Caminho Mínimo de Fonte Única em Grafos com Pesos Negativos

Letícia Rodrigues Bueno

UFABC

Problemas de Caminho Mínimo

- **Caminho mínimo de fonte única:** algoritmo de Dijsktra;

Problemas de Caminho Mínimo

- **Caminho mínimo de fonte única:** algoritmo de Dijsktra;
- **Caminho mínimo de destino único:** inverta a direção das arestas e aplique algoritmo de Dijstra;

Problemas de Caminho Mínimo

- **Caminho mínimo de fonte única:** algoritmo de Dijsktra;
- **Caminho mínimo de destino único:** inverta a direção das arestas e aplique algoritmo de Dijstra;
- **Caminho mínimo entre quaisquer vértices u e v:** algoritmo de Dijsktra;

Problemas de Caminho Mínimo

- **Caminho mínimo de fonte única:** algoritmo de Dijsktra;
- **Caminho mínimo de destino único:** inverta a direção das arestas e aplique algoritmo de Dijstra;
- **Caminho mínimo entre quaisquer vértices u e v:** algoritmo de Dijsktra;
- **Caminho mínimo em grafos com pesos negativos:** algoritmo de Bellman-Ford;

Problemas de Caminho Mínimo

- **Caminho mínimo de fonte única:** algoritmo de Dijsktra;
- **Caminho mínimo de destino único:** inverta a direção das arestas e aplique algoritmo de Dijstra;
- **Caminho mínimo entre quaisquer vértices u e v:** algoritmo de Dijsktra;
- **Caminho mínimo em grafos com pesos negativos:** algoritmo de Bellman-Ford;
- **Caminho mínimo de todos os vértices para todos os vértices:** algoritmo de Floyd-Warshall em tempo $O(n^3)$.

Caminho Mínimo em Grafos com Pesos Negativos

- pode haver arestas com pesos negativos;

Caminho Mínimo em Grafos com Pesos Negativos

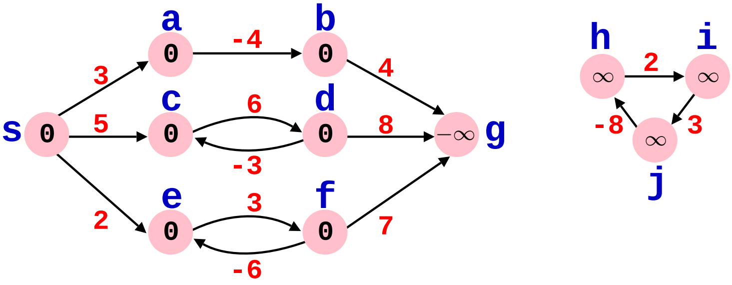
- pode haver arestas com pesos negativos;
- se não há ciclo de peso negativo acessível a partir da origem:

Caminho Mínimo em Grafos com Pesos Negativos

- pode haver arestas com pesos negativos;
- se não há ciclo de peso negativo acessível a partir da origem: **algoritmo de Dijsktra**;

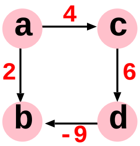
Caminho Mínimo em Grafos com Pesos Negativos

- pode haver arestas com pesos negativos;
- se não há ciclo de peso negativo acessível a partir da origem: **algoritmo de Dijsktra**;
- Exemplo de ciclo de peso negativo:



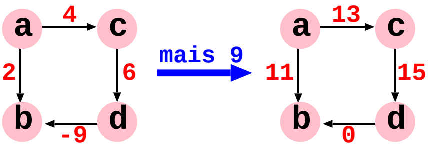
Caminho Mínimo em Grafos com Pesos Negativos

- E se adicionarmos uma constante ao peso de cada aresta?



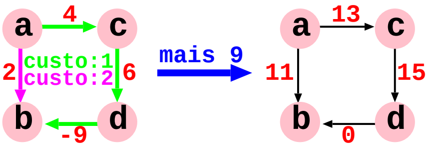
Caminho Mínimo em Grafos com Pesos Negativos

- E se adicionarmos uma constante ao peso de cada aresta?



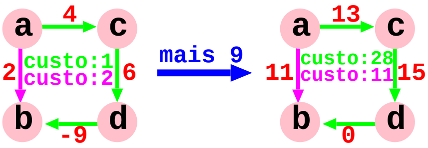
Caminho Mínimo em Grafos com Pesos Negativos

- E se adicionarmos uma constante ao peso de cada aresta?



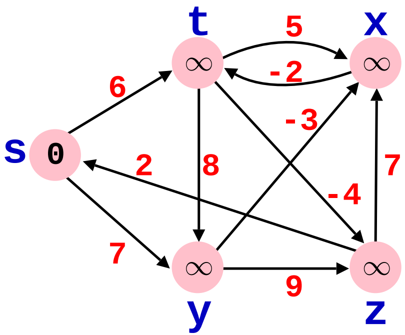
Caminho Mínimo em Grafos com Pesos Negativos

- E se adicionarmos uma constante ao peso de cada aresta?



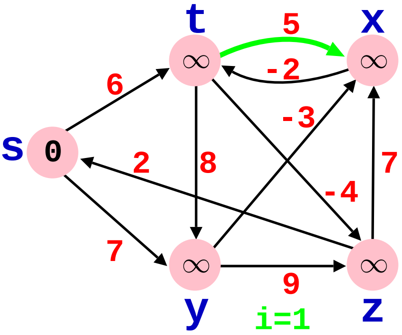
Caminho Mínimo em Grafos com Pesos Negativos

- Algoritmo de Bellman-Ford:



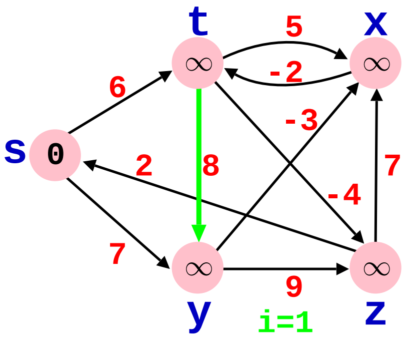
Caminho Mínimo em Grafos com Pesos Negativos

- Algoritmo de Bellman-Ford:



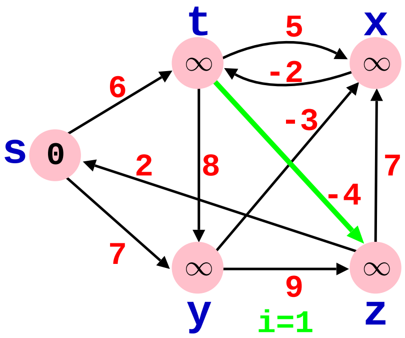
Caminho Mínimo em Grafos com Pesos Negativos

- Algoritmo de Bellman-Ford:



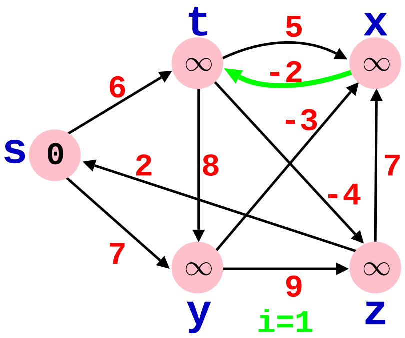
Caminho Mínimo em Grafos com Pesos Negativos

- Algoritmo de Bellman-Ford:



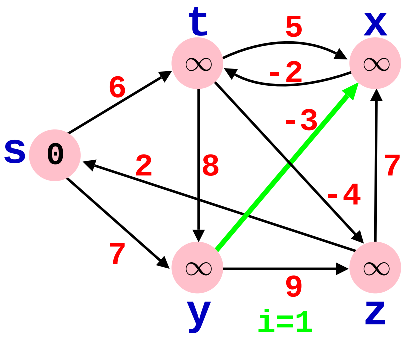
Caminho Mínimo em Grafos com Pesos Negativos

- Algoritmo de Bellman-Ford:



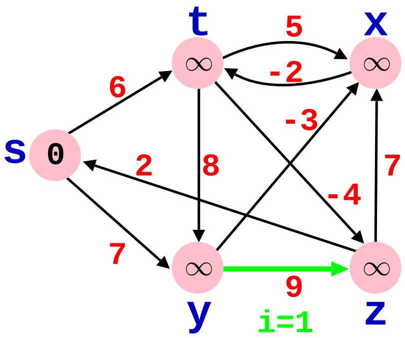
Caminho Mínimo em Grafos com Pesos Negativos

- Algoritmo de Bellman-Ford:



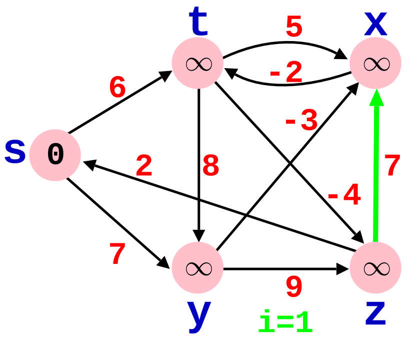
Caminho Mínimo em Grafos com Pesos Negativos

- Algoritmo de Bellman-Ford:



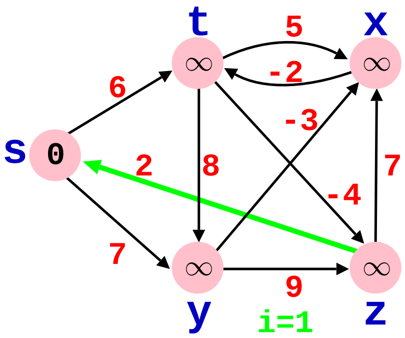
Caminho Mínimo em Grafos com Pesos Negativos

- Algoritmo de Bellman-Ford:



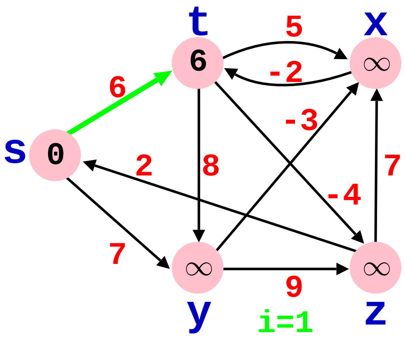
Caminho Mínimo em Grafos com Pesos Negativos

- Algoritmo de Bellman-Ford:



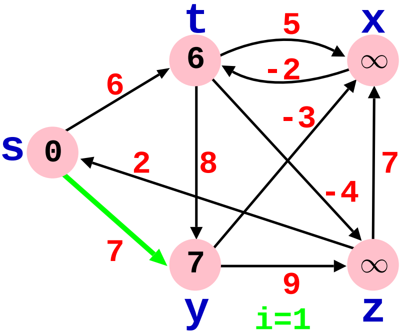
Caminho Mínimo em Grafos com Pesos Negativos

- Algoritmo de Bellman-Ford:



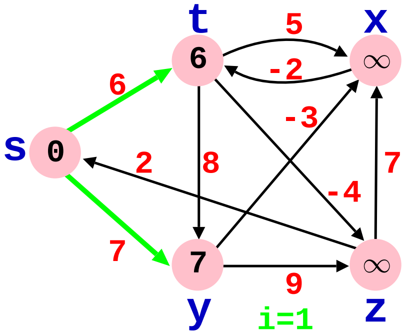
Caminho Mínimo em Grafos com Pesos Negativos

- Algoritmo de Bellman-Ford:



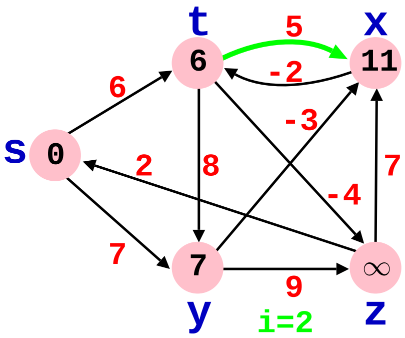
Caminho Mínimo em Grafos com Pesos Negativos

- Algoritmo de Bellman-Ford:



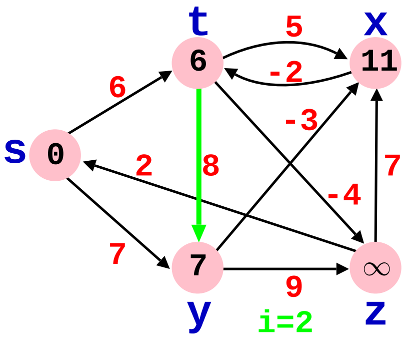
Caminho Mínimo em Grafos com Pesos Negativos

- Algoritmo de Bellman-Ford:



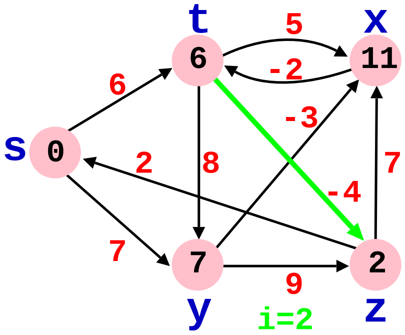
Caminho Mínimo em Grafos com Pesos Negativos

- Algoritmo de Bellman-Ford:



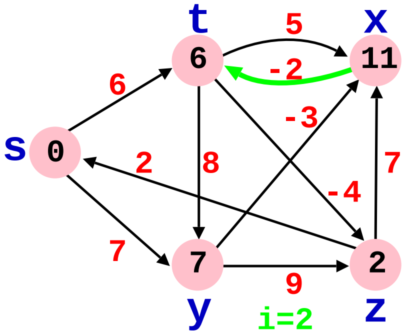
Caminho Mínimo em Grafos com Pesos Negativos

- Algoritmo de Bellman-Ford:



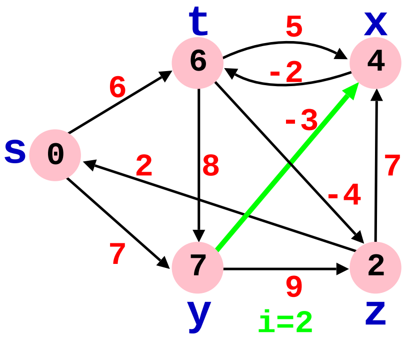
Caminho Mínimo em Grafos com Pesos Negativos

- Algoritmo de Bellman-Ford:



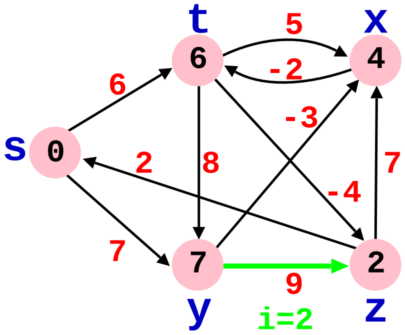
Caminho Mínimo em Grafos com Pesos Negativos

- Algoritmo de Bellman-Ford:



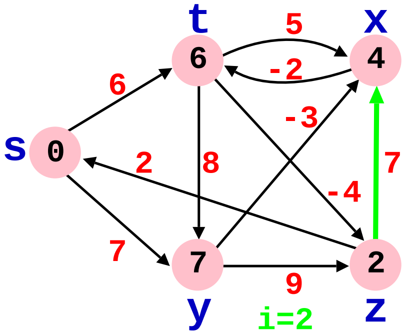
Caminho Mínimo em Grafos com Pesos Negativos

- Algoritmo de Bellman-Ford:



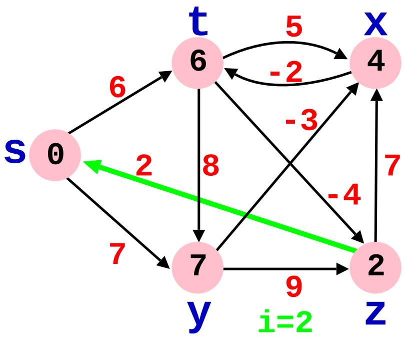
Caminho Mínimo em Grafos com Pesos Negativos

- Algoritmo de Bellman-Ford:



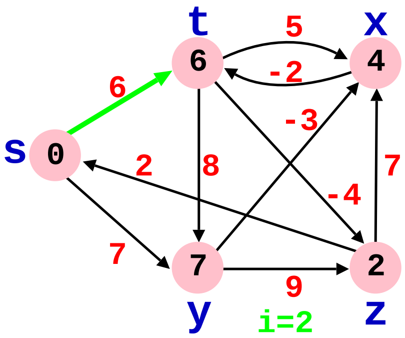
Caminho Mínimo em Grafos com Pesos Negativos

- Algoritmo de Bellman-Ford:



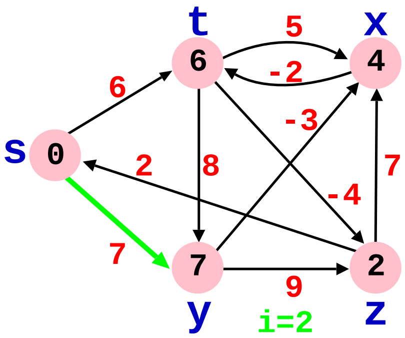
Caminho Mínimo em Grafos com Pesos Negativos

- Algoritmo de Bellman-Ford:



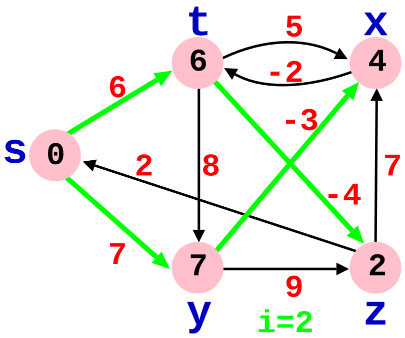
Caminho Mínimo em Grafos com Pesos Negativos

- Algoritmo de Bellman-Ford:



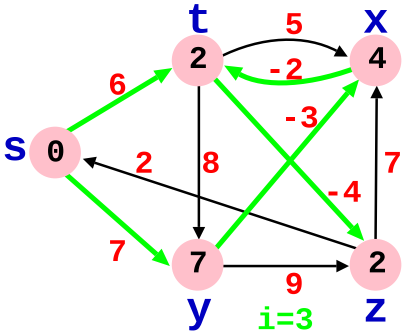
Caminho Mínimo em Grafos com Pesos Negativos

- Algoritmo de Bellman-Ford:



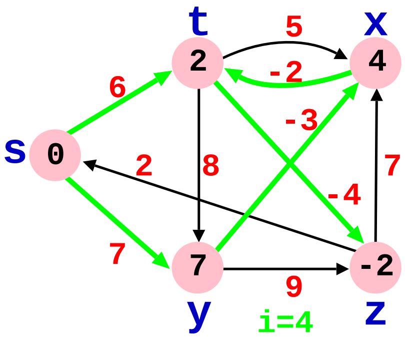
Caminho Mínimo em Grafos com Pesos Negativos

- Algoritmo de Bellman-Ford:



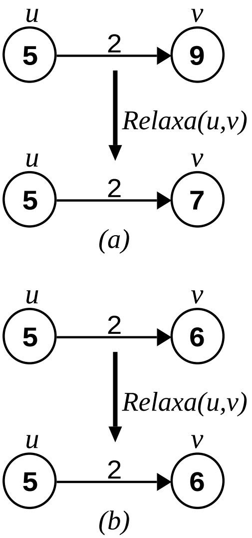
Caminho Mínimo em Grafos com Pesos Negativos

- Algoritmo de Bellman-Ford:



Caminho mínimo de fonte única: algoritmo de Bellman-Ford

```
relaxa(u, v):
    se v.d > u.d + p(u, v) então
        v.d = u.d + p(u, v)
        v.p = u
```



Complexidade do algoritmo de Bellman-Ford

```
bellman-ford(G, s):
    para u em V(G) faça
        u.d = ∞
        u.p = None
    s.d = 0
    s.p = s
    para i=1 a n-1 faça
        para ∀(u,v) ∈ E(G) faça
            relaxa(u, v)
    para ∀(u,v) ∈ E(G) faça
        se v.d > u.d + p(u,v) então
            retorne false;
    retorne true;
```

Complexidade do algoritmo de Bellman-Ford

```
bellman-ford(G, s):
    para u em V(G) faça
        u.d = ∞
        u.p = None
    s.d = 0
    s.p = s
    para i=1 a n-1 faça
        para ∀(u,v) ∈ E(G) faça
            relaxa(u, v)
    para ∀(u,v) ∈ E(G) faça
        se v.d > u.d + p(u,v) então
            retorne false;
    retorne true;
```

Análise da complexidade:

Complexidade do algoritmo de Bellman-Ford

```
1   bellman-ford(G, s):
2       para u em V(G) faça
3           u.d = ∞
4           u.p = None
5       s.d = 0
6       s.p = s
7       para i=1 a n-1 faça
8           para ∀(u,v) ∈ E(G) faça
9               relaxa(u, v)
10      para ∀(u,v) ∈ E(G) faça
11          se v.d > u.d + p(u,v) então
12              retorne false;
13      retorne true;
```

Análise da complexidade:

- laço linha 2: $O(n)$;

Complexidade do algoritmo de Bellman-Ford

```
1   bellman-ford(G, s):
2       para u em V(G) faça
3           u.d = ∞
4           u.p = None
5       s.d = 0
6       s.p = s
7       para i=1 a n-1 faça
8           para ∀(u,v) ∈ E(G) faça
9               relaxa(u, v)
10      para ∀(u,v) ∈ E(G) faça
11          se v.d > u.d + p(u,v) então
12              retorne false;
13      retorne true;
```

Análise da complexidade:

- laço linha 2: $O(n)$;
- laço linha 7: $O(n)$;

Complexidade do algoritmo de Bellman-Ford

```
1  bellman-ford(G, s):
2      para u em V(G) faça
3          u.d = ∞
4          u.p = None
5      s.d = 0
6      s.p = s
7      para i=1 a n-1 faça
8          para ∀(u,v) ∈ E(G) faça
9              relaxa(u, v)
10     para ∀(u,v) ∈ E(G) faça
11         se v.d > u.d + p(u,v) então
12             retorne false;
13     retorne true;
```

Análise da complexidade:

- laço linha 2: $O(n)$;
- laço linha 7: $O(n)$;
- laço linha 8: $O(n \cdot m)$;

Complexidade do algoritmo de Bellman-Ford

```
1   bellman-ford(G, s):
2       para u em V(G) faça
3           u.d = ∞
4           u.p = None
5       s.d = 0
6       s.p = s
7       para i=1 a n-1 faça
8           para ∀(u,v) ∈ E(G) faça
9               relaxa(u, v)
10      para ∀(u,v) ∈ E(G) faça
11          se v.d > u.d + p(u,v) então
12              retorne false;
13      retorne true;
```

Análise da complexidade:

- laço linha 2: $O(n)$;
- laço linha 7: $O(n)$;
- laço linha 8: $O(n \cdot m)$;
- laço linha 10: $O(m)$;

Complexidade do algoritmo de Bellman-Ford

```
1  bellman-ford(G, s):
2      para u em V(G) faça
3          u.d = ∞
4          u.p = None
5      s.d = 0
6      s.p = s
7      para i=1 a n-1 faça
8          para ∀(u,v) ∈ E(G) faça
9              relaxa(u, v)
10     para ∀(u,v) ∈ E(G) faça
11         se v.d > u.d + p(u,v) então
12             retorne false;
13     retorne true;
```

Análise da complexidade:

- laço linha 2: $O(n)$;
- laço linha 7: $O(n)$;
- laço linha 8: $O(n \cdot m)$;
- laço linha 10: $O(m)$;
- **Complexidade total:** $O(n \cdot m)$ que é maior que $O((n + m) \log n)$ (Dijkstra);

Corretude do algoritmo de Bellman-Ford

- **O problema tem subestrutura ótima:** um caminho mínimo entre dois vértices contém outros caminhos mínimos em seu interior.

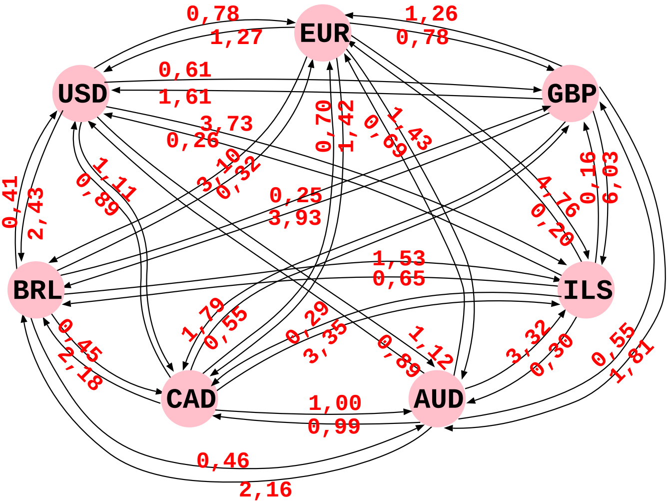
Aplicação do algoritmo de Bellman-Ford: Arbitragem

	USD	CAD	AUD	BRL	ILS	GBP	EUR
USD	**1**	1,11	1,12	2,43	3,73	0,61	0,78
CAD	0,89	**1**	1,00	2,18	3,35	0,55	0,70
AUD	0,89	0,99	**1**	2,16	3,32	0,55	0,69
BRL	0,41	0,45	0,46	**1**	1,53	0,25	0,32
ILS	0,26	0,29	0,30	0,65	**1**	0,16	0,20
GBP	1,61	1,79	1,81	3,93	6,03	**1**	1,26
EUR	1,27	1,42	1,43	3,10	4,76	0,78	**1**

USD = dólar dos EUA
CAD = dólar canadense
AUD = dólar australiano
BRL = real Brasil
ILS = shekel (Israel)
GBP = libra esterlina (Reino Unido)
EUR = euro

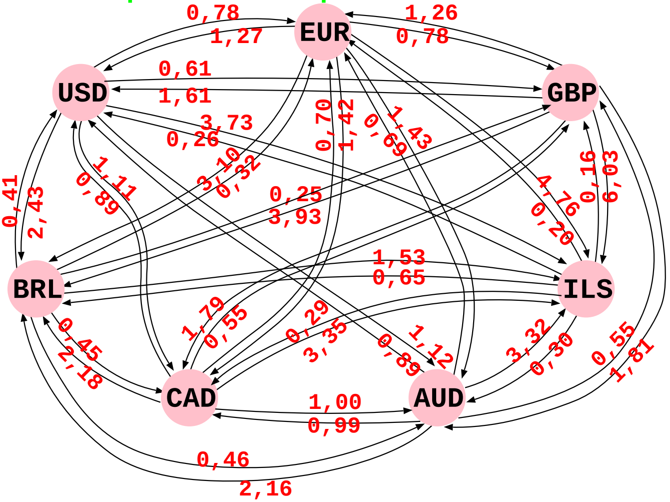
Cotação: 29/10/2014

Aplicação do algoritmo de Bellman-Ford: Arbitragem

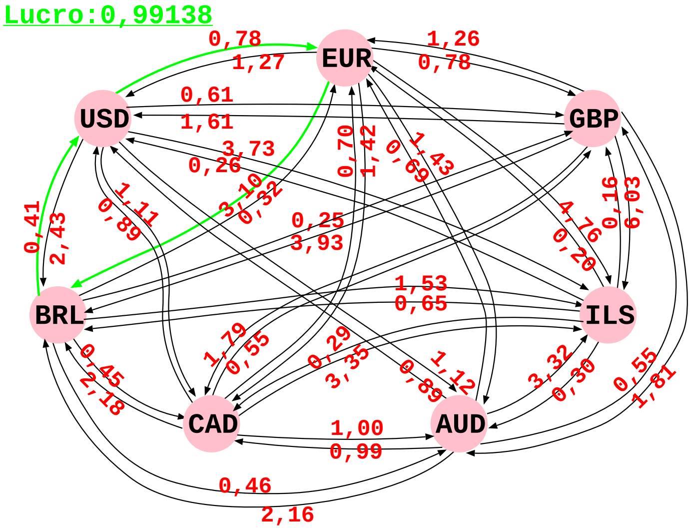


Aplicação do algoritmo de Bellman-Ford: Arbitragem

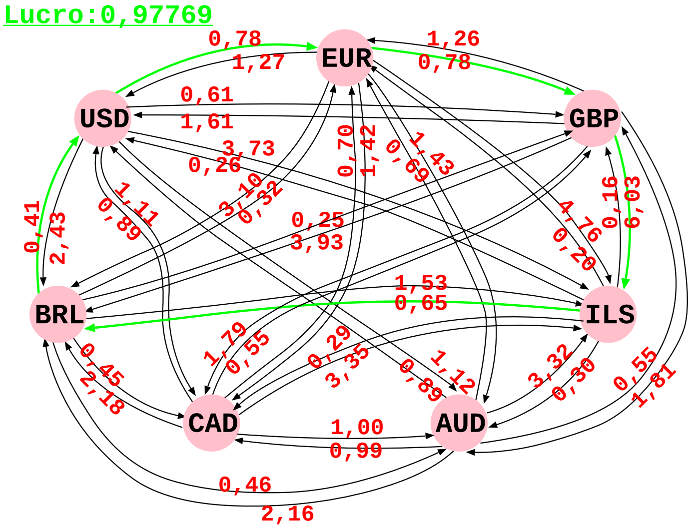
Lucro:se produto dos pesos do ciclo > 1



Aplicação do algoritmo de Bellman-Ford: Arbitragem



Aplicação do algoritmo de Bellman-Ford: Arbitragem



Aplicação do algoritmo de Bellman-Ford: Arbitragem

Observe que

Aplicação do algoritmo de Bellman-Ford: Arbitragem

Observe que

$$n_1 \times n_2 \times \ldots \times n_m > 1$$

Aplicação do algoritmo de Bellman-Ford: Arbitragem

Observe que

$$n_1 \times n_2 \times \ldots \times n_m > 1$$

se e somente se

Aplicação do algoritmo de Bellman-Ford: Arbitragem

Observe que

$$n_1 \times n_2 \times \ldots \times n_m > 1$$

se e somente se

$$\frac{1}{n_1} \times \frac{1}{n_2} \times \ldots \times \frac{1}{n_m} < 1.$$

Aplicação do algoritmo de Bellman-Ford: Arbitragem

Observe que

$$n_1 \times n_2 \times \ldots \times n_m > 1$$

se e somente se

$$\frac{1}{n_1} \times \frac{1}{n_2} \times \ldots \times \frac{1}{n_m} < 1.$$

Aplicando log nos dois lados da desigualdade:

Aplicação do algoritmo de Bellman-Ford: Arbitragem

Observe que

$$n_1 \times n_2 \times \ldots \times n_m > 1$$

se e somente se

$$\frac{1}{n_1} \times \frac{1}{n_2} \times \ldots \times \frac{1}{n_m} < 1.$$

Aplicando log nos dois lados da desigualdade:

$$\log \frac{1}{n_1} \times \log \frac{1}{n_2} \times \ldots \times \log \frac{1}{n_m} < 0$$

Aplicação do algoritmo de Bellman-Ford: Arbitragem

Observe que

$$n_1 \times n_2 \times \ldots \times n_m > 1$$

se e somente se

$$\frac{1}{n_1} \times \frac{1}{n_2} \times \ldots \times \frac{1}{n_m} < 1.$$

Aplicando log nos dois lados da desigualdade:

$$\log \frac{1}{n_1} \times \log \frac{1}{n_2} \times \ldots \times \log \frac{1}{n_m} < 0$$

que é o mesmo que

$$-\log n_1 \times -\log n_2 \times \ldots \times -\log n_m < 0.$$

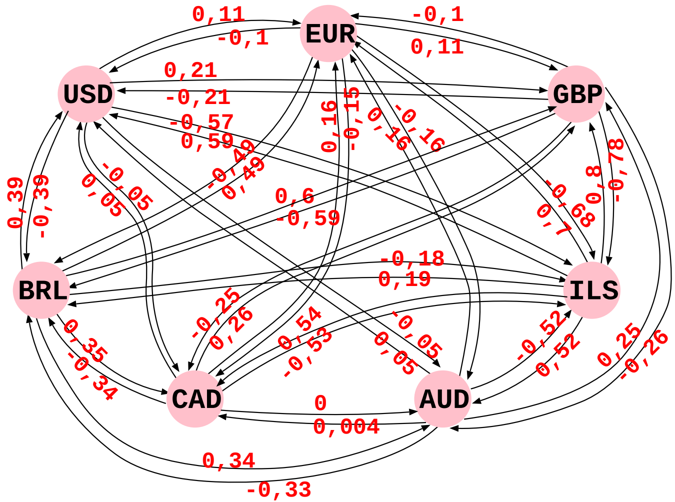
Aplicação do algoritmo de Bellman-Ford: Arbitragem

Substituindo os pesos das arestas:

	USD	CAD	AUD	BRL	ILS	GBP	EUR
USD	**0**	-0,05	-0,05	-0,39	-0,57	0,21	0,11
CAD	0,05	**0**	0	-0,34	-0,53	0,26	0,15
AUD	0,05	0,00	**0**	-0,33	-0,52	0,26	0,16
BRL	0,39	0,35	0,34	**0**	-0,18	0,6	0,49
ILS	0,59	0,54	0,52	0,19	**0**	0,80	0,7
GBP	-0,21	-0,25	-0,26	-0,59	-0,78	**0**	-0,1
EUR	-0,1	-0,15	-0,16	-0,49	-0,68	0,11	**0**

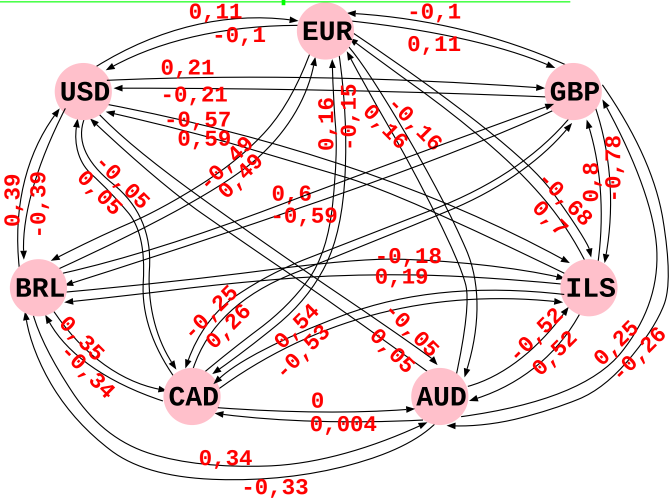
USD = dólar dos EUA
CAD = dólar canadense
AUD = dólar australiano
BRL = real Brasil
ILS = shekel (Israel)
GBP = libra esterlina (Reino Unido)
EUR = euro

Aplicação do algoritmo de Bellman-Ford: Arbitragem

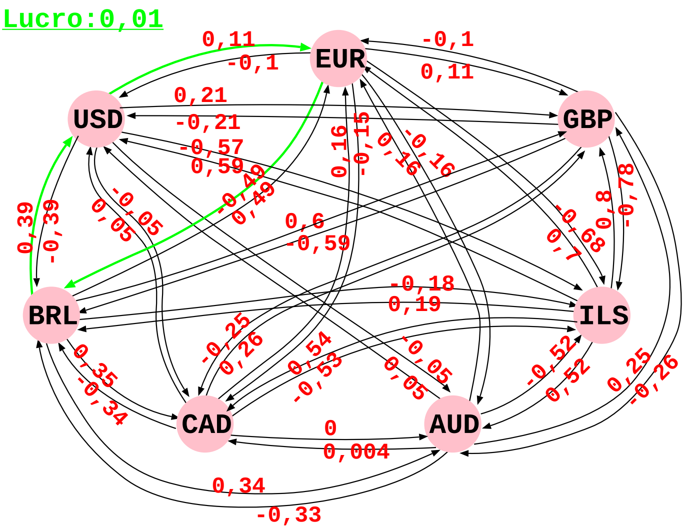


Aplicação do algoritmo de Bellman-Ford: Arbitragem

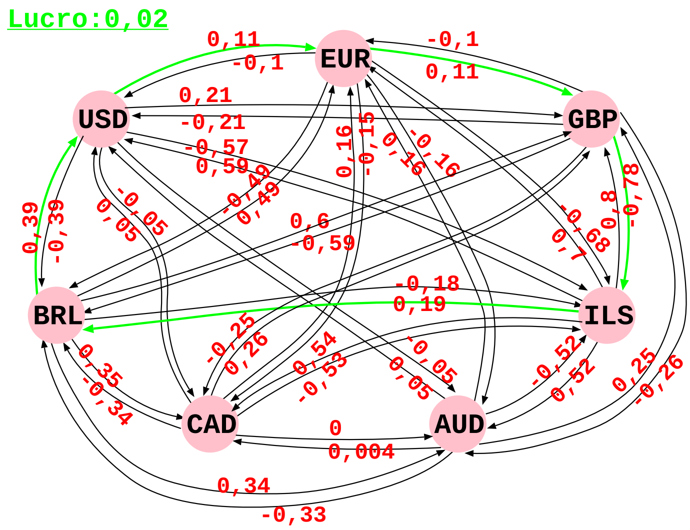
<u>Lucro:se soma dos pesos do ciclo < 0</u>



Aplicação do algoritmo de Bellman-Ford: Arbitragem



Aplicação do algoritmo de Bellman-Ford: Arbitragem



Aplicação do algoritmo de Bellman-Ford: Arbitragem

Resolução pelo algoritmo de Bellman-Ford:

Aplicação do algoritmo de Bellman-Ford: Arbitragem

Resolução pelo algoritmo de Bellman-Ford:

- adicione um vértice v_0 com uma aresta de peso 0 para cada vértice;

Aplicação do algoritmo de Bellman-Ford: Arbitragem

Resolução pelo algoritmo de Bellman-Ford:

- adicione um vértice v_0 com uma aresta de peso 0 para cada vértice;
- a adição de v_0 não pode criar novos ciclos;

Aplicação do algoritmo de Bellman-Ford: Arbitragem

Resolução pelo algoritmo de Bellman-Ford:

- adicione um vértice v_0 com uma aresta de peso 0 para cada vértice;
- a adição de v_0 não pode criar novos ciclos;
- todos os ciclos do grafo são acessíveis a partir de v_0;

Bibliografia Utilizada

CORMEN, T. H.; LEISERSON, C. E.; RIVEST, R. L. e STEIN, C. *Introduction to Algorithms*, 3ª edição, MIT Press, 2009.