Caminos más Cortos desde un Origen a muchos Destinos

Profesor: Julio César López
jlopez@eisc.univalle.edu.co

November 28, 2002

Contenido

Generalidades

- En un **problema de caminos más cortos**, dado un grafo dirigido con peso $G = (V, E)$, se tiene una función de peso $w : E \to \mathcal{R}$ que asigna valores reales a las aristas.
- El **peso** de un camino $p = \langle v_0, v_1, \ldots, v_k \rangle$ es la suma de los pesos de las aristas que lo constituyen:

$$w(p) = \sum_{i=1}^{k} w(v_{i-1}, v_i)$$

Generalidades (cont.)

- Se define el **peso del camino más corto** de u a v como

$$\delta(u,v) = \begin{cases} \min\{w(p) : u \overset{p}{\rightsquigarrow} v\} & \text{si existe un camino de } u \text{ a } v \\ \infty & \text{de lo contrario} \end{cases}$$

- Un **camino más corto** del vértice v al vértice u es entonces definido como cualquier camino p con peso $w(p) = \delta(u,v)$.

Contents

Page 5 of 42

Full Screen

Quit

Generalidades (cont.)

El **problema del camino más corto desde un origen a muchos destinos** es el siguiente:

> Dado un grafo $G = (V, E)$ se desea encontrar un camino más corto desde un vértice **origen** dado $s \in V$ a cada vértice $v \in V$.

Contents

Page 6 of 42

Full Screen

Quit

Generalidades (cont.)

- Algunas veces, en los problemas de caminos más cortos existen aristas cuyo peso es negativo.
- Si el grafo $G = (V, E)$ contiene ciclos con peso no negativo alcanzable desde el origen s, entonces para todo $v \in V$, el peso del camino más corto $\delta(s, v)$ continua bien definido, incluso si tiene valor negativo.

Contents

Page 7 of 42

Full Screen

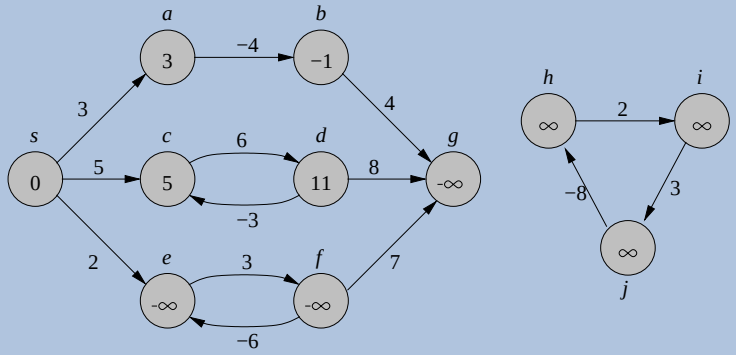
Quit

Generalidades (cont.)

- Sin embargo, si existe un ciclo con peso negativo alcanzable desde s, el peso del camino más corto no esta bien definido.
- Ningún camino de s a un vértice en el ciclo puede ser un camino más corto (siempre habrá un camino "más corto").
- Si existe un ciclo con peso negativo en algún camino de s a v, se define $\delta(s, v) = -\infty$.

Generalidades (cont.)

Ejemplo:



- Como los vértices e y f forman un ciclo con peso negativo alcanzable desde s, ellos tienen pesos de caminos más cortos de $-\infty$.
- Como el vértice g es alcanzable desde un vértice cuyo peso de camino más corto es $-\infty$, entonces él también tiene peso de camino más corto de $-\infty$.
- Los vértices h, i y j no son alcanzables desde s, luego su peso de camino más corto es ∞ (aún cuando están en un ciclo de peso negativo.

Relajación

Para cada vértice $v \in V$ se tiene un atributo $d[v]$, el cual es un límite superior en el peso del camino más corto desde el origen s a v. El valor $d[v]$ se denomina el **peso estimado del camino más corto**.

Se inicializan los pesos estimados de los caminos más cortos y predecesores con el siguiente procedimiento ($\Theta(V)$):

INITIALIZE-SINGLE-SOURCE(G,s)
1 **for** each vertex $v \in V[G]$
2 **do** $d[v] \leftarrow \infty$
3 $\pi[v] \leftarrow$ NIL
4 $d[s] \leftarrow 0$

Relajación (cont.)

El proceso de **relajación** de una arista (u, v) consiste en examinar si se puede mejorar el camino más corto a v pasando por u y, si es el caso, actualizar $d[v]$ y $\pi[v]$.

La relajación puede decrementar el valor del peso estimado del camino más corto $d[v]$ y actualizar el campo $\pi[v]$ del predecesor de v.

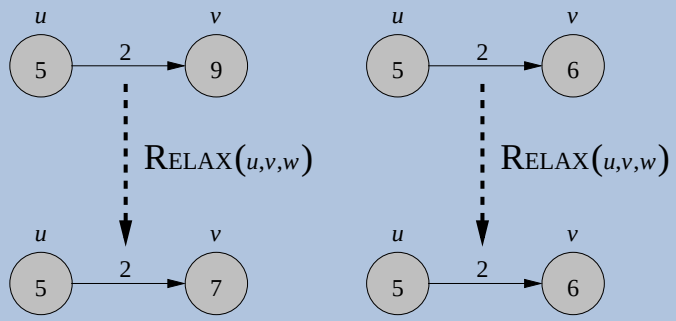
Relajación (cont.)

El siguiente código realiza una relajación de la arista (u, v):

RELAX(u,v,w)
1 **if** $d[v] > d[u] + w(u, v)$
2 **then** $d[v] \leftarrow d[u] + w(u, v)$
3 $\pi[v] \leftarrow u$

Relajación (cont.)

La siguiente figura muestra dos ejemplos de relajación de una arista, uno en el cual el peso estimado del camino más corto decrece y el otro en el que no hay cambios en el peso estimado:



Contents
Page 12 of 42
Full Screen
Quit

Contents

Page 13 of 42

Full Screen

Quit

Relajación (cont.)

Lema 1 (Desigualdad Triangular)
Para cualquier arista $(u, v) \in E$, se tiene que $\delta(s, v) \leq \delta(s, u) + w(u, v)$.

Relajación (cont.)

Lema 2 (Propiedad del Límite Superior)
Siempre se tiene que $d[v] \geq \delta(s, v)$ para todo $v \in V$, y una vez que $d[v]$ tenga el valor $\delta(s, v)$, nunca cambia.

Relajación (cont.)

Corolario 3 (Propiedad de Ningún Camino)
Si no existe camino desde s a v, entonces siempre se tiene que $d[v] = \delta(s, v) = \infty$.

Contents

Page 16 of 42

Full Screen

Quit

Relajación (cont.)

Lema 4 (Propiedad de Convergencia)
Si $s \rightsquigarrow u \rightarrow v$ es un camino más corto en G para algún $u, v \in V$, y si $d[u] = \delta(s, u)$ antes de la relajación de la arista (u, v), entonces $d[v] = \delta(s, v)$ a partir de ese momento..

Contents

Page 17 of 42

Full Screen

Quit

Relajación (cont.)

Lema 5
(Propiedad de Relajación de Camino)
Si $p = \langle v_0, v_1, \ldots, v_k \rangle$ es un camino más corto desde $s = v_0$ a v_k, y las aristas de p están relajadas en el orden $(v_0, v_1), (v_1, v_2), \ldots, (v_{k-1}, v_k)$, entonces $d[v_k] = \delta(s, v_k)$.

Esta propiedad se cumple sin tener en cuenta otras relajaciones que ocurran, incluso si ellas están mezcladas con relajaciones de aristas de p.

Relajación (cont.)

Lema 6
(Propiedad del Subgrafo Predecesor)
Una vez que $d[v] = \delta(s, v)$ para todo $v \in V$, el subgrafo predecesor es un árbol de caminos más cortos con raíz s.

Contents

Page 19 of 42

Full Screen

Quit

El Algoritmo Bellman-Ford

El **algoritmo Bellman-Ford** resuelve el problema de caminos más cortos desde un origen a muchos destinos en el caso general en el cual los pesos de las aristas pueden ser negativos.

Dado un grafo dirigido con peso $G = (V, E)$ con origen s y función de peso $w : E \to \mathcal{R}$, el algoritmo retorna un valor booleano que indica si existe o no un ciclo de peso negativo que es alcanzable desde el origen.

Si no hay un ciclo de peso negativo, el algoritmo produce los caminos más cortos y sus pesos.

El Algoritmo Bellman-Ford (cont.)

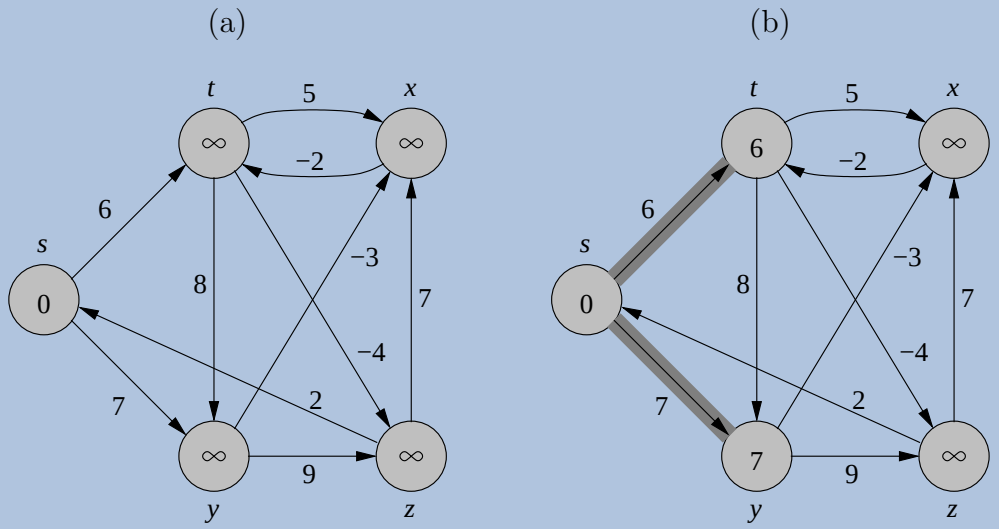
El siguiente es el código del algoritmo:

```
BELLMAN-FORD(G,w,s)
1  INITIALIZE-SINGLE-SOURCE(G,s)
2  for i ← 1 to |V[G]| − 1
3        do for each edge (u, v) ∈ E[G]
4              do RELAX(u,v,w)
5  for each edge (u, v) ∈ E[G]
6        do if d[v] > d[u] + w(u, v)
7              then return FALSE
8  return TRUE
```

Este algoritmo corre en un tiempo $O(VE)$ ya que la inicialización toma un tiempo $\Theta(V)$, cada uno de los $|V| - 1$ pasos sobre las aristas toma un tiempo $\Theta(E)$, y el último ciclo **for** toma un tiempo $O(E)$.

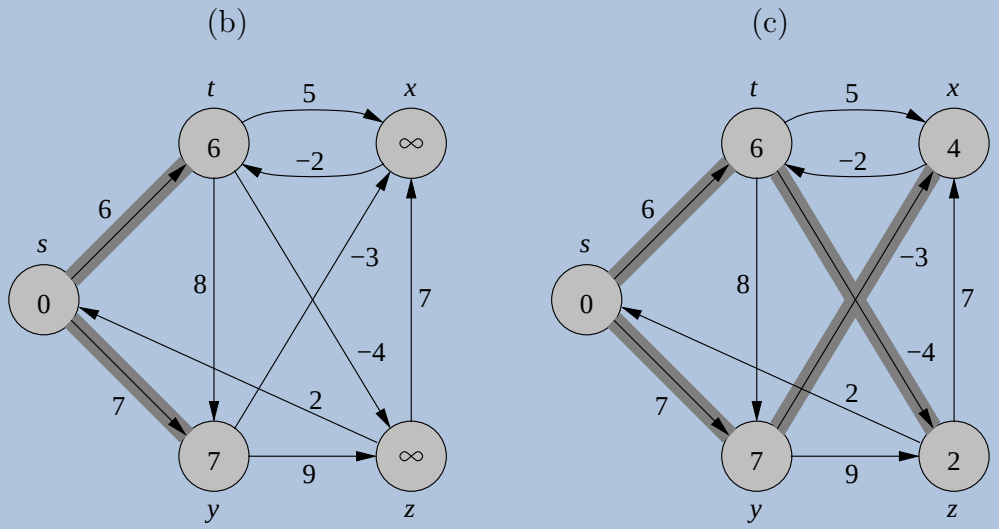
El Algoritmo Bellman-Ford (cont.)

El siguiente ejemplo muestra la ejecución del algoritmo en un grafo de 5 vértices:



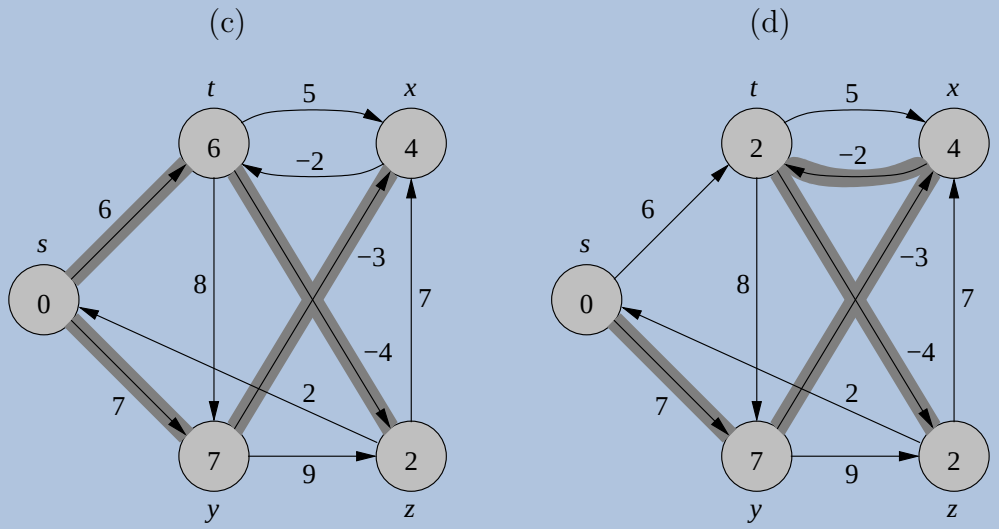
Contents
Page 21 of 42
Full Screen
Quit

El Algoritmo Bellman-Ford (cont.)



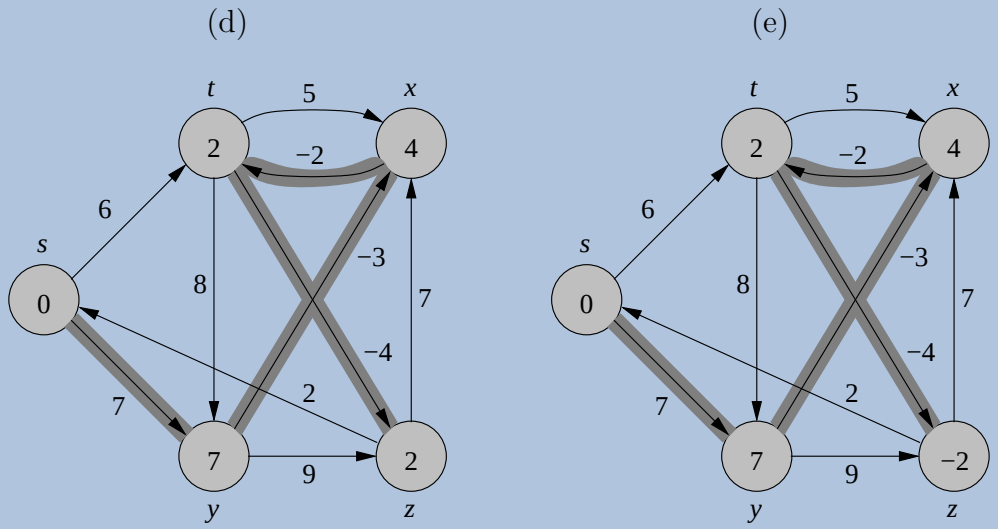
Contents
Page 22 of 42
Full Screen
Quit

El Algoritmo Bellman-Ford (cont.)



Contents
Page 23 of 42
Full Screen
Quit

El Algoritmo Bellman-Ford (cont.)



El Algoritmo Bellman-Ford (cont.)

Lema 7

Suponga que el procedimiento BELLMAN-FORD se aplica a un grafo dirigido con peso $G = (V, E)$ con origen s y función de peso $w : E \to \mathcal{R}$.

Si G contiene ciclos con peso no negativo que son alcanzables desde s, entonces el algoritmo retorna TRUE, se tiene que $d[v] = \delta(s, v)$ para todo $v \in V$, y el subgrafo predecesor G_π es un árbol de rutas más cortas con raíz s.

Si G contiene ciclos con peso negativo alcanzable desde s, entonces el algoritmo retorna FALSE.

Contents

Page 26 of 42

Full Screen

Quit

Caminos más Cortos desde un Origen a muchos Destinos en Grafos Dirigidos Acíclicos

Cuando se relajan las aristas de un gda (grafo dirigido acíclico) con peso $G = (V, E)$ de acuerdo al orden topológico de sus vértices, se puede calcular caminos más cortos desde un origen a muchos destinos en un tiempo $\Theta(V + E)$.

Los caminos más cortos están siempre bien definidos en un gda, ya que si existen aristas con peso negativo, por definicion no pueden existir ciclos.

Contents

Page 27 of 42

Full Screen

Quit

Caminos más Cortos desde un Origen a muchos Destinos en GDA (cont.)

El procedimiento es el siguiente:

DAG-SHORTEST-PATHS(G,w,s)
1 topologically sort the vertices of G
2 INITIALIZE-SINGLE-SOURCE(G,s)
3 **for** each vertex u,
taken in topologically sorted order
4 **do for** each vertex $v \in Adj[u]$
5 **do** RELAX(u,v,w)

Contents

Page 28 of 42

Full Screen

Quit

Caminos más Cortos desde un Origen a muchos Destinos en GDA (cont.)

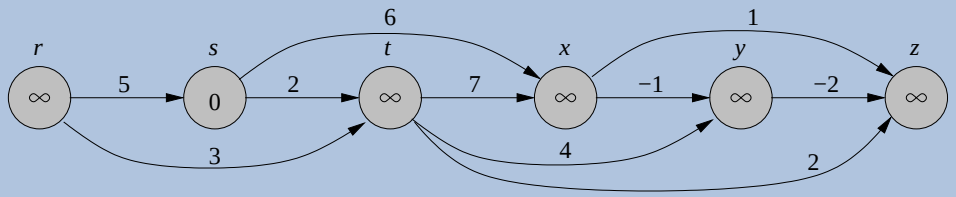
- El orden topológico puede ser realizado en un tiempo $\Theta(V + E)$.
- El llamado a INITIALIZE-SINGLE-SOURCE toma un tiempo $\Theta(V)$.
- El ciclo interno hace $|E|$ iteraciones que toman un tiempo $\Theta(1)$ cada una, mientras que el ciclo externo hace una sola iteración por vértice.

Por lo tanto el tiempo total de ejecución de este algoritmo es $\Theta(V + E)$.

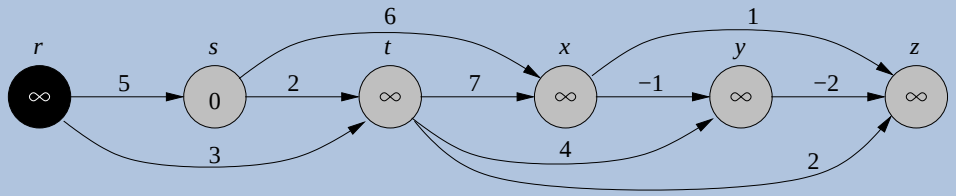
Caminos más Cortos desde un Origen a muchos Destinos en GDA (cont.)

El siguiente ejemplo muestra la ejecución de este algoritmo:

(a)

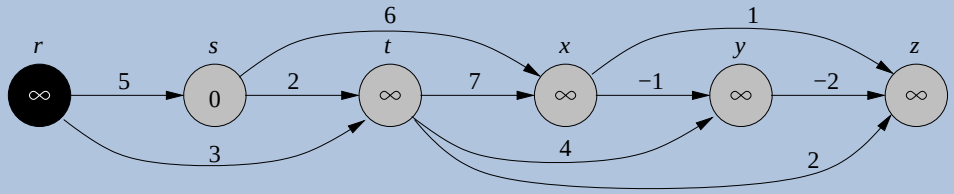


(b)

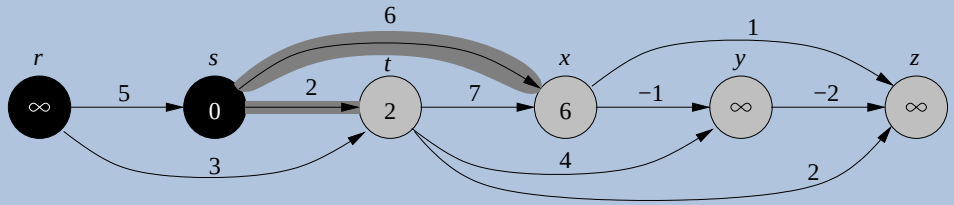


Caminos más Cortos desde un Origen a muchos Destinos en GDA (cont.)

(b)



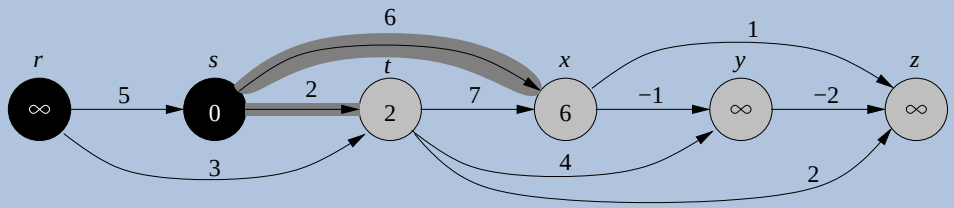
(c)



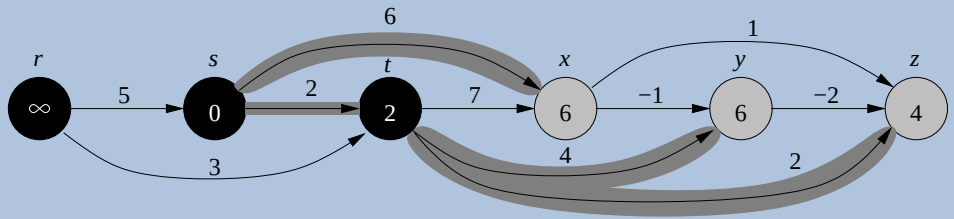
Contents
Page 30 of 42
Full Screen
Quit

Caminos más Cortos desde un Origen a muchos Destinos en GDA (cont.)

(c)

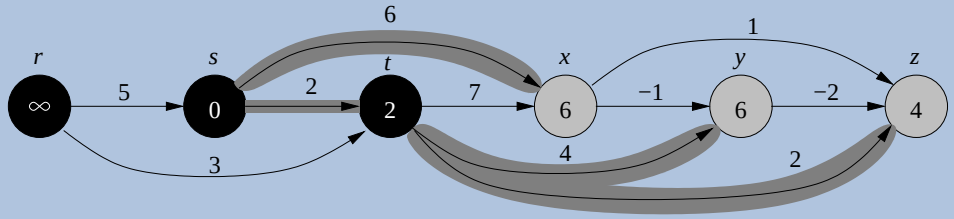


(d)

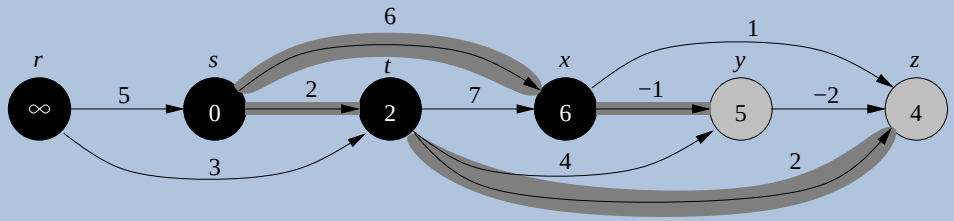


Caminos más Cortos desde un Origen a muchos Destinos en GDA (cont.)

(d)

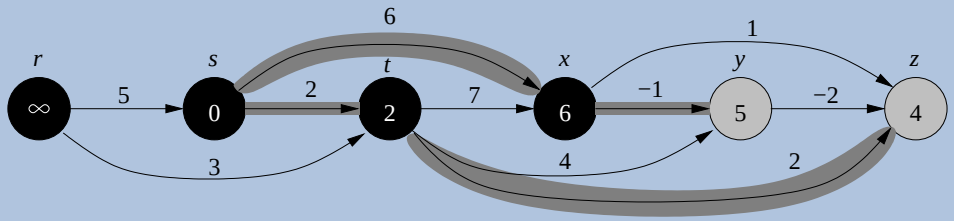


(e)

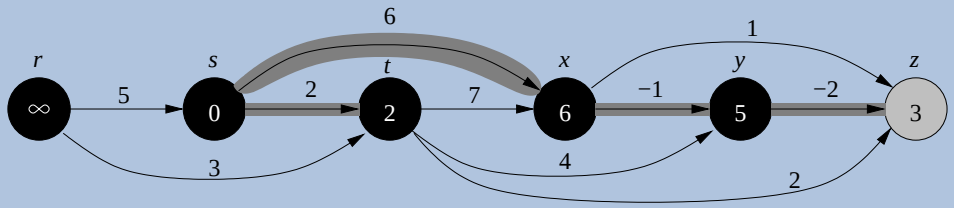


Caminos más Cortos desde un Origen a muchos Destinos en GDA (cont.)

(e)

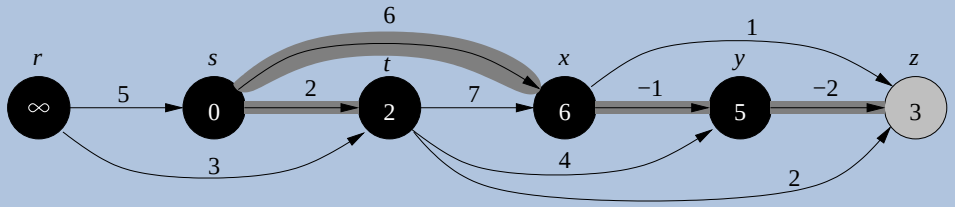


(f)

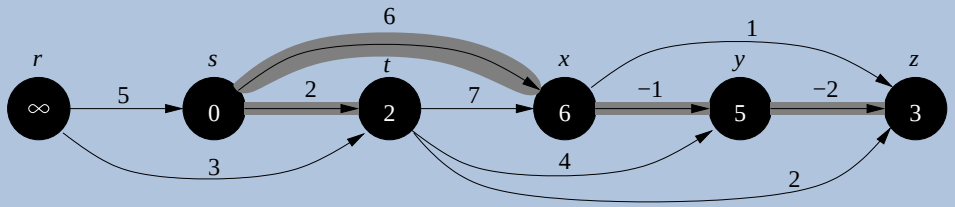


Caminos más Cortos desde un Origen a muchos Destinos en GDA (cont.)

(f)



(g)



Caminos más Cortos desde un Origen a muchos Destinos en GDA (cont.)

Teorema 8
Si un grafo dirigido con peso $G = (V, E)$ tiene un vértice origen s y no tiene ciclos, entonces al finalizar el procedimiento DAG-SHORTEST-PATH, $d[v] = \delta(s, v)$ para todo $v \in V$, y el subgrafo predecesor G_π es un árbol de caminos más cortos.

El Algoritmo de Dijkstra

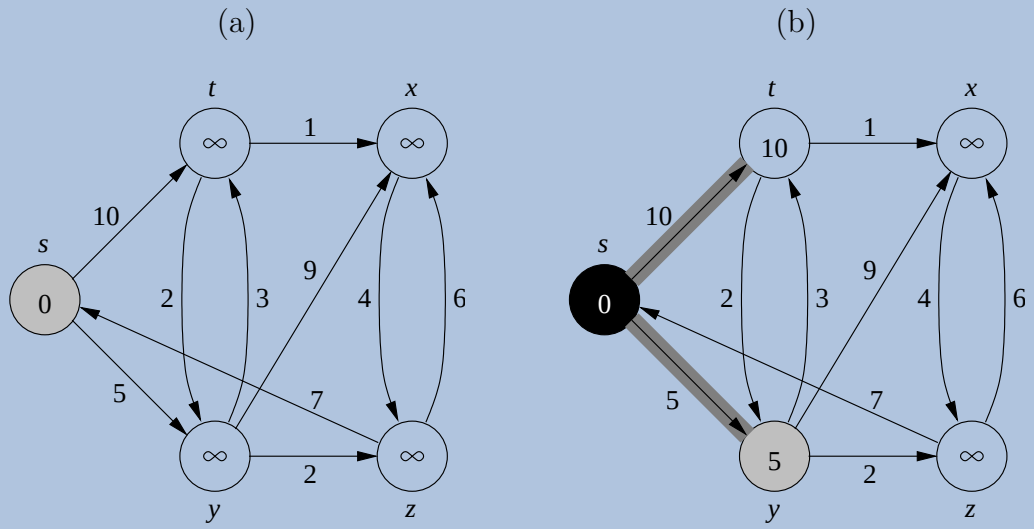
El algoritmo de Dijkstra resuelve el problema de caminos más cortos desde un origen a muchos destinos en un grafo dirigido con peso $G = (V, E)$ para el caso en el cual todas las aristas tienen un peso no negativo.

El algoritmo es el siguiente:

```
DIJKSTRA(G,w,s)
1  INITIALIZE-SINGLE-SOURCE(G,s)
2  S ← ∅
3  Q ← V[G]
4  while Q ≠ ∅
5        do u ← EXTRACT-MIN(Q)
6           S ← S ∪ {u}
7           for each vertex v ∈ Adj[u]
8                 do RELAX(u,v,w)
```

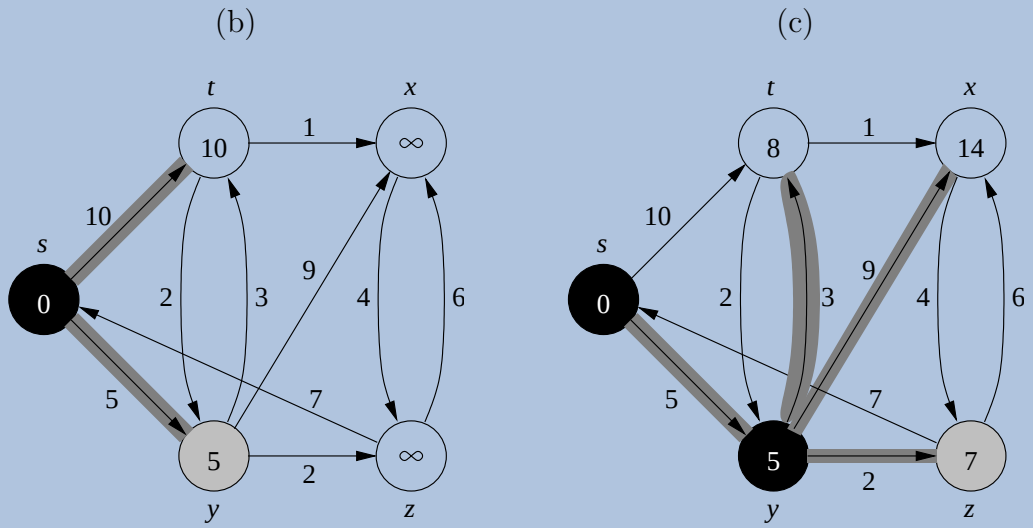
El Algoritmo de Dijkstra (cont.)

El algoritmo de Dijkstra en ejecución se muestra en el siguiente ejemplo:

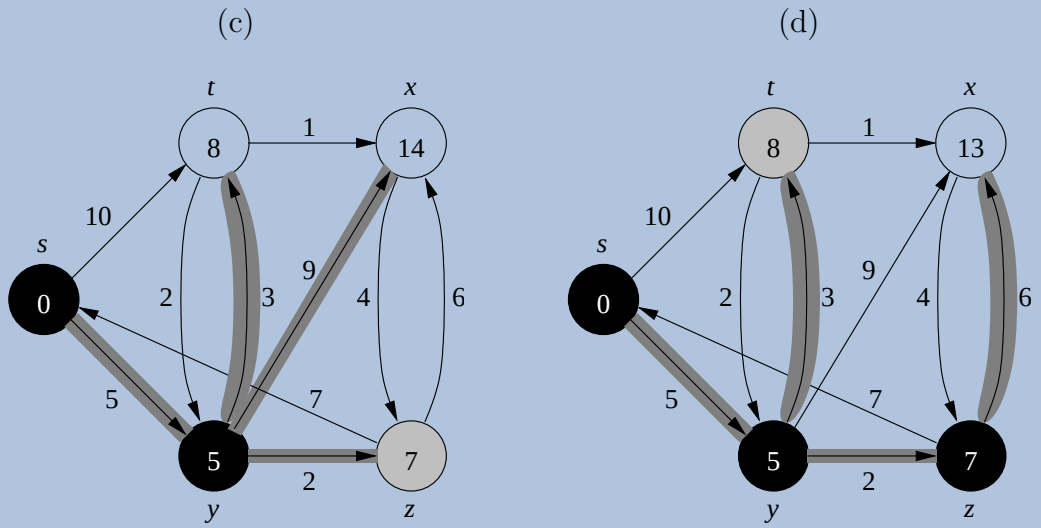


Contents
Page 37 of 42
Full Screen
Quit

El Algoritmo de Dijkstra (cont.)



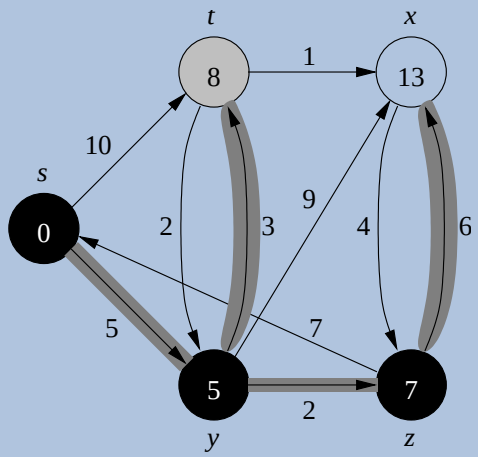
El Algoritmo de Dijkstra (cont.)



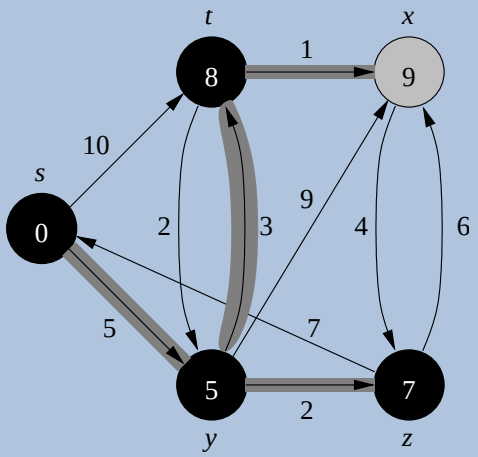
Contents
Page 39 of 42
Full Screen
Quit

El Algoritmo de Dijkstra (cont.)

(d)

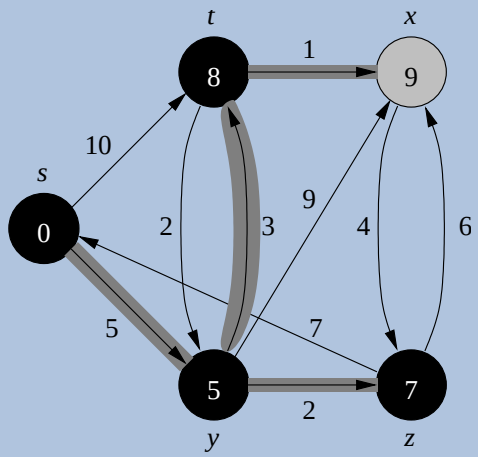


(e)

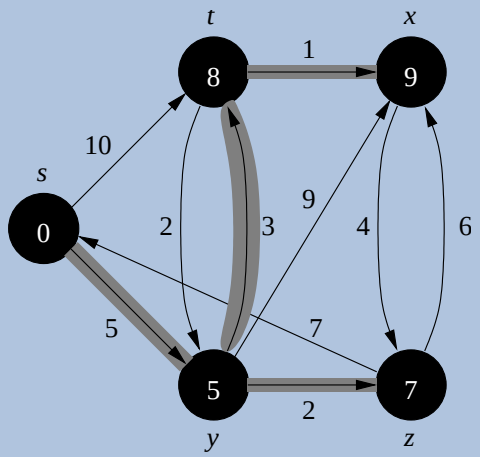


El Algoritmo de Dijkstra (cont.)

(e)



(f)



Contents

Page 42 of 42

Full Screen

Quit

El Algoritmo de Dijkstra (cont.)

Teorema 9
El algoritmo de Dijkstra corriendo en un grafo dirigido con peso $G = (V, E)$ con una función de peso no negativo w y origen s, termina con $d[u] = \delta(s, u)$ para todo $v \in V$.