Centro de
Investigación en
Computación

Examples of Greedy Algorithms

Ejemplos de Algoritmos Greedy

Tema 4

Course

Analysis and design of algorithms

Instructor

Acosta Bermejo Raúl et al.

Lecture notes

Instituto
Politécnico
Nacional

Table of contents (outline)

Tabla de contenido

1. Shortest path
    1. Algoritmo de Dijkstra
    2. Algoritmo de Bellmand-Ford
2. Shortest path All pairs of nodes
    1. Floyd-Warshall
3. Minimum Spanning Tree
    1. Algoritmo de Prim
    2. Algoritmo de Kruskal
4. Clustering

Shortest path in graph

Algoritmo de Dijkstra

Tema

Shortest path in graph

Algoritmo de Dijkstra

Pseudo-código

```
   Dijkstra(G,s)
1.    for each vertex u ∈ G.V()
2.        u.setd(∞)
3.        u.setparent(NIL)
4.        s.setd(0)
5.    S ← ∅              // Set S is used to explain the algorithm
6.    Q.init(G.V())   // Q is a priority queue ADT
7.     while not Q.isEmpty()
8.      u ← Q.extractMin()
9.      S ← S U {u}
10.     for each v ∈ u.adjacent() do
11.        Relax(u, v, G)
12.        Q.modifyKey(v)
```

Input: Graph *G*, start vertex *s*

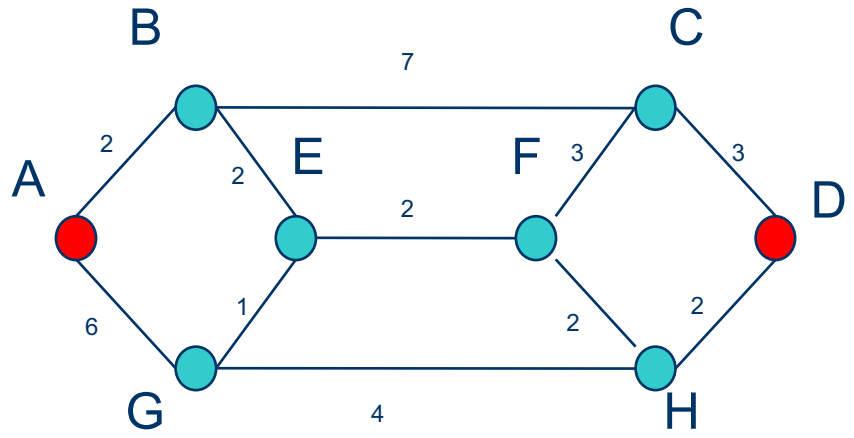
Relaxing Edges

Shortest path in graph

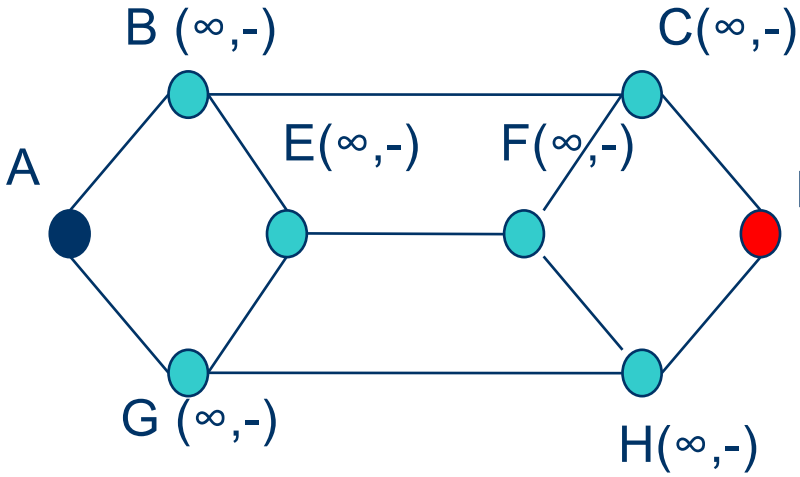
Ejemplo (1/9)

Considérese el siguiente grafo que representa a una red.

- Permanente
- Tentativo



Grafo Inicial

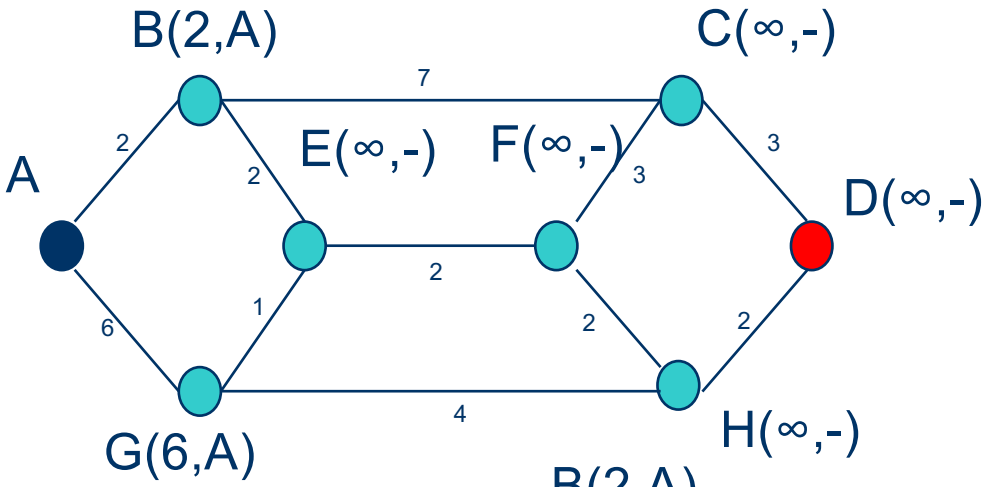


Inicialización

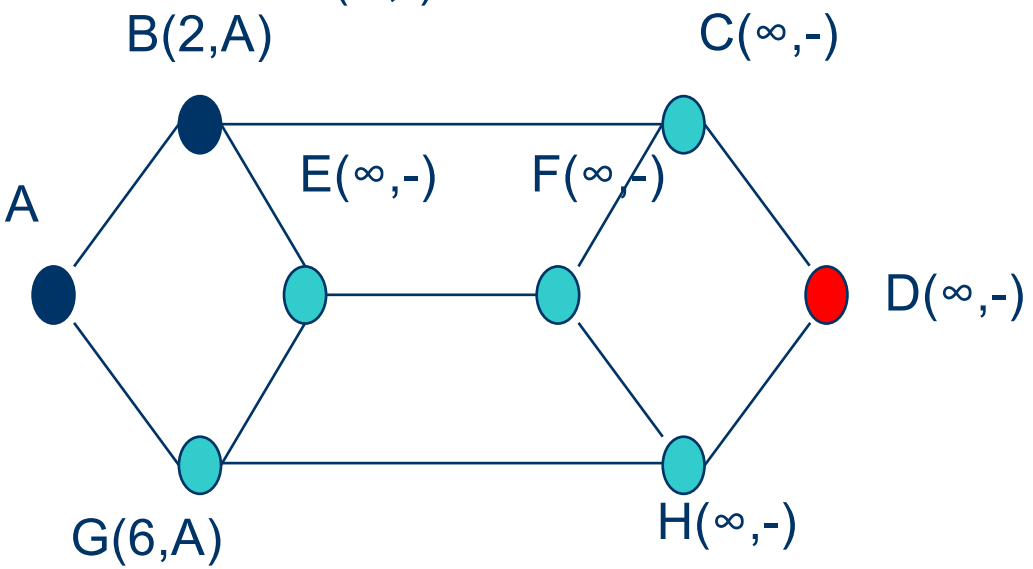
Shortest path in graph

Ejemplo (2/9)

Nodo A
valores



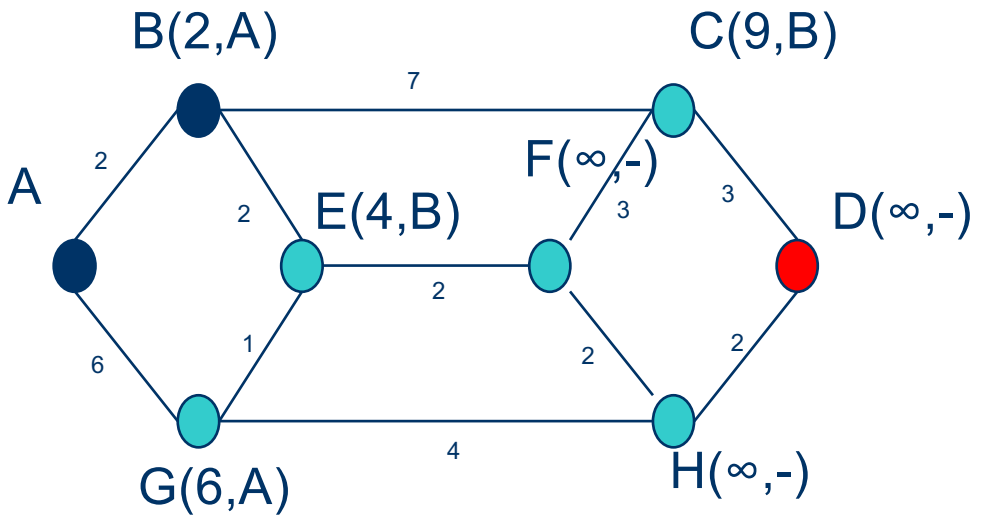
Nodo B
análisis



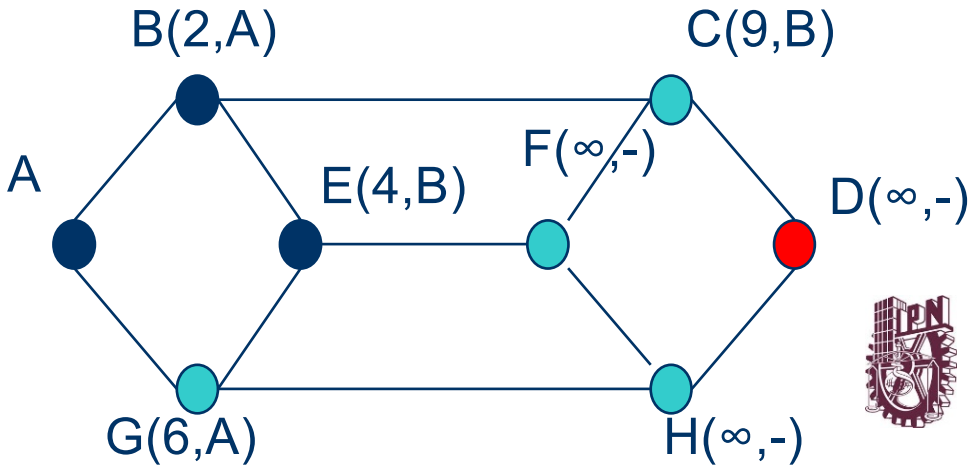
Shortest path in graph

Ejemplo (3/9)

Nodo B
valores



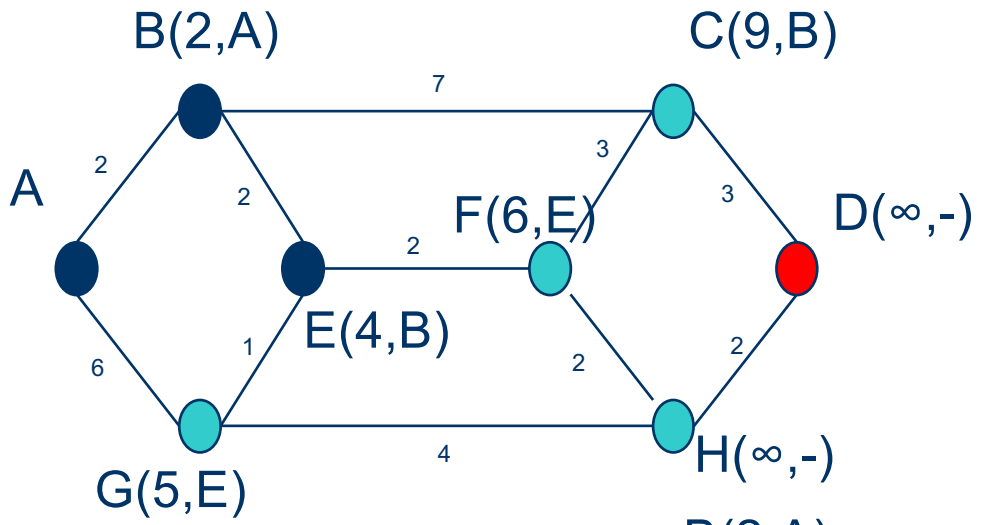
Nodo E
análisis



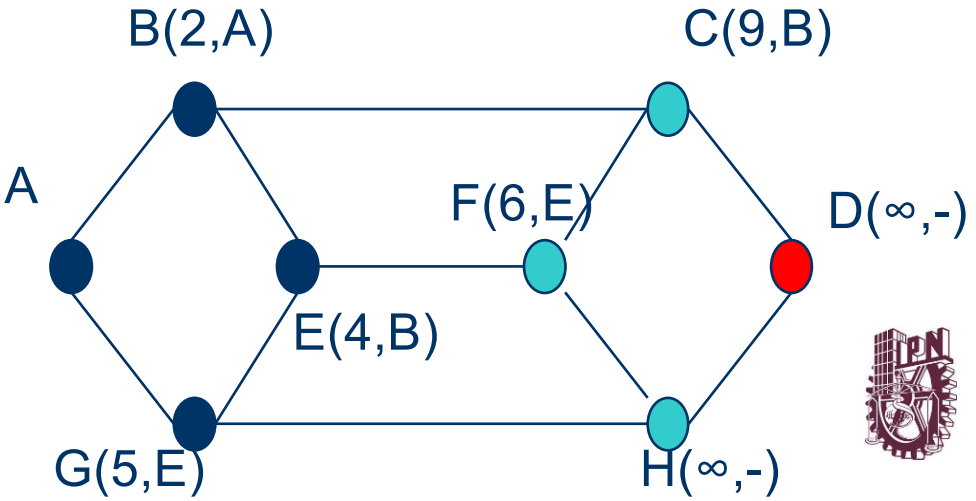
Shortest path in graph

Ejemplo (4/9)

Nodo E
valores



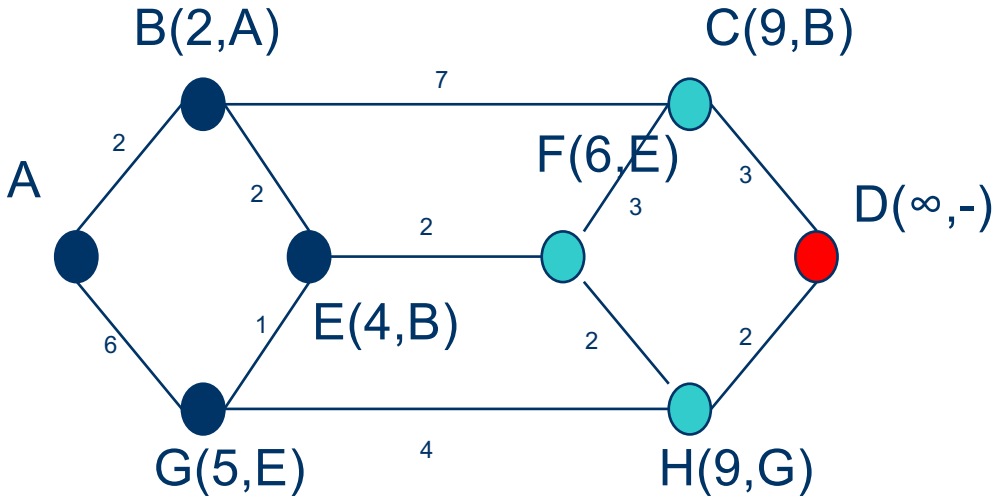
Nodo G
análisis



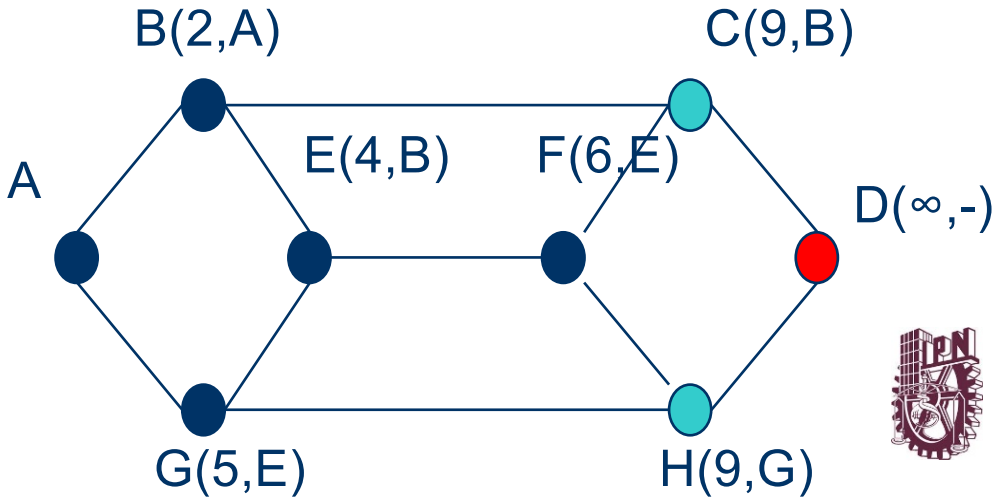
Shortest path in graph

Ejemplo (5/9)

Nodo G
valores



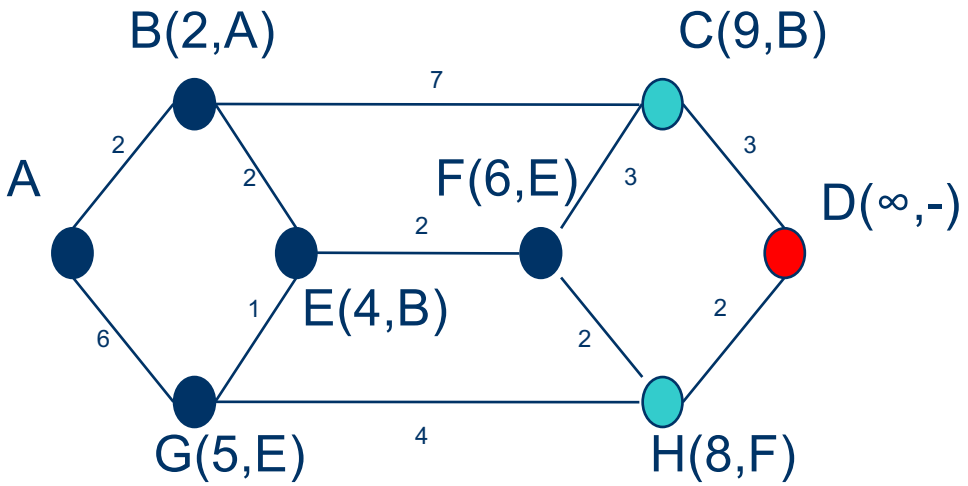
Nodo F
análisis



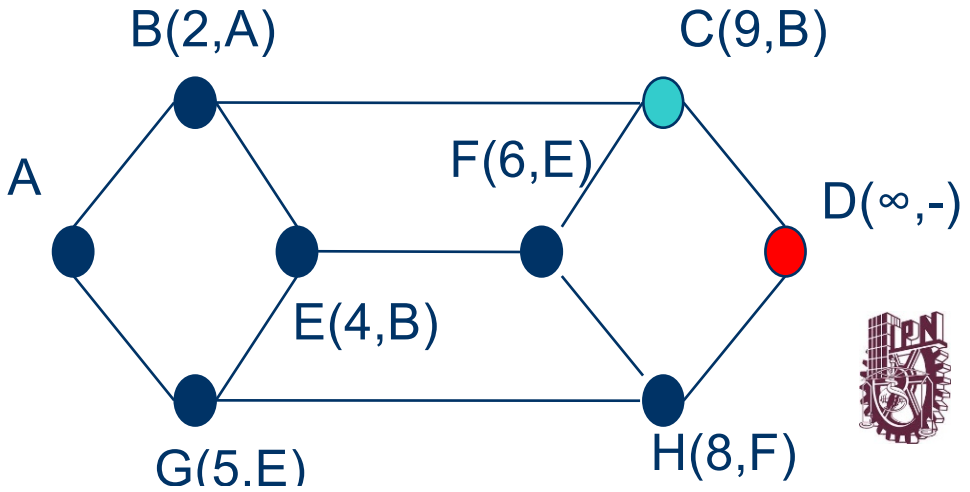
Shortest path in graph

Ejemplo (6/9)

Nodo F
valores



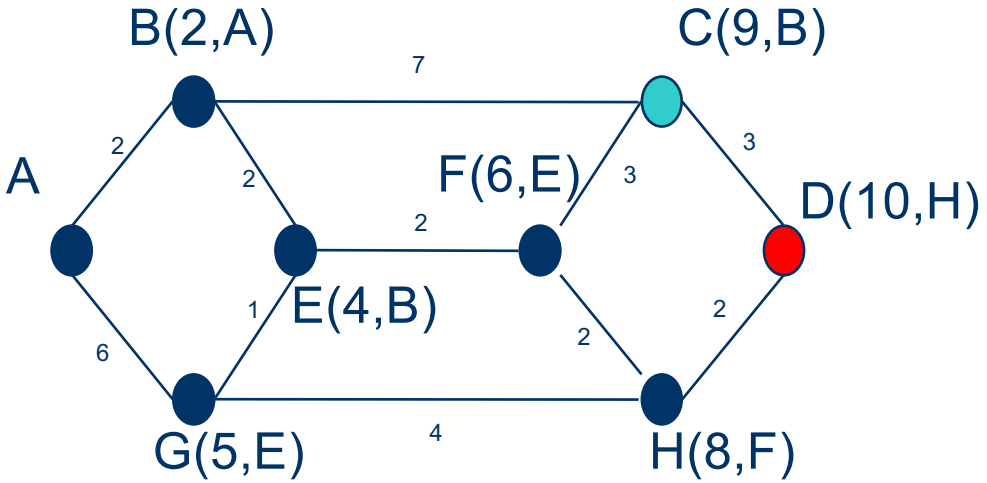
Nodo H
análisis



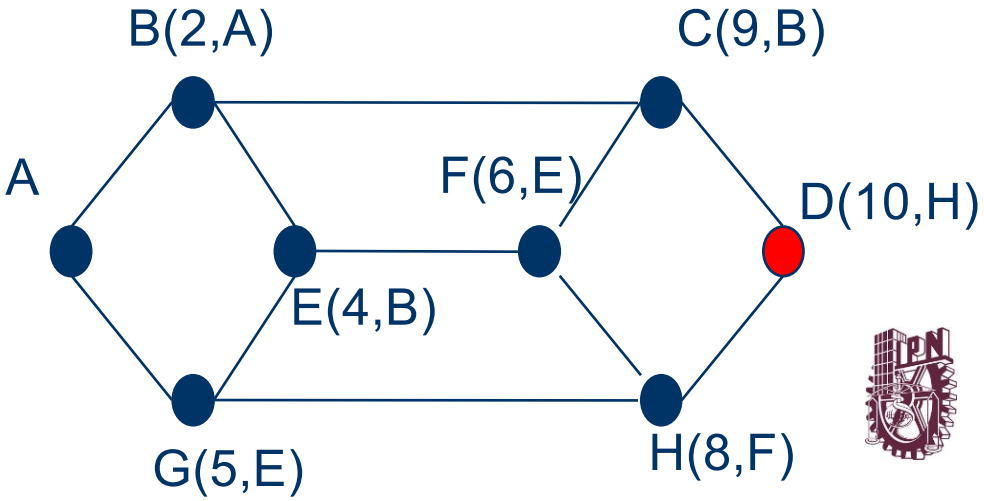
Shortest path in graph

Ejemplo (7/9)

Nodo H
valores



Nodo C
análisis

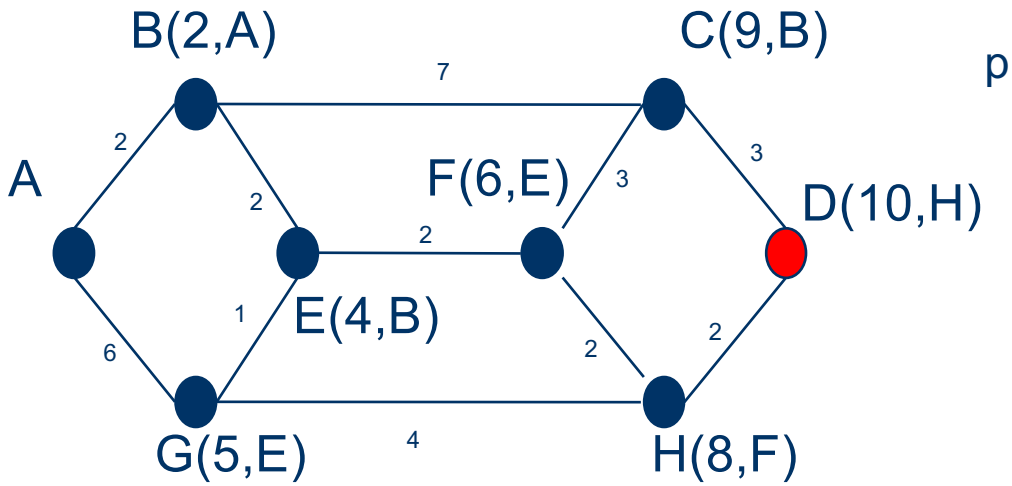


Shortest path in graph

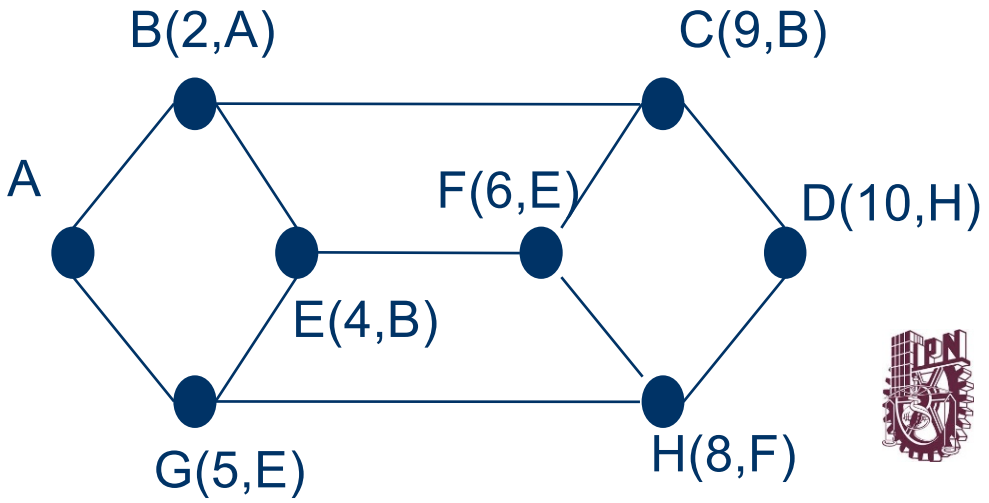
Ejemplo (8/9)

La trayectoria se lee de destino a origen basándose en el predecesor de cada nodo
D-H-F-E-B-A

Nodo C
valores



Nodo D
Análisis y Destino



Shortest path in graph

Algoritmo de Bellman-Ford

Tema

Shortest path in graph

Algoritmo de Bellman Ford

```
Relax (u,v,G)
if v.distance() > u.distance()+G.weight(u,v) then
   v.setDistance(u.distance()+ G.weight (u,v))
   v.setparent(u)
fi
```

Pseudo-código

```
 Bellman-Ford(G,s)
1.    for v ∈ G.V()
2.       v.setDistance(∞)
3.       v.setParent(NIL)
4.    s.setDistance(0)

5.    for i=1 to |G.V()|-1 do
6.      for each edge(u,v) ∈ G.E() do
7.            Relax (u,v,G)

8.    for each edge (u,v) in G.E() do
9.       if v.d() > u.d() + G.w(u,v) then
10.           return false
11.   return true
```

Inicialización

Verificación de ciclos negativos

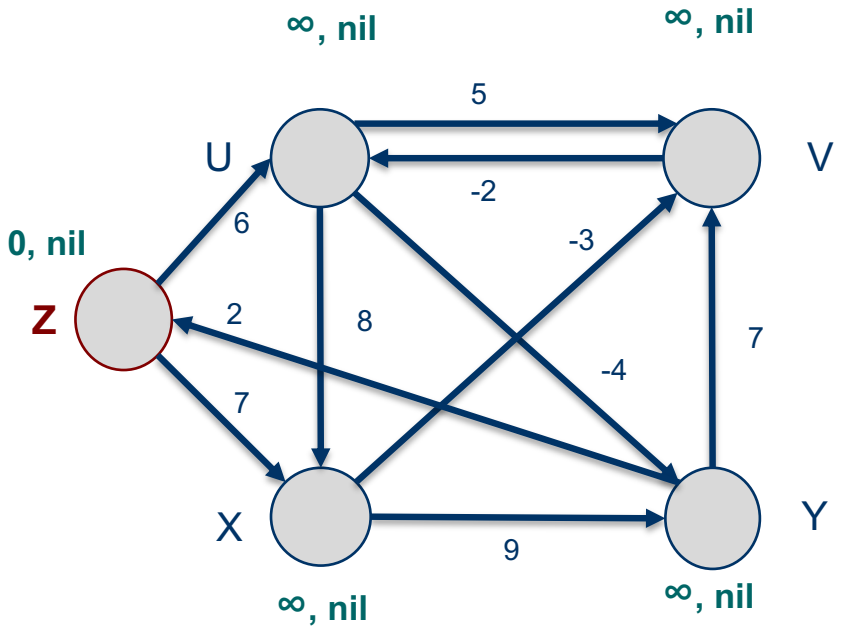
Shortest path in graph

Algoritmo de Bellman Ford

Lista de adyacencia

z: u, x
u: v, y, x
x: v, y
v: u
y: v, z

Realizar una corrida del algoritmo con el siguiente ejemplo:
Source = Z, Target = Y



1. Inicialización (líneas 1-4)

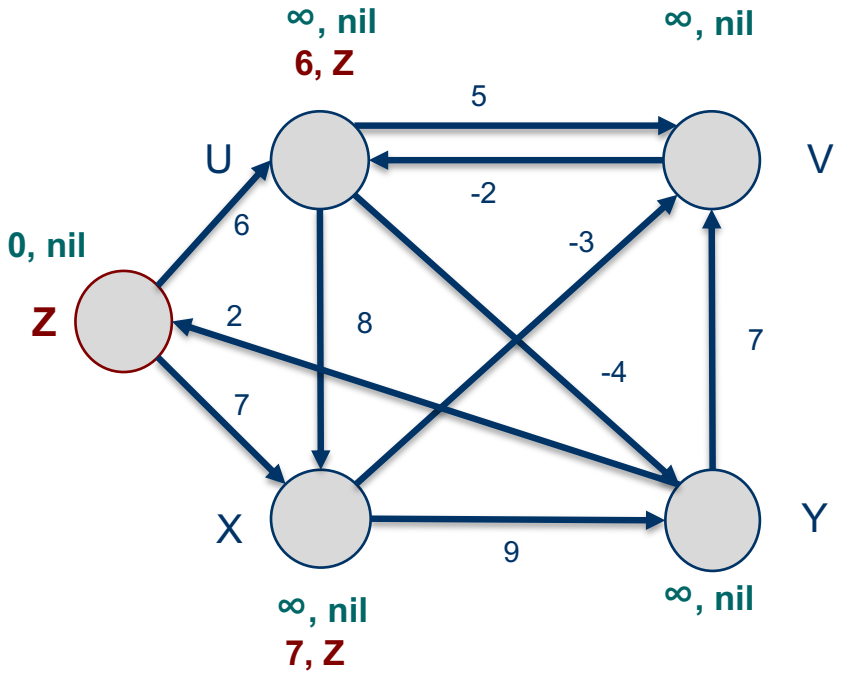
Shortest path in graph

Algoritmo de Bellman Ford

Lista de adyacencia

z: u, x
u: v, y, x
x: v, y
v: u
y: v, z

Realizar una corrida del algoritmo con el siguiente ejemplo:
Source = Z, Target = Y



1. Inicialización (lineas 1-4)
2. Cálculo pesos (lineas 5-7)

edge(Z,U)

Distance(U)∞ > Distance(Z) **0** + w(Z,U) **6**
Distance(U) = **0** + **6** = 6
Predecesor(U) = **Z**

edge(Z,X)

Distance(X) ∞ > Distance(Z) **0** + w(Z,X) **7**
Distance(X) = **0** + **7** = 7
Predecesor(X) = **Z**

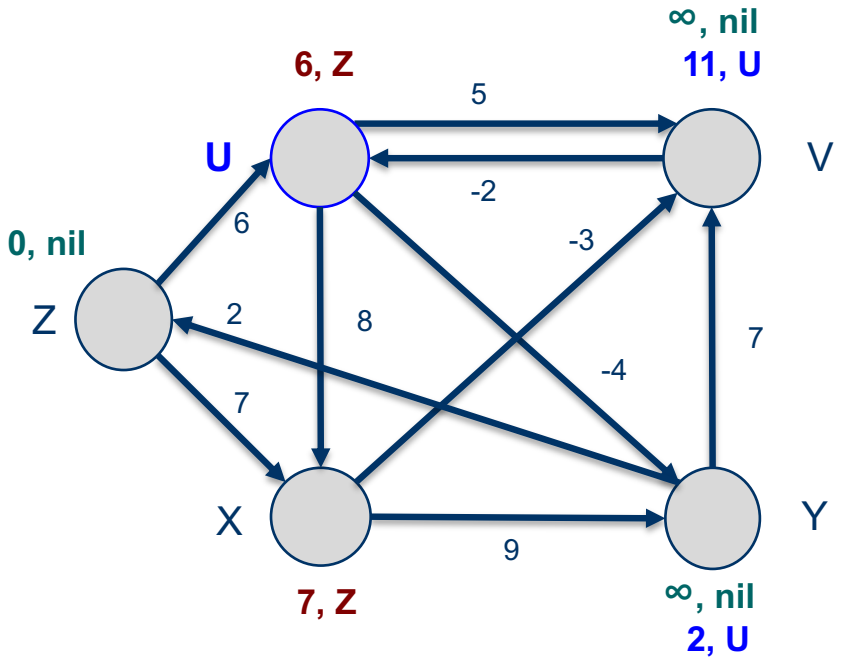
Shortest path in graph

Algoritmo de Bellman Ford

Lista de adyacencia

z: u, x
u: v, y, x
x: v, y
v: u
y: v, z

Realizar una corrida del algoritmo con el siguiente ejemplo:
Source = Z, Target = Y



edge(U,V)
Distance(V) ∞ > Distance(U) **6** + w(U,V) 5
Distance(V) = **6** + 5 = **11**
Predecesor(V) = **U**

edge(U,Y)
Distance(Y) ∞ > Distance(U) **6** + w(U,Y) -4
Distance(Y) = **6** + (-4) = **2**
Predecesor(Y) = **U**

edge(U,X)
Distance(X) **7** > Distance(U) **6** + w(U,X) 8
Falso

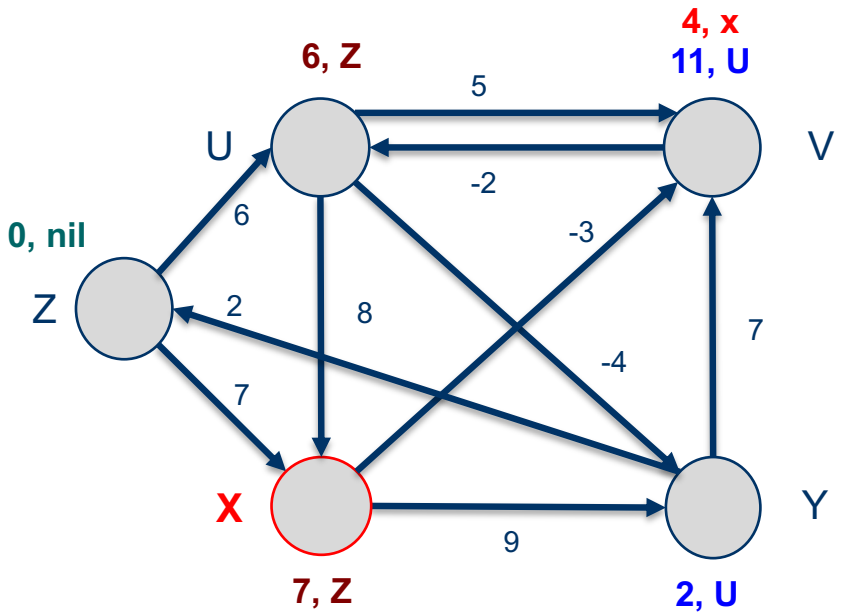
Shortest path in graph

Algoritmo de Bellman Ford

Source = Z, Target = Y

Lista de adyacencia

z: u, x
u: v, y, x
x: v, y
v: u
y: v, z



edge(X,V)

Distance(V) **11** > Distance(X) **7** + w(X,V) -3
Distance(V) = 7 + (-3) = **4**
Predecesor(V) = **X**

edge(X,Y)

Distance(Y)∞ > Distance(X) **7** + w(X,Y) 9
Falso

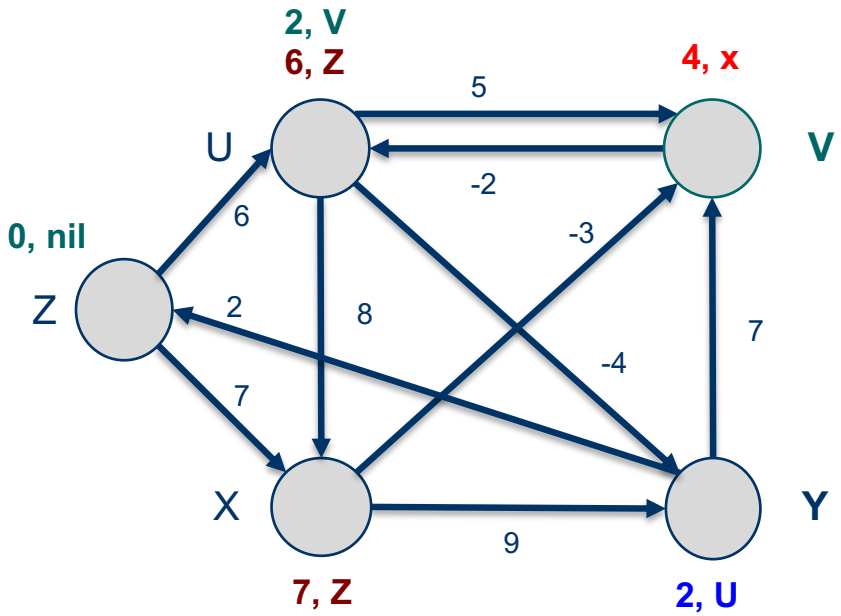
Shortest path in graph

Algoritmo de Bellman Ford

Lista de adyacencia

z: u, x
u: v, y, x
x: v, y
v: u
y: v, z

Source = Z, Target = Y



edge(V,U)
Distance(U) **6** > Distance(V) **4** + w(V,U) -2
Distance(U) = 4 + (-2) = **2**
Predecesor(U) = **V**

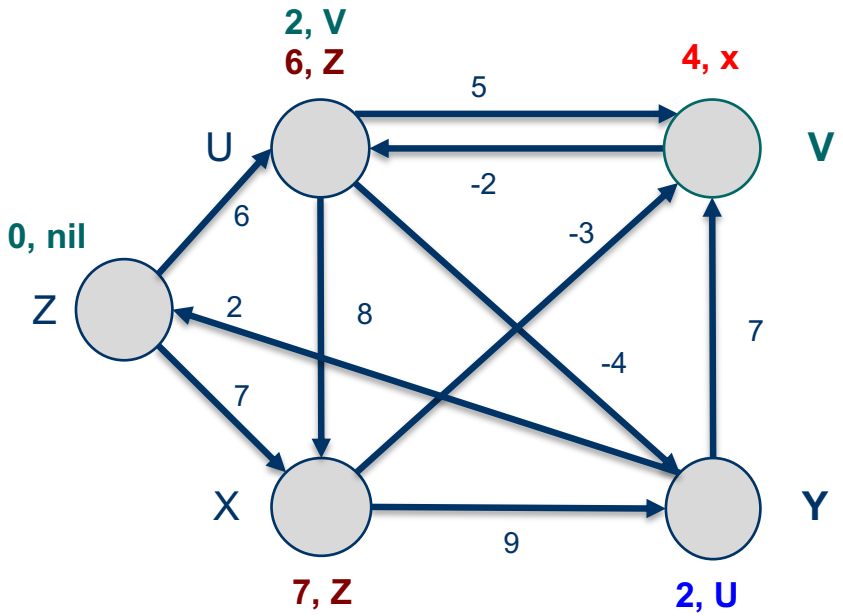
Shortest path in graph

Algoritmo de Bellman Ford

Lista de adyacencia

z: u, x
u: v, y, x
x: v, y
v: u
y: v, z

Source = Z, Target = Y



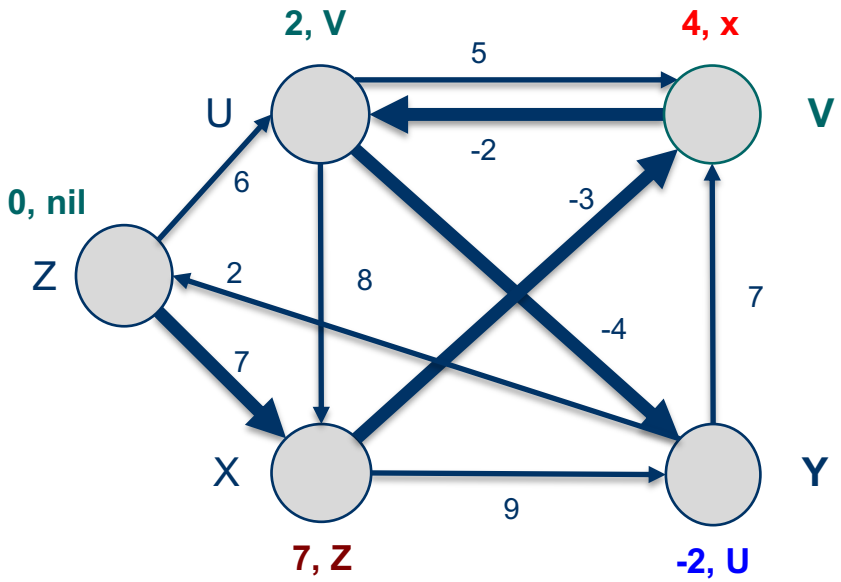
edge(Y,V)
Distance(V) **4** > Distance(Y) **2** + w(Y,V) 7
Falso

edge(Y,Z)
Distance(Z) **0** > Distance(Y) **2** + w(Y,Z) 2
Falso

Shortest path in graph

Algoritmo de Bellman Ford

Source = Z, Target = Y



El camino más corto es

Z, X, V, U, Y

7+(-3)+(-2)+(-4) = -2

Un truco que se puede Hacer es reasignar pesos para que no haya valores Negativos, por ejemplo sumar 4 a todos.

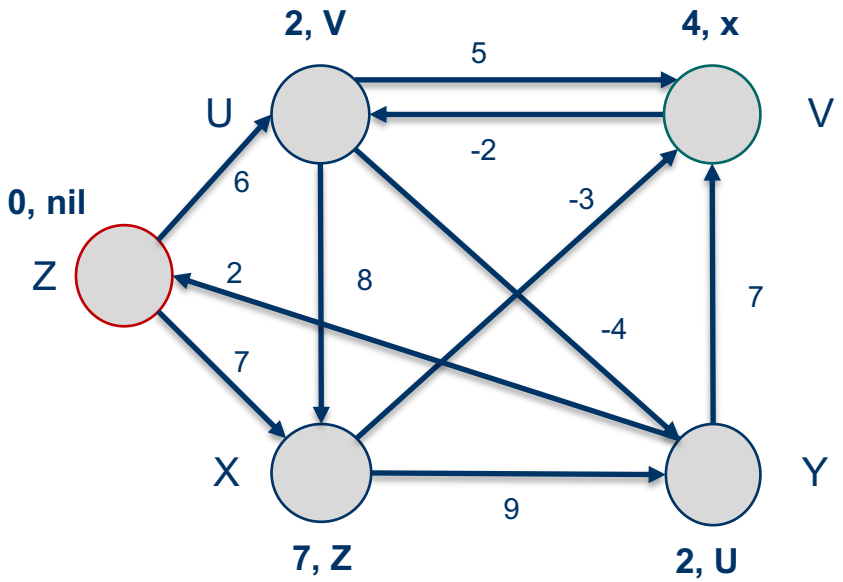
Shortest path in graph

Algoritmo de Bellman Ford

Lista de adyacencia

z: u, x
u: v, y, x
x: v, y
v: u
y: v, z

Verificar **ciclos negativos**



edge(Z,U)

Distance(U) **2** > Distance(Z) **0** + w(Z,U) **6**

False

edge(Z,X)

Distance(X) **7** > Distance(Z) **0** + w(Z,X) **7**

False

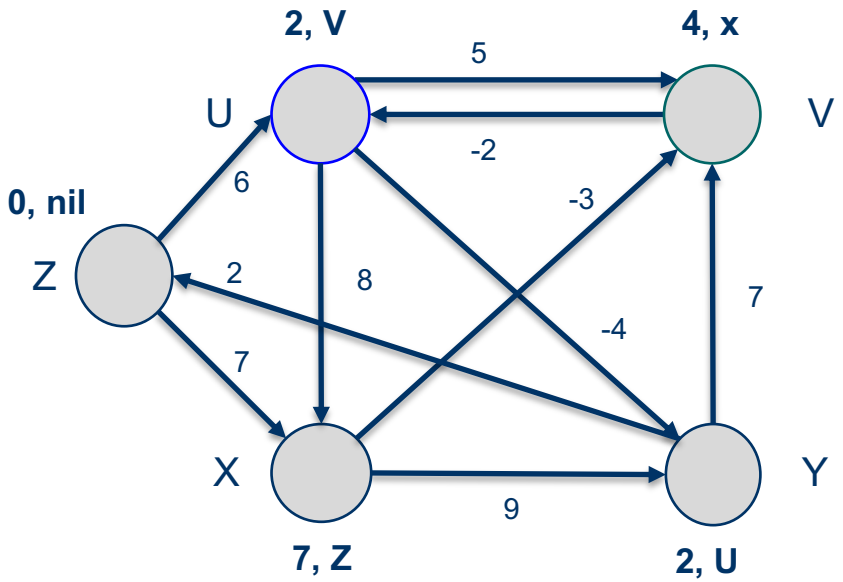
Shortest path in graph

Algoritmo de Bellman Ford

Lista de adyacencia

z: u, x
u: v, y, x
x: v, y
v: u
y: v, z

Verificar ciclos negativos



edge(U,V)
Distance(V) **4** > Distance(U) **2** + w(U,V) **5**
False

edge(U,Y)
Distance(Y) **2** > Distance(U) **2** + w(U,Y) **-4**
False

edge(U,X)
Distance(X) **7** > Distance(U) **2** + w(U,X) **8**
False

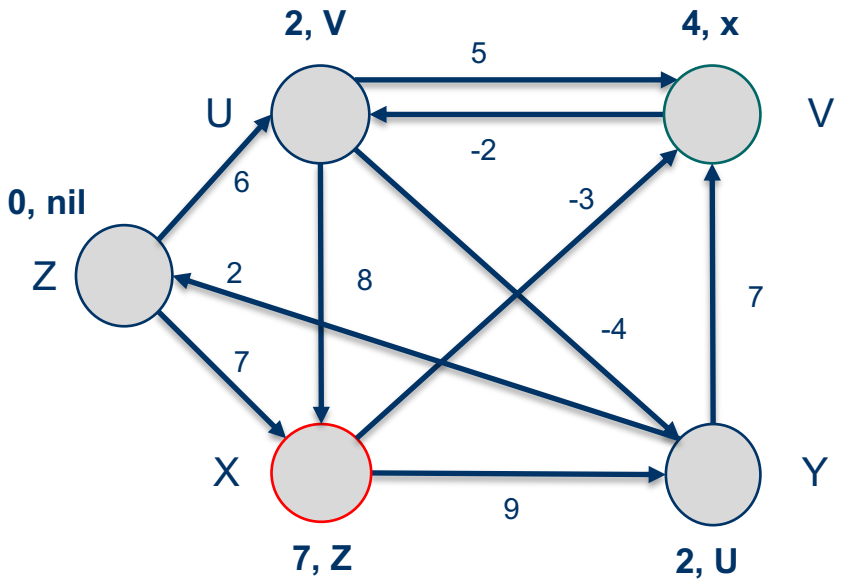
Shortest path in graph

Algoritmo de Bellman Ford

Lista de adyacencia

z: u, x
u: v, y, x
x: v, y
v: u
y: v, z

Verificar ciclos negativos



edge(X,V)

Distance(V) **4** > Distance(X) **7** + w(X,V) **-3**

False

edge(X,Y)

Distance(Y) **2** > Distance(X) **7** + w(X,Y) **9**

False

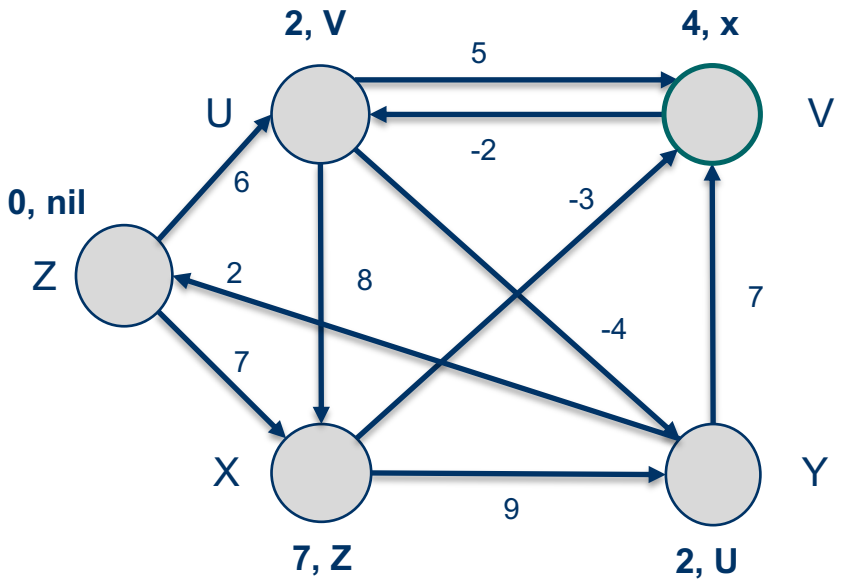
Shortest path in graph

Algoritmo de Bellman Ford

Lista de adyacencia

z: u, x
u: v, y, x
x: v, y
v: u
y: v, z

Verificar ciclos negativos



edge(V,U)
Distance(U) **2** > Distance(V) **4** + w(V,U) -2
False

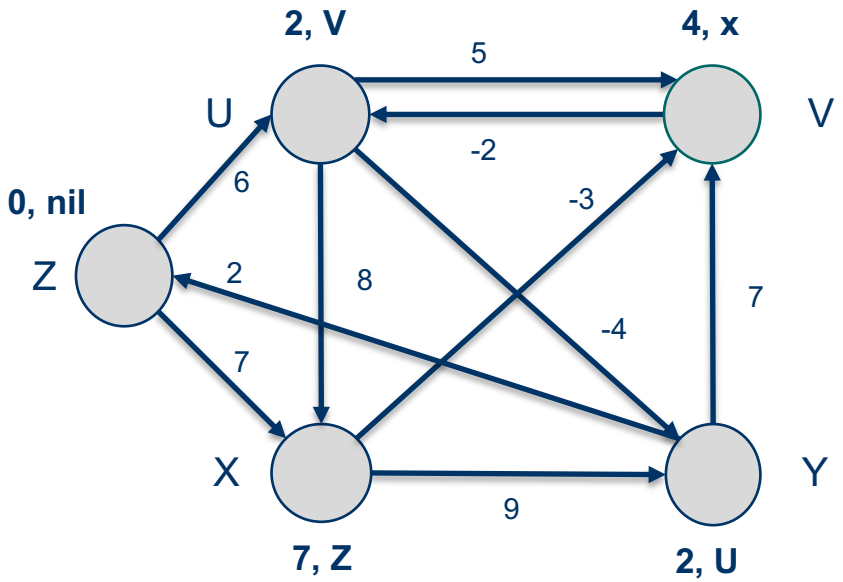
Shortest path in graph

Algoritmo de Bellman Ford

Lista de adyacencia

z: u, x
u: v, y, x
x: v, y
v: u
y: v, z

Verificar ciclos negativos



edge(Y,V)
Distance(V) **4** > Distance(Y) **2** + w(Y,V) **7**
False

edge(Y,Z)
Distance(Z) **0** > Distance(Y) **2** + w(Y,Z) **2**
False

Shortest path in graph

Algoritmo de Floyd-Warshall

Tema

Shortest path in graph

Algoritmo de Floyd-Warshall

NO viene en el temario sin embargo es una de tantas variantes a estudiar.

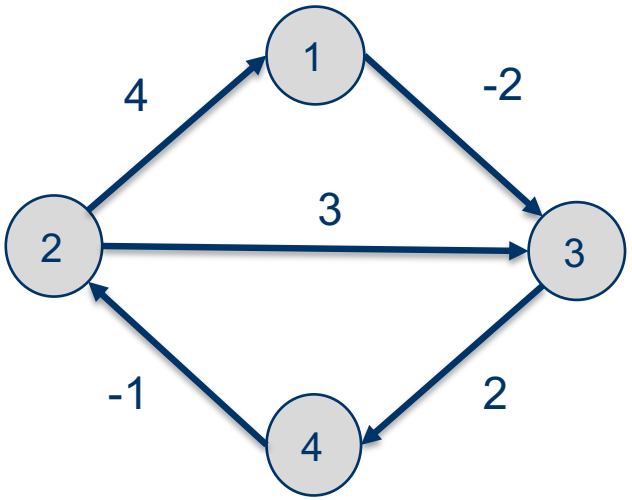
```
1.  let dist be a |V| × |V| array of minimum distances initialized to ∞ (infinity)
2.  for each vertex v
3.      dist[v][v] ← 0

4.  for each edge (u,v)
5.      dist[u][v] ← w(u,v)  // the weight of the edge (u,v)

6.  for k=1 to |V|
7.      for i=1 to |V|
8.          for j=1 to |V|
9.              if dist[i][j] > dist[i][k] + dist[k][j]
10.                 dist[i][j] ← dist[i][k] + dist[k][j]
11.             fi
```

dist[i][j] = min(dist[i][j], dist[i][k]+dist[k][j])

Shortest path in graph

Algoritmo de Floyd-Warshall

Example



Initialization line 1

Table of dist

Node	1	2	3	4
1	∞	∞	∞	∞
2	∞	∞	∞	∞
3	∞	∞	∞	∞
4	∞	∞	∞	∞

lines 2-3

Node	1	2	3	4
1	0	∞	∞	∞
2	∞	0	∞	∞
3	∞	∞	0	∞
4	∞	∞	∞	0

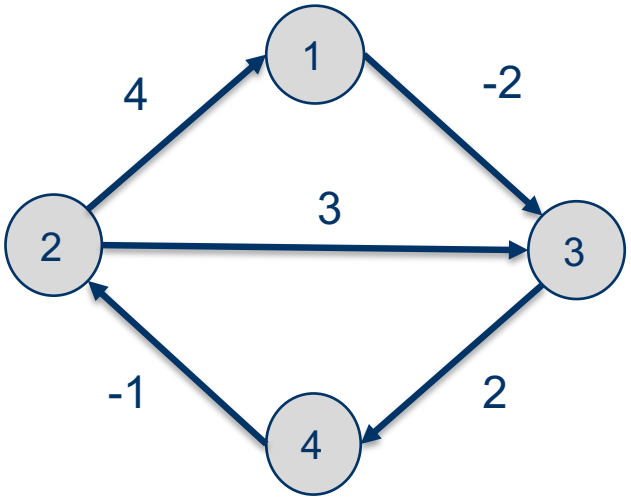
Cuantos ciclos y que aristas prueba?

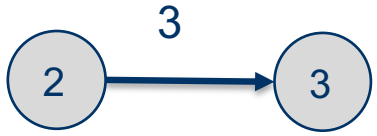
Shortest path in graph

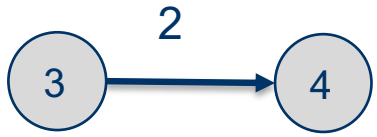
Algoritmo de Floyd-Warshall

Example

First cicle
K=1







Node	1	2	3	4
1	0	∞	-2	∞
2	4	0	3	∞
3	∞	∞	0	2
4	∞	-1	∞	0

Shortest path in graph

Algoritmo de Floyd-Warshall

Example

The Floyd–Warshall algorithm typically only provides the lengths of the paths between all pairs of vertices. With simple modifications, it is possible to create a method to reconstruct the actual path between any two endpoint vertices.

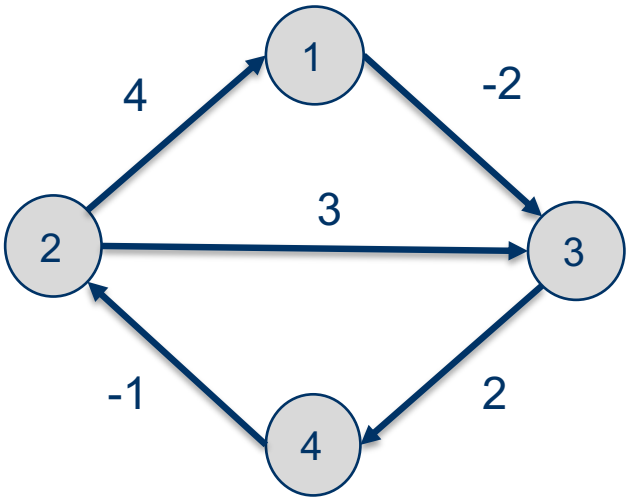


Table of paths

Node	1	2	3	4
1	1	2	3	4
2	1	2	3	4
3	1	2	3	4
4	1	2	3	4

Shortest path in graph

Algoritmo de Floyd-Warshall

Example

You start with the dist table (making some operations) and you update path
table.

Table of dist

Node	1	2	3	4
1	0	∞	-2	∞
2	4	0	3	∞
3	∞	∞	0	2
4	∞	-1	∞	0

Table of paths

Node	1	2	3	4
1	1	2	3	4
2	1	2	3	4
3	1	2	3	4
4	1	2	3	4

Shortest path in graph

Algoritmo de Floyd-Warshall

Example

First vertice (1).

Table of dist

Node	1	2	3	4
1	0	∞	-2	∞
2	4	0	3 2	∞
3	∞	∞	0	2
4	∞	-1	∞	0

4+∞ < 0

4+(-2) <3

4+∞ < ∞

∞ 3th row

∞ 4 th row

Table of paths

Node	1	2	3	4
1	1	2	3	4
2	1	2	1	4
3	1	2	3	4
4	1	2	3	4

- Recorremos la franja verde izq. (vertical).
- Se suma su valor (izq) con el de la franja verde superior.
- Si se mejora (min) se actualiza.

Shortest path in graph

Algoritmo de Floyd-Warshall

Example

2nd vertice (2).

Table of dist

Node	1	2	3	4
1	0	∞	-2	∞
2	4	0	3	∞
3	∞	∞	0	2
4	3	-1	2	0

$4+\infty < 0$
$3+\infty < 3$
$\infty+\infty < \infty$

$4+\infty < \infty$
$3+\infty < \infty$
$\infty +\infty < 2$

$4+(-1) < \infty$
$3+(-1) < \infty$
$\infty+(-1) < \infty$

Table of paths

Node	1	2	3	4
1	1	2	3	4
2	1	2	1	4
3	1	2	3	4
4	2	2	2	4

Shortest path in graph

Algoritmo de Floyd-Warshall

Example

3th vertice (3).

Table of dist

Node	1	2	3	4
1	0	∞	-2	∞ 0
2	4	0	3	∞ 5
3	∞	∞	0	2
4	∞	-1	∞	0

∞+(-2) < 0
∞+(-2) < ∞
(-2)+2 < ∞

∞+3 < 4
∞+3 < 0
3+2 < ∞

Table of paths

Node	1	2	3	4
1	1	2	3	3
2	1	2	3	3
3	1	2	3	4
4	2	2	2	4

4th no change

Shortest path in graph

Algoritmo de Floyd-Warshall

Example

4th vertice (4).

∞ always give ∞
and doesn't change

-1+2 < ∞

Table of dist

Node	1	2	3	4
1	0	∞	-2	∞
2	4	0	3	∞
3	∞	1	0	2
4	∞	-1	∞	0

Table of paths

Node	1	2	3	4
1	1	2	3	4
2	1	2	3	4
3	1	4	3	4
4	2	2	2	4

Shortest path in graph

Algoritmo de Floyd-Warshall

2do ejemplo

Hallar el camino mínimo desde el vértice 3 hasta 4 en el grafo con la siguiente matriz de distancias:

Table of dist

Node	1	2	3	4	5	6
1	0	3	5	1	∞	∞
2	3	0	∞	∞	9	∞
3	5	∞	0	7	7	1
4	1	∞	7	0	∞	4
5	∞	9	7	∞	0	∞
6	∞	∞	1	4	∞	0

Resp. 5

Minimum Spaning Tree

Algoritmo de Prim

Tema

MST Prim's algorithm

Algoritmo de Prim

The algorithm may informally be described as performing the following steps:

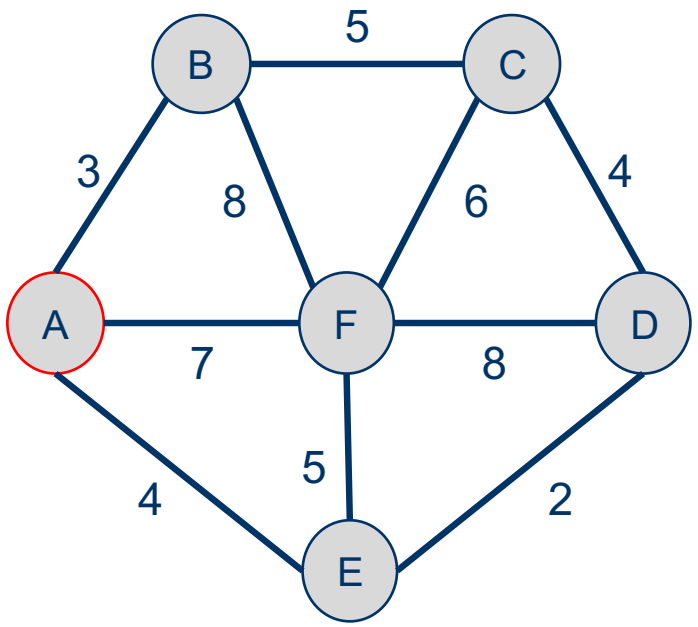
1. Select any vertex.
   Initialize a tree T with that vertex.
2. Select the shortest edge connected to that vertex.

3. Select the shortest edge connected to any vertex already connected.
   It is better if you sort edges (out of T) in ascending order.
4. Repeat step 3 until all vertices have been connected.

MST Prim's algorithm

Algoritmo de Prim

Example



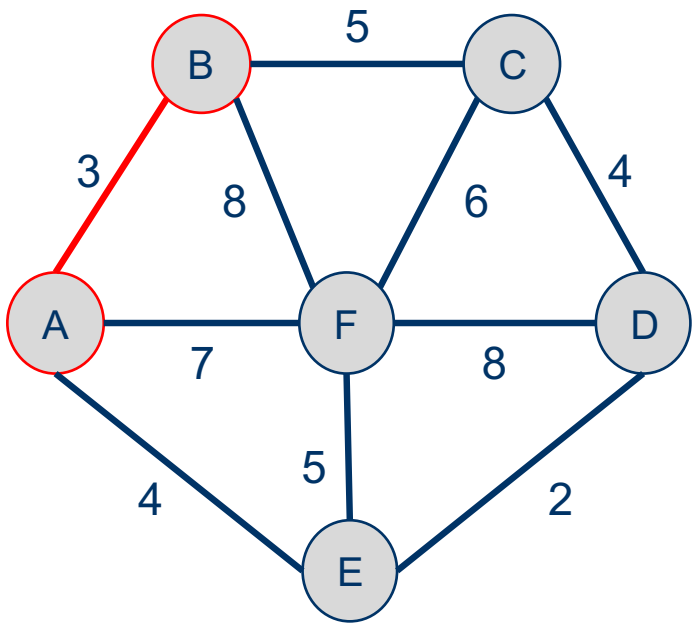
Select any vertex

A

MST Prim's algorithm

Algoritmo de Prim

Example



Select any vertex

A

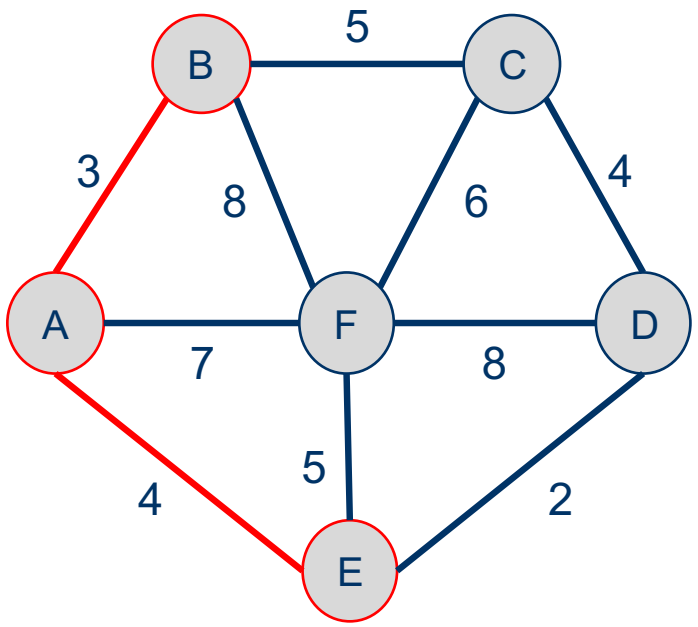
Select the shortest edge connected to that vertex

AB 3

MST Prim's algorithm

Algoritmo de Prim

Example



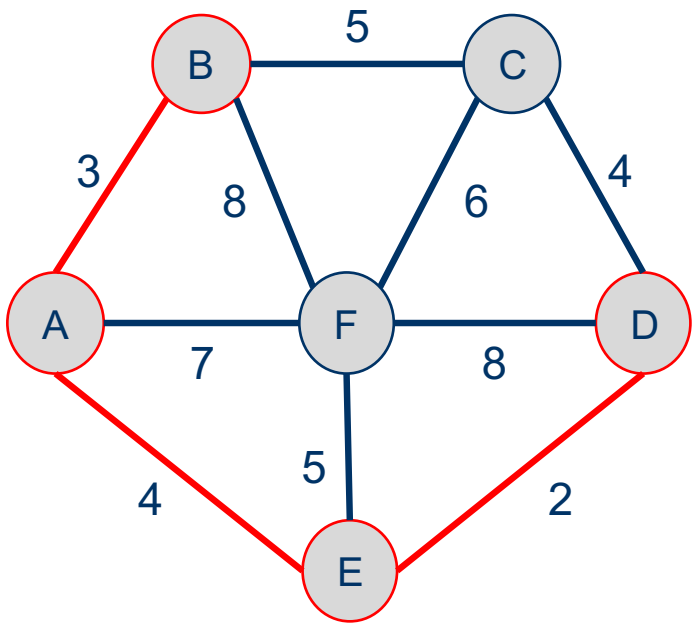
Select the shortest edge connected to any vertex already connected.

AE 4

MST Prim's algorithm

Algoritmo de Prim

Example



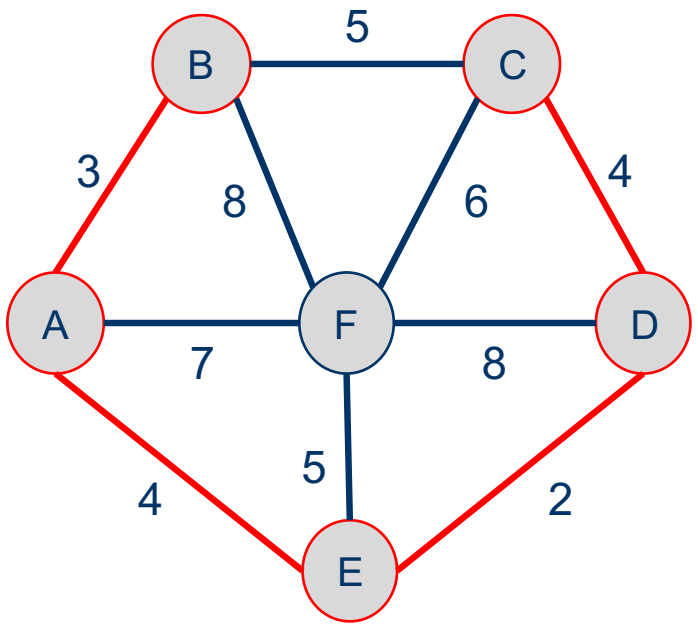
Select the shortest edge connected to any vertex already connected.

ED 2

MST Prim's algorithm

Algoritmo de Prim

Example



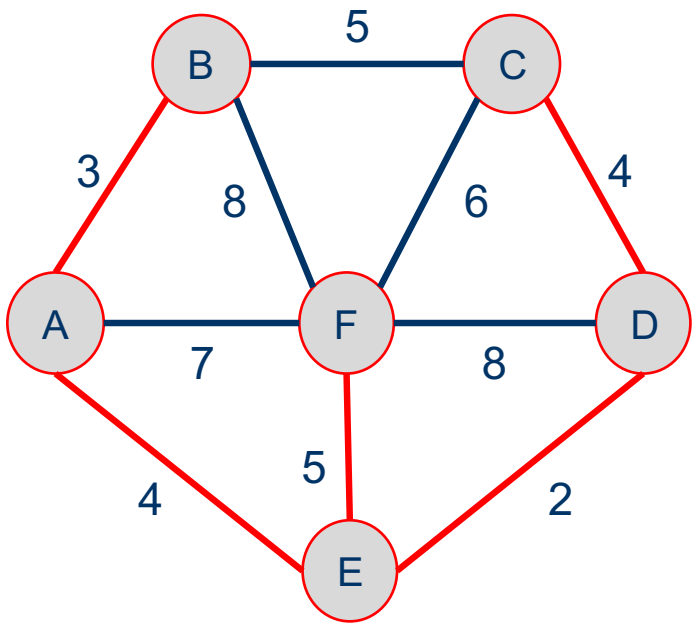
Select the shortest edge connected to any vertex already connected.

DC 4

MST Prim's algorithm

Algoritmo de Prim

Example



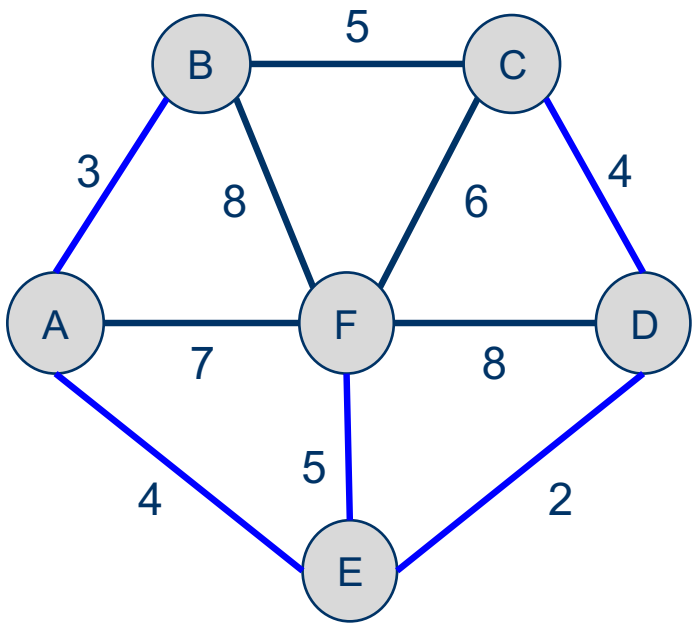
Select the shortest edge connected to any vertex already connected.

EF 5

MST Prim's algorithm

Algoritmo de Prim

Example



All vertices have been connected.

The solution is

AB 3
AE 4
ED 2
DC 4
EF 5

Total weight of tree: 18

Minimum Spaning Tree

Algoritmo de Kruskal

Tema

MST Kruskal's algorithm

Algoritmo de Kruskal

1928 – 2010
Maplewood, Nueva Jersey
Fue un matemático y estadístico estadounidense.

Escrito por Joseph Kruskal y publicado en *Proceedings of the American Mathematical Society*, pp. 48–50 en 1956.

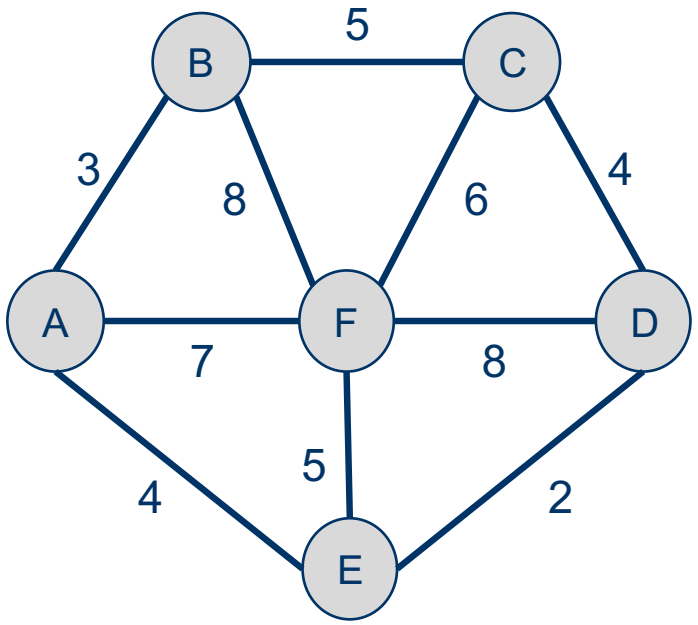
The algorithm may informally be described as performing the following steps:

1. Select the shortest edge in a network.
2. Select the next shortest edge which does not create a cycle.
3. Repeat step 2 until all vertices have been connected.

MST Kruskal's algorithm

Algoritmo de Kruskal

Example



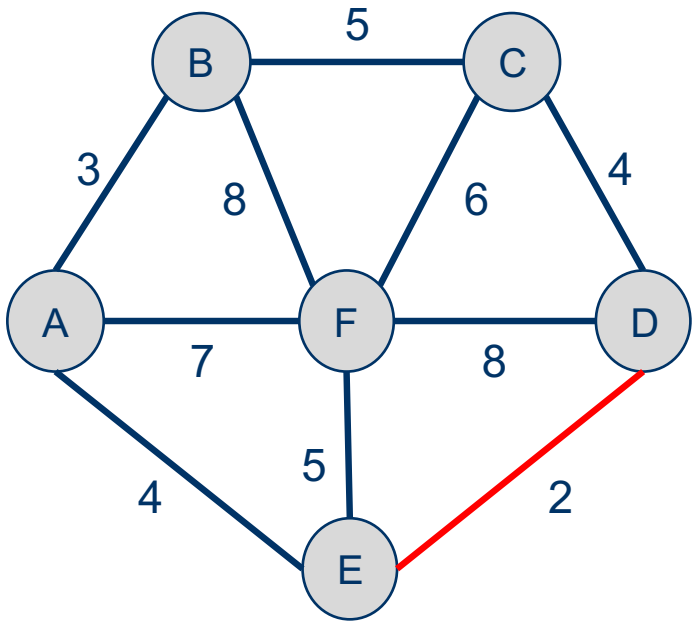
List the edges in order of size:

ED 2
AB 3
AE 4
CD 4
BC 5
EF 5
CF 6
AF 7
BF 8
CF 8

MST Kruskal's algorithm

Algoritmo de Kruskal

Example



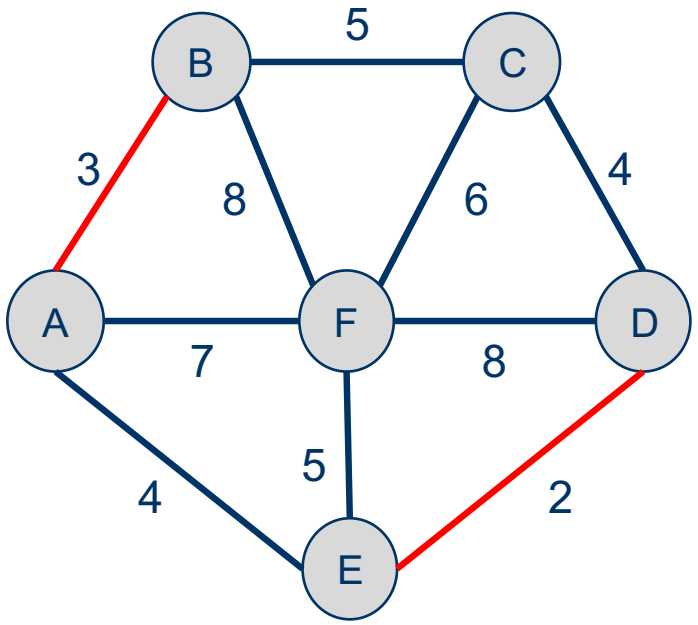
List the edges in order of size:

ED 2
AB 3
AE 4
CD 4
BC 5
EF 5
CF 6
AF 7
BF 8
CF 8

MST Kruskal's algorithm

Algoritmo de Kruskal

Example



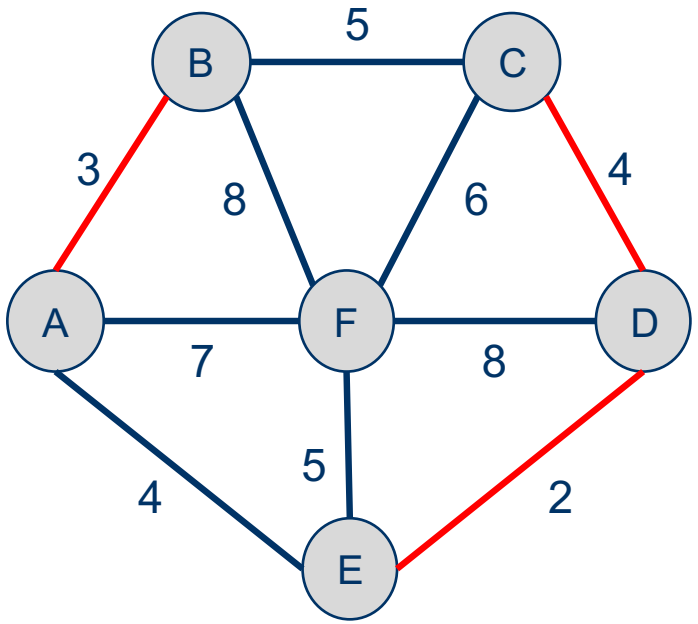
List the edges in order of size:

ED 2
AB 3
AE 4
CD 4
BC 5
EF 5
CF 6
AF 7
BF 8
CF 8

MST Kruskal's algorithm

Algoritmo de Kruskal

Example



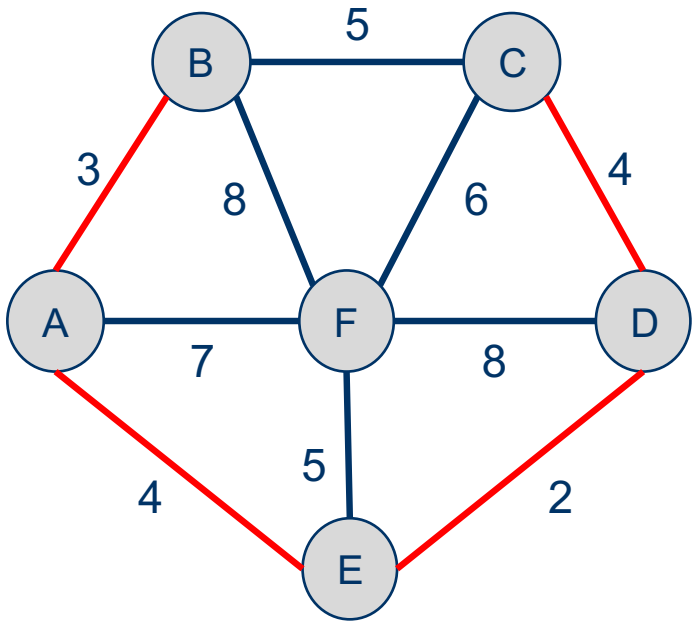
List the edges in order of size:

ED 2
AB 3
AE 4 pick one of
CD 4
BC 5
EF 5
CF 6
AF 7
BF 8
CF 8

MST Kruskal's algorithm

Algoritmo de Kruskal

Example



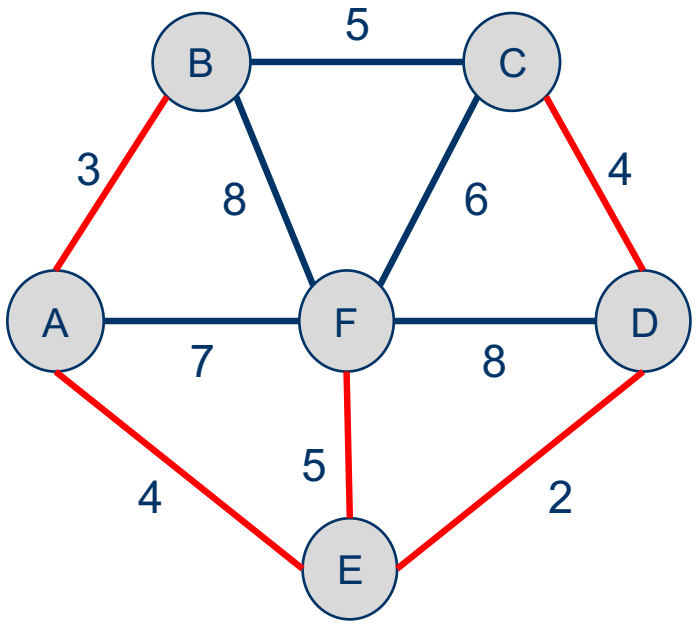
List the edges in order of size:

ED 2
AB 3
AE 4
CD 4
BC 5
EF 5
CF 6
AF 7
BF 8
CF 8

MST Kruskal's algorithm

Algoritmo de Kruskal

Example



List the edges in order of size:

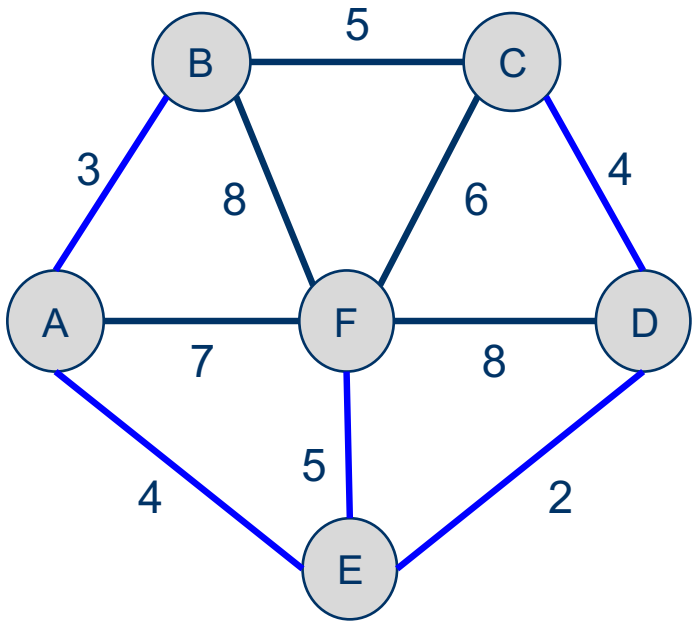
ED 2
AB 3
AE 4
CD 4
BC 5 Forms a cycle
EF 5
CF 6
AF 7
BF 8
CF 8

And now all vertices are connected

MST Kruskal's algorithm

Algoritmo de Kruskal

Example



All vertices have been connected.

The solution is

ED 2
AB 3
CD 4
AE 4
EF 5

Total weight of tree: 18

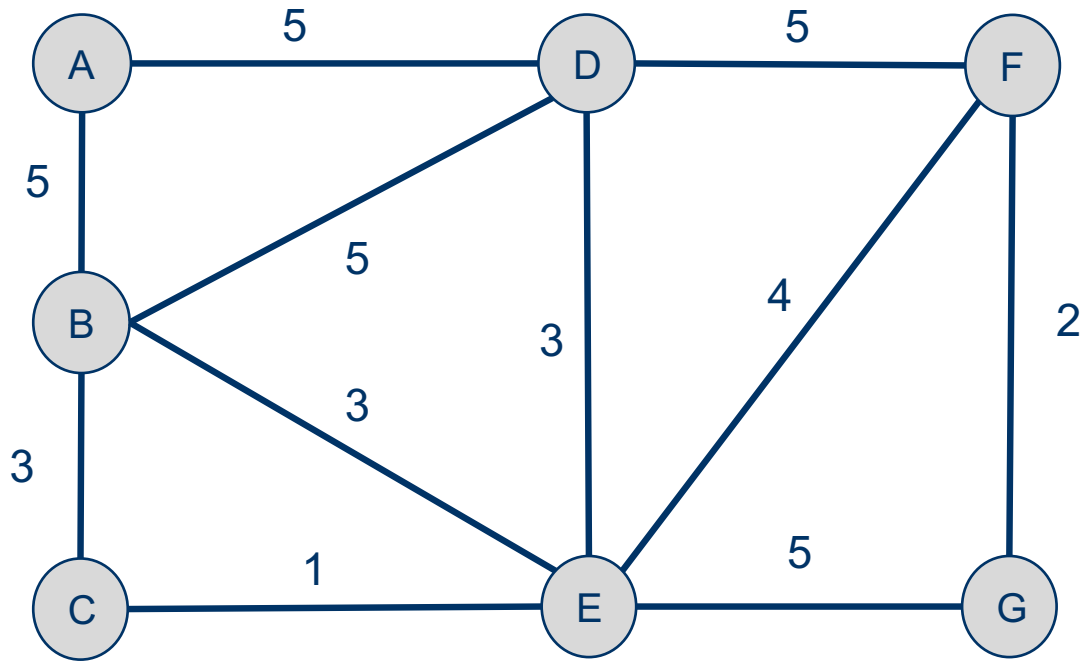
Minimum Spaning Tree

Ejercicios

Encontrar el peso mínimo
y el árbol

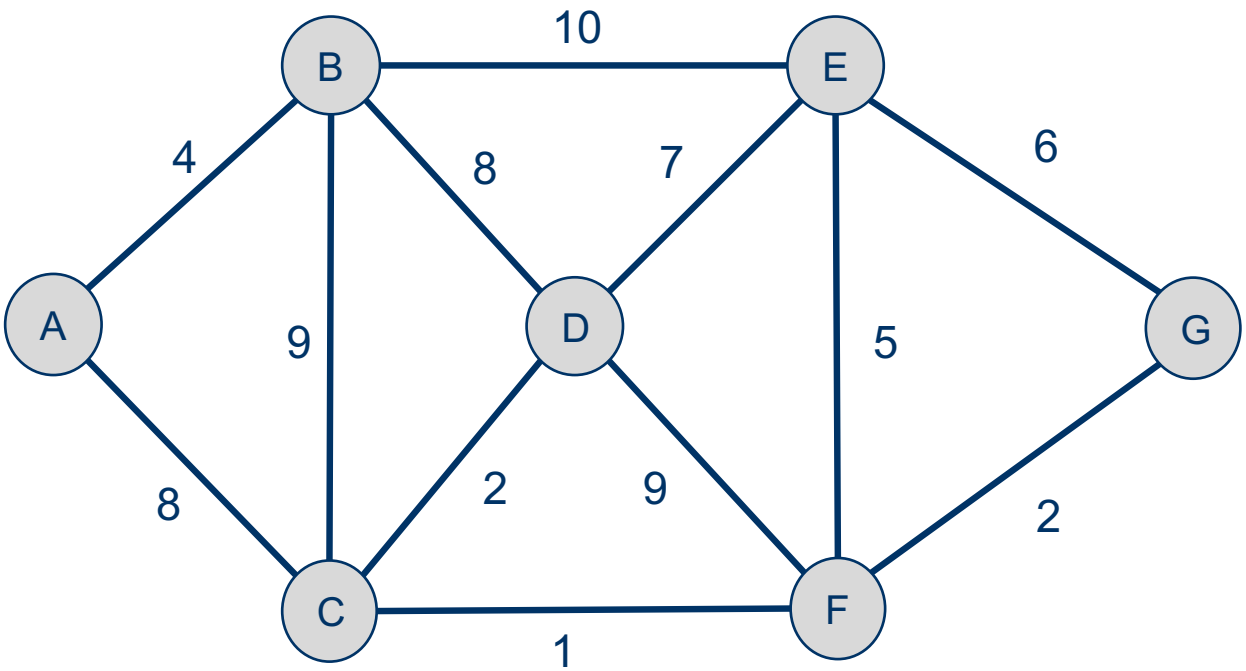
Examples 1

Ejercicio



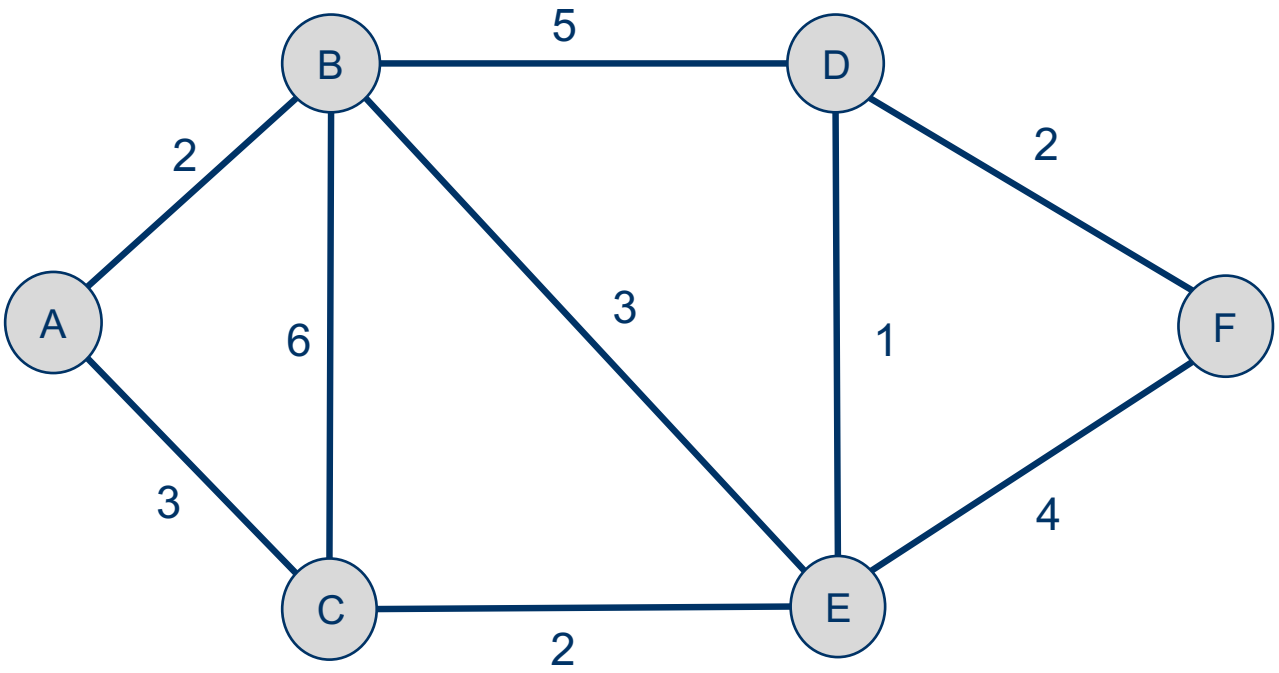
Examples 2

Ejercicio



Examples 3

Ejercicio



Solutions

Soluciones

Ejercicios

1. 7 nodos y 11 aristas
    1. Peso = 18
    2. Aristas = 6 = { (A,B), (D,E), (F,G), (C,E), (E,F), (B,E) }
2. 7 nodos y 12 aristas
    1. Peso = 22
    2. Aristas = 6 = {(A,B), (B,D), (D,C), (C,F), (F,E), (F,G) }
3. 6 nodos y 9 aristas.
    1. Peso = 10
    2. Aristas = 5 = { (A,B), (B,E), (C,E), (D,E), (D,F) }

The end

Contacto

Raúl Acosta Bermejo

http:www.cic.ipn.mx
http://www.ciseg.cic.ipn.mx/

racostab@ipn.mx
racosta@cic.ipn.mx

57-29-60-00
Ext. 56652

A recordar

Lo más importante es: