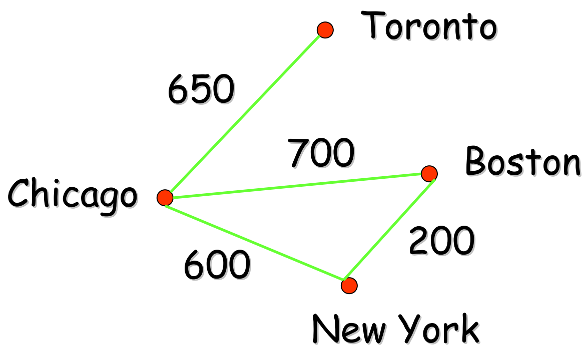
Problema do Caminho Mais Curto

" Podemos afectar pesos" aos arcos de um grafo, por exemplo, para representar uma distância entre cidades numa rede ferroviária:



Problema do Caminho Mais Curto
Shortest Path Problems

Estes grafos ponderados podem ser usados para modelar redes de computadores com tempos de resposta, ou com custos de ligação.

Uma das questões mais interessantes que podemos investigar com estes grafos é:

Qual é o caminho mais curto entre dois vértices no grafo, ou seja, o caminho com a soma mínima de pesos?

Isto corresponde à ligação mais rápida, ou à ligação mais económica numa rede de computadores.

Problema do Caminho Mais Curto

Algoritmos:

- Distância mais curta de um vértice origem e todos os outros vértices do grafo
  - Dijkstra – custos não negativos; $O(n^2)$
  - Ford – custos gerais
    (sem ciclos comprimento negativo)
  - Algoritmo de Partição– custos gerais; $O(nm)$
    (sem ciclos comprimento negativo)
- Distância mais curta entre todos os pares de vértices do grafo
  - Floyd – custos gerais; $O(n^3)$
    (sem ciclos comprimento negativo)

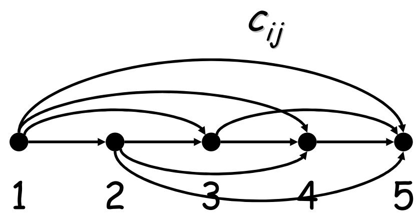
Problema do Caminho Mais Curto

Aplicações:

- Concepção de redes de comunicações
- Problemas de transporte
- Problemas de distribuição
- Substituição de equipamento
- Método do caminho crítico
- Dimensão dos lotes de produção
- Sub-problema de outros algoritmos
  (carteiro chinês, caixeiro viajante)

Problema do Caminho Mais Curto

Substituição de equipamento:



O Algoritmo de Dijkstra

O algoritmo de Dijkstra (rotulação permanente – *label setting algorithm*) é um procedimento iterativo que determina o caminho mais curto entre um vértice origem $\underline{s}$ e todos os outros vértices do grafo.

Associa um rótulo a cada vértice, que corresponde à distância mais curta entre o vértice e a origem $\underline{s}$.

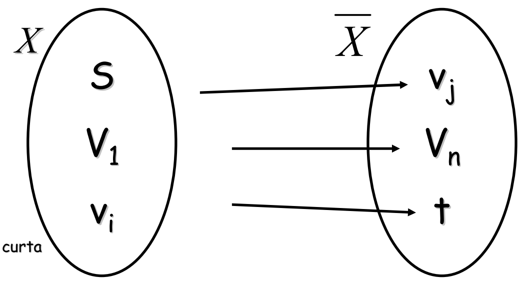
Os rótulos são temporários, e em cada iteração um rotúlo transforma-se em rótulo permanente, (encontrada a distância mais curta para esse vértice).

Faz uso da propriedade de não existir custos negativos.

O Algoritmo de Dijkstra

Teorema: o algoritmo de Dijkstra determina correctamente a distância mais curta do vértice <u>*s*</u> para cada um dos restantes vértices do grafo.

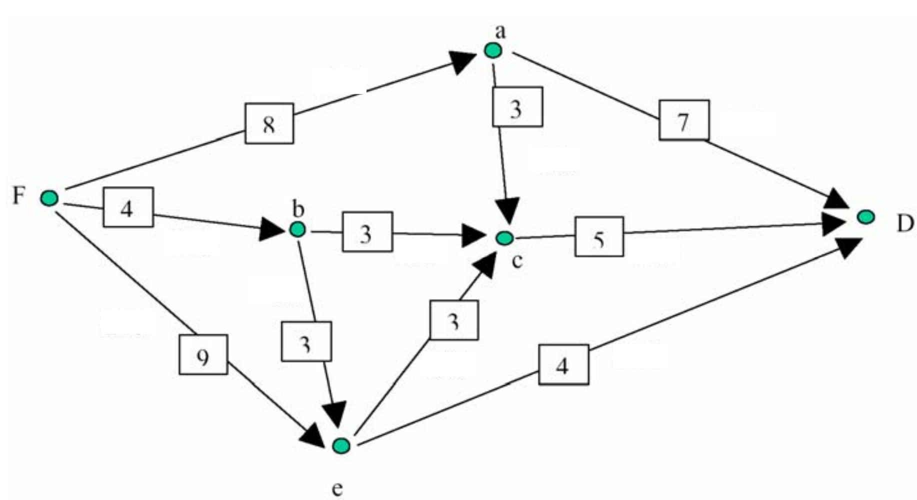
Prova:



Vértices com distância + curta encontrada

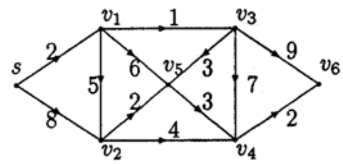
O Algoritmo de Dijkstra

Distância + curta de F a D



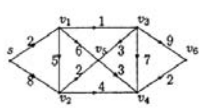
O Algoritmo de Dijkstra

input: digrafo $G = (V, A)$, com custos não negativos
output: caminho mais curto entre o vértice s e todos os outros vértices
begin
  $X := \{s\}$; $\bar{X} := V - \{s\}$; $d(s) := 0$; $pred(s) := -$;
  for all $j \in \bar{X}$ **do**
    if $(s,j) \in A$ **then** $d(j) := c_{sj}$; $pred(j) := s$
      else $d(j) := +\infty$; $pred(j) := 0$
  while $X \neq V$ **do**
    begin
      (**comentário:** selecção de vértice i)
      seja i o vértice de $\bar{X}$: $d(i) = \min_{j \in \bar{X}}\{d(j)\}$
      (**comentário:** rotulação permanente)
      $X = X \cup \{i\}$; $\bar{X} := \bar{X} - \{i\}$;
      (**comentário:** análise dos sucess. de i com rótulos temporários)
      for all $j \in \bar{X} : (i,j) \in A$ **do**
        if $d(j) > d(i) + c_{ij}$ **then** $d(j) := d(i) + c_{ij}$; $pred(j) := i$;
    end;
end;



Algoritmo de Dijkstra

O Algoritmo de Dijkstra

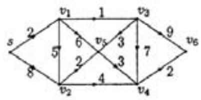


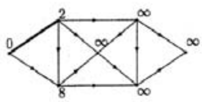
input: digrafo $G = (V, A)$, com custos não negativos
output: caminho mais curto entre o vértice s e todos os outros vértices
begin
  $X := \{s\}$; $\bar{X} := V - \{s\}$; $d(s) := 0$; $pred(s) := -$;
  for all $j \in \bar{X}$ **do**
    if $(s,j) \in A$ **then** $d(j) := c_{sj}$; $pred(j) := s$
      else $d(j) := +\infty$; $pred(j) := 0$
  while $X \neq V$ **do**
    begin
      (**comentário:** selecção de vértice i)
      seja i o vértice de $\bar{X}$: $d(i) = \min_{j \in \bar{X}}\{d(j)\}$
      (**comentário:** rotulação permanente)
      $X = X \cup \{i\}$; $\bar{X} := \bar{X} - \{i\}$;
      (**comentário:** análise dos sucess. de i com rótulos temporários)
      for all $j \in \bar{X} : (i,j) \in A$ **do**
        if $d(j) > d(i) + c_{ij}$ **then** $d(j) := d(i) + c_{ij}$; $pred(j) := i$;
    end;
end;

Algoritmo de Dijkstra

i	X	$d(s)$	$d(v_1)$	$d(v_2)$	$d(v_3)$	$d(v_4)$	$d(v_5)$	$d(v_6)$
	$\{s\}$	0	2	8	∞	∞	∞	∞

O Algoritmo de Dijkstra





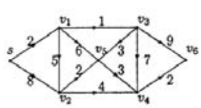
Seleccionar v_1

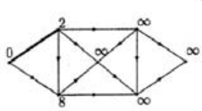
input: digrafo $G = (V, A)$, com custos não negativos
output: caminho mais curto entre o vértice s e todos os outros vértices
begin
  $X := \{s\}$; $\bar{X} := V - \{s\}$; $d(s) := 0$; $pred(s) := -$;
  for all $j \in \bar{X}$ **do**
    if $(s, j) \in A$ **then** $d(j) := c_{sj}$; $pred(j) := s$
      else $d(j) := +\infty$; $pred(j) := 0$
  while $X \neq V$ **do**
    begin
      (**comentário:** selecção de vértice i)
      seja i o vértice de $\bar{X}$: $d(i) = \min_{j \in \bar{X}}\{d(j)\}$
      (**comentário:** rotulação permanente)
      $X = X \cup \{i\}$; $\bar{X} := \bar{X} - \{i\}$;
      (**comentário:** análise dos sucess. de i com rótulos temporários)
      for all $j \in \bar{X} : (i, j) \in A$ **do**
        if $d(j) > d(i) + c_{ij}$ **then** $d(j) := d(i) + c_{ij}$; $pred(j) := i$;
    end;
end;

Algoritmo de Dijkstra

i	X	$d(s)$	$d(v_1)$	$d(v_2)$	$d(v_3)$	$d(v_4)$	$d(v_5)$	$d(v_6)$
	$\{s\}$	0	2	8	∞	∞	∞	∞

O Algoritmo de Dijkstra





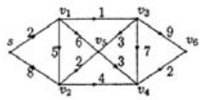
Seleccionar v_1

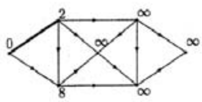
input: digrafo $G = (V, A)$, com custos não negativos
output: caminho mais curto entre o vértice s e todos os outros vértices
begin
  $X := \{s\}$; $\bar{X} := V - \{s\}$; $d(s) := 0$; $pred(s) := -$;
  for all $j \in \bar{X}$ **do**
    if $(s, j) \in A$ **then** $d(j) := c_{sj}$; $pred(j) := s$
      else $d(j) := +\infty$; $pred(j) := 0$
  while $X \neq V$ **do**
    begin
      (**comentário:** selecção de vértice i)
      seja i o vértice de $\bar{X}$: $d(i) = \min_{j \in \bar{X}}\{d(j)\}$
      (**comentário:** rotulação permanente)
      $X = X \cup \{i\}$; $\bar{X} := \bar{X} - \{i\}$;
      (**comentário:** análise dos sucess. de i com rótulos temporários)
      for all $j \in \bar{X} : (i, j) \in A$ **do**
        if $d(j) > d(i) + c_{ij}$ **then** $d(j) := d(i) + c_{ij}$; $pred(j) := i$;
    end;
end;

Algoritmo de Dijkstra

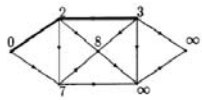
i	X	$d(s)$	$d(v_1)$	$d(v_2)$	$d(v_3)$	$d(v_4)$	$d(v_5)$	$d(v_6)$
	$\{s\}$	0	2	8	∞	∞	∞	∞
v_1	$\{s, v_1\}$			7	3		8	

O Algoritmo de Dijkstra





Seleccionar v_1



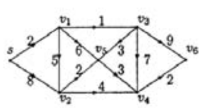
Seleccionar v_3

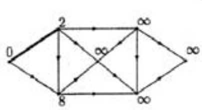
```
input:  digrafo G = (V, A), com custos não negativos
output:  caminho mais curto entre o vértice s e todos os outros vértices
begin
   X := {s}; X̄ := V − {s}; d(s) := 0; pred(s) := −;
   for all j ∈ X̄ do
      if (s, j) ∈ A then d(j) := c_sj; pred(j) := s
                    else d(j) := +∞; pred(j) := 0
   while X ≠ V do
      begin
         (comentário: selecção de vértice i)
         seja i o vértice de X̄ : d(i) = min_{j∈X̄} {d(j)}
         (comentário: rotulação permanente)
         X = X ∪ {i}; X̄ := X̄ − {i};
         (comentário: análise dos sucess. de i com rótulos temporários)
         for all j ∈ X̄ : (i, j) ∈ A do
            if d(j) > d(i) + c_ij then d(j) := d(i) + c_ij; pred(j) := i;
      end;
end;
```

Algoritmo de Dijkstra

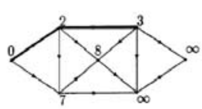
i	X	$d(s)$	$d(v_1)$	$d(v_2)$	$d(v_3)$	$d(v_4)$	$d(v_5)$	$d(v_6)$
	$\{s\}$	0	2	8	∞	∞	∞	∞
v_1	$\{s, v_1\}$			7	3		8	

O Algoritmo de Dijkstra





Seleccionar v_1



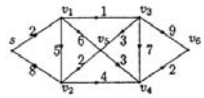
Seleccionar v_3

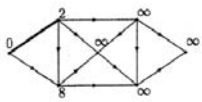

```
input:  digrafo G = (V, A), com custos não negativos
output:  caminho mais curto entre o vértice s e todos os outros vértices
begin
   X := {s}; X̄ := V − {s}; d(s) := 0; pred(s) := −;
   for all j ∈ X̄ do
      if (s, j) ∈ A then d(j) := c_sj; pred(j) := s
                    else d(j) := +∞; pred(j) := 0
   while X ≠ V do
      begin
         (comentário: selecção de vértice i)
         seja i o vértice de X̄ : d(i) = min_{j∈X̄} {d(j)}
         (comentário: rotulação permanente)
         X = X ∪ {i}; X̄ := X̄ − {i};
         (comentário: análise dos sucess. de i com rótulos temporários)
         for all j ∈ X̄ : (i, j) ∈ A do
            if d(j) > d(i) + c_ij then d(j) := d(i) + c_ij; pred(j) := i;
      end;
end;
```

Algoritmo de Dijkstra

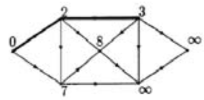
i	X	$d(s)$	$d(v_1)$	$d(v_2)$	$d(v_3)$	$d(v_4)$	$d(v_5)$	$d(v_6)$
	$\{s\}$	0	2	8	∞	∞	∞	∞
v_1	$\{s, v_1\}$			7	3		8	
v_3	$\{s, v_1, v_3\}$					10	6	12

O Algoritmo de Dijkstra

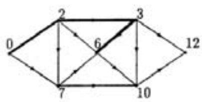




Seleccionar v_1



Seleccionar v_3



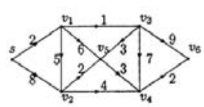
Seleccionar v_5

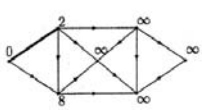
input: digrafo $G = (V, A)$, com custos não negativos
output: caminho mais curto entre o vértice s e todos os outros vértices
begin
  $X := \{s\}$; $\bar{X} := V - \{s\}$; $d(s) := 0$; $pred(s) := -$;
  for all $j \in \bar{X}$ **do**
    if $(s, j) \in A$ **then** $d(j) := c_{sj}$; $pred(j) := s$
      else $d(j) := +\infty$; $pred(j) := 0$
  while $X \neq V$ **do**
    begin
      (**comentário:** selecção de vértice i)
      seja i o vértice de $\bar{X}$: $d(i) = \min_{j \in \bar{X}}\{d(j)\}$
      (**comentário:** rotulação permanente)
      $X = X \cup \{i\}$; $\bar{X} := \bar{X} - \{i\}$;
      (**comentário:** análise dos sucess. de i com rótulos temporários)
      for all $j \in \bar{X} : (i, j) \in A$ **do**
        if $d(j) > d(i) + c_{ij}$ **then** $d(j) := d(i) + c_{ij}$; $pred(j) := i$;
    end;
end;

Algoritmo de Dijkstra

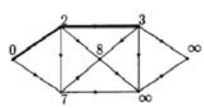
i	X	$d(s)$	$d(v_1)$	$d(v_2)$	$d(v_3)$	$d(v_4)$	$d(v_5)$	$d(v_6)$
	$\{s\}$	0	2	8	∞	∞	∞	∞
v_1	$\{s, v_1\}$			7	3		8	
v_3	$\{s, v_1, v_3\}$					10	6	12

O Algoritmo de Dijkstra

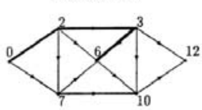




Seleccionar v_1



Seleccionar v_3



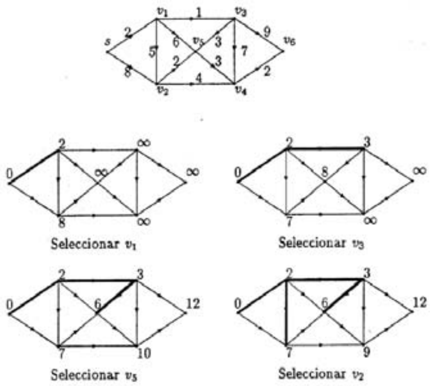
Seleccionar v_5

input: digrafo $G = (V, A)$, com custos não negativos
output: caminho mais curto entre o vértice s e todos os outros vértices
begin
  $X := \{s\}$; $\bar{X} := V - \{s\}$; $d(s) := 0$; $pred(s) := -$;
  for all $j \in \bar{X}$ **do**
    if $(s, j) \in A$ **then** $d(j) := c_{sj}$; $pred(j) := s$
      else $d(j) := +\infty$; $pred(j) := 0$
  while $X \neq V$ **do**
    begin
      (**comentário:** selecção de vértice i)
      seja i o vértice de $\bar{X}$: $d(i) = \min_{j \in \bar{X}}\{d(j)\}$
      (**comentário:** rotulação permanente)
      $X = X \cup \{i\}$; $\bar{X} := \bar{X} - \{i\}$;
      (**comentário:** análise dos sucess. de i com rótulos temporários)
      for all $j \in \bar{X} : (i, j) \in A$ **do**
        if $d(j) > d(i) + c_{ij}$ **then** $d(j) := d(i) + c_{ij}$; $pred(j) := i$;
    end;
end;

Algoritmo de Dijkstra

i	X	$d(s)$	$d(v_1)$	$d(v_2)$	$d(v_3)$	$d(v_4)$	$d(v_5)$	$d(v_6)$
	$\{s\}$	0	2	8	∞	∞	∞	∞
v_1	$\{s, v_1\}$			7	3		8	
v_3	$\{s, v_1, v_3\}$					10	6	12
v_5	$\{s, v_1, v_3, v_5\}$					9		

O Algoritmo de Dijkstra

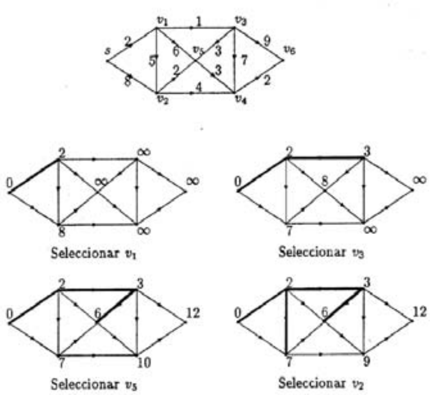


input: digrafo $G = (V, A)$, com custos não negativos
output: caminho mais curto entre o vértice s e todos os outros vértices
begin
  $X := \{s\}$; $\bar{X} := V - \{s\}$; $d(s) := 0$; $pred(s) := -$;
  for all $j \in \bar{X}$ **do**
    if $(s, j) \in A$ **then** $d(j) := c_{sj}$; $pred(j) := s$
      else $d(j) := +\infty$; $pred(j) := 0$
  while $X \neq V$ **do**
    begin
      (**comentário:** selecção de vértice i)
      seja i o vértice de $\bar{X} : d(i) = \min_{j \in \bar{X}}\{d(j)\}$
      (**comentário:** rotulação permanente)
      $X = X \cup \{i\}$; $\bar{X} := \bar{X} - \{i\}$;
      (**comentário:** análise dos sucess. de i com rótulos temporários)
      for all $j \in \bar{X} : (i, j) \in A$ **do**
        if $d(j) > d(i) + c_{ij}$ **then** $d(j) := d(i) + c_{ij}$; $pred(j) := i$;
    end;
end;

Algoritmo de Dijkstra

i	X	$d(s)$	$d(v_1)$	$d(v_2)$	$d(v_3)$	$d(v_4)$	$d(v_5)$	$d(v_6)$
	$\{s\}$	0	2	8	∞	∞	∞	∞
v_1	$\{s, v_1\}$			7	3		8	
v_3	$\{s, v_1, v_3\}$					10	6	12
v_5	$\{s, v_1, v_3, v_5\}$					9		

O Algoritmo de Dijkstra

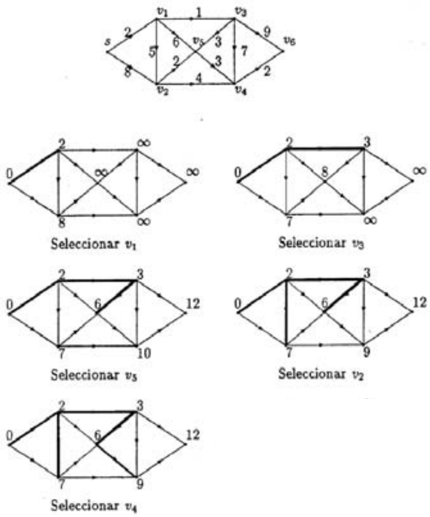


input: digrafo $G = (V, A)$, com custos não negativos
output: caminho mais curto entre o vértice s e todos os outros vértices
begin
  $X := \{s\}$; $\bar{X} := V - \{s\}$; $d(s) := 0$; $pred(s) := -$;
  for all $j \in \bar{X}$ **do**
    if $(s, j) \in A$ **then** $d(j) := c_{sj}$; $pred(j) := s$
      else $d(j) := +\infty$; $pred(j) := 0$
  while $X \neq V$ **do**
    begin
      (**comentário:** selecção de vértice i)
      seja i o vértice de $\bar{X} : d(i) = \min_{j \in \bar{X}}\{d(j)\}$
      (**comentário:** rotulação permanente)
      $X = X \cup \{i\}$; $\bar{X} := \bar{X} - \{i\}$;
      (**comentário:** análise dos sucess. de i com rótulos temporários)
      for all $j \in \bar{X} : (i, j) \in A$ **do**
        if $d(j) > d(i) + c_{ij}$ **then** $d(j) := d(i) + c_{ij}$; $pred(j) := i$;
    end;
end;

Algoritmo de Dijkstra

i	X	$d(s)$	$d(v_1)$	$d(v_2)$	$d(v_3)$	$d(v_4)$	$d(v_5)$	$d(v_6)$
	$\{s\}$	0	2	8	∞	∞	∞	∞
v_1	$\{s, v_1\}$			7	3		8	
v_3	$\{s, v_1, v_3\}$					10	6	12
v_5	$\{s, v_1, v_3, v_5\}$					9		
v_2	$\{s, v_1, v_3, v_5, v_2\}$							

O Algoritmo de Dijkstra

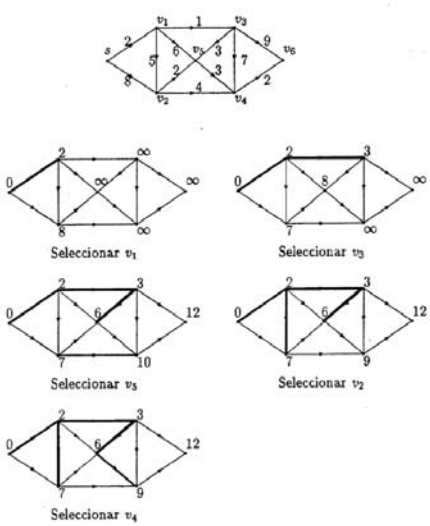


input: digrafo $G = (V, A)$, com custos não negativos
output: caminho mais curto entre o vértice s e todos os outros vértices
begin
  $X := \{s\}$; $\bar{X} := V - \{s\}$; $d(s) := 0$; $pred(s) := -$;
  for all $j \in \bar{X}$ **do**
    if $(s, j) \in A$ **then** $d(j) := c_{sj}$; $pred(j) := s$
      else $d(j) := +\infty$; $pred(j) := 0$
  while $X \neq V$ **do**
    begin
      (**comentário:** selecção de vértice i)
      seja i o vértice de $\bar{X} : d(i) = \min_{j \in \bar{X}} \{d(j)\}$
      (**comentário:** rotulação permanente)
      $X = X \cup \{i\}$; $\bar{X} := \bar{X} - \{i\}$;
      (**comentário:** análise dos sucess. de i com rótulos temporários)
      for all $j \in \bar{X} : (i, j) \in A$ **do**
        if $d(j) > d(i) + c_{ij}$ **then** $d(j) := d(i) + c_{ij}$; $pred(j) := i$;
    end;
end;

Algoritmo de Dijkstra

i	X	$d(s)$	$d(v_1)$	$d(v_2)$	$d(v_3)$	$d(v_4)$	$d(v_5)$	$d(v_6)$
	$\{s\}$	0	2	8	∞	∞	∞	∞
v_1	$\{s, v_1\}$			7	3		8	
v_3	$\{s, v_1, v_3\}$					10	6	12
v_5	$\{s, v_1, v_3, v_5\}$					9		
v_2	$\{s, v_1, v_3, v_5, v_2\}$							

O Algoritmo de Dijkstra

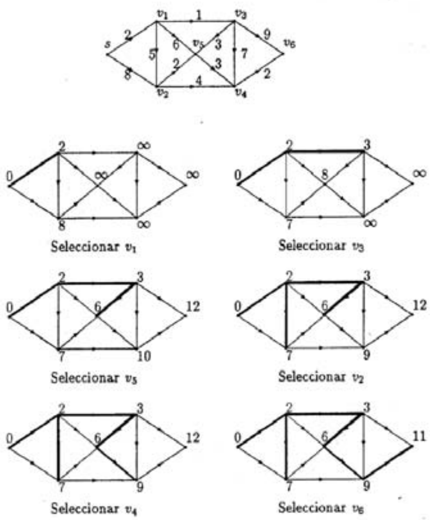


input: digrafo $G = (V, A)$, com custos não negativos
output: caminho mais curto entre o vértice s e todos os outros vértices
begin
  $X := \{s\}$; $\bar{X} := V - \{s\}$; $d(s) := 0$; $pred(s) := -$;
  for all $j \in \bar{X}$ **do**
    if $(s, j) \in A$ **then** $d(j) := c_{sj}$; $pred(j) := s$
      else $d(j) := +\infty$; $pred(j) := 0$
  while $X \neq V$ **do**
    begin
      (**comentário:** selecção de vértice i)
      seja i o vértice de $\bar{X} : d(i) = \min_{j \in \bar{X}} \{d(j)\}$
      (**comentário:** rotulação permanente)
      $X = X \cup \{i\}$; $\bar{X} := \bar{X} - \{i\}$;
      (**comentário:** análise dos sucess. de i com rótulos temporários)
      for all $j \in \bar{X} : (i, j) \in A$ **do**
        if $d(j) > d(i) + c_{ij}$ **then** $d(j) := d(i) + c_{ij}$; $pred(j) := i$;
    end;
end;

Algoritmo de Dijkstra

i	X	$d(s)$	$d(v_1)$	$d(v_2)$	$d(v_3)$	$d(v_4)$	$d(v_5)$	$d(v_6)$
	$\{s\}$	0	2	8	∞	∞	∞	∞
v_1	$\{s, v_1\}$			7	3		8	
v_3	$\{s, v_1, v_3\}$					10	6	12
v_5	$\{s, v_1, v_3, v_5\}$					9		
v_2	$\{s, v_1, v_3, v_5, v_2\}$							
v_4	$\{s, v_1, v_3, v_5, v_2, v_4\}$							11

O Algoritmo de Dijkstra

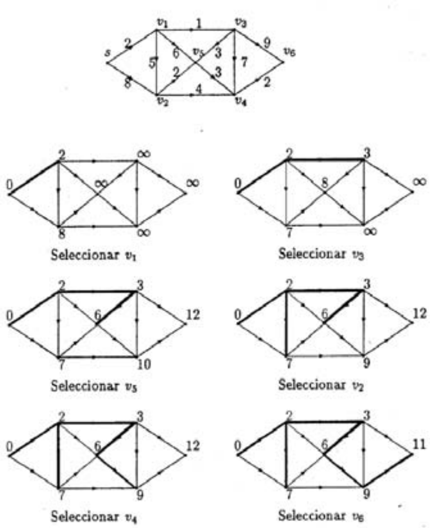



```
input:  digrafo G = (V, A), com custos não negativos
output:  caminho mais curto entre o vértice s e todos os outros vértices
begin
   X := {s};  X̄ := V − {s};  d(s) := 0;  pred(s) := −;
   for all j ∈ X̄ do
      if (s, j) ∈ A then  d(j) := c_sj;  pred(j) := s
                    else  d(j) := +∞;  pred(j) := 0
   while  X ≠ V do
      begin
         (comentário: selecção de vértice i)
         seja i o vértice de X̄ : d(i) = min_{j∈X̄} {d(j)}
         (comentário: rotulação permanente)
         X = X ∪ {i};  X̄ := X̄ − {i};
         (comentário: análise dos sucess. de i com rótulos temporários)
         for all j ∈ X̄ : (i, j) ∈ A do
            if d(j) > d(i) + c_ij then  d(j) := d(i) + c_ij;  pred(j) := i;
      end;
end;
```

Algoritmo de Dijkstra

i	X	$d(s)$	$d(v_1)$	$d(v_2)$	$d(v_3)$	$d(v_4)$	$d(v_5)$	$d(v_6)$
	$\{s\}$	0	2	8	∞	∞	∞	∞
v_1	$\{s, v_1\}$			7	3		8	
v_3	$\{s, v_1, v_3\}$					10	6	12
v_5	$\{s, v_1, v_3, v_5\}$					9		
v_2	$\{s, v_1, v_3, v_5, v_2\}$							
v_4	$\{s, v_1, v_3, v_5, v_2, v_4\}$							11

O Algoritmo de Dijkstra




```
input:  digrafo G = (V, A), com custos não negativos
output:  caminho mais curto entre o vértice s e todos os outros vértices
begin
   X := {s};  X̄ := V − {s};  d(s) := 0;  pred(s) := −;
   for all j ∈ X̄ do
      if (s, j) ∈ A then  d(j) := c_sj;  pred(j) := s
                    else  d(j) := +∞;  pred(j) := 0
   while  X ≠ V do
      begin
         (comentário: selecção de vértice i)
         seja i o vértice de X̄ : d(i) = min_{j∈X̄} {d(j)}
         (comentário: rotulação permanente)
         X = X ∪ {i};  X̄ := X̄ − {i};
         (comentário: análise dos sucess. de i com rótulos temporários)
         for all j ∈ X̄ : (i, j) ∈ A do
            if d(j) > d(i) + c_ij then  d(j) := d(i) + c_ij;  pred(j) := i;
      end;
end;
```

Algoritmo de Dijkstra

i	X	$d(s)$	$d(v_1)$	$d(v_2)$	$d(v_3)$	$d(v_4)$	$d(v_5)$	$d(v_6)$
	$\{s\}$	0	2	8	∞	∞	∞	∞
v_1	$\{s, v_1\}$			7	3		8	
v_3	$\{s, v_1, v_3\}$					10	6	12
v_5	$\{s, v_1, v_3, v_5\}$					9		
v_2	$\{s, v_1, v_3, v_5, v_2\}$							
v_4	$\{s, v_1, v_3, v_5, v_2, v_4\}$							11
v_6	$\{s, v_1, v_3, v_5, v_2, v_4, v_6\}$							

Figura 3.5: Aplicação do algoritmo de Dijkstra

O Algoritmo de Ford

Modelos com custos negativos não podem ser estudados pelo algoritmo de Dijkstra.

Modelo do caminho mais longo.

(caminho mais curto aplicado aos custos simétricos)

Problema: após a atribuição de uma distância pode ser descoberto outro caminho mais curto.

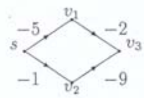
Solução: necessário corrigir rótulo e re-avaliar sucessores – (algoritmo de correcção de rótulos – *label correcting algorithms*)

O Algoritmo de Ford

input: digrafo $G = (V, A)$, com custos arbitrários
output: caminho mais curto entre o vértice s e todos os outros vértices
begin
  $X := \{s\};\ \bar{X} := V - \{s\};\ d(s) := 0;\ pred(s) := -;$
  for all $j \in \bar{X}$ **do**
    $d(j) := +\infty;\ pred(j) := 0;$
  while $X \neq \emptyset$ **do**
    begin
      seleccionar um vértice $i \in X$
      efectuar teste para detectar circuitos negativos
      $X = X - \{i\};\ \bar{X} := \bar{X} \cup \{i\};$
      (**comentário:** análise dos sucessores do vértice i)
      for all $j : (i, j) \in A$ **do**
        if $d(j) > d(i) + c_{ij}$ **then**
          $d(j) := d(i) + c_{ij};\ pred(j) := i;$
          (**comentário:** adicionar j a X se já lá não estiver)
          if $j \in \bar{X}$ **then**
            $X = X \cup \{j\}; \bar{X} = \bar{X} - \{j\};$
    end;
end;

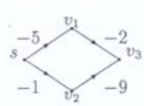
Figura 3.6: Algoritmo de Ford

O Algoritmo de Ford



i	$d(s)$	$d(v_1)$	$d(v_2)$	$d(v_3)$	X
	0	∞	∞	∞	$\{s\}$

O Algoritmo de Ford



i	$d(s)$	$d(v_1)$	$d(v_2)$	$d(v_3)$	X
	0	∞	∞	∞	$\{s\}$

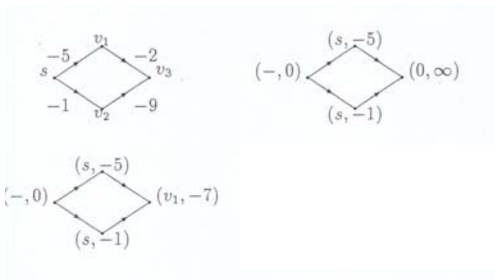
O Algoritmo de Ford

i	$d(s)$	$d(v_1)$	$d(v_2)$	$d(v_3)$	X
	0	∞	∞	∞	$\{s\}$
s		-5	-1		$\{v_1, v_2\}$

O Algoritmo de Ford

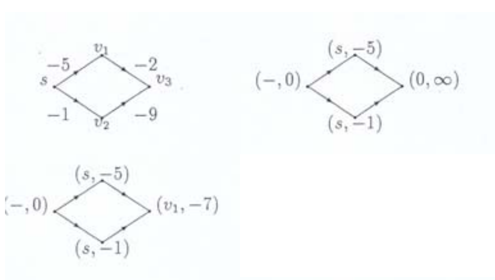
i	$d(s)$	$d(v_1)$	$d(v_2)$	$d(v_3)$	X
	0	∞	∞	∞	$\{s\}$
s		-5	-1		$\{v_1, v_2\}$

O Algoritmo de Ford



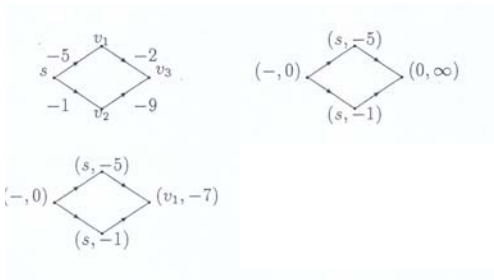
i	$d(s)$	$d(v_1)$	$d(v_2)$	$d(v_3)$	X
	0	∞	∞	∞	$\{s\}$
s		-5	-1		$\{v_1, v_2\}$
v_1					$\{v_2,$

O Algoritmo de Ford



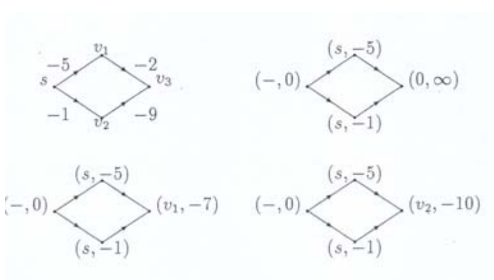
i	$d(s)$	$d(v_1)$	$d(v_2)$	$d(v_3)$	X
	0	∞	∞	∞	$\{s\}$
s		-5	-1		$\{v_1, v_2\}$
v_1				-7	$\{v_2, v_3\}$

O Algoritmo de Ford



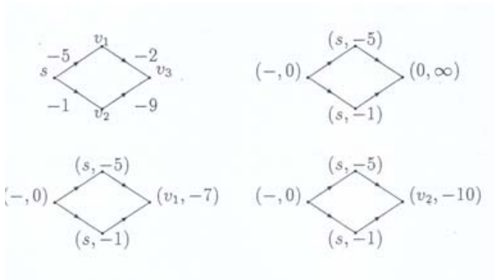
i	$d(s)$	$d(v_1)$	$d(v_2)$	$d(v_3)$	X
	0	∞	∞	∞	$\{s\}$
s		-5	-1		$\{v_1, v_2\}$
v_1				-7	$\{v_2, v_3\}$
v_2					$\{v_3$

O Algoritmo de Ford



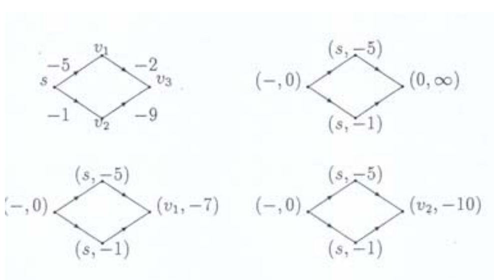
i	$d(s)$	$d(v_1)$	$d(v_2)$	$d(v_3)$	X
	0	∞	∞	∞	$\{s\}$
s		-5	-1		$\{v_1, v_2\}$
v_1				-7	$\{v_2, v_3\}$
v_2				-10	$\{v_3\}$

O Algoritmo de Ford



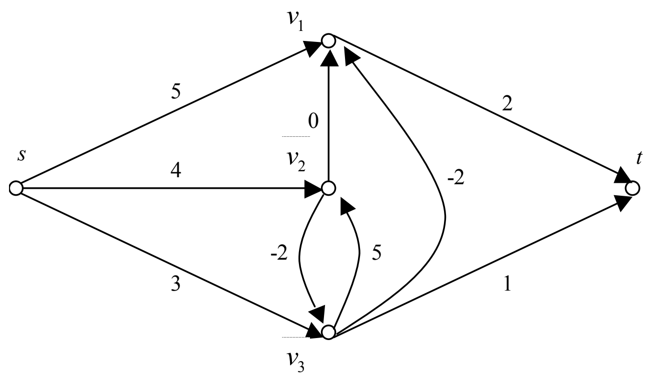
i	$d(s)$	$d(v_1)$	$d(v_2)$	$d(v_3)$	X
	0	∞	∞	∞	$\{s\}$
s		-5	-1		$\{v_1, v_2\}$
v_1				-7	$\{v_2, v_3\}$
v_2				-10	$\{v_3\}$
v_3					

O Algoritmo de Ford



i	$d(s)$	$d(v_1)$	$d(v_2)$	$d(v_3)$	X
	0	∞	∞	∞	$\{s\}$
s		-5	-1		$\{v_1, v_2\}$
v_1				-7	$\{v_2, v_3\}$
v_2				-10	$\{v_3\}$
v_3					$\emptyset$

O Algoritmo de Ford



O Algoritmo de Partição

No algoritmo de Ford, um vértice pode ser analisado mais do que uma vez. -> Não é possível definir a complexidade polinomial do algoritmo. A prática revela um comportamento competitivo.

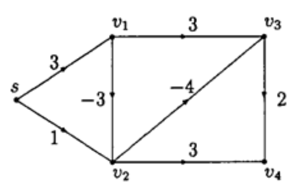
Uma forma eficiente de "manusear" os vértices do conjunto X consiste em particionar a lista em dois subconjuntos. Este procedimento estabelece um limite para o número de iterações.

Complexidade: Polinomial $O(mn)$

O Algoritmo de Partição

```
input:   digrafo G = (V, A), com custos arbitrários
output:  caminho mais curto entre o vértice s e todos os outros vérti
begin
   k := 0;
   NOW := {s}; NEXT := ∅;
   d(s) := 0; pred(s) := −;
   for all j ∉ NOW do
      d(j) := +∞; pred(j) := 0;
   while NOW ≠ ∅ do
      begin
         while NOW ≠ ∅ do
            begin
               seleccionar um vértice i;
               efectuar teste para detectar circuitos negativos;
               NOW := NOW − {i};
               for all j : (i, j) ∈ A do
                  if d(j) > d(i) + c_ij then
                        d(j) := d(i) + c_ij; pred(j) := i;
                        if j ∉ NOW then NEXT := NEXT ∪ {j};
            end;
         k := k + 1;
         NOW := NEXT;
         NEXT := ∅;
      end
end;
```


O Algoritmo de Partição



k	NOW	i	$d(s)$	$d(v_1)$	$d(v_2)$	$d(v_3)$	$d(v_4)$	NEXT
0	$\{s\}$		0	∞	∞	∞	∞	$\emptyset$
	$\{s\}$	s		3	1			$\{v_1, v_2\}$
1	$\{v_1, v_2\}$	v_1			0	6		$\{v_3\}$
	$\{v_2\}$	v_2				-4	3	$\{v_3, v_4\}$
2	$\{v_3, v_4\}$	v_3					2	$\emptyset$
	$\{v_4\}$	v_4						$\emptyset$