INSTITUTO FEDERAL
Sudeste de Minas Gerais
Campus Manhuaçu

Estruturas de Dados II
Algoritmo de Bellman-Ford

Prof. Leonardo C. R. Soares - *leonardo.soares@ifsudestemg.edu.br*
Instituto Federal do Sudeste de Minas Gerais

6 de setembro de 2024

[a] Este material é fortemente baseado nas notas de aula do professor Marco Antonio Moreira de Carvalho - UFOP

Algoritmo de Bellman-Ford

Arestas de peso negativo

Para além das distâncias geográficas, caminhos mais curtos podem modelar diversas outras situações reais, incluindo aquelas que para serem modeladas necessitam de arestas cujo peso é negativo:

- Movimentações financeiras, nas quais é possível obter lucro ou prejuízo, principalmente quando há utilização de câmbio;
- Um taxista que recebe mais dinheiro do que gasta com combustível a cada viagem: se o táxi roda vazio, ele gasta mais do que recebe;
- Um entregador que necessita atravessar um pedágio e pode acabar pagando mais do que recebe para entregar encomendas;
- A energia gerada e consumida durante uma reação química.

Algoritmo de Bellman-Ford

Ford-Moore-Bellman?

Alguns autores denominam o algoritmo como Algoritmo de *Ford-Moore-Bellman* em homenagem aos três autores que propuseram o mesmo algoritmo em anos diferentes:

- Lester Ford (1956);
- Edward Moore (1957);
- Richard Bellman (1958).

Algoritmo de Bellman-Ford

Princípio

Ao invés de fechar um vértice por iteração, como o algoritmo de Dijkstra, o algoritmo de Bellman-Ford examina todos os vértices de um grafo **orientado** por iteração até que atualizações não sejam possíveis.

Em um grafo com n vértices, qualquer caminho possui no máximo $n - 1$ arestas, portanto, cada vértice é examinado no máximo $n - 1$ vezes.

Com esta estratégia, é possível calcular caminhos mínimos em grafos com arestas de **peso negativo**.

Assim como o algoritmo de Dijkstra, baseia-se no princípio de relaxação: uma aproximação da distância da origem até cada vértice é gradualmente atualizada por valores mais precisos até que a solução ótima seja atingida.

Algoritmo de Bellman-Ford

Princípio

Se, em alguma iteração do algoritmo os caminhos até cada um dos vértices permanecerem inalterados, não haverá atualizações nas próximas iterações e o algoritmo pode terminar.

Algoritmo de Bellman-Ford

Princípio

Se, em alguma iteração do algoritmo os caminhos até cada um dos vértices permanecerem inalterados, não haverá atualizações nas próximas iterações e o algoritmo pode terminar.

Entretanto, se houver atualizações na última iteração do algoritmo, é sinal de que há pelo menos um ciclo negativo no grafo, dado que algum caminho terá n arestas ou mais. Para este caso, o algoritmo não será capaz de retornar o caminho mínimo, a maioria das implementações retorna *false* para este caso e *true* caso contrário.

Algoritmo de Bellman-Ford

Terminologia

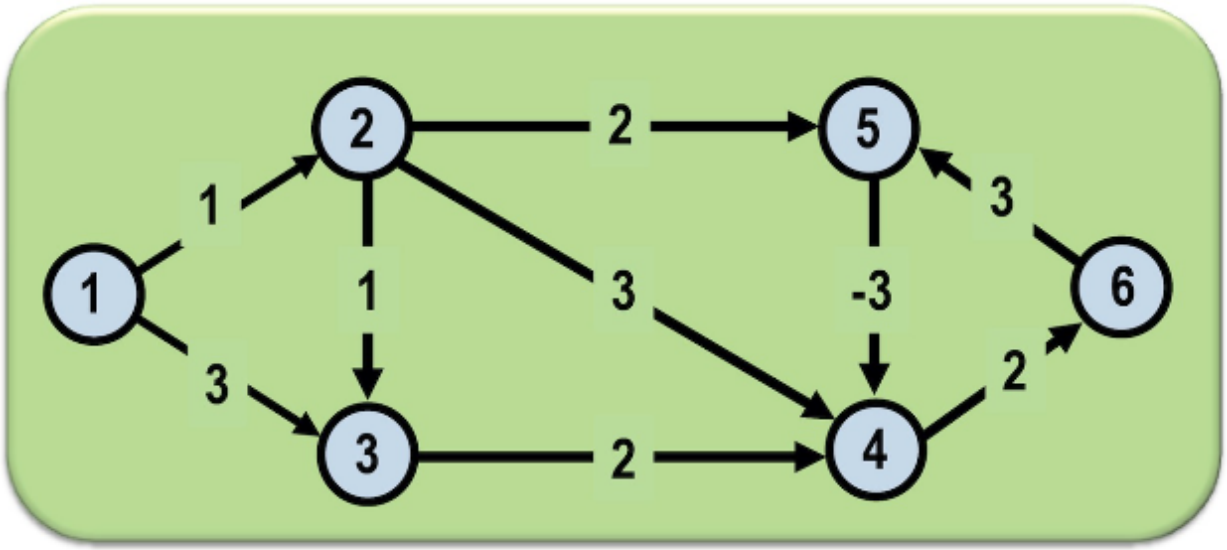
- $\Gamma^{-}(i)$: Conjunto de vértices antecessores do vértice atual;
- $dt[i]$: Vetor que armazena a distância entre o vértice de origem e o vértice i;
- $rot[i]$: Vetor que armazena o índice do vértice anterior ao vértice i, no caminho cuja distância está armazenada em $dt[i]$;
- $altera$: Variável *booleana* que indica se houve alguma atualização na iteração atual.

Algoritmo de Bellman-Ford

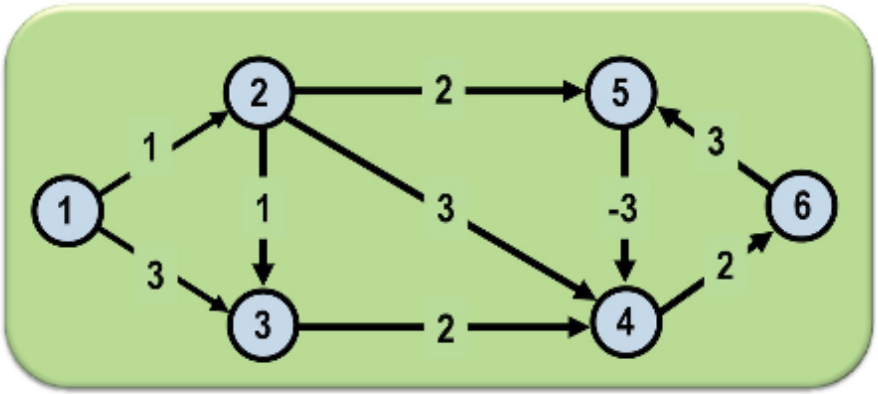
Entrada: Grafo $G = (V, E)$ e matriz de pesos D=$\{d_{ij}\}$ para todos os arcos (i, j)

```
1  dt[1] ← 0; rot[1] ← ∞; //considerando o vértice 1 como o inicial
2  para i ← 2 até n faça
3  |   se ∃ (1, i) ∈ E então rot[i] ← 1; dt[i] ← d_1i;
4  |   senão rot[i] ← 0; dt[i] ← ∞;
5  fim
6  para k ← 1 até n-1 faça
7  |   altera ← falso;
8  |   para i ← 2 até n faça
9  |   |   para j ∈ Γ⁻(i) faça
10 |   |   |   se dt[i] > dt[j]+d_ji então
11 |   |   |   |   dt[i] ← dt[j]+d_ji;
12 |   |   |   |   rot[i] ← j;
13 |   |   |   |   altera ← verdadeiro; //indica que houve alteração
14 |   |   |   fim
15 |   |   fim
16 |   fim
17 |   se altera = falso então  k ← n ;
18 fim
```

Exemplo



Exemplo



dt

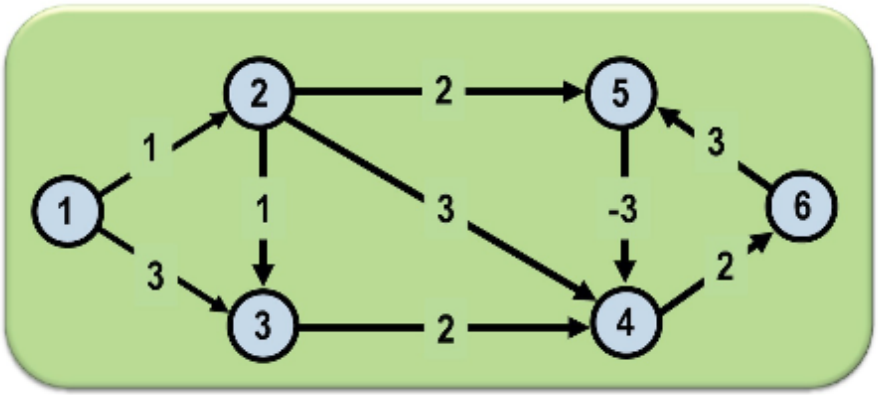
2	3	4	5	6
1	3	∞	∞	∞

rot

2	3	4	5	6
1	1	0	0	0

Vetores após a inicialização do algoritmo.

Exemplo



i=2, $\Gamma^-(i)$={1};

- j=1, $dt[1]+d_{12} = 1$

dt

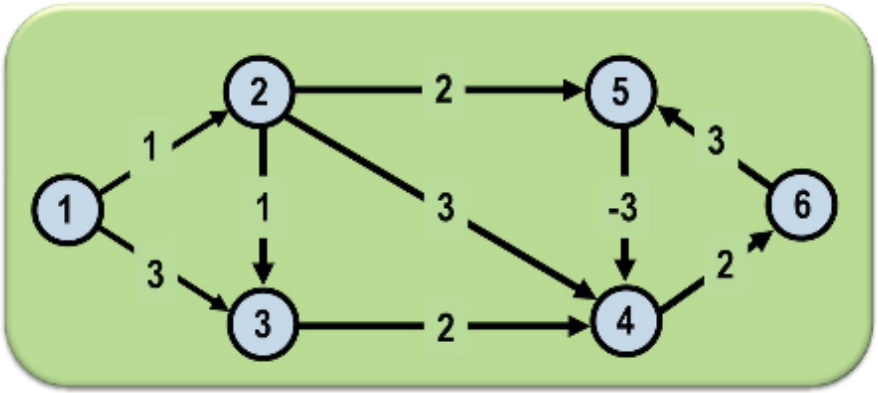
2	3	4	5	6
1	3	∞	∞	∞

rot

2	3	4	5	6
1	1	0	0	0

Iteração k=1 (continua...)

Exemplo



i=3, $\Gamma^{-}(i)$={1, 2};

- j=1, $dt[1]+d_{13} = 3$
- j=2, $dt[2]+d_{23} = 2$

dt

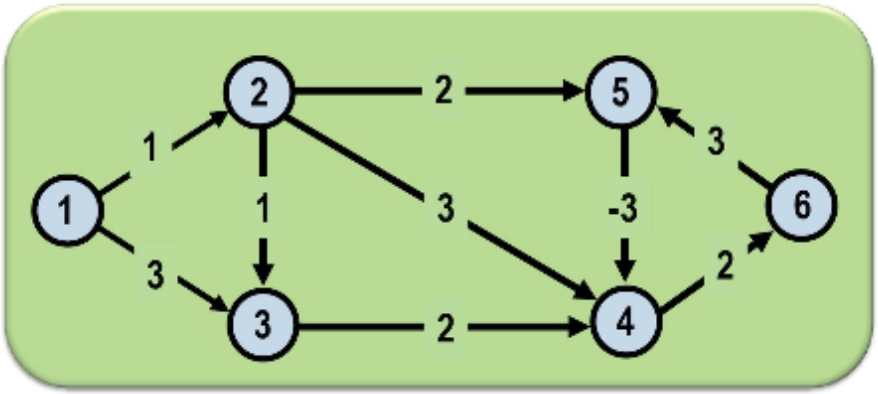
2	3	4	5	6
1	2	∞	∞	∞

rot

2	3	4	5	6
1	2	0	0	0

Iteração k=1 (continua...)

Exemplo



i=4, $\Gamma^-(i)$={2, 3, 5};

- j=2, $dt[2]+d_{24} = 4$
- j=3, $dt[3]+d_{34} = 4$
- j=5, $dt[5]+d_{54} = \infty$

dt

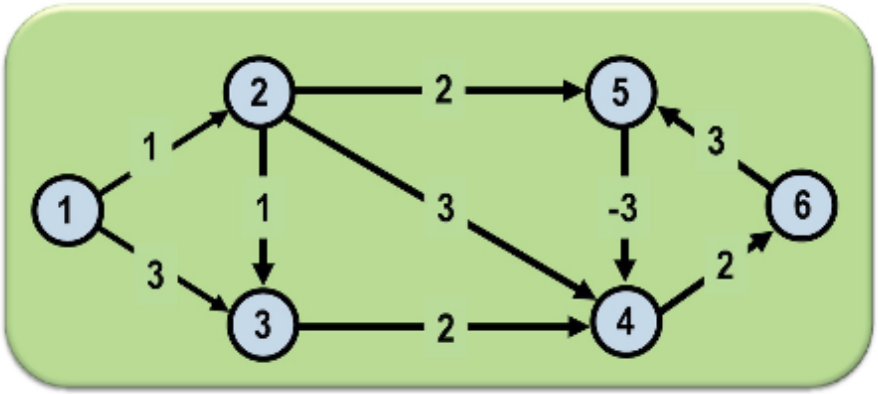
2	3	4	5	6
1	2	4	∞	∞

rot

2	3	4	5	6
1	2	2	0	0

Iteração k=1 (continua...)

Exemplo



i=5, $\Gamma^-(i)$={2, 6};

- j=2, $dt[2]+d_{25} = 3$
- j=6, $dt[6]+d_{65} = \infty$

dt

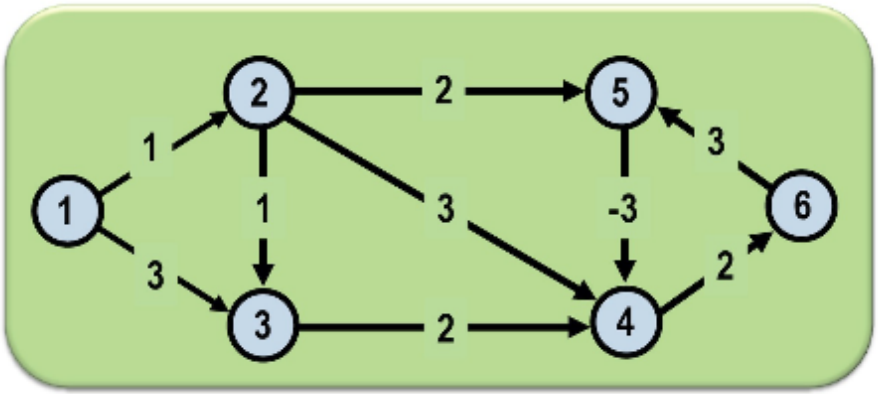
2	3	4	5	6
1	2	4	3	∞

rot

2	3	4	5	6
1	2	2	2	0

Iteração k=1 (continua...)

Exemplo



i=6, $\Gamma^-(i)$={4};

- j=4, $dt[4]+d_{46} = 6$

dt

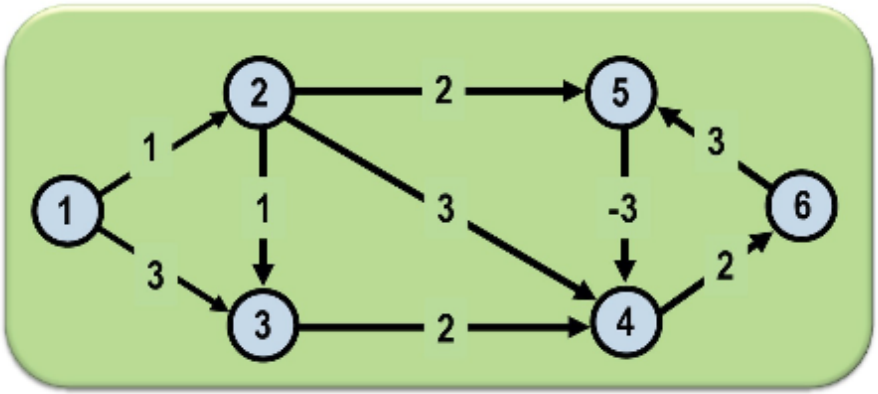
2	3	4	5	6
1	2	4	3	6

rot

2	3	4	5	6
1	2	2	2	4

Iteração k=1 (final)

Exemplo



i=2, $\Gamma^-(i)$={1};

- j=1, $dt[1]+d_{12} = 1$

dt

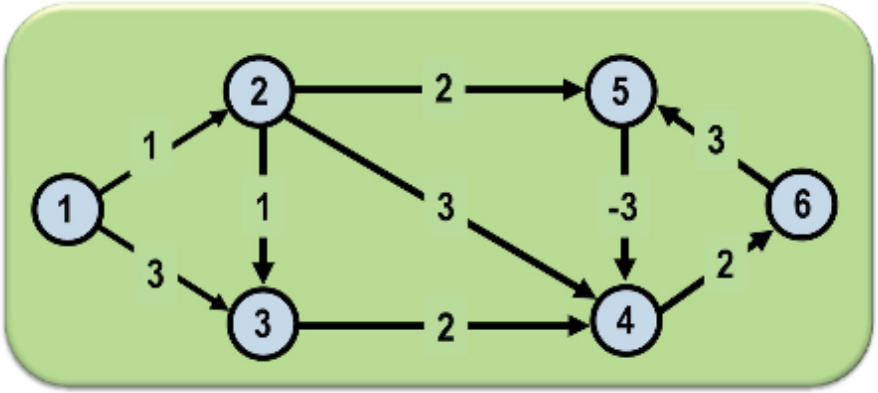
2	3	4	5	6
1	2	4	3	6

rot

2	3	4	5	6
1	2	2	2	4

Iteração k=2 (continua...)

Exemplo



i=3, $\Gamma^{-}(i)$={1, 2};

- j=1, $dt[1]+d_{13} = 3$
- j=2, $dt[2]+d_{23} = 2$

dt

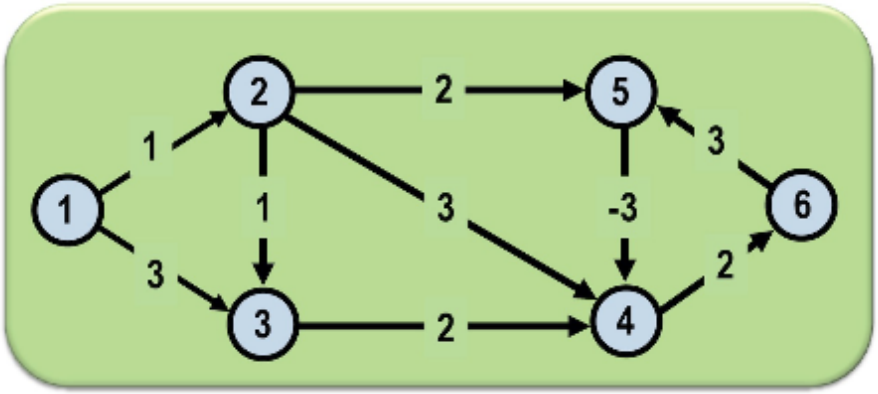
2	3	4	5	6
1	2	4	3	6

rot

2	3	4	5	6
1	2	2	2	4

Iteração k=2 (continua...)

Exemplo



i=4, $\Gamma^-(i)$={2, 3, 5};

- j=2, $dt[2]+d_{24} = 4$
- j=3, $dt[3]+d_{34} = 4$
- j=5, $dt[5]+d_{54} = 0$

dt

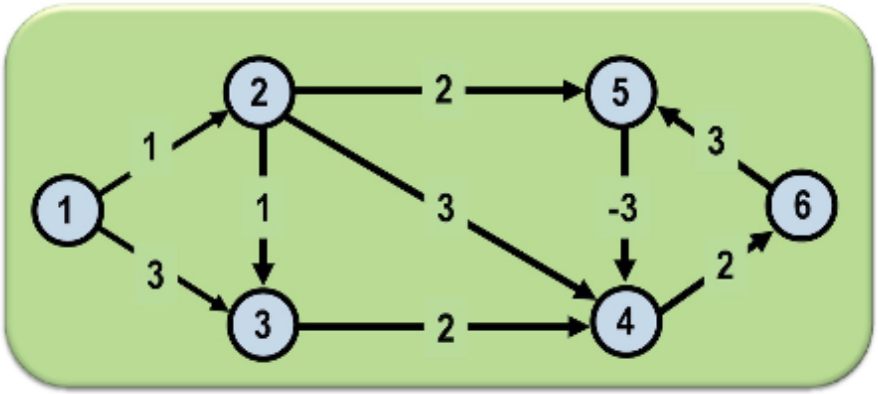
2	3	4	5	6
1	2	0	3	6

rot

2	3	4	5	6
1	2	5	2	4

Iteração k=2 (continua...)

Exemplo



i=5, $\Gamma^-(i)$={2, 6};

- j=2, $dt[2]+d_{25} = 3$
- j=6, $dt[6]+d_{65} = 9$

dt

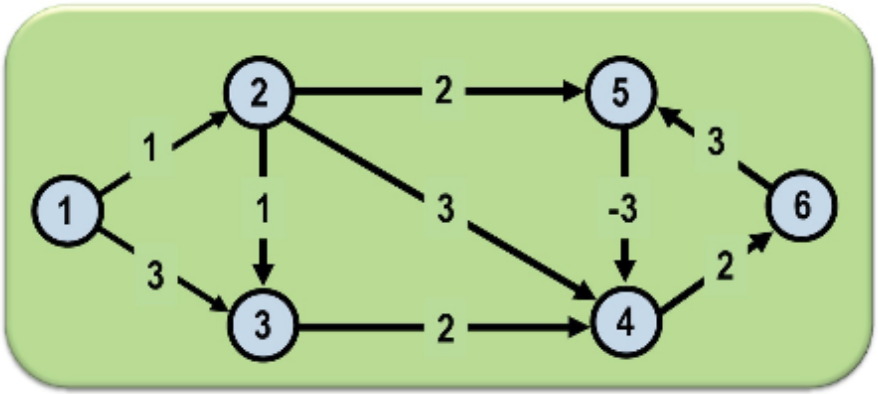
2	3	4	5	6
1	2	0	3	6

rot

2	3	4	5	6
1	2	5	2	4

Iteração k=2 (continua...)

Exemplo



i=6, $\Gamma^-(i)$={4};

- j=4, $dt[4]+d_{46} = 2$

dt

2	3	4	5	6
1	2	0	3	2

rot

2	3	4	5	6
1	2	5	2	4

Iteração k=2 (final)

Exemplo

Final

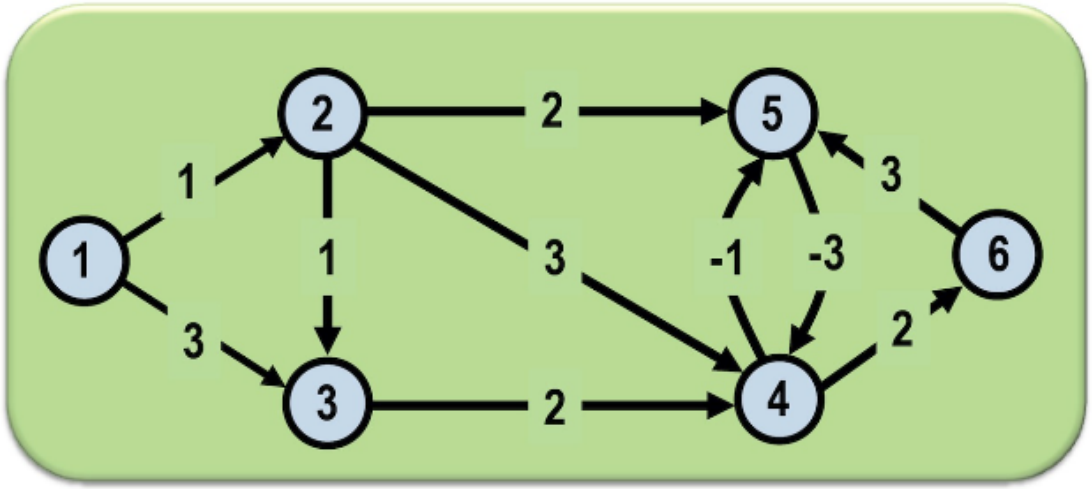
Na próxima iteração, em que $k = 3$, nenhuma alteração é realizada e o algoritmo se encerra indicando que foi possível encontrar o caminho mínimo.

Ciclos de custo negativo

Bellman-Ford - Detecção

Em caminhos sem ciclos, o caminho mais longo consiste em $n - 1$ arestas, ou iterações no laço principal do algoritmo.

Caso ocorra alguma atualização de distância na iteração n do algoritmo, é detectado um ciclo de peso negativo.



Exemplo de ciclo negativo entre os vértices 4 e 5.

Bellman-Ford

Complexidade

Em uma implementação simples, o laço principal é repetido no máximo $n-1$ vezes. A cada iteração, são calculados caminhos com k arestas entre a origem o os demais vértices. Para cada um dos $n-1$ vértices, todos os seus antecessores são examinados. O vértice original não é atualizado, logo $n-2$ antecessores são analisados no máximo. Logo, a complexidade é limitada por $\mathcal{O}(n^3)$.

Bellman-Ford

Complexidade

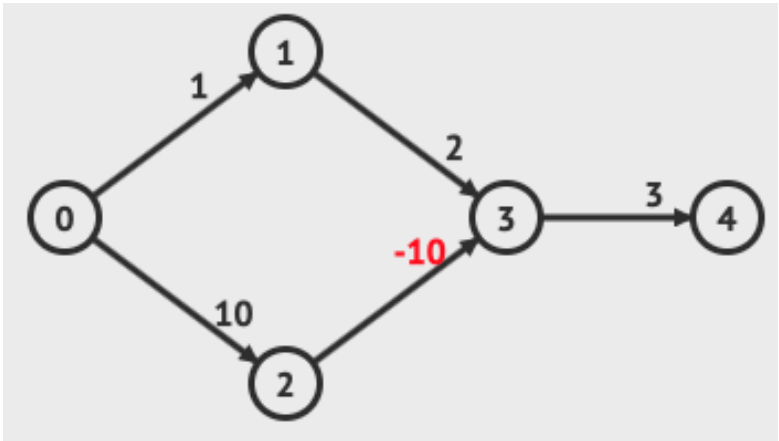
Em uma implementação simples, o laço principal é repetido no máximo $n-1$ vezes. A cada iteração, são calculados caminhos com k arestas entre a origem o os demais vértices. Para cada um dos $n-1$ vértices, todos os seus antecessores são examinados. O vértice original não é atualizado, logo $n-2$ antecessores são analisados no máximo. Logo, a complexidade é limitada por $\mathcal{O}(n^3)$.

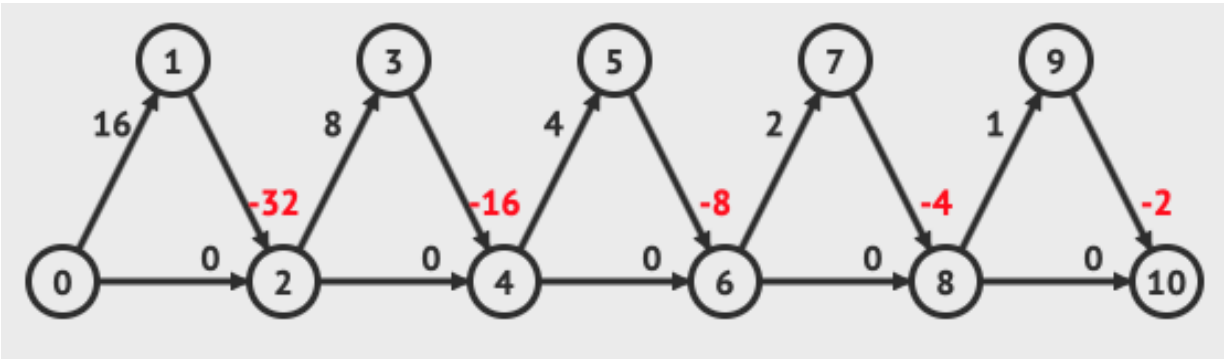
Em 1970, Jin Yen[a] propôs uma implementação deste método de complexidade $\mathcal{O}(nm)$.

[a] Yen, Jin Y. (1970). "An algorithm for finding shortest routes from all source nodes to a given destination in general networks". Quarterly of Applied Mathematics 27: 526–530.

Exercício

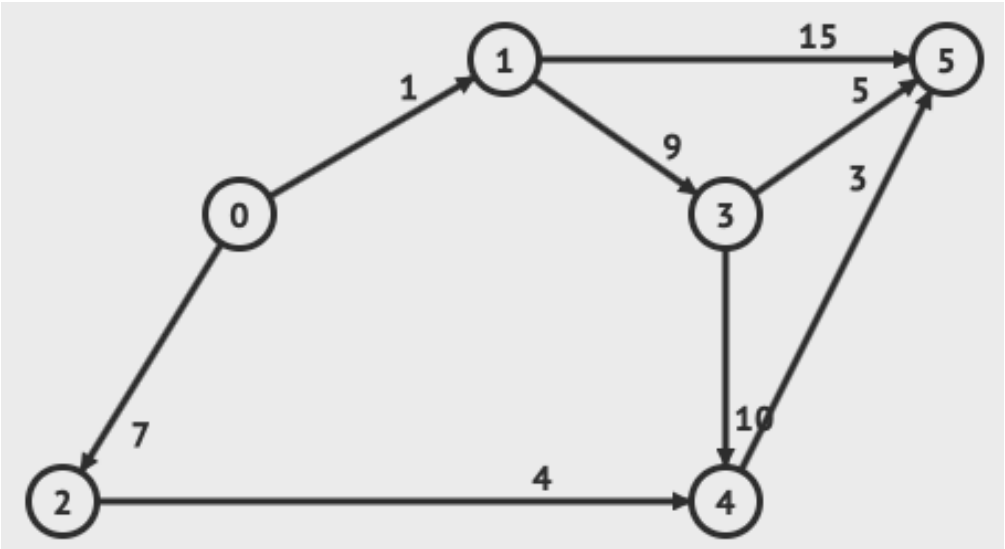
Execute o algoritmo de Bellman-Ford para o grafo abaixo. Considere o vértice de saída como 0:





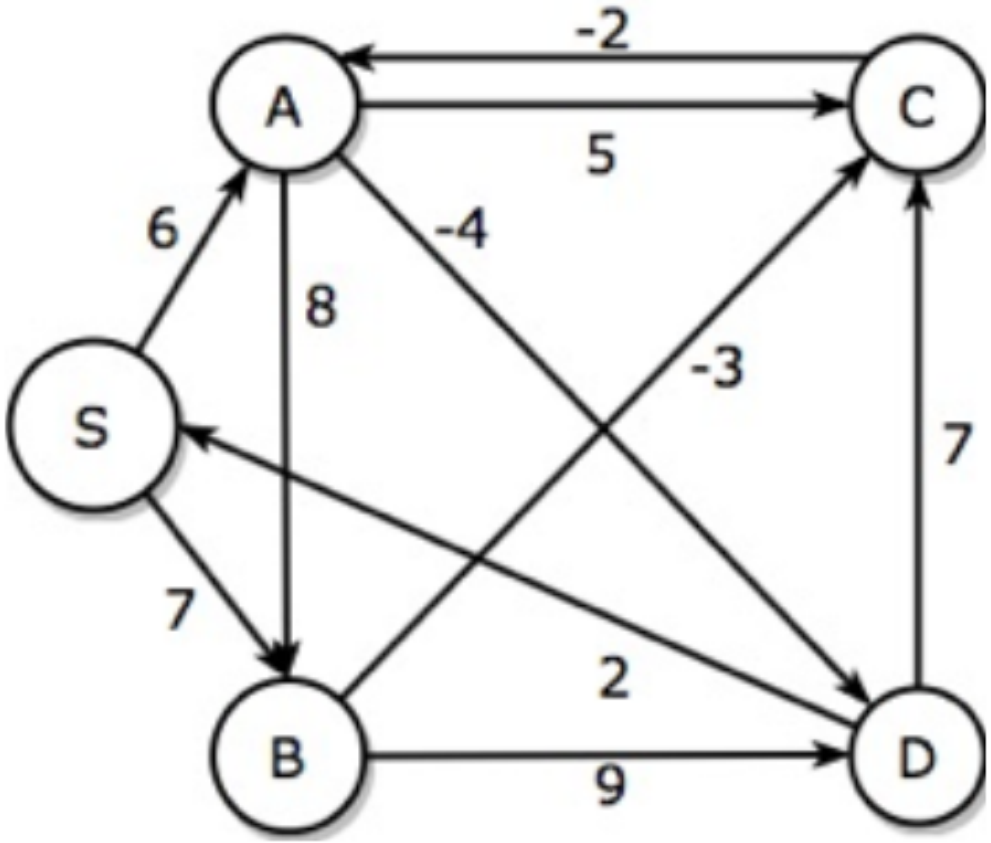
Exercício

Execute o algoritmo de Bellman-Ford para o grafo abaixo. Considere o vértice de saída como 0:



Exercício

Execute o algoritmo de Bellman-Ford para o grafo abaixo. Considere o vértice de saída como S:




thank you
danke
рахмат
謝謝
ngiyabonga
teşekkür ederim
gracias
Баярлалаа
спасибо
dank je
merci
obrigado
bedankt
dziękuję
sukriya
kop khun krap
go raibh maith agat
mochchakkeram
arigatô
dakujem
grazie
terima kasih
감사합니다
ありがとう
ευχαριστώ
мерси
tapadh leat
takk
taiku
sagolun
hvala
mauruuru
faafetai lava
kiitos
dankie
dhanyavad
nanni
nandri
enkosi
bayarlalaa
gracie
koszonom
blagodaram
vinaka
mersi
barka
welalin
tack
misaotra
matondo
paldies
grazzi
mahalo
хвала
asante
manana
obrigada
tenki
murakoze
chokrane
mamnun
дякую
djiere dieuf
tau
chnorakaloutioun
gratias ago
gracies
sulpay
trugarez
shukriya
merce
dhanyavadagalu
diolch
tanemirt
rahmet
xiexie
najis tuke
rahmat
kam sah hamnida
didi madloba
mèsi
dėkuji
sobodi
akun
dankon
spas