Grafos
Caminho mais curto

Algoritmo de Dijkstra
Bellman-Ford

Leonardo Vieira, Luís P. Ribeiro, Gerson Barbosa, Carlos Romani, Phyllipe Lima,
Paulo H. Barchi, Pedro A. Santos

Visão Geral

- Grafos - Caminho mais curto
- Dijkstra
  - Introdução
  - Aplicações
  - Algoritmo - Análise
  - Exemplos
  - Execuções
- Bellman-Ford
  - Introdução
  - Aplicações
  - Algoritmo - Análise
  - Exemplos

Grafos - Caminho mais curto

- Arestas com peso ou custo.
- Grafo direcionado → Caminhos direcionados.
- Pesos não são necessariamente distâncias.
- Nem todos os vértices são (necessariamente) atingíveis.
- Pesos negativos introduzem complicações.
- Caminhos mais curtos normalmente são simples.
- Caminhos mais curtos não são necessariamente únicos.
- Arestas paralelas e auto-laços (arestas que ligam o vértice ao próprio vértice) podem estar presentes.

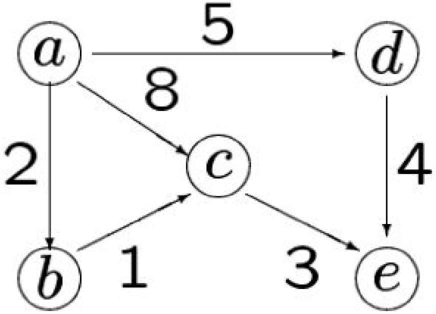
Grafos - Caminho mais curto

- Áreas de aplicação
  - Pesquisa operacional
  - Tráfego de estradas
  - Otimização de trajetos
  - Linhas de transmissão elétricas
  - Conexões de rede (transporte, comunicação)
  - Planejamento de rotas e tempo de viagens
  - Roteamento e programação de veículos
  - Substituição de equipamento

Grafos - Caminho mais curto

- Considere um dígrafo sem peso G = (V,E).
    - O caminho mais curto entre dois vértices será o de menor comprimento (ou menor número de arestas).
    - Neste problema pode se usar o BFS (Breadth-First Search). Que é um algoritmo para se encontrar o menor número de arestas de um único vértice (única fonte) até os demais.
    - BFS processa os vértices em ordem crescente a partir do vértice raiz.

Grafos - Caminho mais curto

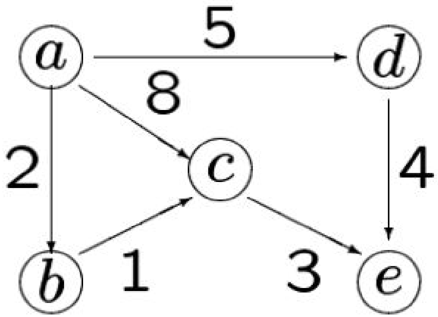
- Considere um dígrafo com peso G = (V,E).
    - Existe uma função w : E -> R, mapeando valores reais (um peso) para uma aresta.
    - Tendo a aresta e = (u,v), o seu peso é dado por w(e)
    - O comprimento de um caminho p = <v0,v1,v2,...,vn> é a soma dos pesos das arestas envolvidas
        - comprimento(p) = Somatório i = 1 até k (vi-1,vi)
    - A distância de u para v é o comprimento do menor caminho de u para v. Caso não exista caminho, será infinito
    - 

Exemplo: comprimento(<a,b,c,e>) = 6

Grafos - Caminho mais curto

- Problema do caminho mais curto de fonte única
  - Dado um dígrafo sem peso negativo, G = (V,E), e um vértice único de origem, s pertence a V, qual a distancia de s em relação a todos os outros vértices no dígrafo?
- Observações:
  - Todo subcaminho de um caminho mais curto também é um caminho mais curto!

Exemplo: Se <a,b,c,e> é um caminho mais curto, então, <a,b,c> também é um caminho mais curto



Algoritmo de Dijkstra

Algoritmo de Dijkstra - Introdução

- 1956 - Mathematical Center, Amsterdam.
- Demonstrar capacidade de um novo computador: ARMA.
- Objetivo: escolher problema e solução que leigos pudessem entender.
- Algoritmo do caminho mais curto.
- Trabalha apenas com pesos positivos
- 1957 - Novo problema:
    - Minimizar a quantidade de cabos necessária para conectar os pinos da parte traseira da máquina.
    - Solução: redescoberta do algoritmo conhecido como Algoritmo de Prim para encontrar árvore geradora mínima.
- 1959 - Publicação: “Algoritmo de Dijkstra”
    - Soluciona o problema de caminho mais curto com fonte única.

Algoritmo de Dijkstra - Introdução

Numerische Mathematik 1, 269—271 (1959)

A Note on Two Problems in Connexion with Graphs

By

E. W. Dijkstra

Algoritmo de Dijkstra - Aplicações

- Redes de computadores
- Problemas de logística
- Otimização de trajetos (problema do caixeiro-viajante)
- Sistemas de Informações Geográficas (SIG)
    - TerraNetwork
        - Boost Graph Library (BGL)
        - TerraLib
        - TerraView

Fonte: Ribeiro Filho, et. al., 2006.

Algoritmo de Dijkstra - Funcionamento

Ideia básica por trás do algoritmo:

- Gerar uma lista em ordem crescente de distância até o vértice de origem s.
- Construir uma árvore de caminho mínimo aresta por aresta. A cada passo uma nova aresta é adicionada, que corresponde a construção do caminho mais curto em relação ao novo vértice.

Algoritmo de Dijkstra - Funcionamento

Ideia concreta por trás do algoritmo:

- Manter e estimar uma distância d[v] do comprimento de s até v. Para cada vértice no dígrafo.
- Inicialmente d[s] = 0 e os demais tem valor infinito. A partir de agora o algoritmo processará os vértices em alguma ordem.
- Processar o vértice u: Achar novos caminhos e atualizar d[v] para todo v adjacente a u. Este processo se chama relaxamento.

Algoritmo de Dijkstra - Funcionamento

Ideia concreta por trás do algoritmo:

Pergunta 1: Como achar novos caminhos e fazer o processo de atualização?

Pergunta 2: Em que ordem se deve processar os vértices?

Algoritmo de Dijkstra - Funcionamento

Resposta para a pergunta 1:

Novos caminhos: Dado que um vértice u está sendo processado, todos os vértices adjacentes,v, serão examinados. Para cada vértice v, um caminho é adicionado entre s(origem) e v. Ou seja, caminho de s para u mais uma nova aresta.

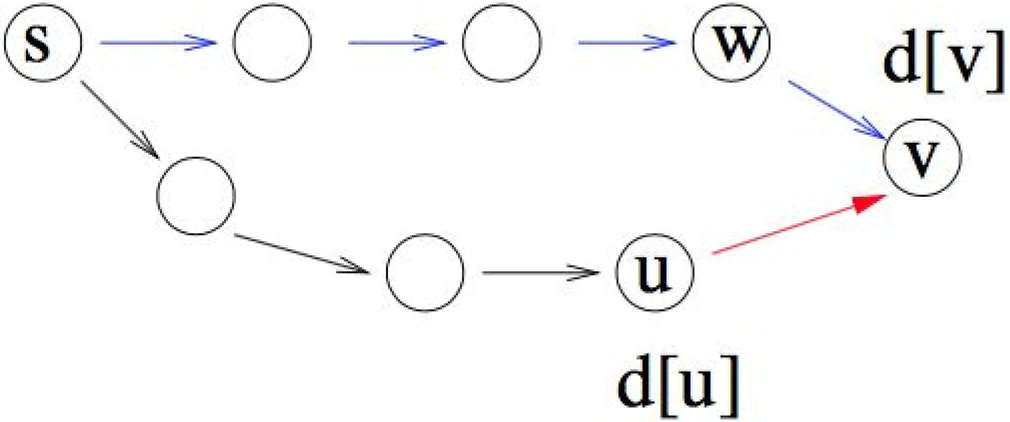
Relaxamento: Se a distância do novo caminho de s para v é menor que d[v], então atualize d[v] com essa nova distância.

Algoritmo de Dijkstra - Funcionamento

Relaxamento: Considere uma aresta e = (u,v) e o peso w(e). u já foi processado, então d[u] já é sabido, e existe uma estimativa de d[v].

novo_d[v] = d[u] + w(e). Se novo_d[v] < d[v], então substitui d[v].

pred[v] = u



Algoritmo de Dijkstra - Funcionamento

Algoritmo de relaxamento:

```
F(u,v){
    if(d[u] + w(u,v) < d[v]){
        d[v] = d[u] + w(u,v);
        pred[v] = u;
    }
}
```

Algoritmo de Dijkstra - Funcionamento

Resposta para a pergunta 2: Utiliza-se uma fila de prioridade para decidir qual o próximo vértice a ser processado:

Na fila é salvo o par (vértice,chave), onde a chave é d[v] (distância de s até v). O vértice com menor valor de chave é o próximo a ser processado.

Algoritmo de Dijkstra - Análise

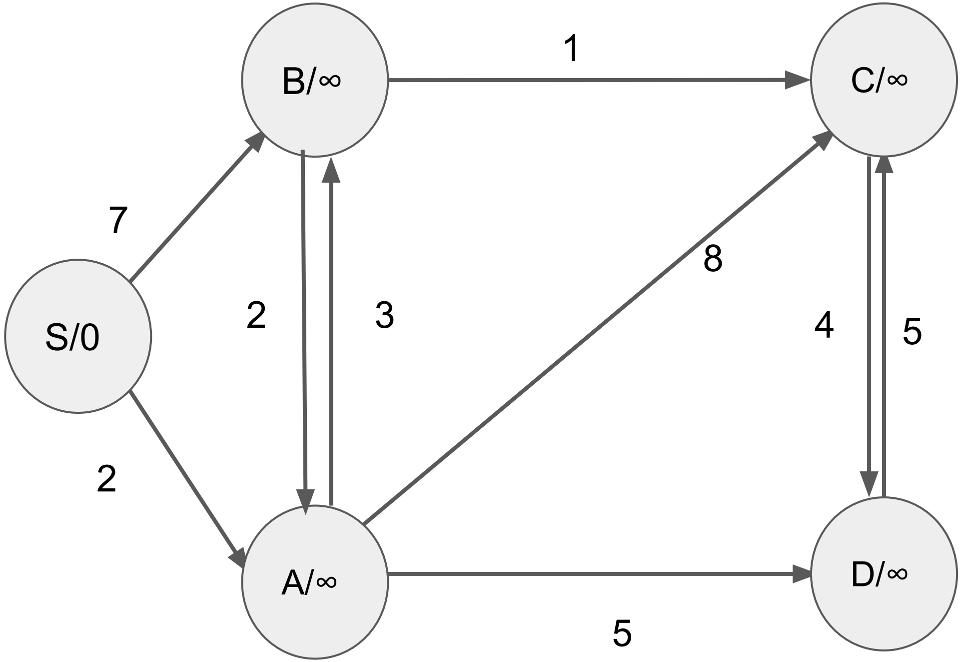
```
DIJKSTRA(G, w, s)
    INITIALIZE-SINGLE-SOURCE(G, s)
    S = Ø
    Q = G.V
    while Q ≠ Ø
        u = EXTRACT-MIN(Q)
        S = S ∪ {u}
        for each vertex v ∈ G.Adj[u]
            RELAX(u, v, w)
```

O(V)

O(|A|log(V))

O(|V|+|A|log(V))

Algoritmo de Dijkstra - Exemplo

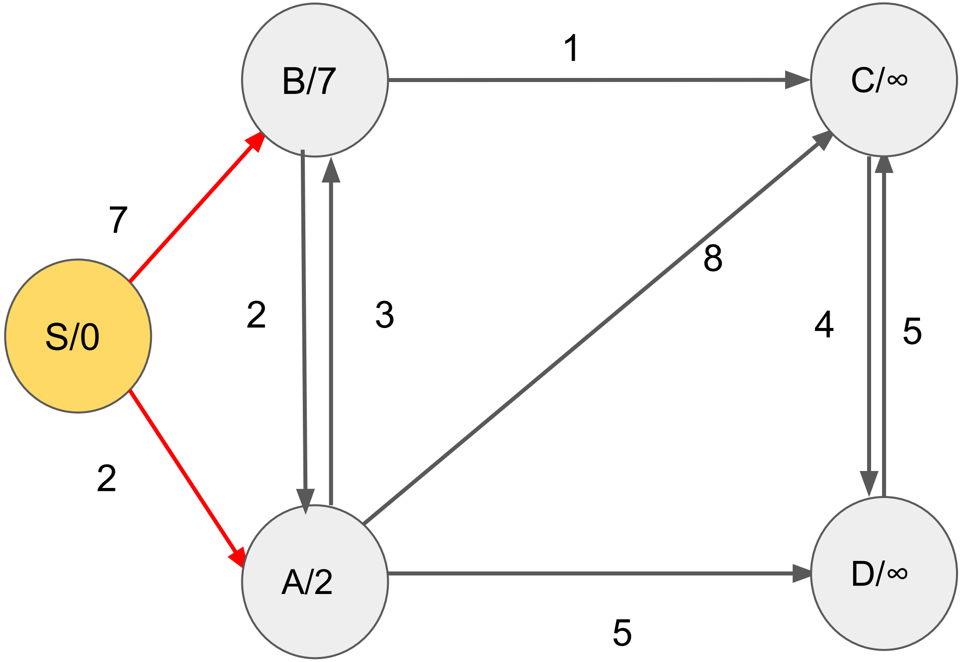


Passo 1: Inicialização

Vértices	S	A	B	C	D
Distância	0	∞	∞	∞	∞
Precedente	nil	nil	nil	nil	nil

Fila	S	A	B	C	D
Chave	0	∞	∞	∞	∞

Algoritmo de Dijkstra - Exemplo

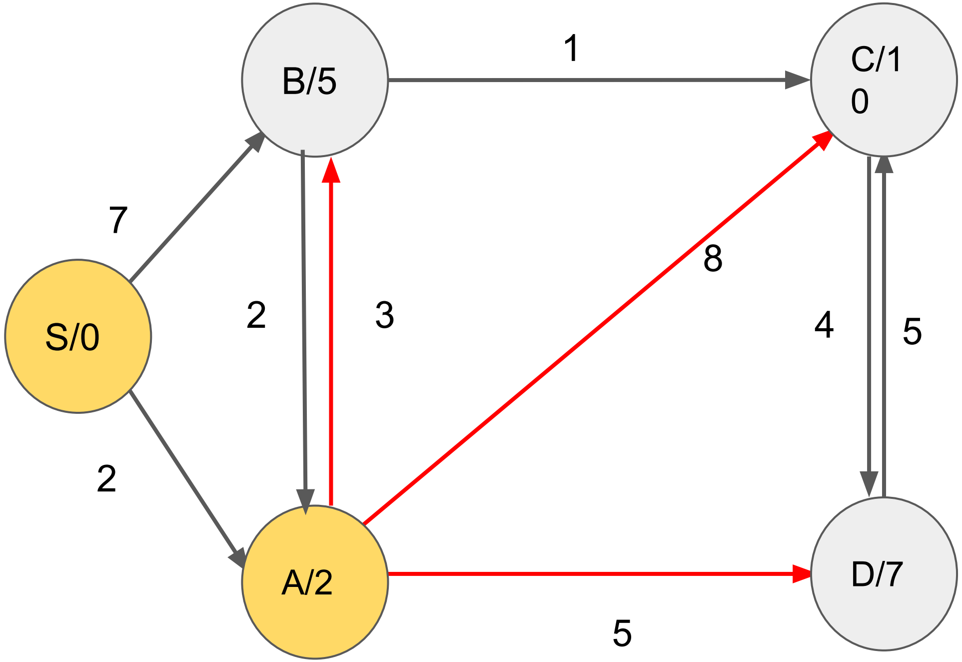


Passo 2: Adj(S) = {A,B}

Vértices	S	A	B	C	D
Distância	0	2	7	∞	∞
Precedente	nil	s	s	nil	nil

Fila	A	B	C	D
Chave	2	7	∞	∞

Algoritmo de Dijkstra - Exemplo

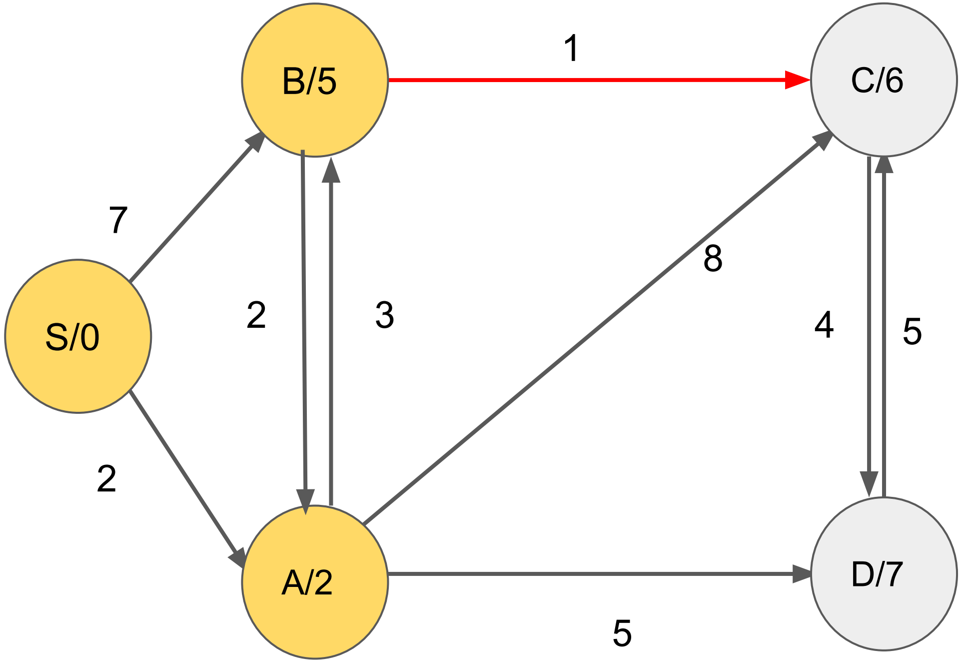


Passo 3: Adj(A) = {B,C,D}

Vértices	S	A	B	C	D
Distância	0	2	5	10	7
Precedente	nil	S	A	A	A

Fila	B	C	D
Chave	5	10	7

Algoritmo de Dijkstra - Exemplo

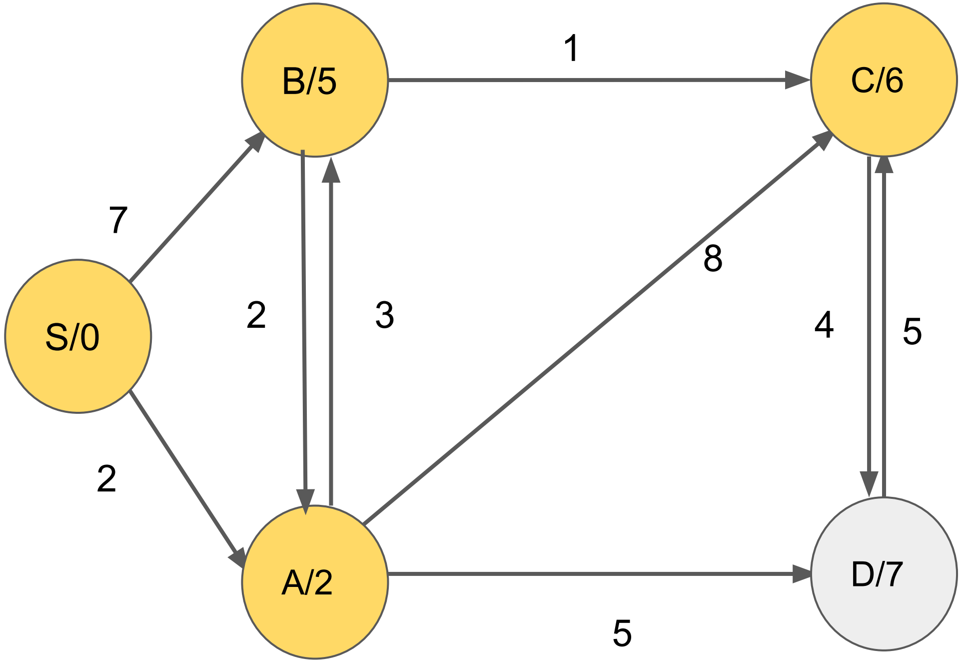


Passo 4: Adj(B) = {A,C}

Vértices	S	A	B	C	D
Distância	0	2	5	6	7
Precedente	nil	S	A	B	A

Fila	C	D
Chave	6	7

Algoritmo de Dijkstra - Exemplo

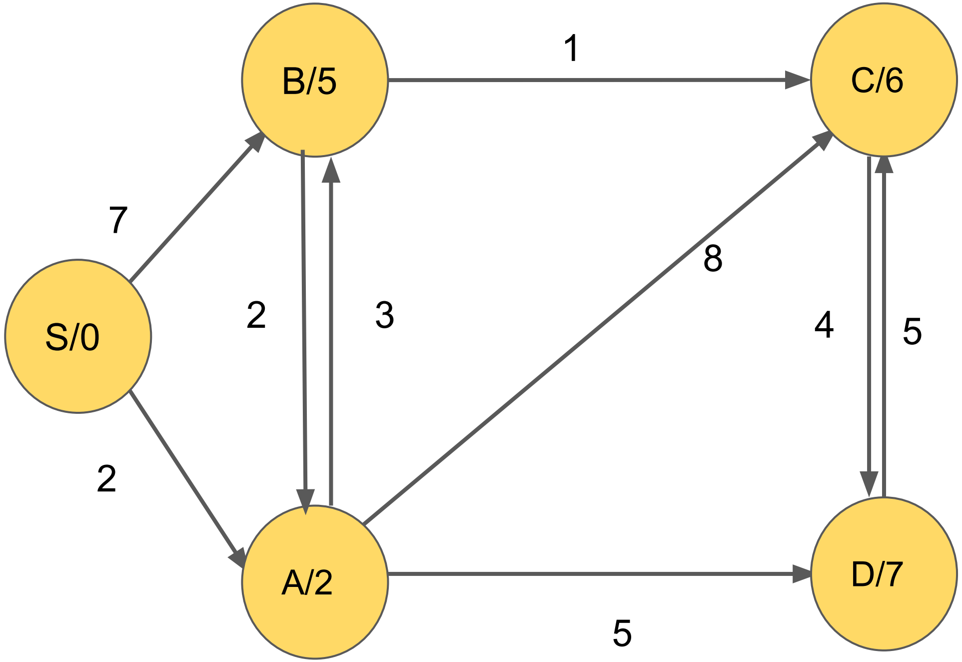


Passo 5: Adj(C) = {D}

Vértices	S	A	B	C	D
Distância	0	2	5	6	7
Precedente	nil	S	A	B	A

Fila	D
Chave	7

Algoritmo de Dijkstra - Exemplo

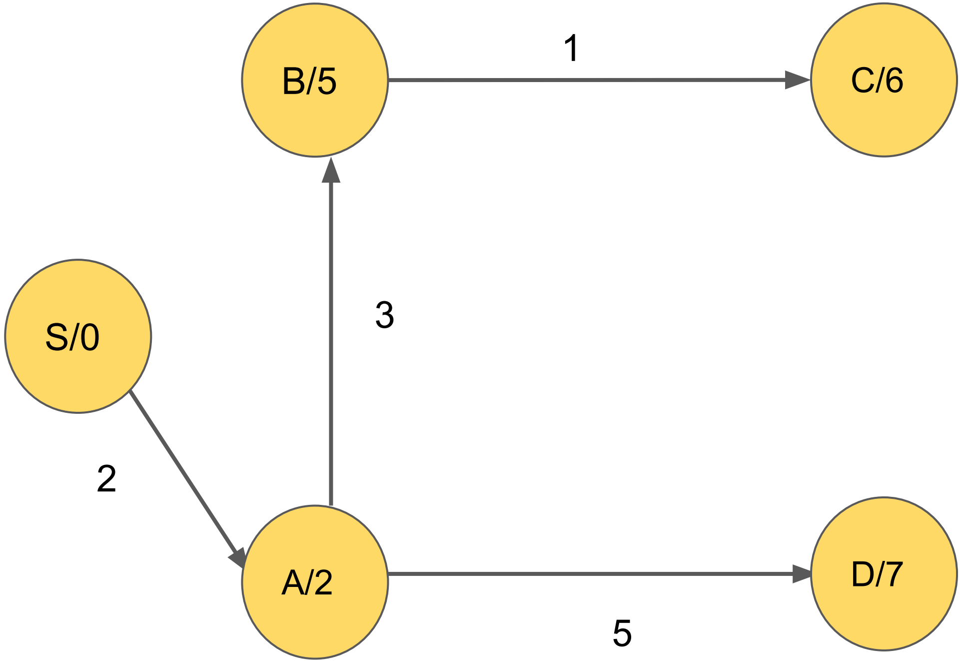


Passo 5: Adj(D) = {C}

Vértices	S	A	B	C	D
Distância	0	2	5	6	7
Precedente	nil	S	A	B	A

Algoritmo de Dijkstra - Exemplo

Árvore de Caminho Mínimo: T = (V,A).



Vértices	S	A	B	C	D
Precedente	nil	S	A	B	A

Bellman-Ford - Introdução

- Algoritmo proposto por Richard E. Bellman e Lester Randolph Ford Jr. em 1958.
- Possui o mesmo propósito do algoritmo de Dijkstra, calcula o caminho mais curto de um vértice a todos os outros vértices em um dígrafo ponderado.
- Mais lento que o algoritmo de Dijkstra.
- Porém mais flexível trabalhando com pesos negativos associados às arestas do grafo.
- Analisa a ocorrência de um ciclo negativo.

Bellman-Ford - Aplicações

- Protocolos de Roteamento Vetor-Distância.
- Problema "Triangular Arbitrage"

Exemplo de aplicação - “Triangular Arbitrage”

As an example, suppose you have \$1 million and you are provided with the following exchange rates: EUR/USD = 0.8631, EUR/GBP = 1.4600 and USD/GBP = 1.6939.

With these exchange rates there is an arbitrage opportunity:

Step 1. Sell dollars for euros: \$1 million x 0.8631 = €863,100

Step 2. Sell euros for pounds: €863,100/1.4600 = £591,164.40

Step 3. Sell pounds for dollars: £591,164.40 x 1.6939 = \$1,001,373

Step 4. Subtract the initial investment from the final amount: \$1,001,373 - \$1,000,000 = \$1,373

Bellman-Ford - Funcionamento

- Inicialização
  - onde se padroniza a distância para cada nó, sendo para a origem igual a 0.
- Relaxamento
  - consiste em verificar se existe um caminho mais curto a partir de um vértice para outros que já foram encontrados até o momento, passando por todos os vértices.
- Checagem de ciclo negativo
  - aplicamos a técnica de relaxamento mais uma vez, e se conseguirmos reduzir ainda mais a distância, significa que existe ciclo negativo.

Bellman-Ford - Algoritmo

```
BELLMAN-FORD(G, w, s)
    INITIALIZE-SINGLE-SOURCE(G, s)
    for i = 1 to |G.V| - 1
        for each edge (u, v) ∈ G.E
            RELAX(u, v, w)
    for each edge (u, v) ∈ G.E
        if v.d > u.d + w(u, v)
            return FALSE
    return TRUE
```

Inicialização

Relaxamento

Checagem de ciclo negativo

Bellman-Ford - Análise

```
BELLMAN-FORD(G, w, s)
INITIALIZE-SINGLE-SOURCE(G, s)
for i = 1 to |G.V| - 1
    for each edge (u, v) ∈ G.E
        RELAX(u, v, w)
for each edge (u, v) ∈ G.E
    if v.d > u.d + w(u, v)
        return FALSE
return TRUE
```

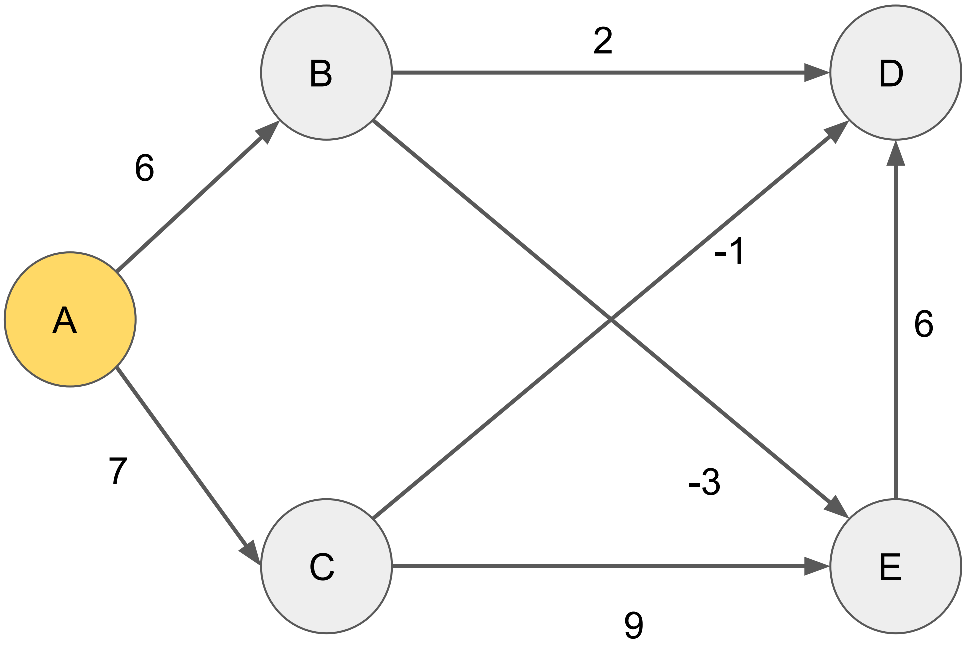
O(|V|)

O(|V||A|)

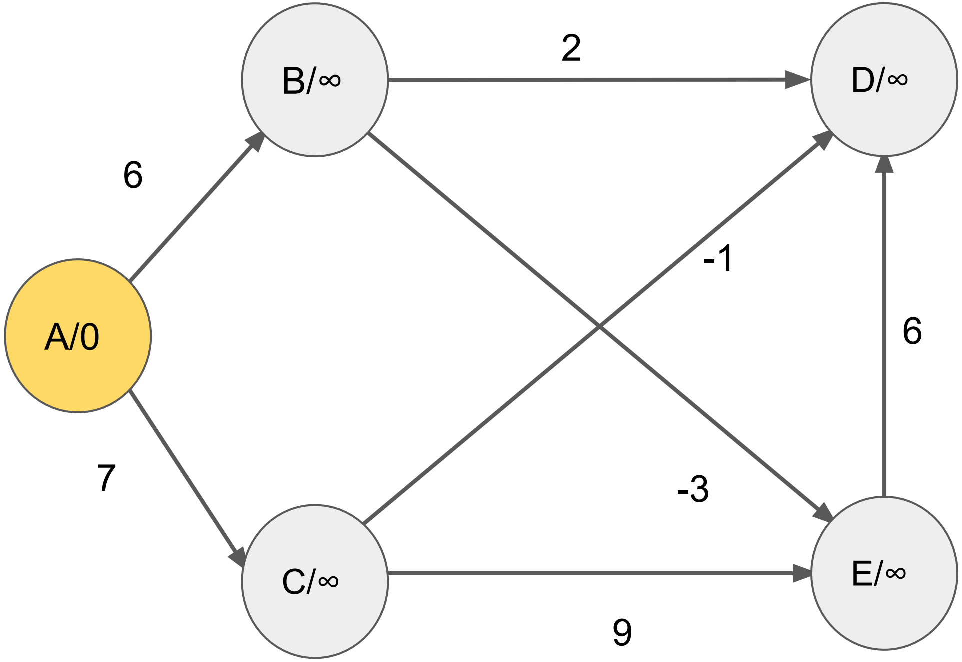
O(|A|)

O(|V||A|)

Bellman-Ford - Exemplos



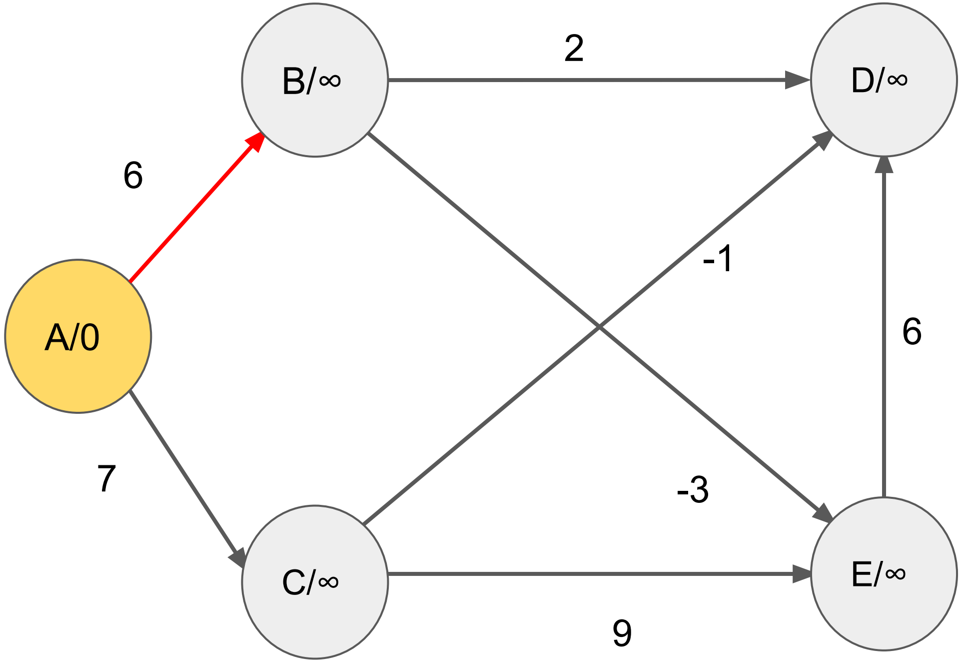
Bellman-Ford - Exemplos



Passo 1: Inicialização

Vértices	A	B	C	D	E
Distância	0	∞	∞	∞	∞
Precedente					

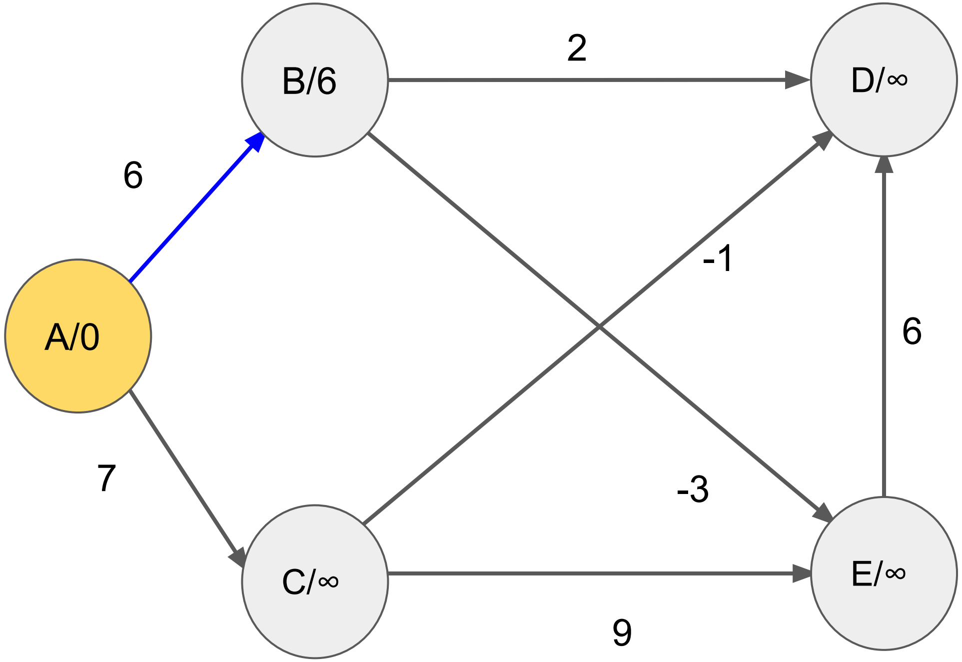
Bellman-Ford - Exemplos



Passo 2: Relaxamento

Vértices	A	B	C	D	E
Distância	0	∞	∞	∞	∞
Precedente					

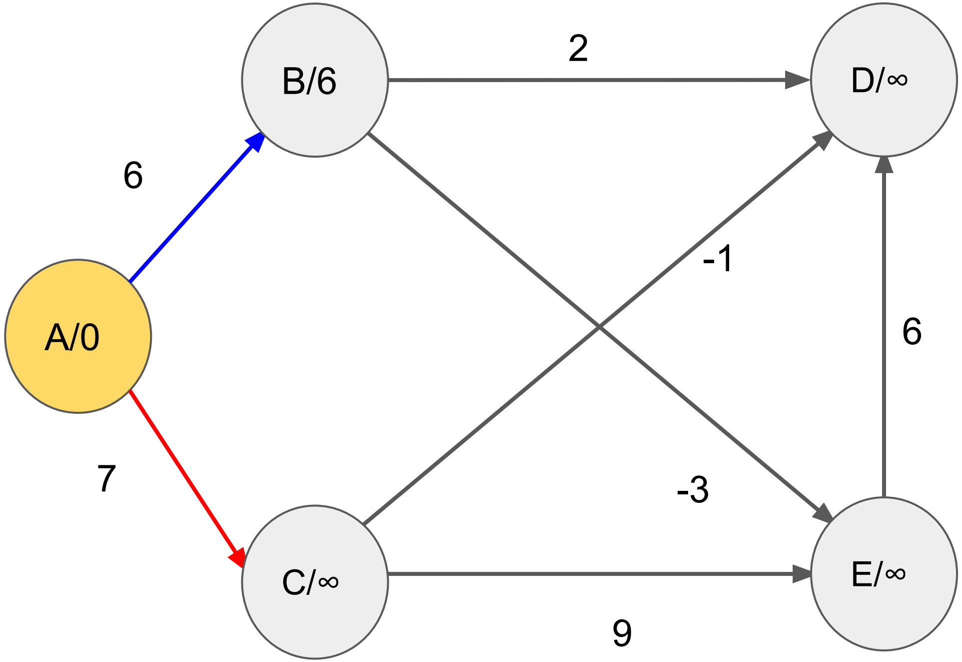
Bellman-Ford - Exemplos



Passo 2: Relaxamento

Vértices	A	B	C	D	E
Distância	0	6	∞	∞	∞
Precedente		A			

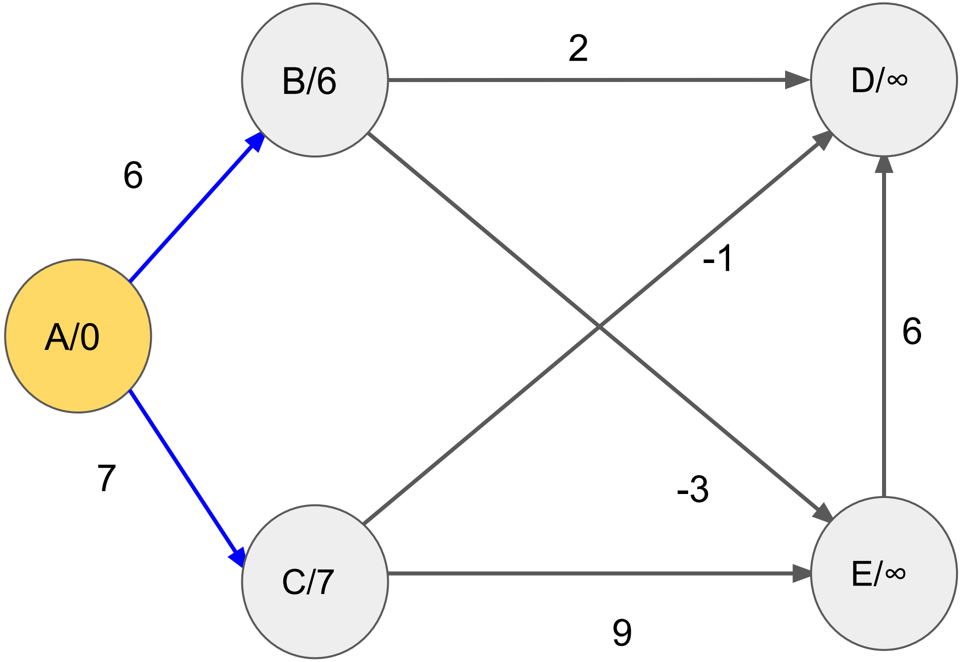
Bellman-Ford - Exemplos



Passo 2: Relaxamento

Vértices	A	B	C	D	E
Distância	0	6	∞	∞	∞
Precedente		A			

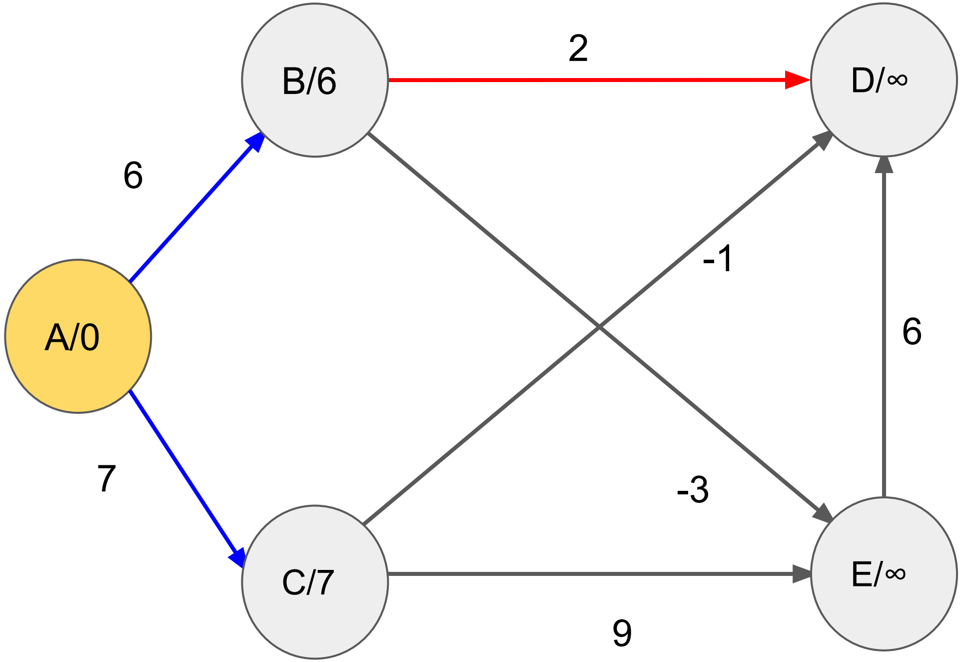
Bellman-Ford - Exemplos



Passo 2: Relaxamento

Vértices	A	B	C	D	E
Distância	0	6	7	∞	∞
Precedente		A	A		

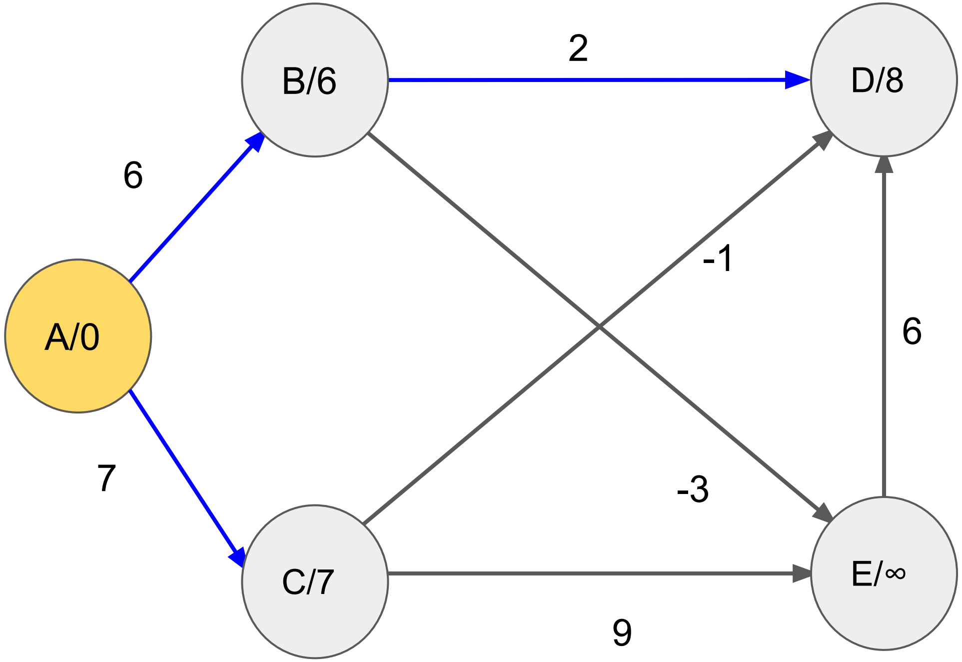
Bellman-Ford - Exemplos



Passo 2: Relaxamento

Vértices	A	B	C	D	E
Distância	0	6	7	∞	∞
Precedente		A	A		

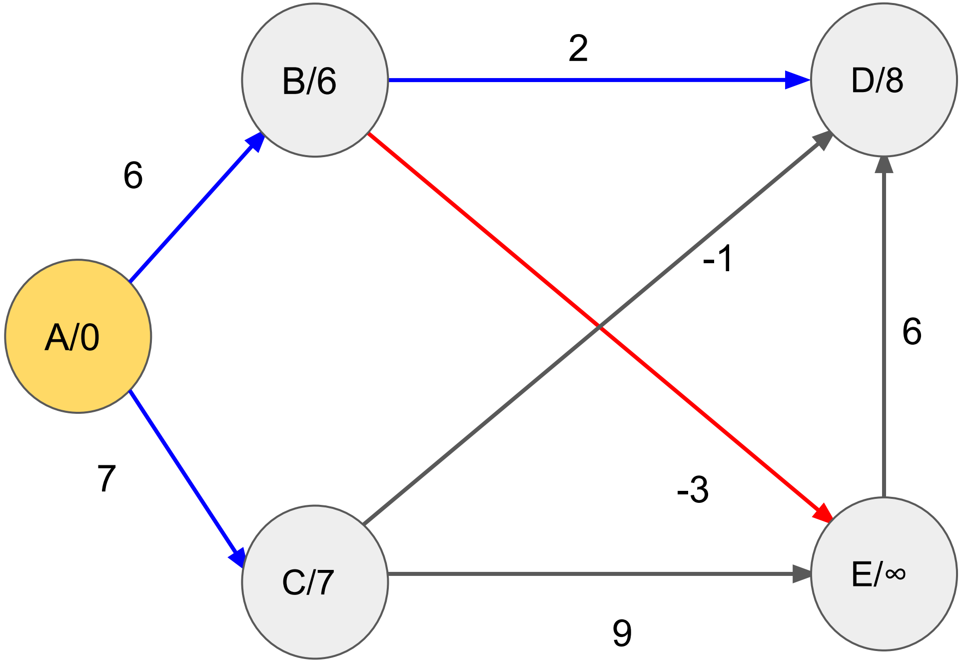
Bellman-Ford - Exemplos



Passo 2: Relaxamento

Vértices	A	B	C	D	E
Distância	0	6	7	8	∞
Precedente		A	A	B	

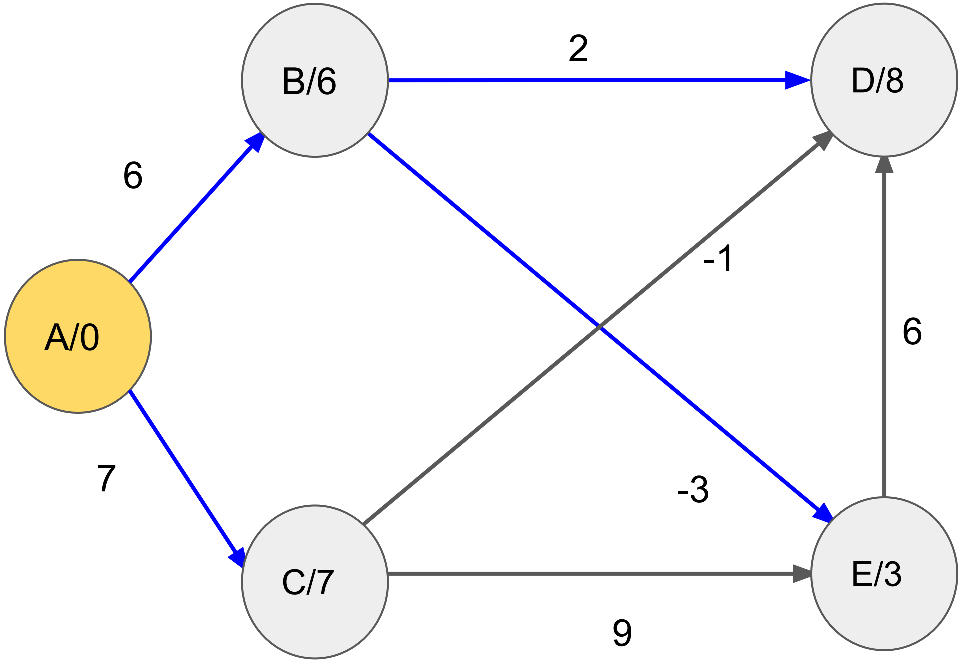
Bellman-Ford - Exemplos



Passo 2: Relaxamento

Vértices	A	B	C	D	E
Distância	0	6	7	8	∞
Precedente		A	A	B	

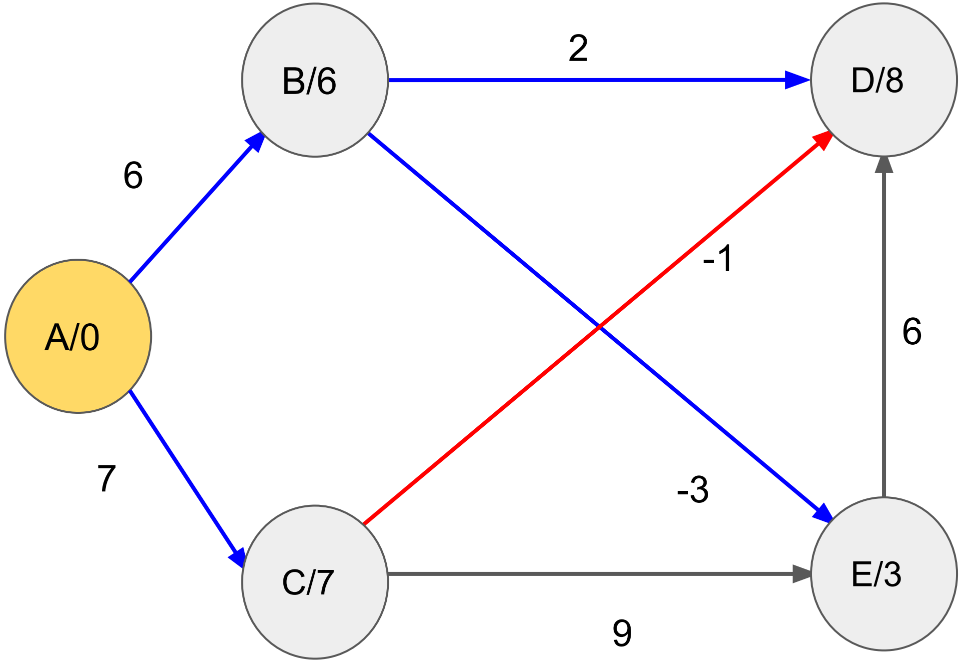
Bellman-Ford - Exemplos



Passo 2: Relaxamento

Vértices	A	B	C	D	E
Distância	0	6	7	8	3
Precedente		A	A	B	B

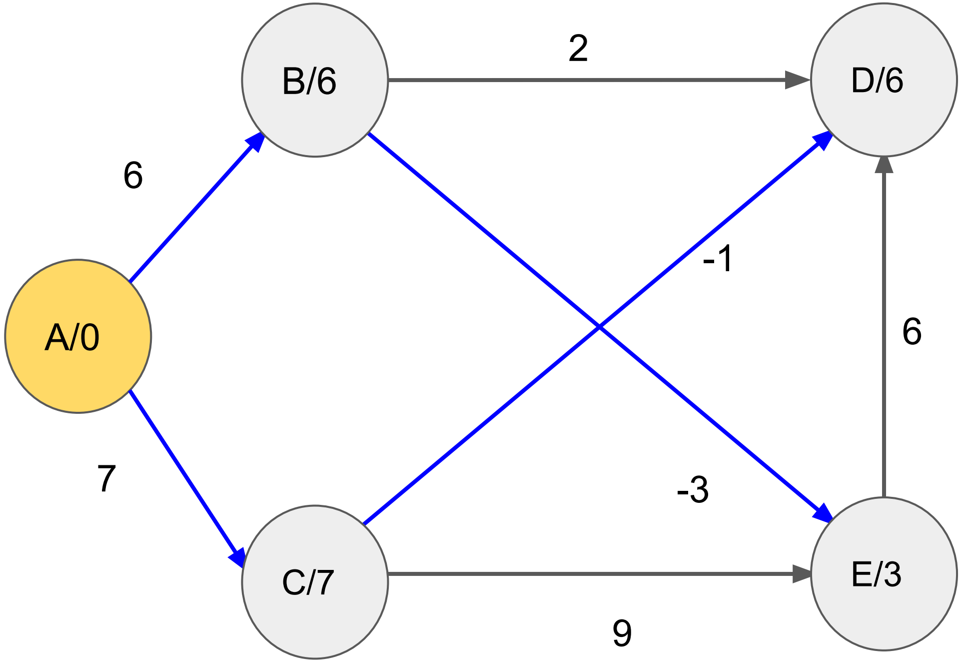
Bellman-Ford - Exemplos



Passo 2: Relaxamento

Vértices	A	B	C	D	E
Distância	0	6	7	8	3
Precedente		A	A	B	B

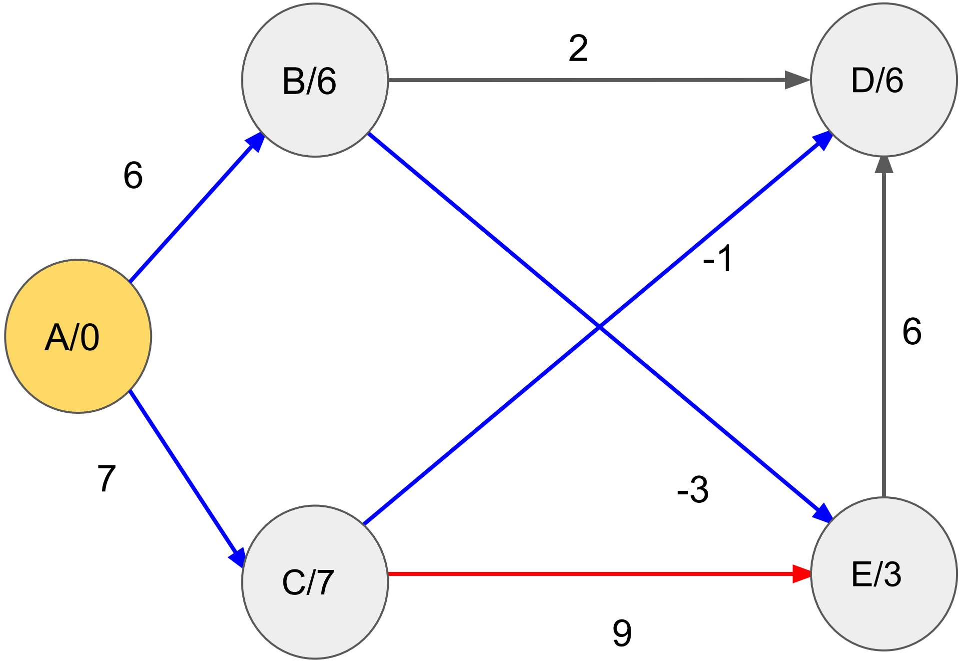
Bellman-Ford - Exemplos



Passo 2: Relaxamento

Vértices	A	B	C	D	E
Distância	0	6	7	6	3
Precedente		A	A	C	B

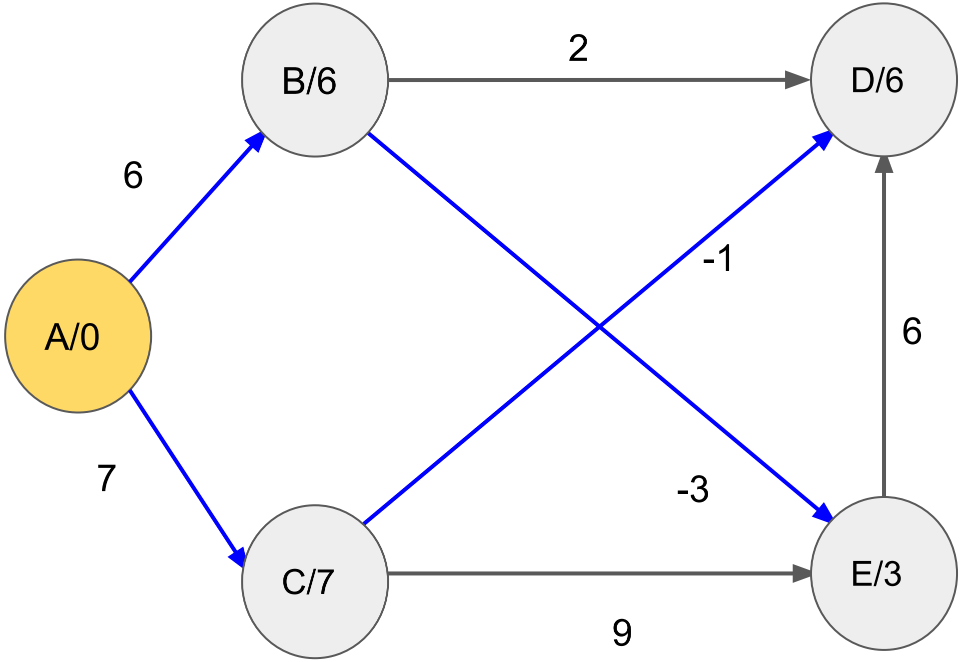
Bellman-Ford - Exemplos



Passo 2: Relaxamento

Vértices	A	B	C	D	E
Distância	0	6	7	6	3
Precedente		A	A	C	B

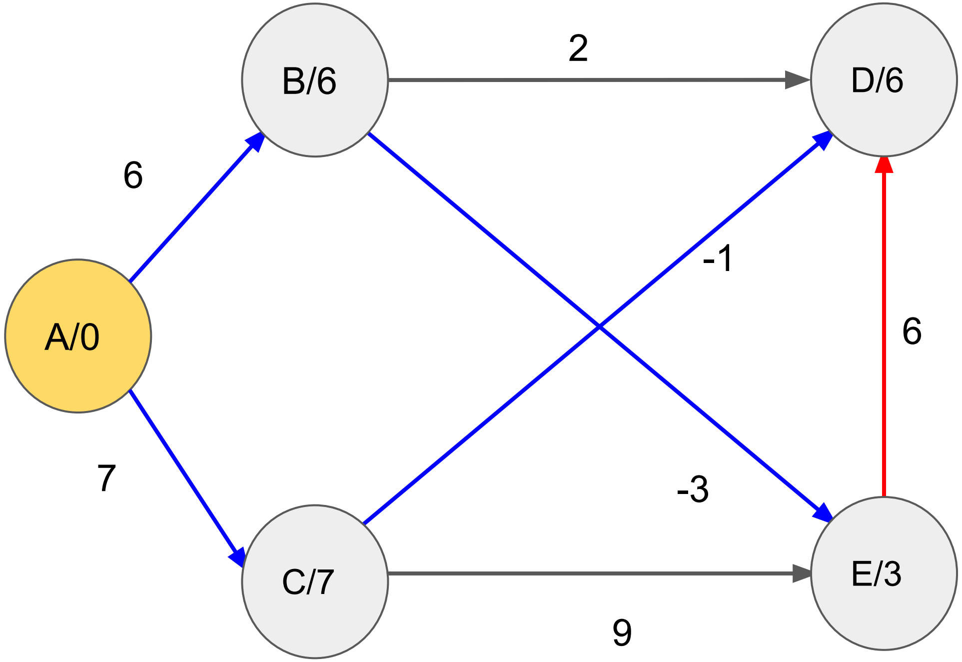
Bellman-Ford - Exemplos



Passo 2: Relaxamento

Vértices	A	B	C	D	E
Distância	0	6	7	6	3
Precedente		A	A	C	B

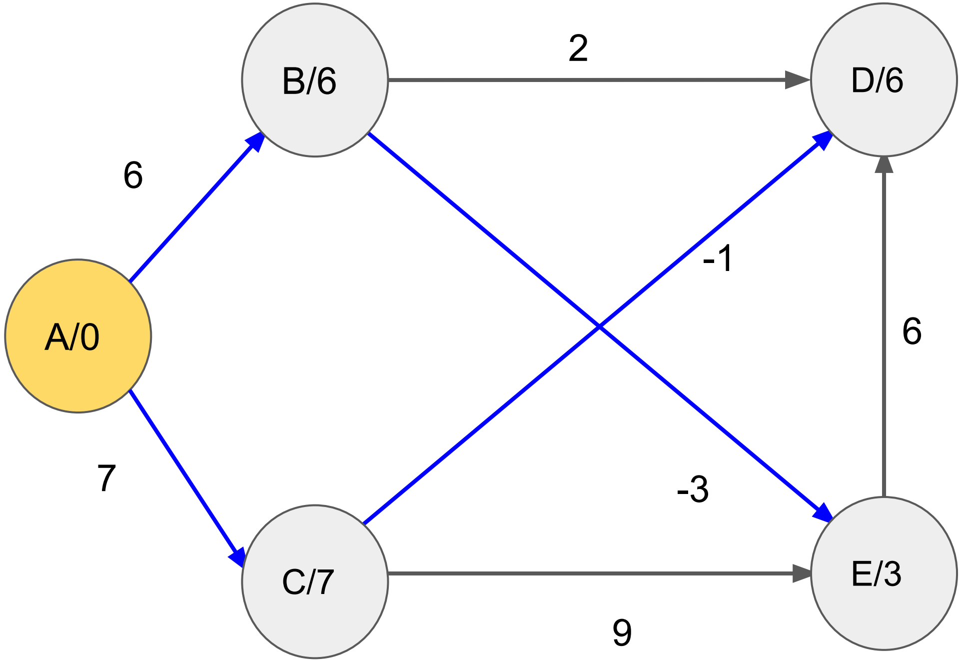
Bellman-Ford - Exemplos



Passo 2: Relaxamento

Vértices	A	B	C	D	E
Distância	0	6	7	6	3
Precedente		A	A	C	B

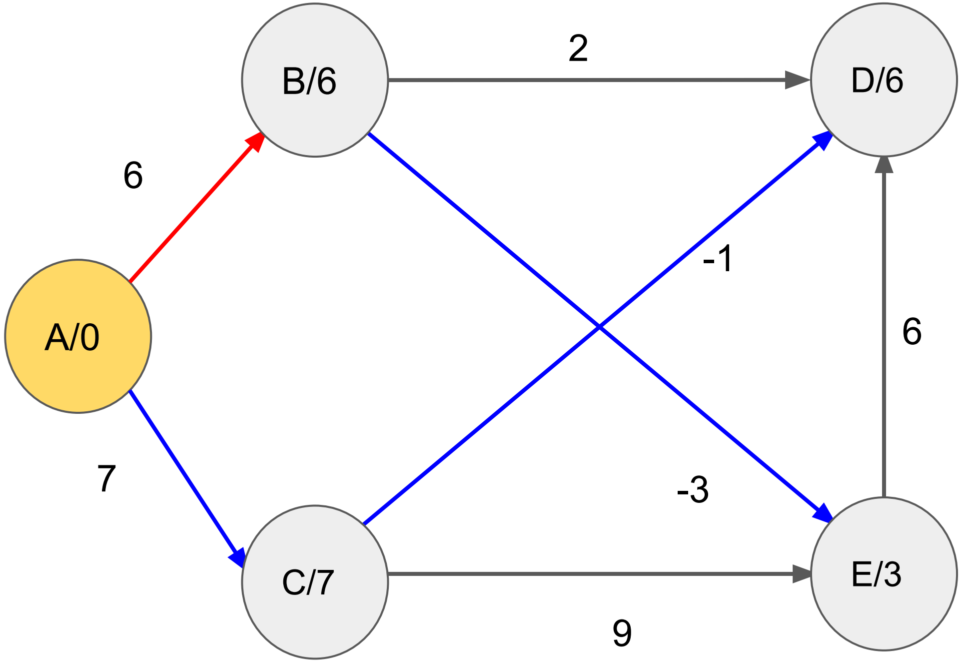
Bellman-Ford - Exemplos



Passo 2: Relaxamento

Vértices	A	B	C	D	E
Distância	0	6	7	6	3
Precedente		A	A	C	B

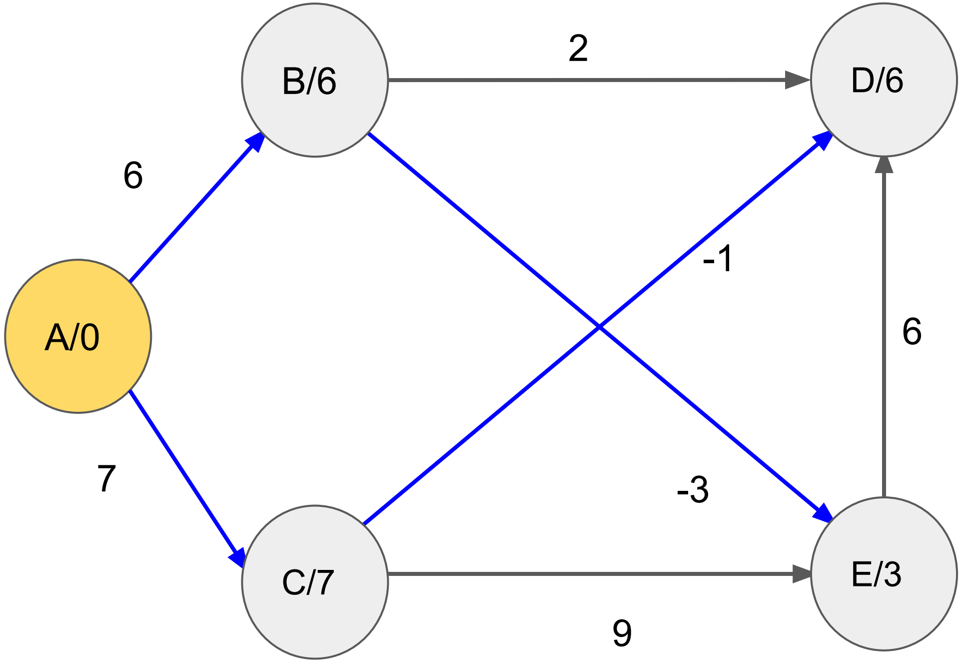
Bellman-Ford - Exemplos



Passo 2: Relaxamento

Vértices	A	B	C	D	E
Distância	0	6	7	6	3
Precedente		A	A	C	B

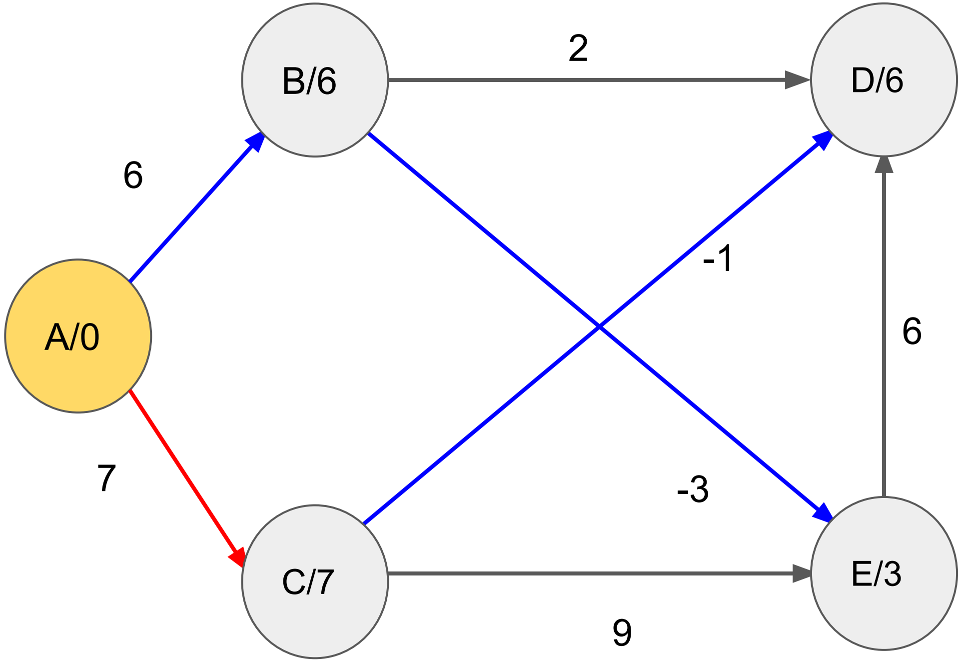
Bellman-Ford - Exemplos



Passo 2: Relaxamento

Vértices	A	B	C	D	E
Distância	0	6	7	6	3
Precedente		A	A	C	B

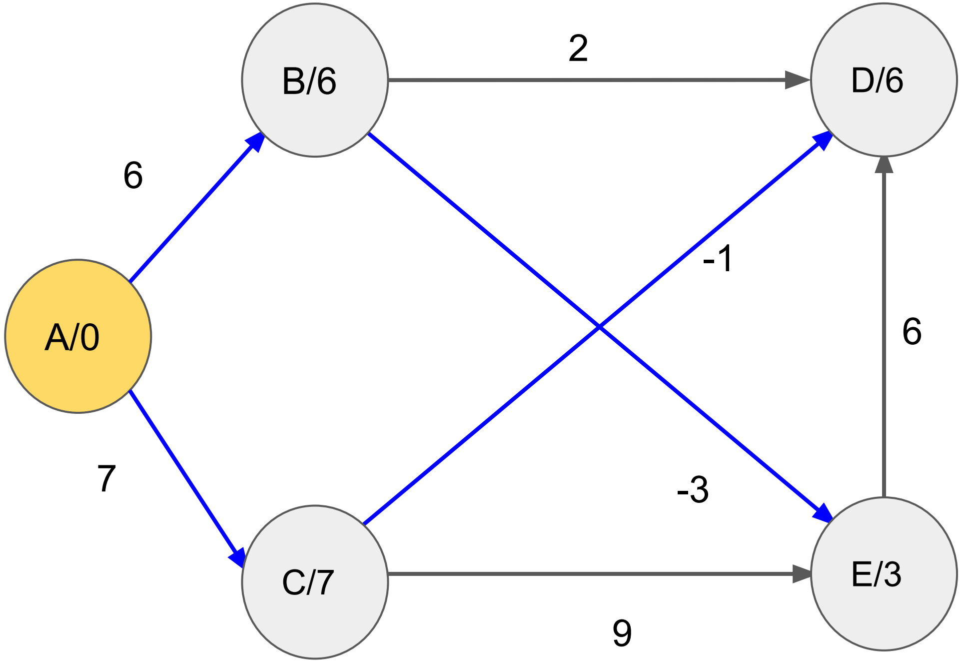
Bellman-Ford - Exemplos



Passo 2: Relaxamento

Vértices	A	B	C	D	E
Distância	0	6	7	6	3
Precedente		A	A	C	B

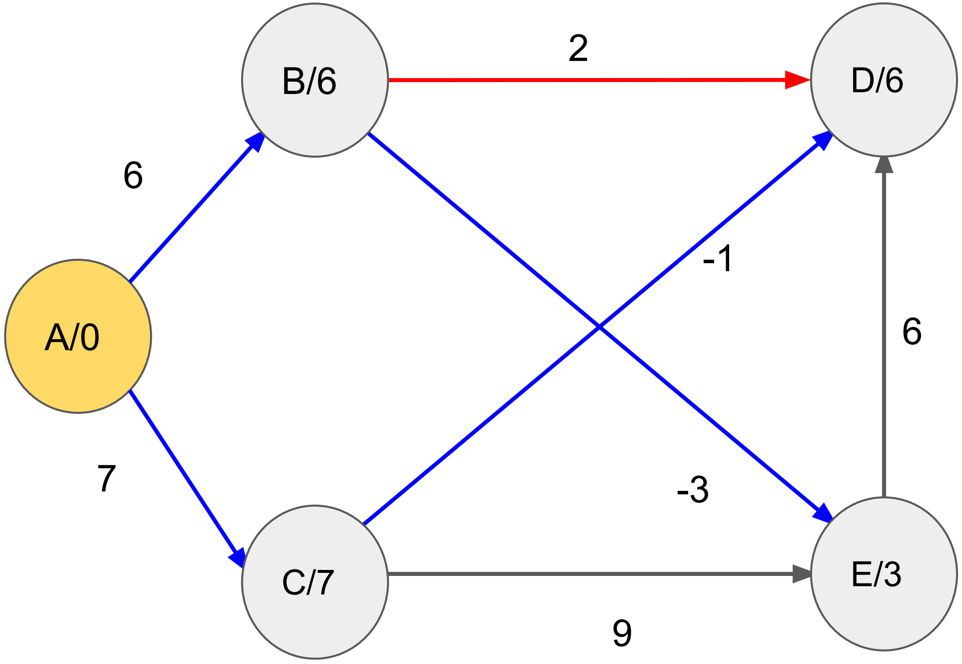
Bellman-Ford - Exemplos



Passo 2: Relaxamento

Vértices	A	B	C	D	E
Distância	0	6	7	6	3
Precedente		A	A	C	B

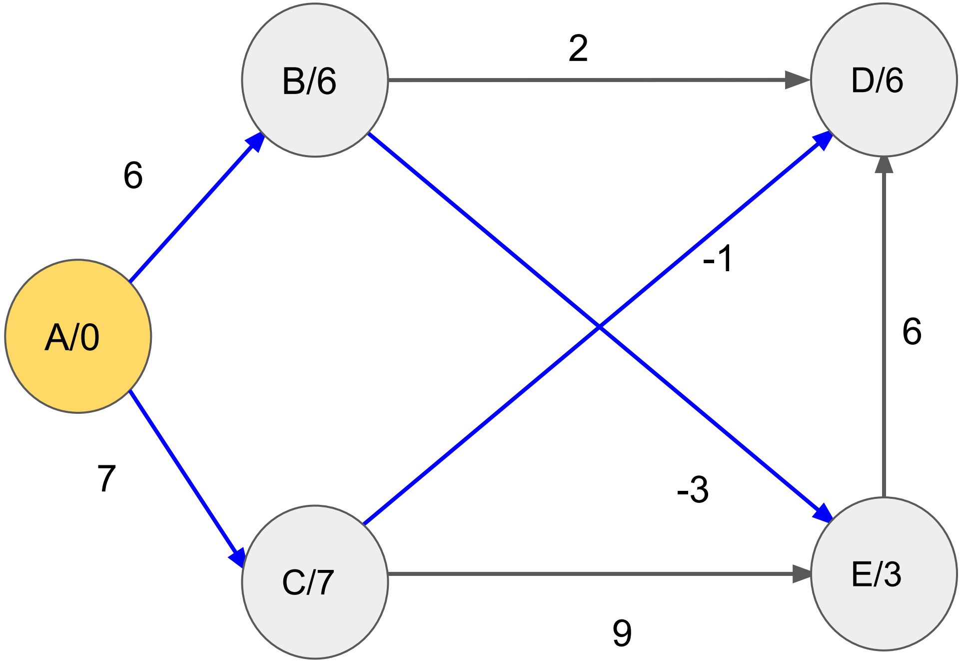
Bellman-Ford - Exemplos



Passo 2: Relaxamento

Vértices	A	B	C	D	E
Distância	0	6	7	6	3
Precedente		A	A	C	B

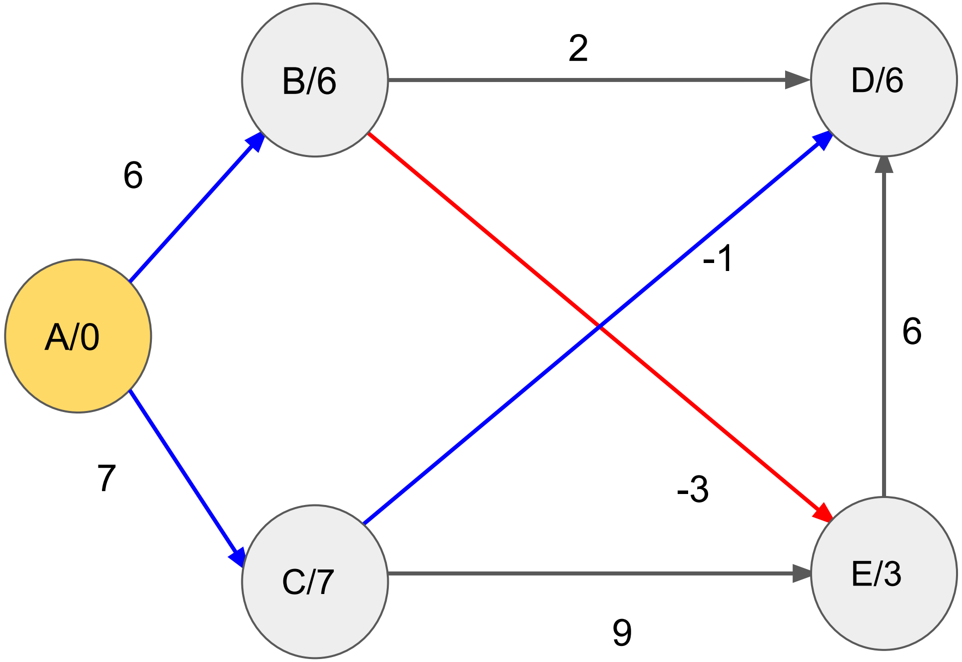
Bellman-Ford - Exemplos



Passo 2: Relaxamento

Vértices	A	B	C	D	E
Distância	0	6	7	6	3
Precedente		A	A	C	B

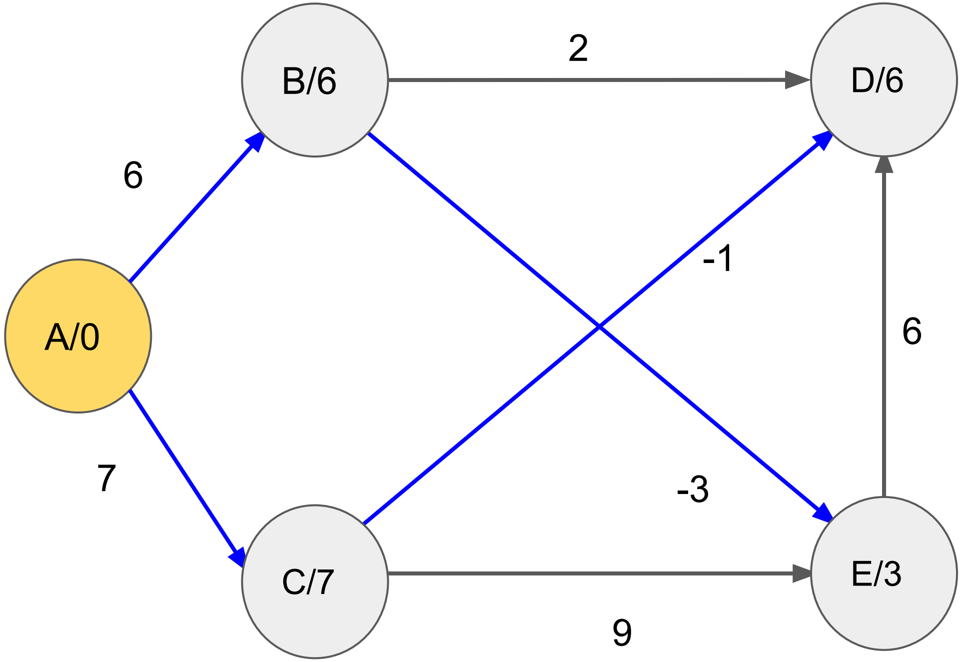
Bellman-Ford - Exemplos



Passo 2: Relaxamento

Vértices	A	B	C	D	E
Distância	0	6	7	6	3
Precedente		A	A	C	B

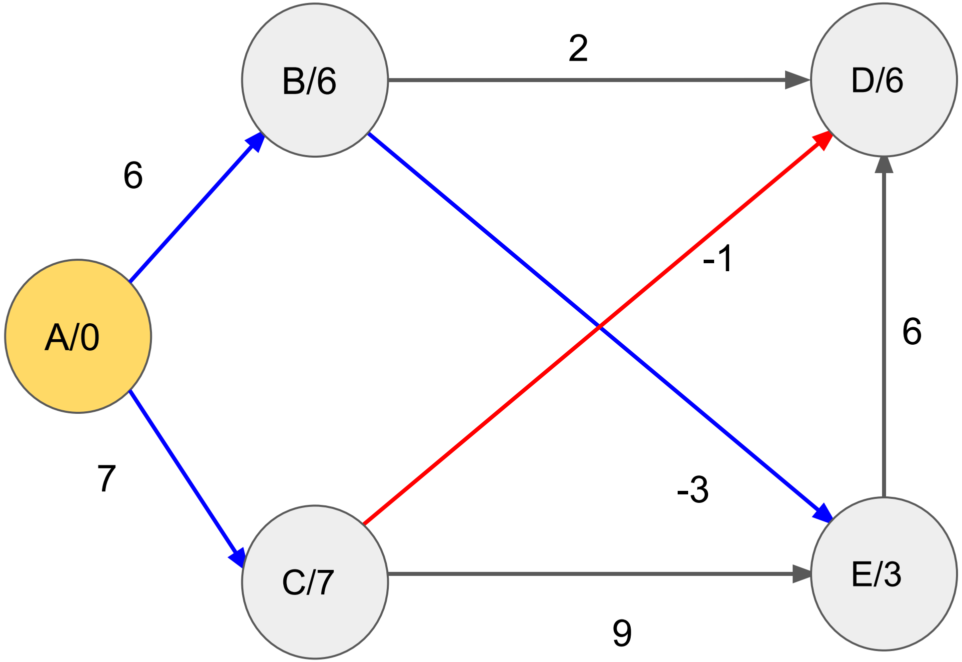
Bellman-Ford - Exemplos



Passo 2: Relaxamento

Vértices	A	B	C	D	E
Distância	0	6	7	6	3
Precedente		A	A	C	B

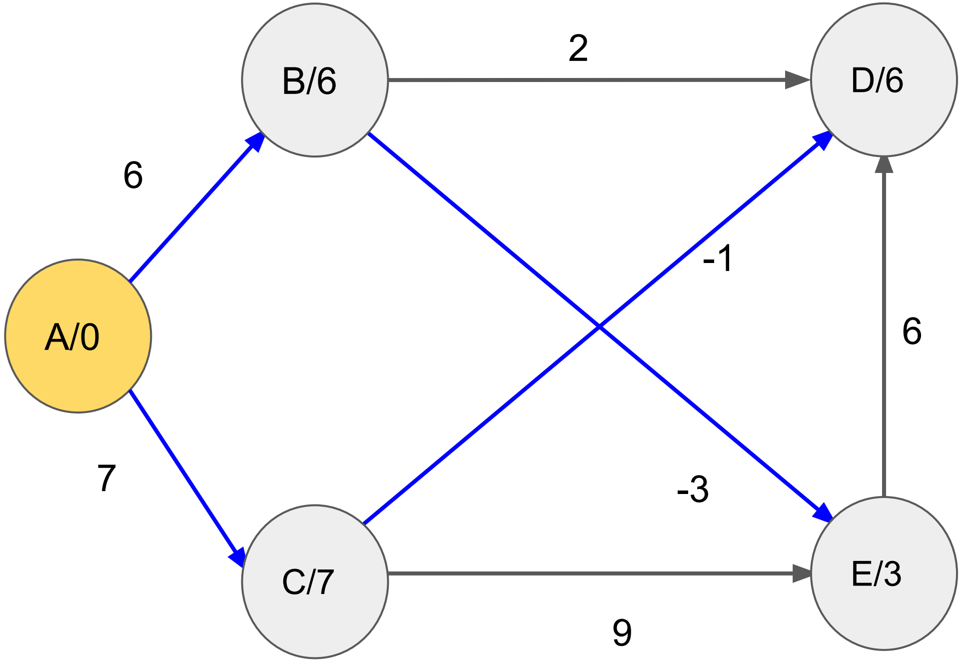
Bellman-Ford - Exemplos



Passo 2: Relaxamento

Vértices	A	B	C	D	E
Distância	0	6	7	6	3
Precedente		A	A	C	B

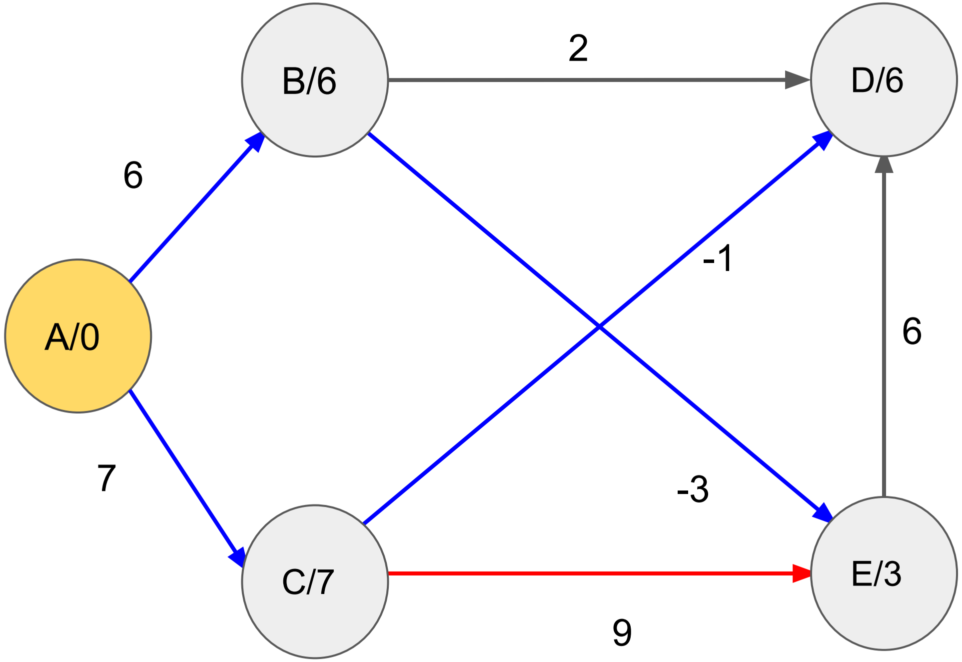
Bellman-Ford - Exemplos



Passo 2: Relaxamento

Vértices	A	B	C	D	E
Distância	0	6	7	6	3
Precedente		A	A	C	B

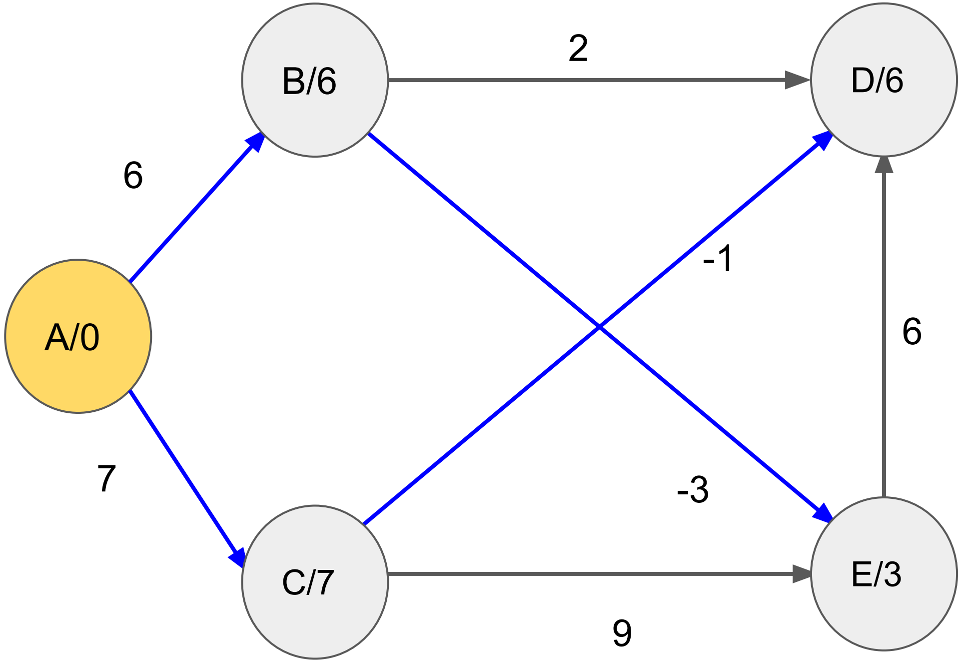
Bellman-Ford - Exemplos



Passo 2: Relaxamento

Vértices	A	B	C	D	E
Distância	0	6	7	6	3
Precedente		A	A	C	B

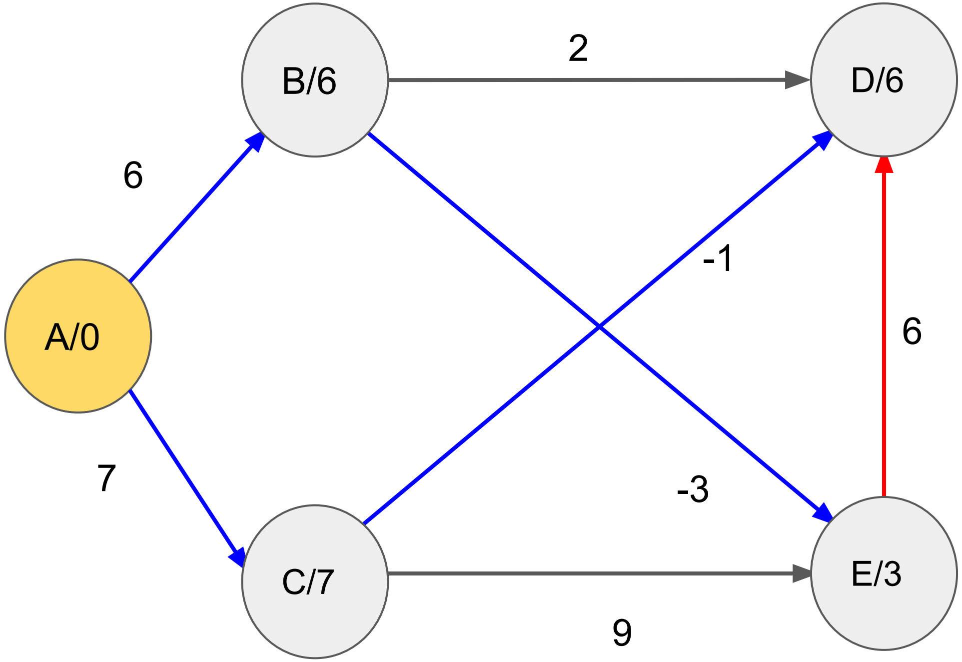
Bellman-Ford - Exemplos



Passo 2: Relaxamento

Vértices	A	B	C	D	E
Distância	0	6	7	6	3
Precedente		A	A	C	B

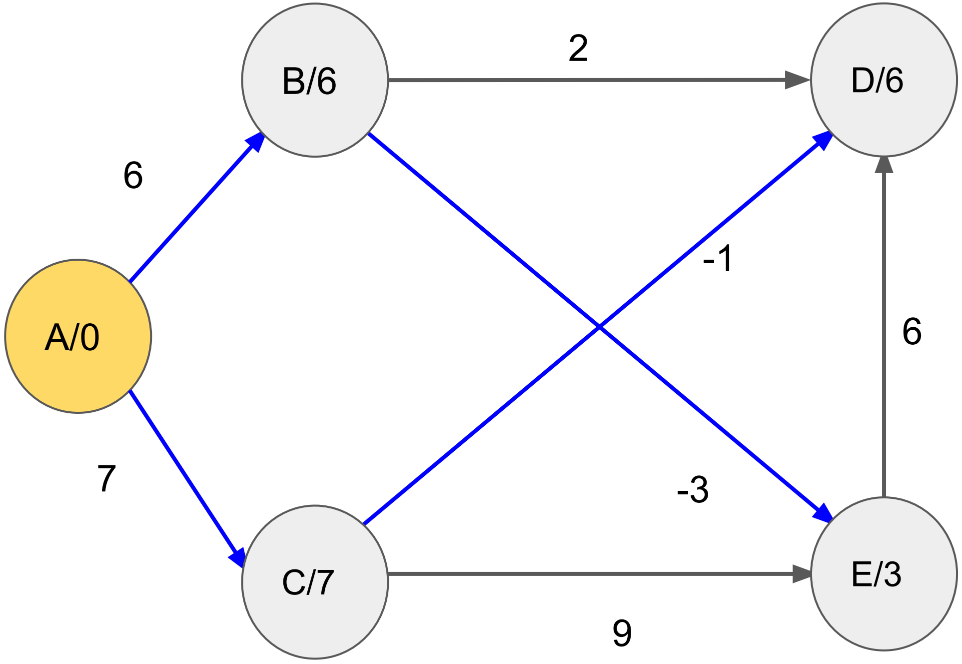
Bellman-Ford - Exemplos



Passo 2: Relaxamento

Vértices	A	B	C	D	E
Distância	0	6	7	6	3
Precedente		A	A	C	B

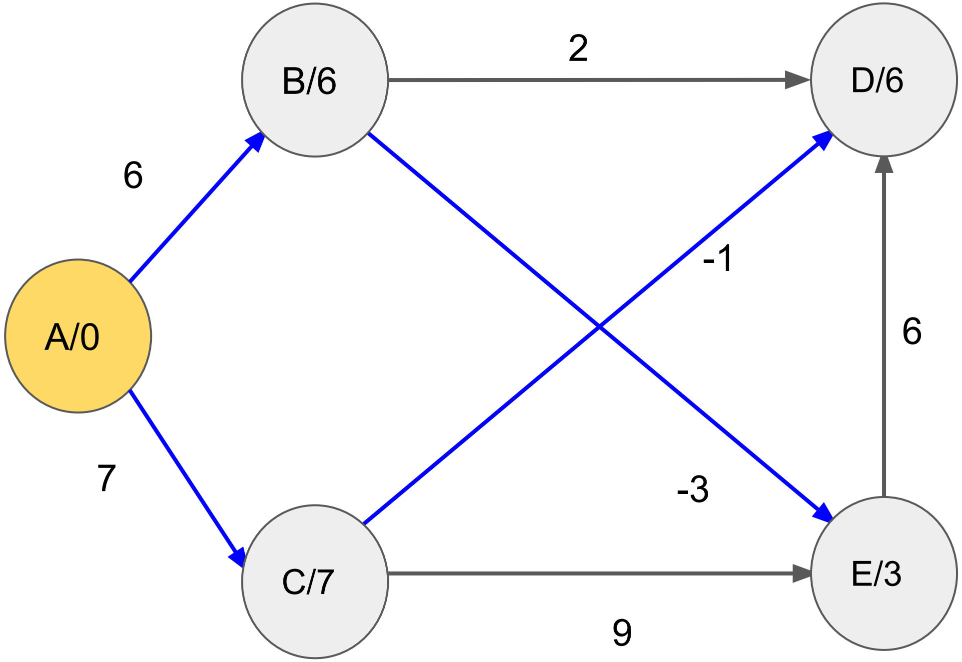
Bellman-Ford - Exemplos



Passo 2: Relaxamento

Vértices	A	B	C	D	E
Distância	0	6	7	6	3
Precedente		A	A	C	B

Bellman-Ford - Exemplos



Passo 3: Checagem de ciclos negativos

Vértices	A	B	C	D	E
Distância	0	6	7	6	3
Precedente		A	A	C	B

Diferenças entre os algoritmos

Dijkstra	**Bellman-Ford**
Mais eficiente $O(\|V\|+\|A\|\log(V))$ Somente pesos não negativos *Greedy*	Mais complexo $O(\|V\|\|A\|)$ Trabalha com pesos negativos *Not greedy*

Referências Bibliográficas

- **Dijkstra**, Edsger W. **A note on two problems in connexion with graphs**. Numerische mathematik 1.1, 1959, p. 269-271.
- **Bellman**, Richard. **On a routing problem**. Quarterly of applied mathematics 16.1, 1958, p. 87-90.
- **Ford Jr**, Lester R. **Network flow theory**. No. P-923. RAND CORP SANTA MONICA CA, 1956.
- Moore, Edward F**. The shortest path through a maze**. Proc. International Symposium on the Theory of Switching. Harvard University Press, 1959.
- Cormen, Thomas H. Introduction to algorithms, MIT press, 2009.
- Ribeiro Filho, G. et al. TerraNetwork: a Urban Street Network Analysis System. 2006. Disponível em: http://www.lac.inpe.br/~terranetwork/geo.pdf