

# Algoritmos y estructuras de datos

P/A: Camino Mínimo.

CEIS

Escuela Colombiana de Ingeniería

2024-2

# Agenda

## ① Camino Mínimo

Conceptos

Problema - Solución

Bellman-Ford

Dijkstra

## ② Aspectos finales

Ejercicios

# Camino Mínimo

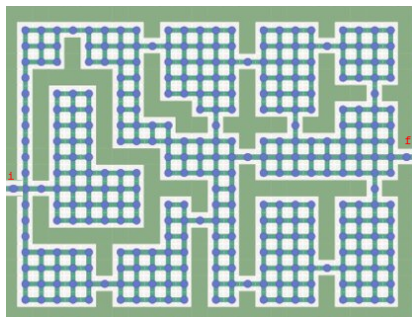
## Camino

Dado un grafo dirigido,  $G = (V, E)$

Un **camino**  $p$  entre dos vértices,  $i$  y  $f$ , es una secuencia de vértices

$p = \langle v_0, v_1, \dots, v_k \rangle$ , con

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



Nodos



Arcos

# Camino Mínimo

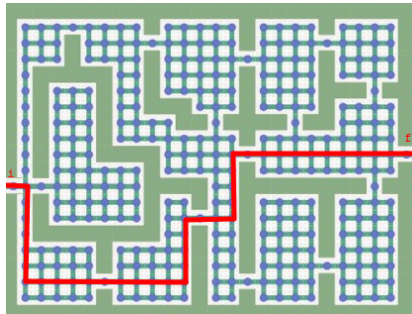
## Camino

Dado un grafo dirigido,  $G = (V, E)$

Un **camino**  $p$  entre dos vértices,  $i$  y  $f$ , es una secuencia de vértices

$p = \langle v_0, v_1, \dots, v_k \rangle$ , con

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



# Camino Mínimo

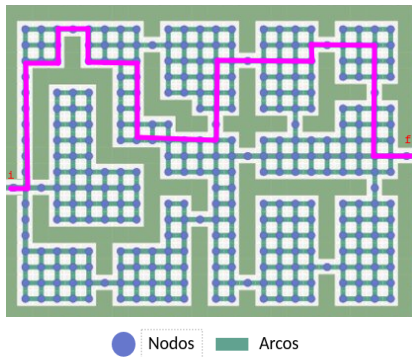
## Camino

Dado un grafo dirigido,  $G = (V, E)$

Un **camino**  $p$  entre dos vértices,  $i$  y  $f$ , es una secuencia de vértices

$p = \langle v_0, v_1, \dots, v_k \rangle$ , 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**  $w(p)$

$p = \langle v_0, v_1, \dots, v_k \rangle$ , es la suma de los  
pesos de los arcos correspondientes.

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

5	4	5	6	7	8	9	10	11	12
4	3	4	5	6	7	8	9	10	11
3	2	3	4	5	6	7	8	9	10
2	1	2	3	4	5	6	7	8	9
1		1	6	11	16	21	20		8
2	1	2	3	4	5	6	7	8	9
3	2	3	4	5	6	7	8	9	10
4				6	7	8	9	10	11
5				7	8	9	10	11	12
6	7	8	9	8	9	10	11	12	13

# Camino Mínimo

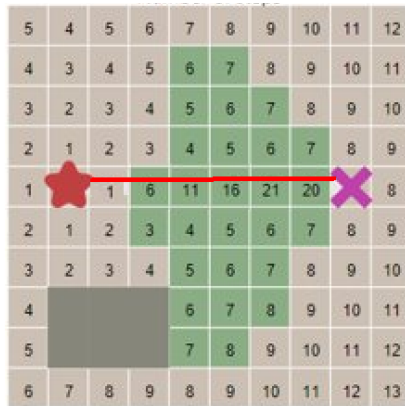
## Peso de un camino

Dado un grafo dirigido,  $G = (V, E)$ ,  
con una función de peso  $w : E \rightarrow R$

El **peso de un camino**  $w(p)$

$p = \langle v_0, v_1, \dots, v_k \rangle$ , 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

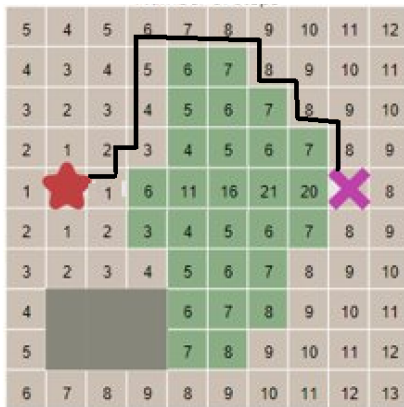
Dado un grafo dirigido,  $G = (V, E)$ ,

con una función de peso  $w : E \rightarrow \mathbb{R}$

**El peso de un camino  $w(p)$**

$p = \langle v_0, v_1, \dots, v_k \rangle$ , 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  $w : E \rightarrow R$

**La ruta de peso más corta de  $u$  a  $v$**

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

5	4	5	6	7	8	9	10	11	12
4	3	4	5	6	7	8	9	10	11
3	2	3	4	5	6	7	8	9	10
2	1	2	3	4	5	6	7	8	9
1		1	6	11	16	21	20		8
2	1	2	3	4	5	6	7	8	9
3	2	3	4	5	6	7	8	9	10
4				6	7	8	9	10	11
5				7	8	9	10	11	12
6	7	8	9	8	9	10	11	12	13

# Problema - Solución

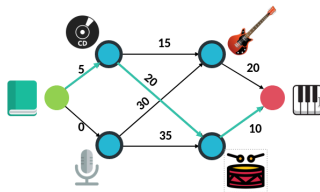
## Rutas más cortas desde una fuente

$s \rightarrow v_*$

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



# Problema - Solución

## Rutas más cortas desde una fuente

$$s \rightarrow v_*$$

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-path problem*

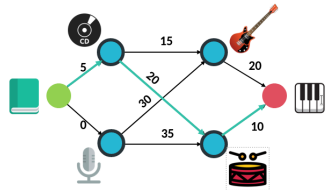
Reversar la dirección de cada arco ( $s \leftarrow v_*$ )

- *Single-pair shortest-path problem*

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

- *All-pairs shortest-paths problem*

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



# 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( $G, s$ )

```
1  for each vertex  $v \in G.V$ 
2       $v.d = \infty$ 
3       $v.\pi = \text{NIL}$ 
4   $s.d = 0$ 
```

# 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  $v.\pi$

RELAX( $u, v, w$ )

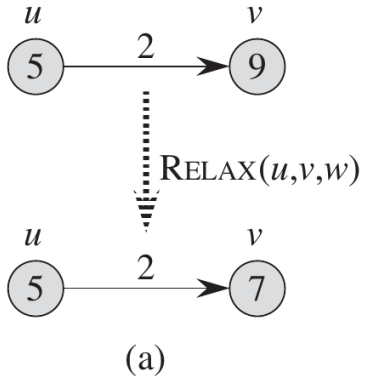
```
1  if  $v.d > u.d + w(u, v)$   
2       $v.d = u.d + w(u, v)$   
3       $v.\pi = u$ 
```

# 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  $v.\pi$

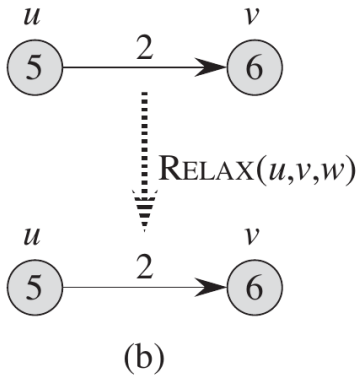


# Camino Mínimo

## 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( $G, w, s$ )

```
1  INITIALIZE-SINGLE-SOURCE( $G, s$ )
2  for  $i = 1$  to  $|G.V| - 1$ 
3      for each edge  $(u, v) \in G.E$ 
4          RELAX( $u, v, w$ )
5  for each edge  $(u, v) \in G.E$ 
6      if  $v.d > u.d + w(u, v)$ 
7          return FALSE
8  return TRUE
```

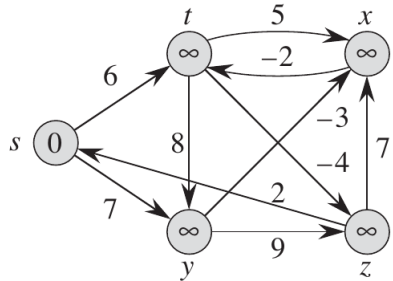


# Camino Mínimo

## Bellman Ford

BELLMAN-FORD( $G, w, s$ )

```
1  INITIALIZE-SINGLE-SOURCE( $G, s$ )
2  for  $i = 1$  to  $|G.V| - 1$ 
3      for each edge  $(u, v) \in G.E$ 
4          RELAX( $u, v, w$ )
5  for each edge  $(u, v) \in G.E$ 
6      if  $v.d > u.d + w(u, v)$ 
7          return FALSE
8  return TRUE
```



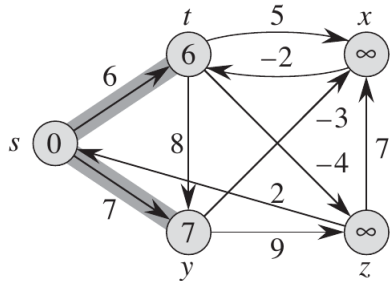
(a)

# Camino Mínimo

## Bellman Ford

BELLMAN-FORD( $G, w, s$ )

```
1  INITIALIZE-SINGLE-SOURCE( $G, s$ )
2  for  $i = 1$  to  $|G.V| - 1$ 
3      for each edge  $(u, v) \in G.E$ 
4          RELAX( $u, v, w$ )
5  for each edge  $(u, v) \in G.E$ 
6      if  $v.d > u.d + w(u, v)$ 
7          return FALSE
8  return TRUE
```



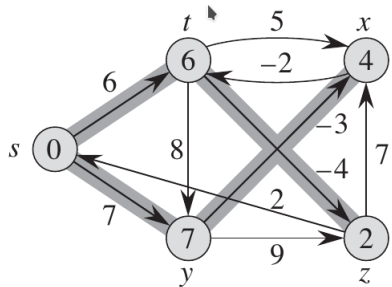
(b)

# Camino Mínimo

## Bellman Ford

BELLMAN-FORD( $G, w, s$ )

```
1 INITIALIZE-SINGLE-SOURCE( $G, s$ )
2 for  $i = 1$  to  $|G.V| - 1$ 
3   for each edge  $(u, v) \in G.E$ 
4     RELAX( $u, v, w$ )
5 for each edge  $(u, v) \in G.E$ 
6   if  $v.d > u.d + w(u, v)$ 
7     return FALSE
8 return TRUE
```



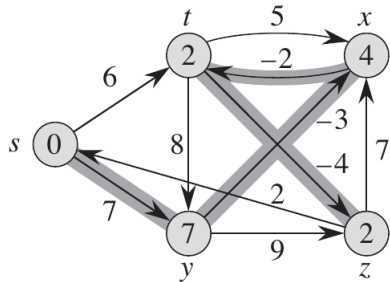
(c)

# Camino Mínimo

## Bellman Ford

BELLMAN-FORD( $G, w, s$ )

```
1  INITIALIZE-SINGLE-SOURCE( $G, s$ )
2  for  $i = 1$  to  $|G.V| - 1$ 
3      for each edge  $(u, v) \in G.E$ 
4          RELAX( $u, v, w$ )
5  for each edge  $(u, v) \in G.E$ 
6      if  $v.d > u.d + w(u, v)$ 
7          return FALSE
8  return TRUE
```



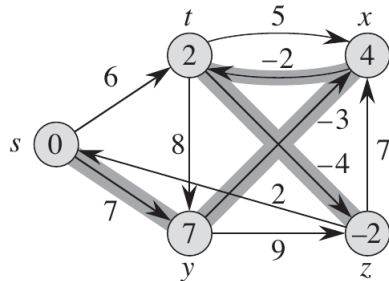
(d)

# Camino Mínimo

## Bellman Ford

BELLMAN-FORD( $G, w, s$ )

```
1  INITIALIZE-SINGLE-SOURCE( $G, s$ )
2  for  $i = 1$  to  $|G.V| - 1$ 
3      for each edge  $(u, v) \in G.E$ 
4          RELAX( $u, v, w$ )
5  for each edge  $(u, v) \in G.E$ 
6      if  $v.d > u.d + w(u, v)$ 
7          return FALSE
8  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( $G, w, s$ )

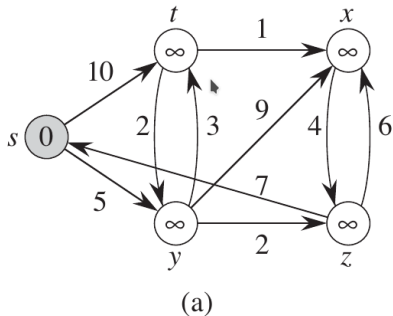
```
1 INITIALIZE-SINGLE-SOURCE( $G, s$ )
2  $S = \emptyset$ 
3  $Q = \emptyset$ 
4 for each vertex  $u \in G.V$ 
5     INSERT( $Q, u$ )
6 while  $Q \neq \emptyset$ 
7      $u = \text{EXTRACT-MIN}(Q)$ 
8      $S = S \cup \{u\}$ 
9     for each vertex  $v$  in  $G.Adj[u]$ 
10        RELAX( $u, v, w$ )
11        if the call of RELAX decreased  $v.d$ 
12            DECREASE-KEY( $Q, v, v.d$ )
```

# Camino Mínimo

## Dijkstra

DIJKSTRA( $G, w, s$ )

```
1 INITIALIZE-SINGLE-SOURCE( $G, s$ )
2  $S = \emptyset$ 
3  $Q = \emptyset$ 
4 for each vertex  $u \in G.V$ 
5     INSERT( $Q, u$ )
6 while  $Q \neq \emptyset$ 
7      $u = \text{EXTRACT-MIN}(Q)$ 
8      $S = S \cup \{u\}$ 
9     for each vertex  $v$  in  $G.Adj[u]$ 
10        RELAX( $u, v, w$ )
11        if the call of RELAX decreased  $v.d$ 
12            DECREASE-KEY( $Q, v, v.d$ )
```

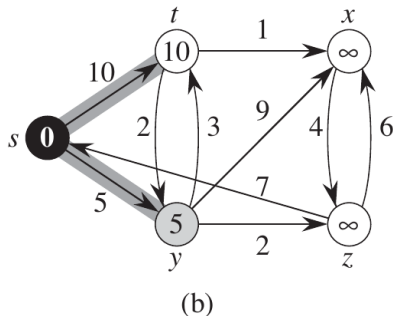


# Camino Mínimo

## Dijkstra

DIJKSTRA( $G, w, s$ )

```
1  INITIALIZE-SINGLE-SOURCE( $G, s$ )
2   $S = \emptyset$ 
3   $Q = \emptyset$ 
4  for each vertex  $u \in G.V$ 
5      INSERT( $Q, u$ )
6  while  $Q \neq \emptyset$ 
7       $u = \text{EXTRACT-MIN}(Q)$ 
8       $S = S \cup \{u\}$ 
9      for each vertex  $v$  in  $G.Adj[u]$ 
10         RELAX( $u, v, w$ )
11         if the call of RELAX decreased  $v.d$ 
12             DECREASE-KEY( $Q, v, v.d$ )
```



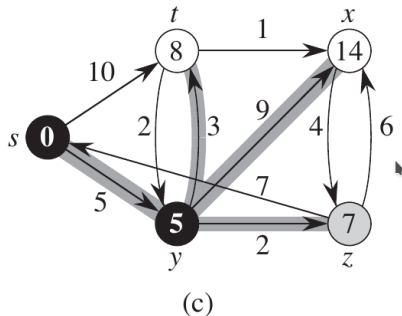


# Camino Mínimo

## Dijkstra

DIJKSTRA( $G, w, s$ )

```
1 INITIALIZE-SINGLE-SOURCE( $G, s$ )
2  $S = \emptyset$ 
3  $Q = \emptyset$ 
4 for each vertex  $u \in G.V$ 
5     INSERT( $Q, u$ )
6 while  $Q \neq \emptyset$ 
7      $u = \text{EXTRACT-MIN}(Q)$ 
8      $S = S \cup \{u\}$ 
9     for each vertex  $v$  in  $G.Adj[u]$ 
10        RELAX( $u, v, w$ )
11        if the call of RELAX decreased  $v.d$ 
12            DECREASE-KEY( $Q, v, v.d$ )
```

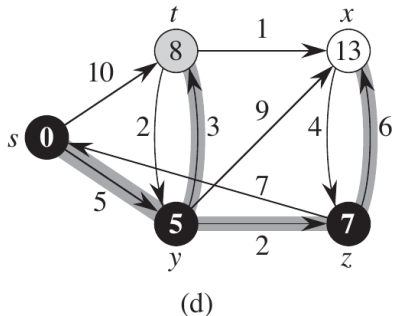


# Camino Mínimo

## Dijkstra

DIJKSTRA( $G, w, s$ )

```
1  INITIALIZE-SINGLE-SOURCE( $G, s$ )
2   $S = \emptyset$ 
3   $Q = \emptyset$ 
4  for each vertex  $u \in G.V$ 
5      INSERT( $Q, u$ )
6  while  $Q \neq \emptyset$ 
7       $u = \text{EXTRACT-MIN}(Q)$ 
8       $S = S \cup \{u\}$ 
9      for each vertex  $v$  in  $G.Adj[u]$ 
10         RELAX( $u, v, w$ )
11         if the call of RELAX decreased  $v.d$ 
12             DECREASE-KEY( $Q, v, v.d$ )
```

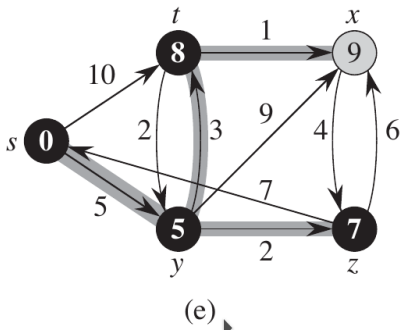


# Camino Mínimo

## Dijkstra

DIJKSTRA( $G, w, s$ )

```
1  INITIALIZE-SINGLE-SOURCE( $G, s$ )
2   $S = \emptyset$ 
3   $Q = \emptyset$ 
4  for each vertex  $u \in G.V$ 
5      INSERT( $Q, u$ )
6  while  $Q \neq \emptyset$ 
7       $u = \text{EXTRACT-MIN}(Q)$ 
8       $S = S \cup \{u\}$ 
9      for each vertex  $v$  in  $G.Adj[u]$ 
10         RELAX( $u, v, w$ )
11         if the call of RELAX decreased  $v.d$ 
12             DECREASE-KEY( $Q, v, v.d$ )
```

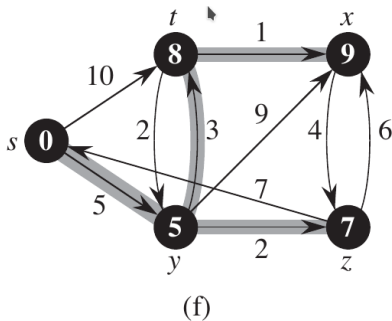


# Camino Mínimo

## Dijkstra

DIJKSTRA( $G, w, s$ )

```
1 INITIALIZE-SINGLE-SOURCE( $G, s$ )
2  $S = \emptyset$ 
3  $Q = \emptyset$ 
4 for each vertex  $u \in G.V$ 
5     INSERT( $Q, u$ )
6 while  $Q \neq \emptyset$ 
7      $u = \text{EXTRACT-MIN}(Q)$ 
8      $S = S \cup \{u\}$ 
9     for each vertex  $v$  in  $G.Adj[u]$ 
10        RELAX( $u, v, w$ )
11        if the call of RELAX decreased  $v.d$ 
12            DECREASE-KEY( $Q, v, v.d$ )
```



- 

