Algoritmos y estructuras de datos

P/A: Camino Mínimo.

CEIS

Escuela Colombiana de Ingeniería

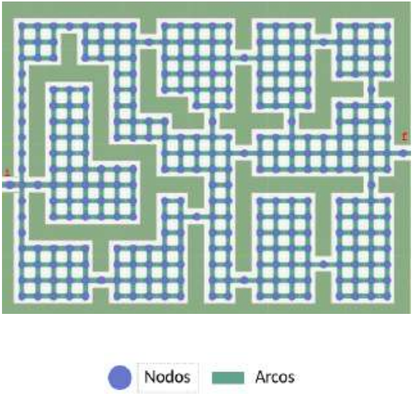
Agenda

Camino Mínimo

Camino

Dado un grafo dirigido, $G = (V, E)$
Un **camino** p entre dos vértices i y f es una secuencia de vértices $p =< v_0, v_1, .., v_k >$, con

- $i = v_0$ y $f = v_k$
- $\forall i : 0..k, v_i \in V$
- $\forall i : 0...k-1, (v_i, v_{i+1}) \in E$

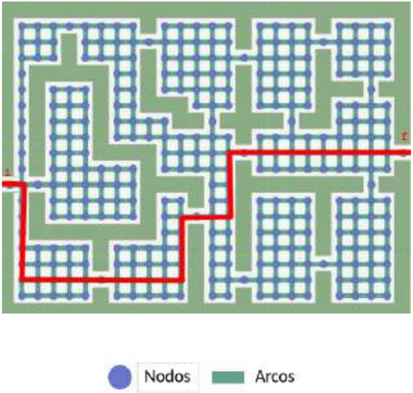


Camino Mínimo

Camino

Dado un grafo dirigido, $G = (V, E)$
Un **camino** p entre dos vértices i y f es una secuencia de vértices $p =< v_0, v_1, .., v_k >$, con

- $i = v_0$ y $f = v_k$
- $\forall i : 0..k, v_i \in V$
- $\forall i : 0...k-1, (v_i, v_{i+1}) \in E$

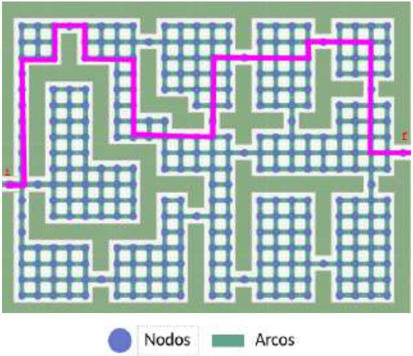


Camino Mínimo

Camino

Dado un grafo dirigido, $G = (V, E)$
Un **camino** p entre dos vértices i y f es una secuencia de vértices $p =< v_0, v_1, .., v_k >$, con

- $i = v_0$ y $f = v_k$
- $\forall i : 0..k, v_i \in V$
- $\forall i : 0...k - 1, (v_i, v_{i+1}) \in E$



Camino Mínimo

Peso de un camino

Dado un grafo dirigido, $G = (V, E)$, con una función de peso $w : E \rightarrow R$
El **peso de un camino** $w(p)$
$p =< v_0, v_1, .., v_k >$, es la suma de los pesos de los arcos correspondientes.

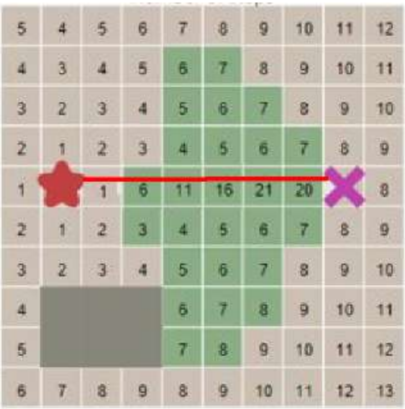
$$w(p) = \sum_{i=1}^{k} w(v_{i-1}, v_i)$$

Camino Mínimo

Peso de un camino

Dado un grafo dirigido, $G = (V, E)$,
con una función de peso $w : E \to R$
El **peso de un camino** $w(p)$
$p =< v_0, v_1, .., v_k >$, es la suma de los pesos de los arcos correspondientes.

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

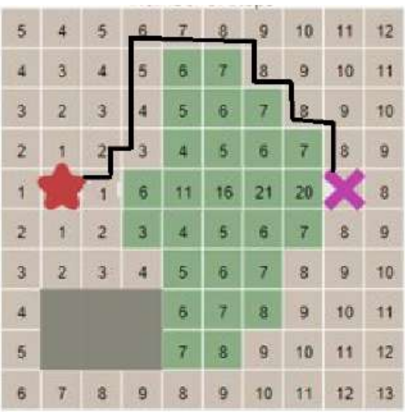


Camino Mínimo

Peso de un camino

Dado un grafo dirigido, $G = (V, E)$, con una función de peso $w : E \to R$
El **peso de un camino** $w(p)$
$p =< v_0, v_1, .., v_k >$, es la suma de los pesos de los arcos correspondientes.

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



Camino Mínimo

Ruta más corta

Dado un grafo dirigido, $G = (V, E)$, con una función de peso $w : E \to R$
La ruta de peso más corta de u a v

$$\delta(u,v) = \begin{cases} \min\{w(p) : u \overset{p}{\leadsto} v\} & \text{if there is a path from } u \text{ to } v\ , \\ \infty & \text{otherwise}\ . \end{cases}$$

Problema - Solución

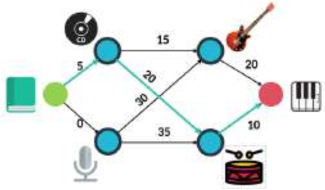
Rutas más cortas desde una fuente

$s \longrightarrow v_*$

Dado un vértice fuente $s \in V$, encontrar la ruta mas corta a cada uno de los diferentes vértices $v \in V$.

Otros problemas

- *Single-destination shortest-paths problem*
- *Single-pair shortest-path problem*
- *All-pairs shortest-paths problem*



Problema - Solución

Rutas más cortas desde una fuente

$s \longrightarrow v_*$

Dado un vértice fuente $s \in V$, encontrar la ruta mas corta a cada uno de los diferentes vértices $v \in V$.

Otros problemas

- *Single-destination shortest-paths problem*

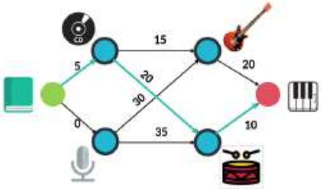
  Reversar la dirección de cada arco ($s \longleftarrow v_*$)

- *Single-pair shortest-path problem*

  Calcular las rutas más cortas desde u ($u \longrightarrow v_*$)

- *All-pairs shortest-paths problem*

  Calcular las rutas más cortas desde cada vértice$u$ ($u \longrightarrow v_*$)



Camino Mínimo

Relajación

Los algoritmos de solución usan la técnica de **relajación**.

- $\forall v \in V$, $v.d$ es el peso de la ruta más corta estimada hacia v.
- $\forall v \in V$, $v.\pi$ es predecesor de v.

```
INITIALIZE-SINGLE-SOURCE(G, s)
for each vertex v ∈ G.V
    v.d = ∞
    v.π = NIL
s.d = 0
```

Relajación

El proceso de relajación del arco (u, v) consiste en:

- Verificar si se puede mejorar la ruta más corta a v, hallada hasta el momento, pasando por u
- Si se puede, actualizar $v.d$ y $v.\pi$

RELAX(u, v, w)

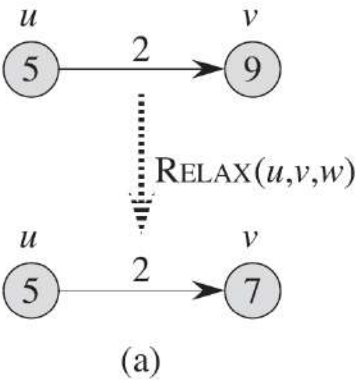
if $v.d > u.d + w(u, v)$
    $v.d = u.d + w(u, v)$
    $v.\pi = u$

Camino Mínimo

Relajación

El proceso de relajación del arco (u, v) consiste en:

- Verificar si se puede mejorar la ruta más corta a v, hallada hasta el momento, pasando por u
- Si se puede, actualizar $v.d$ y $v.\pi$

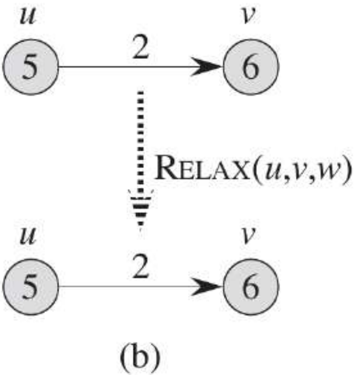


Camino Mínimo

Relajación

El proceso de relajación del arco (u, v) consiste en:

- Verificar si se puede mejorar la ruta más corta a v, hallada hasta el momento, pasando por u
- Si se puede, actualizar $v.d$ y $v.\pi$



Camino mínimo

Bellman Ford

1. Relajar los arcos, haciendo que disminuya progresivamente un estimado $v.d$ hasta lograr el camino mínimo de s a v.
2. Retorna si desde la fuente se llega a hay un ciclo de peso negativo

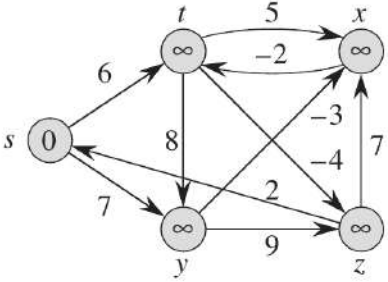
$$\Theta(VE)$$

```
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
```

Camino Mínimo

Bellman Ford

```
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
```

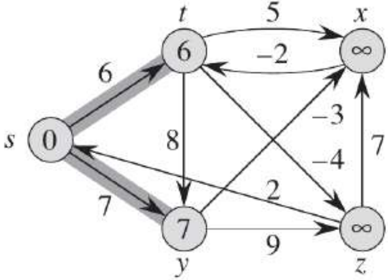


(a)

Camino Mínimo

Bellman Ford

```
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
```

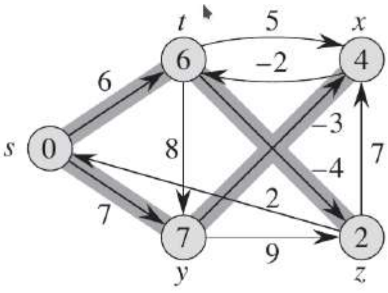


(b)

Camino Mínimo

Bellman Ford

```
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
```

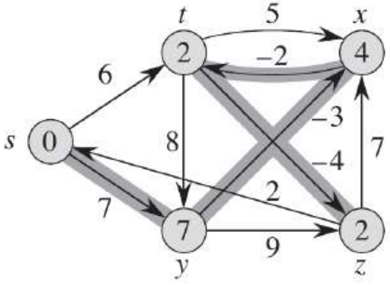


(c)

Camino Mínimo

Bellman Ford

```
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
```

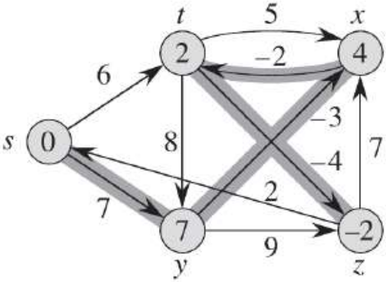


(d)

Camino Mínimo

Bellman Ford

```
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
```



(e)

Camino mínimo

Dijkstra

INV: S tiene los vértices de los que se conoce la ruta mínima a la fuente s

1. Seleccionar u de $V - S$ con ruta mínima estimada
2. Adicionar u a S
3. Relajar todos los arcos que parten de u
4. Repetir el paso 1 hasta cubrir todos los vértices

Los pesos no pueden ser negativos

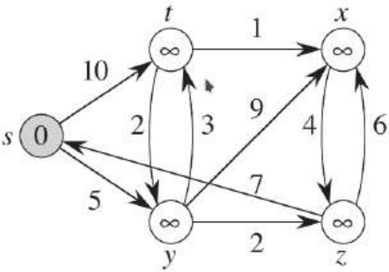
$$\Theta(V^2)$$

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

Camino Mínimo

Dijkstra

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

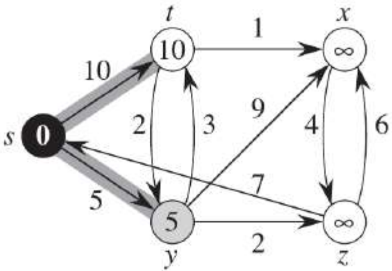


(a)

Camino Mínimo

Dijkstra

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

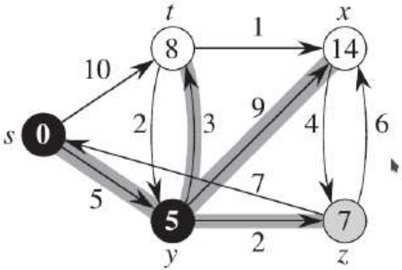


(b)

Camino Mínimo

Dijkstra

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

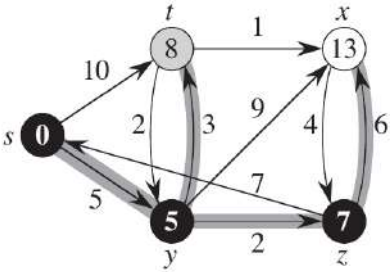


(c)

Camino Mínimo

Dijkstra

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

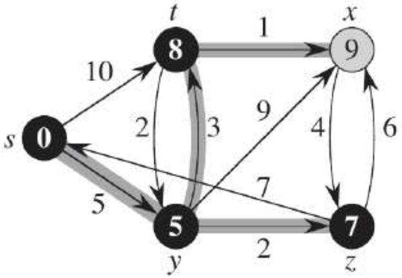


(d)

Camino Mínimo

Dijkstra

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

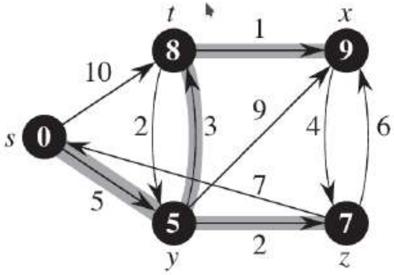


(e)

Camino Mínimo

Dijkstra

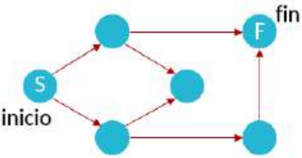
```
DIJKSTRA(G, w, s)
  INITIALIZE-SINGLE-SOURCE(G, s)
  S = ∅
  Q = G.V
  while Q ≠ ∅
      u = EXTRACT-MIN(Q)
      S = S ∪ {u}
      for each vertex v ∈ G.Adj[u]
          RELAX(u, v, w)
```



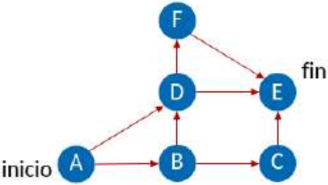
(f)

Ejercicios

1. Ejecute el algoritmo BFS en los siguientes grafos para obtener la respuesta solicitada:



Encuentre la longitud de la ruta más corta de inicio a fin



Encuentre la longitud de la ruta más corta de A a E

2. Elabore una lista del orden en que las siguientes actividades pueden realizarse (una tarea es dependiente de otra si existe un arco entre ellas)

3. Para cada uno de los siguientes grafos, ¿cuál es el peso de la ruta más corta de principio a fin?