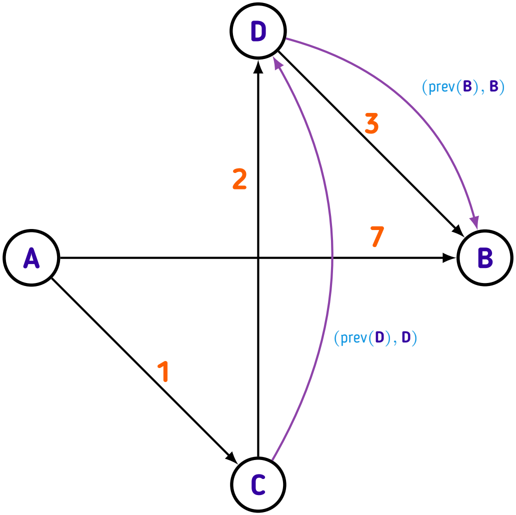
	A	B	C	D
dist$(u, \mathbf{A})$	0	**6**	1	3

	A	B	C	D
pred(u)	A	D	A	C



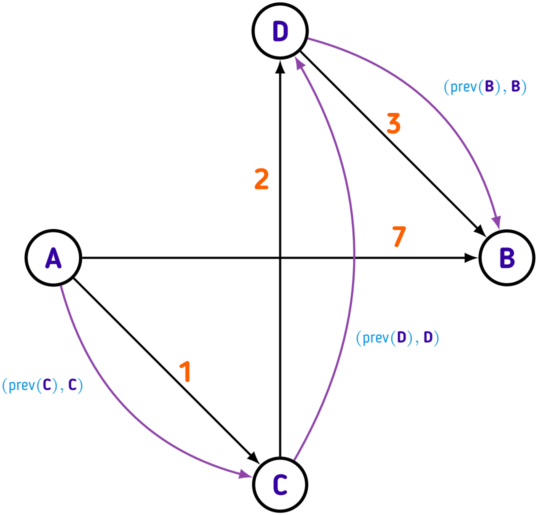
	A	B	C	D
$\text{dist}(u, \textbf{A})$	0	6	1	3

	A	B	C	D
$\text{pred}(u)$	A	D	A	C



	A	B	C	D
$\text{dist}(u, \textbf{A})$	0	6	1	3

	A	B	C	D
$\text{pred}(u)$	A	D	A	C



	A	B	C	D
dist$(u,$ **A**$)$	0	6	1	3

	A	B	C	D
pred(u)	A	D	A	C

```cpp
pair<vector<int>, vector<int>>
bellman_ford(int s, int N, const vector<edge>& edges)
{
    vector<int> dist(N + 1, oo), pred(N + 1, oo);

    dist[s] = 0;
    pred[s] = s;

    for (int i = 1; i <= N - 1; i++)
        for (auto [u, v, w] : edges)
            if (dist[v] > dist[u] + w) {
                dist[v] = dist[u] + w;
                pred[v] = u;
            }

    return { dist, pred };
}
```

```cpp
vector<ii> path(int s, int u, const vector<int>& pred)
{
    vector<ii> p;
    int v = u;

    do {
        p.push_back(ii(pred[v], v));
        v = pred[v];
    } while (v != s);

    reverse(p.begin(), p.end());

    return p;
}
```

Caminhos mínimos e ciclos

Caminhos mínimos e ciclos

Seja

$$p = \{(a, u_1), (u_1, u_2), \ldots, (v, u_r), \ldots, (u_s, v), \ldots, (u_t, b)\}$$

um caminho de a **a** b **e** $\omega(c)$ **o custo do ciclo** $c = \{(v, u_r), \ldots, (u_s, v)\}$**, isto é**

$$\omega(c) = \sum_{e \in c} w(e)$$

Caminhos mínimos e ciclos

Seja

$$p = \{(a, u_1), (u_1, u_2), \ldots, (v, u_r), \ldots, (u_s, v), \ldots, (u_t, b)\}$$

um caminho de a **a** b **e** $\omega(c)$ **o custo do ciclo** $c = \{(v, u_r), \ldots, (u_s, v)\}$**, isto é**

$$\omega(c) = \sum_{e \in c} w(e)$$

custo da aresta e

Caminhos mínimos e ciclos

Seja

$$p = \{(a, u_1), (u_1, u_2), \ldots, (v, u_r), \ldots, (u_s, v), \ldots, (u_t, b)\}$$

um caminho de a **a** b **e** $\omega(c)$ **o custo do ciclo** $c = \{(v, u_r), \ldots, (u_s, v)\}$**, isto é**

$$\omega(c) = \sum_{e \in c} w(e)$$

custo da aresta e

Se p **é caminho mínimo de** a **a** b **então** $\omega(c) = 0$**.**

Caminhos mínimos e ciclos positivos

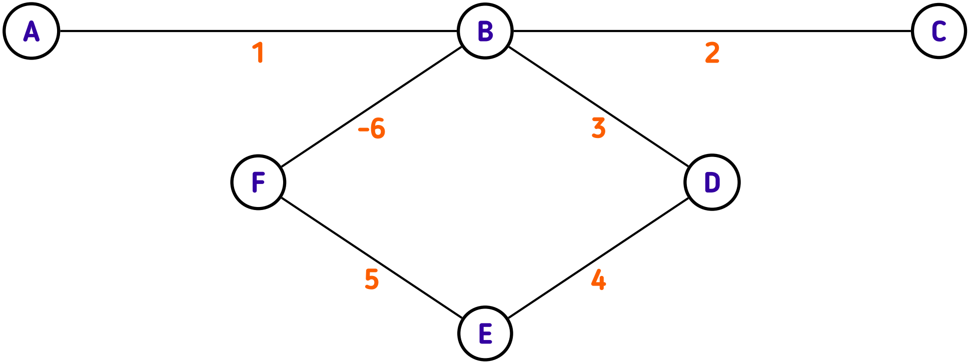
Caminhos mínimos e ciclos positivos

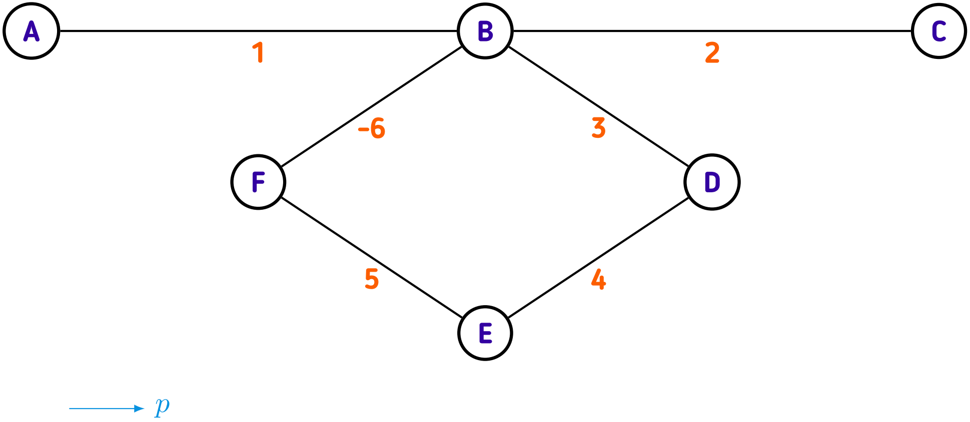
Seja $\omega(c) > 0$ **e**

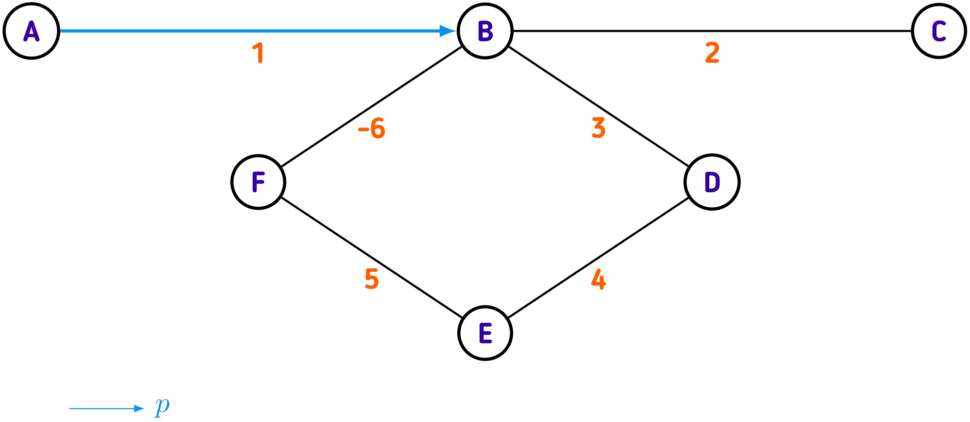
$$q = \{(a, u_1), (u_1, u_2), \ldots, (u_{r-1}, v), (v, u_{s+1}), \ldots, (u_t, b)\},$$

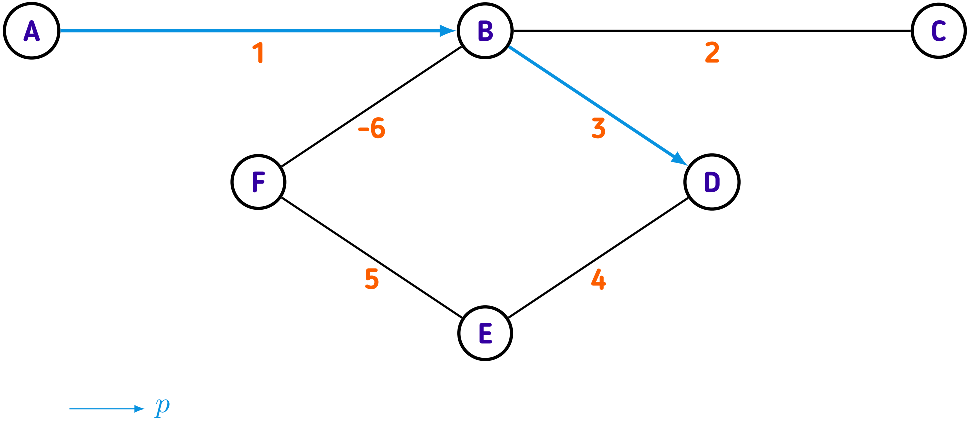
o caminho resultante da exclusão do ciclo c **de** p**. Então** $\omega(q) < \omega(p)$**, pois**

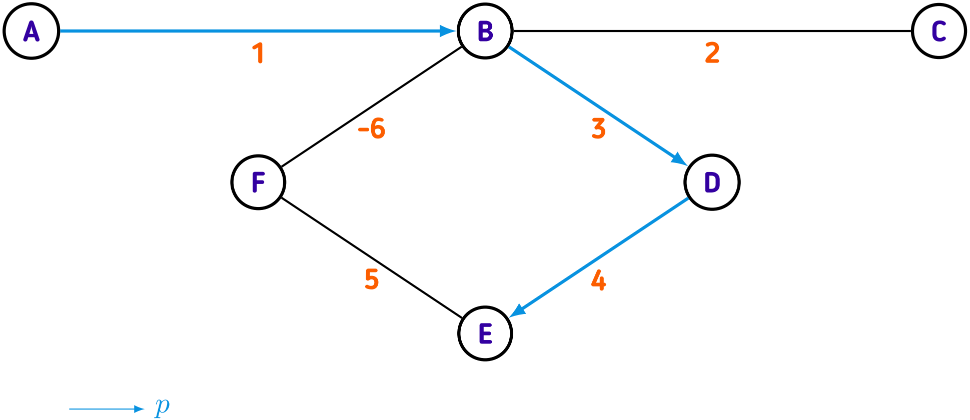
$$\omega(p) = \sum_{e_i \in p} w(e_i) = \sum_{e_j \in q} w(e_j) + \sum_{e_k \in c} w(e_k) = \omega(q) + \omega(c) > \omega(q)$$

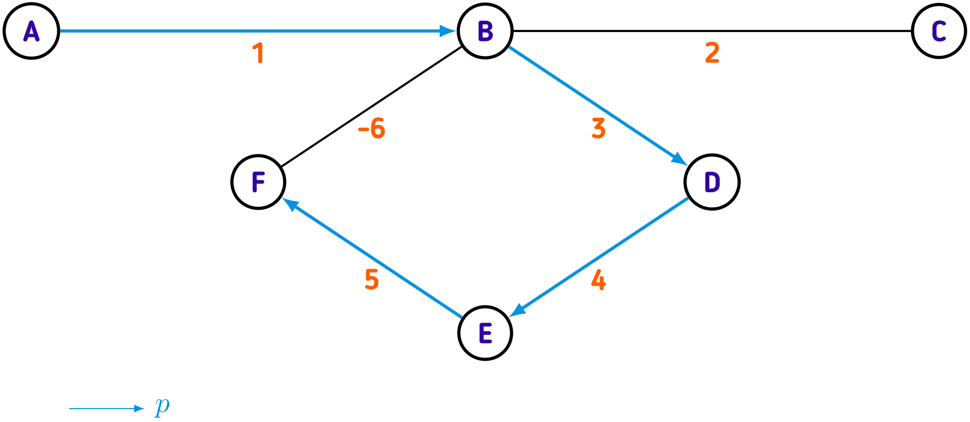

A
B
C
1
2
-6
3
F
D
5
4
E

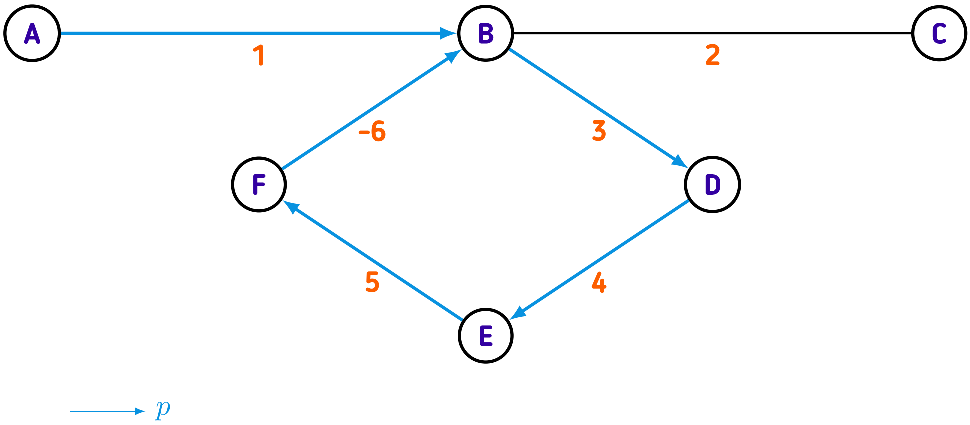

A
B
C
D
E
F
1
2
3
4
5
-6
p

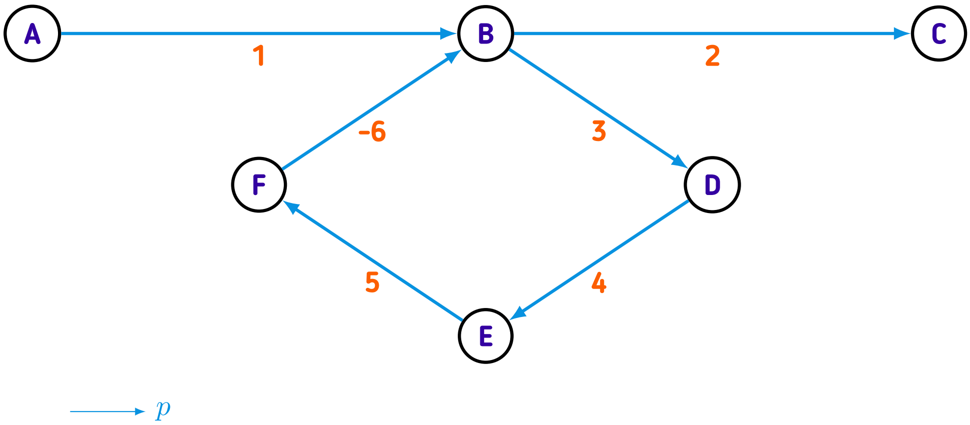

A
B
C
1
2
-6
3
F
D
5
4
E
p

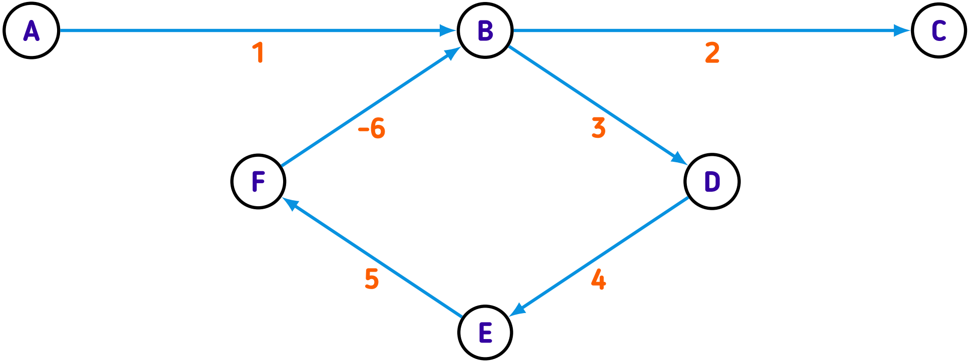

A
B
C
F
D
E
1
2
-6
3
5
4
p


A
B
C
D
E
F
1
2
-6
3
5
4
p

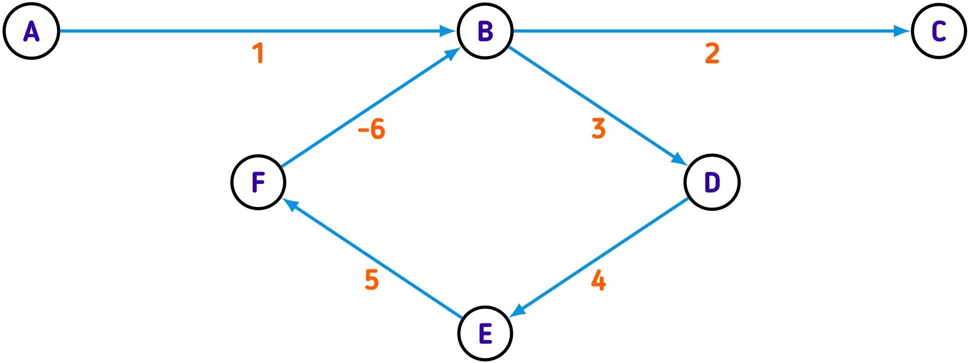

A
B
C
D
E
F
1
2
3
4
5
-6
p


A
B
C
D
E
F
1
2
3
4
5
-6
p


A
B
C
D
E
F
1
2
3
4
5
-6
p

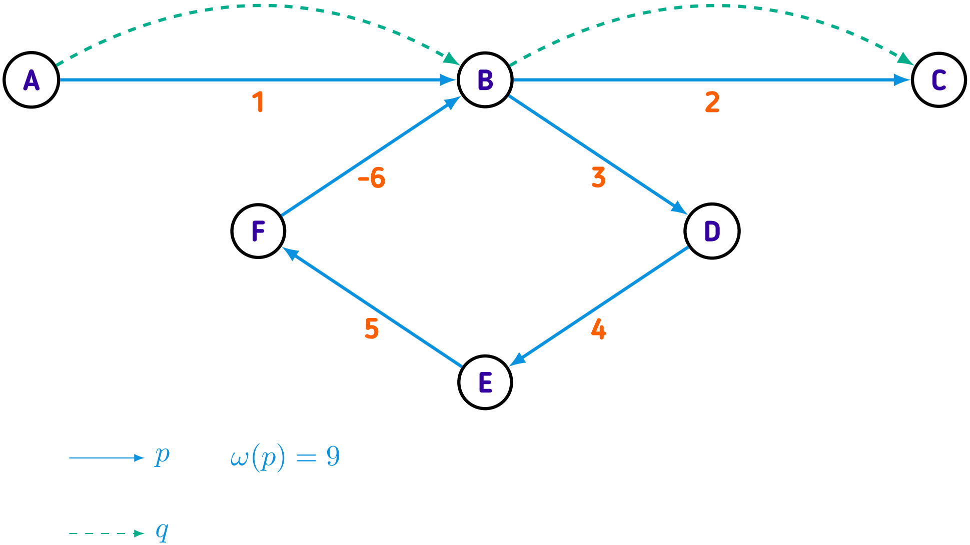


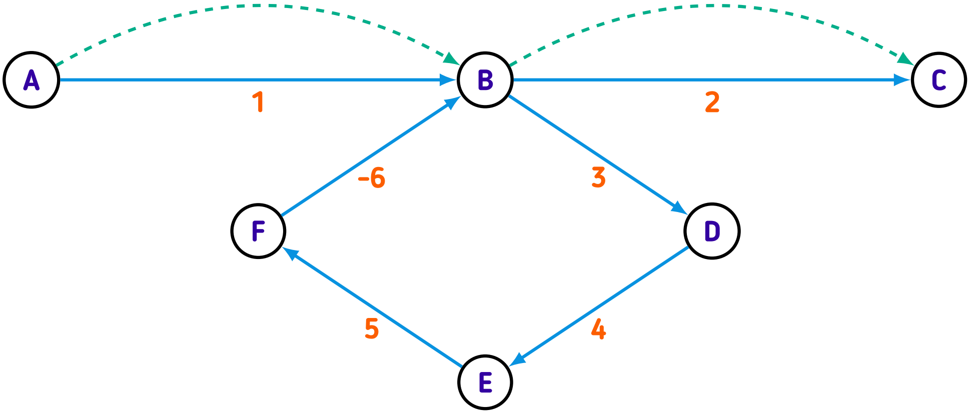
$p \qquad \omega(p) = 9$



p $\quad \omega(p) = 9$

q


A
B
C
D
E
F
1
2
3
4
5
-6
p
ω(p) = 9
q



$p \qquad \omega(p) = 9$

$q \qquad \omega(q) = 3$

Caminhos mínimos e ciclos negativos

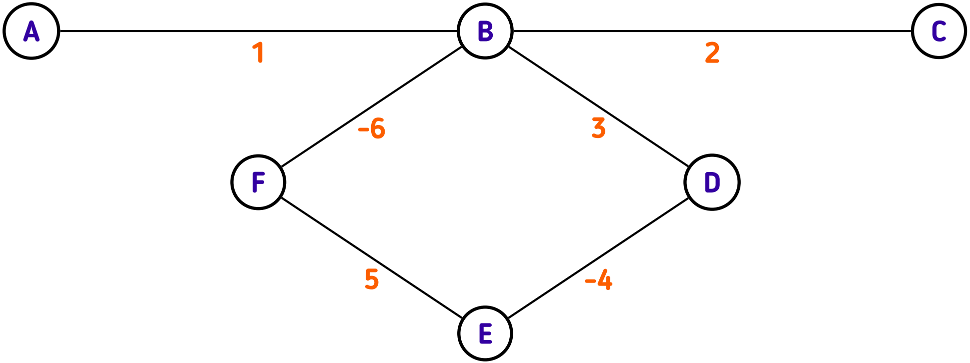
Caminhos mínimos e ciclos negativos

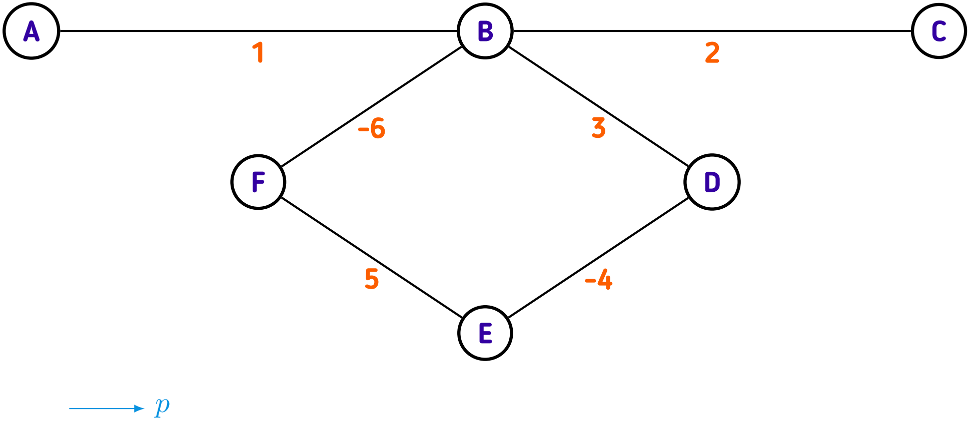
Seja $\omega(c) < 0$ **e**

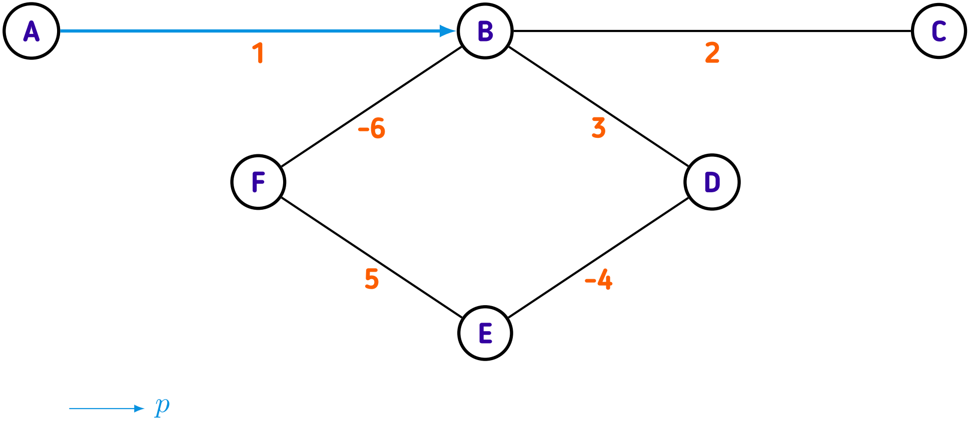
$$q = \{(a, u_1), (u_1, u_2), \ldots, (v, u_r), \ldots, (u_s, v), (v, u_r), \ldots, (u_s, v), \ldots, (u_t, b)\}$$

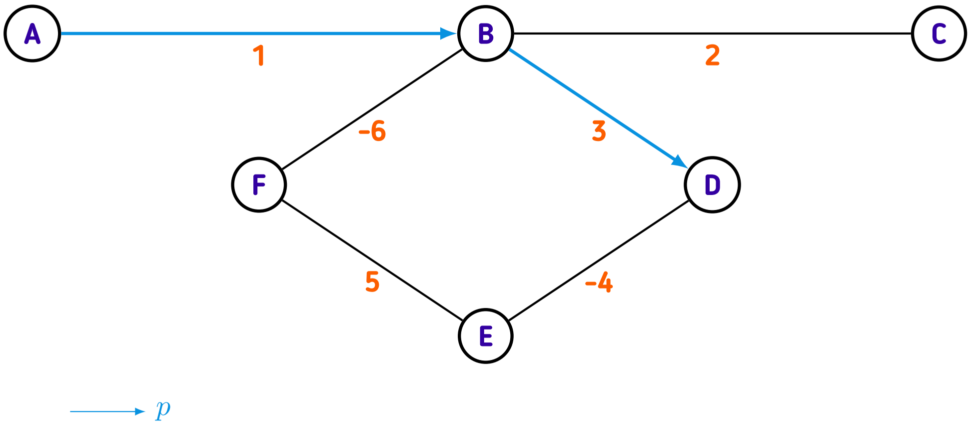
o caminho resultante da duplicação do ciclo c **de** p**. Então** $\omega(q) < \omega(p)$**, pois**

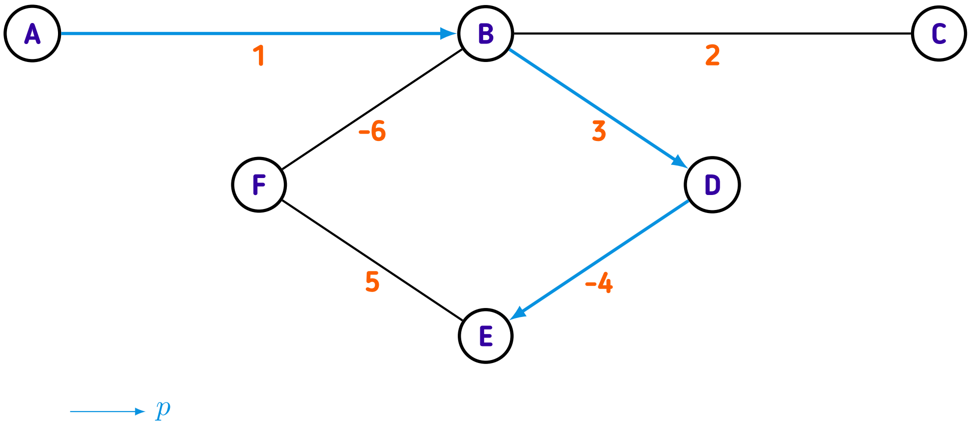
$$\omega(q) = \sum_{e_i \in q} w(e_i) = \sum_{e_j \in p} w(e_j) + \sum_{e_k \in c} w(e_k) = \omega(p) + \omega(c) < \omega(p)$$

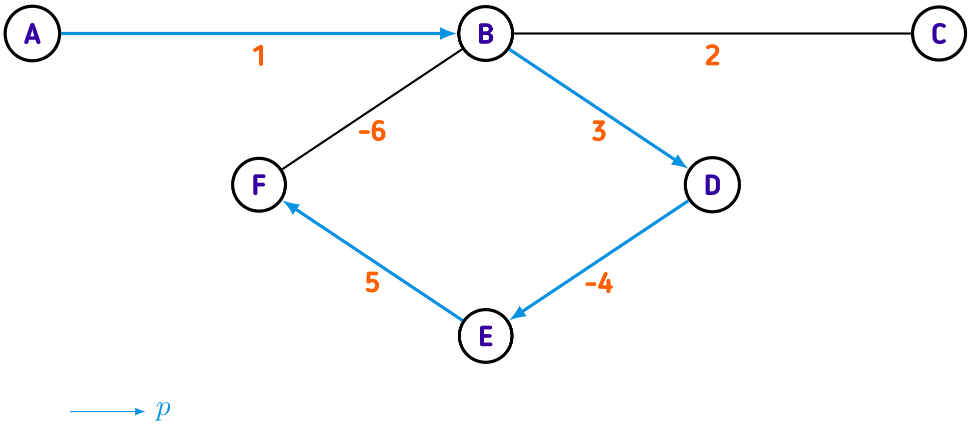

A
B
C
1
2
-6
3
F
D
5
-4
E

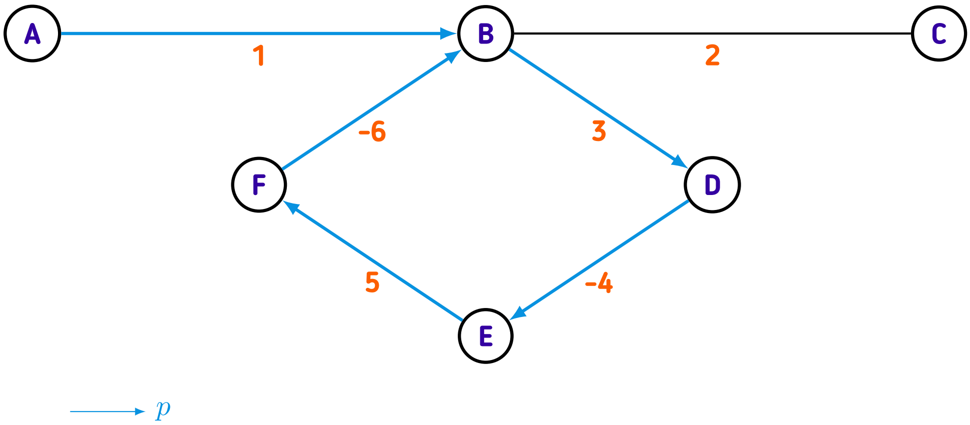

A
B
C
1
2
-6
3
F
D
5
-4
E
p


A
B
C
D
E
F
1
2
-6
3
5
-4
p

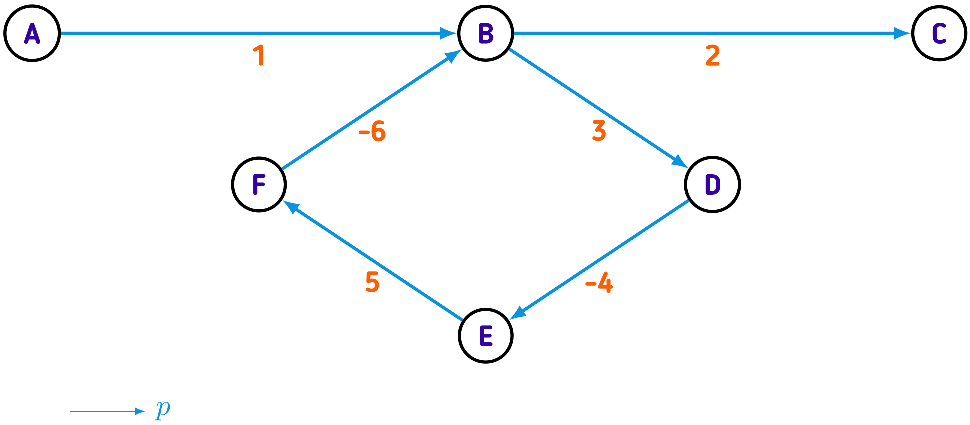

A
B
C
D
E
F
1
2
-6
3
5
-4
p

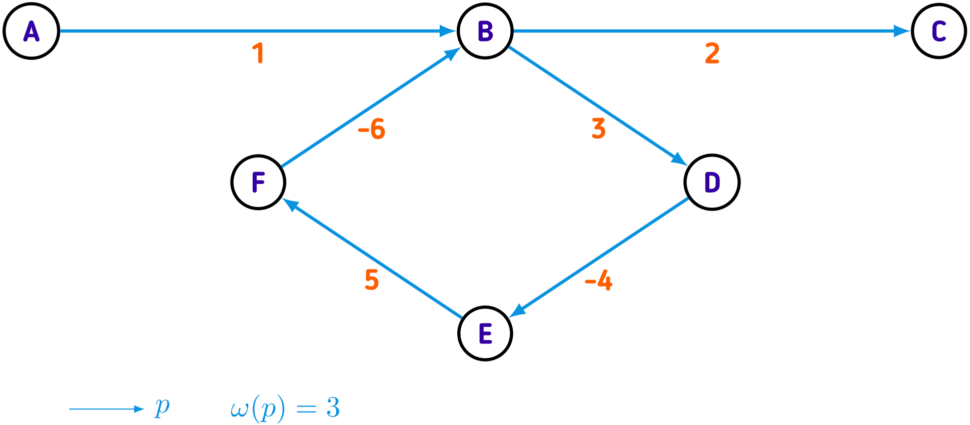

A
B
C
D
E
F
1
2
-6
3
5
-4
p

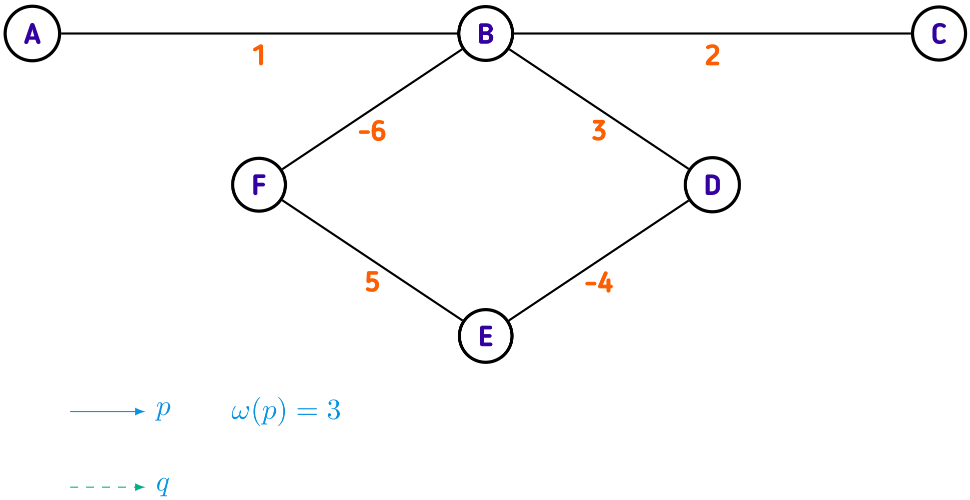

A
B
C
F
D
E
1
2
-6
3
5
-4
p

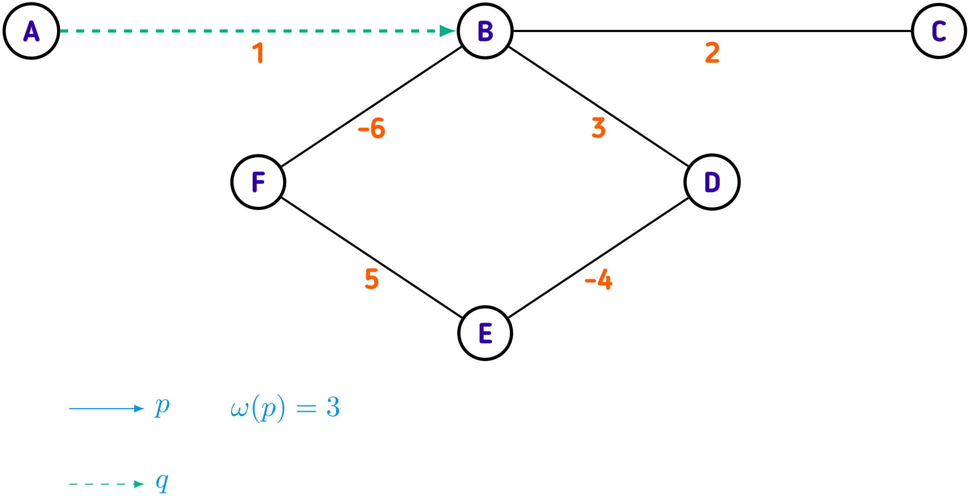


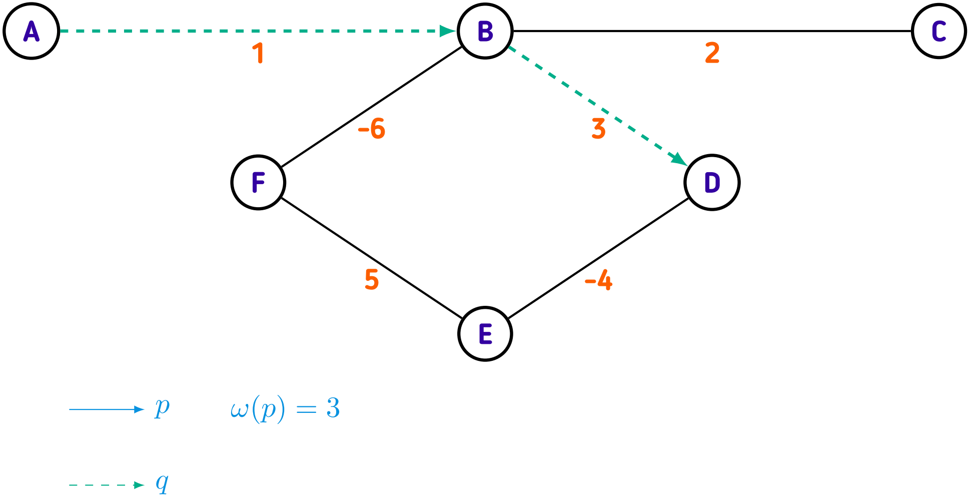
A
B
C
D
E
F
1
2
3
-4
5
-6
p

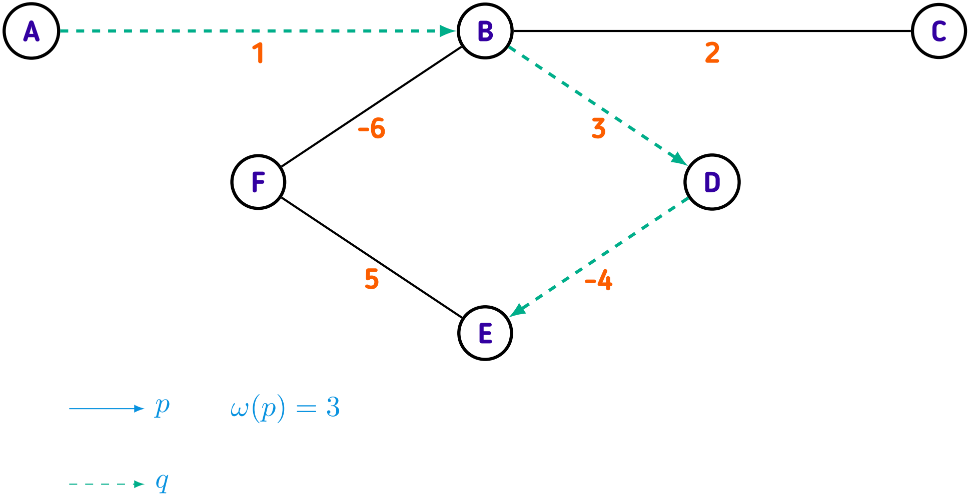

A
B
C
D
E
F
1
2
3
-4
5
-6
p


A
B
C
1
2
-6
3
F
D
5
-4
E
p
ω(p) = 3

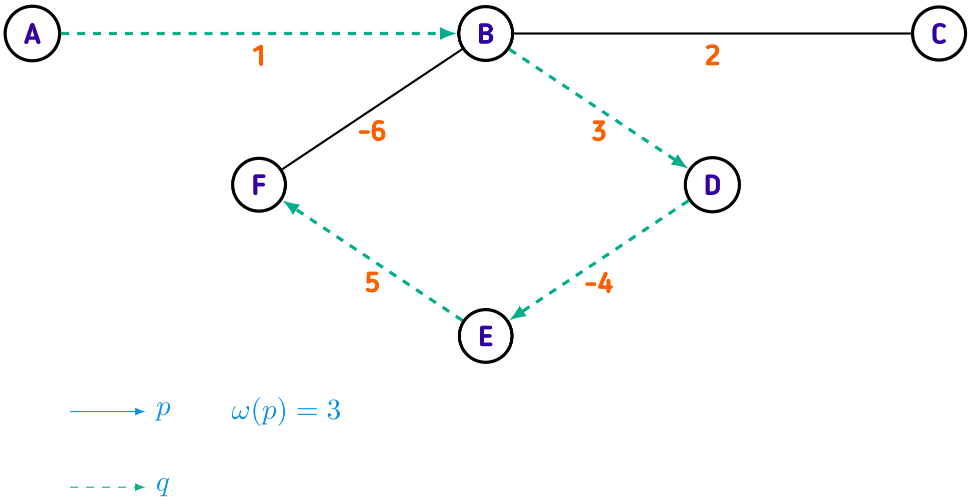

A
B
C
D
E
F
1
2
-6
3
5
-4
p
ω(p) = 3
q

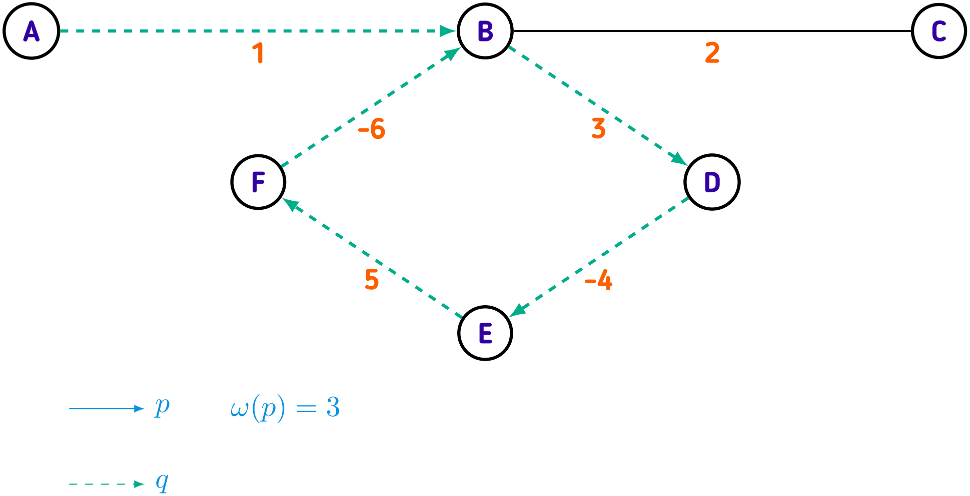

A
B
C
D
E
F
1
2
-6
3
5
-4
p
$\omega(p) = 3$
q

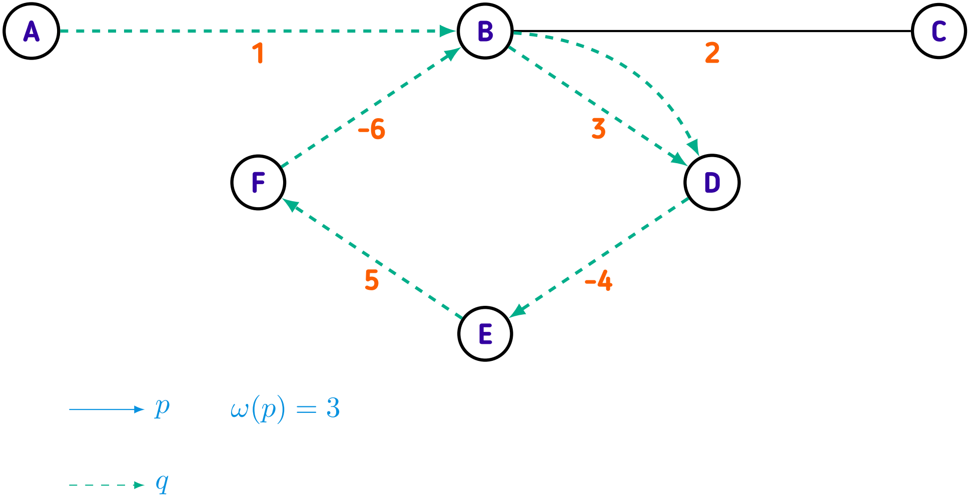

A
B
C
D
E
F
1
2
-6
3
5
-4
p
$\omega(p) = 3$
q

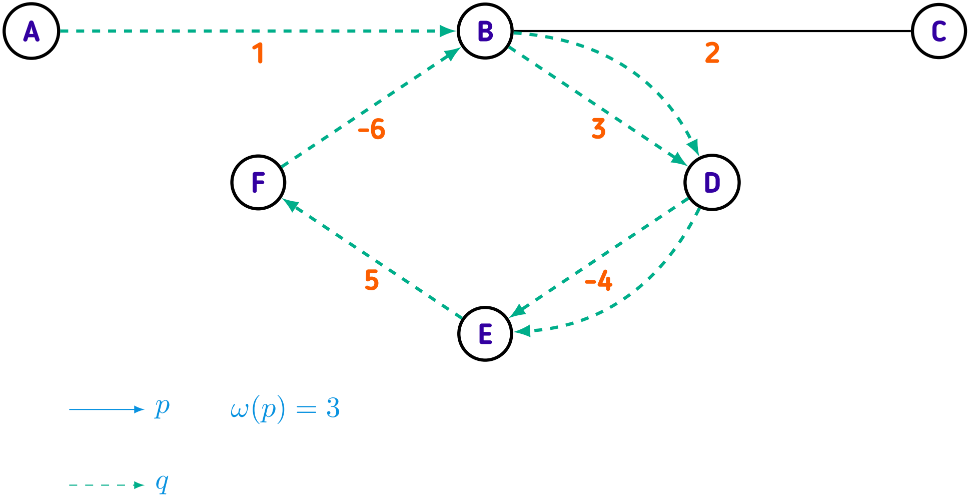


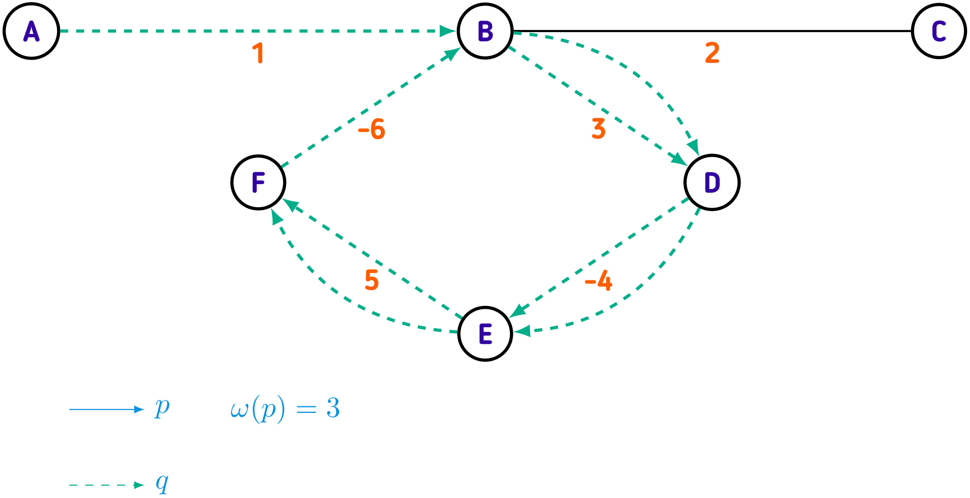
A
B
C
D
E
F
1
2
-6
3
5
-4
p
ω(p) = 3
q

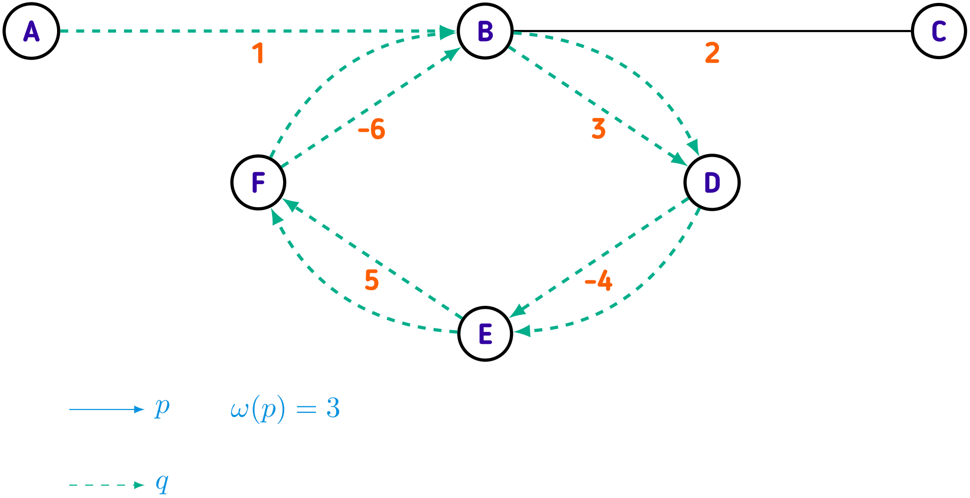

A
B
C
D
E
F
1
2
-6
3
5
-4
p
$\omega(p) = 3$
q


A
B
C
D
E
F
1
2
3
-4
5
-6
p
$\omega(p) = 3$
q

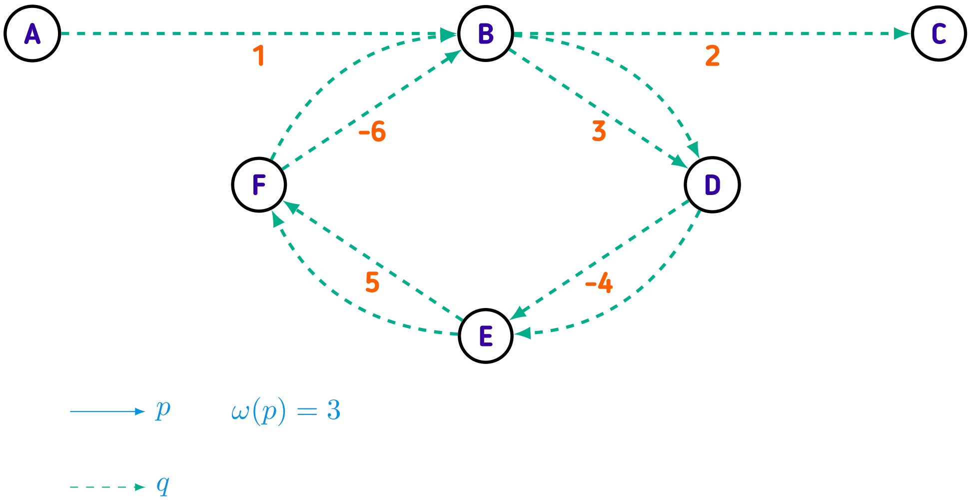

A
B
C
D
E
F
1
2
3
-4
5
-6
p
ω(p) = 3
q

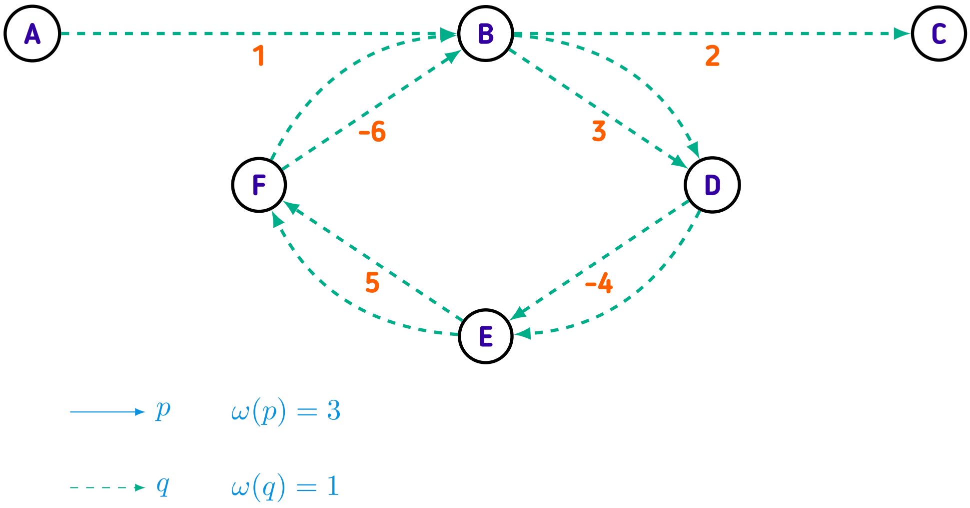

A
B
C
D
E
F
1
2
3
-6
-4
5
p
q
ω(p) = 3


A
B
C
1
2
-6
3
F
D
5
-4
E
p
ω(p) = 3
q



A
B
C
D
E
F
1
2
3
-4
5
-6
p
$\omega(p) = 3$
q


A
B
C
D
E
F
1
2
3
-6
-4
5
p
$\omega(p) = 3$
q


A
B
C
D
E
F
1
2
3
-6
-4
5
p
ω(p) = 3
q
ω(q) = 1

Número de rodadas do algoritmo de Bellman-Ford

Número de rodadas do algoritmo de Bellman-Ford

Teorema. Seja $G(V, E)$ um grafo cujos pesos de suas arestas sejam todos não-negativos. Então para qualquer $v \in V$, o caminho mínimo de s a u identificado pelo algoritmo de Bellman-Ford tem, no máximo, $|V| - 1$ arestas.

Detecção de ciclos negativos

Detecção de ciclos negativos

Teorema. Seja $G(V, E)$ um grafo. Se a $|V|$-ésima rodada do algoritmo de Bellman-Ford atualizar o vetor d ao menos uma vez, então G possui pelo menos um ciclo negativo.

```cpp
bool has_negative_cycle(int s, int N, const vector<edge>& edges)
{
    const int oo { 1000000010 };

    vector<int> dist(N + 1, oo);
    dist[s] = 0;

    for (int i = 1; i <= N - 1; i++)
        for (auto [u, v, w] : edges)
            dist[v] = min(dist[v], dist[u] + w);

    for (auto [u, v, w] : edges)
        if (dist[v] > dist[u] + w)
            return true;

    return false;
}
```

Problemas sugeridos

1. AtCoder Beginner Contest 088 – Problem D: Repainting
2. Codeforces Beta Round #3 – Problem A: Shortest path of the king
3. OJ 10000 – Longest Paths
4. OJ 10959 – The Party, Part I

Referências

1. **HALIM**, **Felix**; **HALIM**, **Steve**. *Competitive Programming 3*, **2010.**

2. **LAAKSONEN**, **Antti**. *Competitive Programmer's Handbook*, **2018.**

3. **SKIENA**, **Steven**; **REVILLA**, **Miguel**. *Programming Challenges*, **2003.**

4. **Wikipédia**, *Bellman-Ford algorithm.* **Acesso em 07/07/2021.**

5. **Wikipédia**, *L. R. Ford Jr.* **Acesso em 07/07/2021.**

6. **Wikipédia**, *Richard E. Bellman.* **Acesso em 07/07/2021.**