

INTELIGÊNCIA ARTIFICIAL

PARTE 4

Buscas em Árvores Mínimas

- Kruskal
- Dijkstra
- Prim



1

Algoritmo de Kruskal

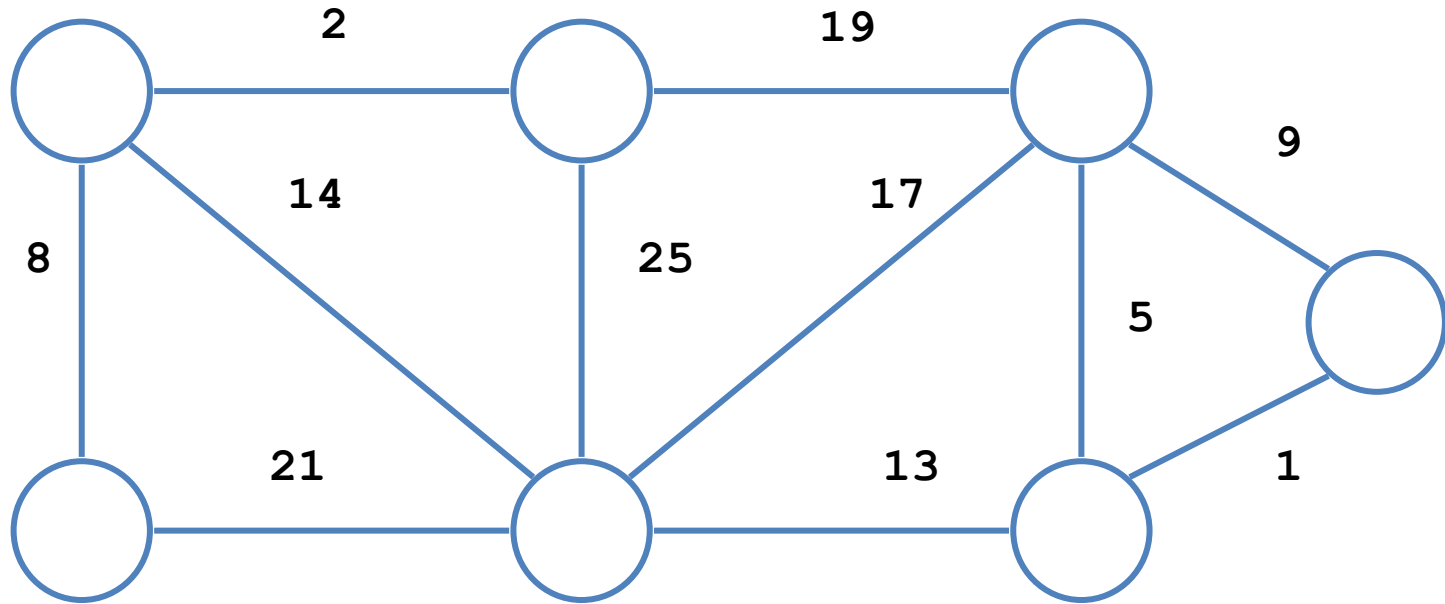
1 – Algoritmo de Kruskal

Características:

- É um algoritmo de busca, guloso, que se baseia no algoritmo de árvore espalhada de peso mínimo.
- Encontra uma aresta segura para adicionar à floresta crescente encontrando, de todas as arestas que conectam duas árvores quaisquer, uma aresta de peso mínimo.
- Encontra um subconjunto das arestas que forma uma árvore com todos os vértices, onde o peso total, dado pela soma dos pesos das arestas da árvore, é minimizado.

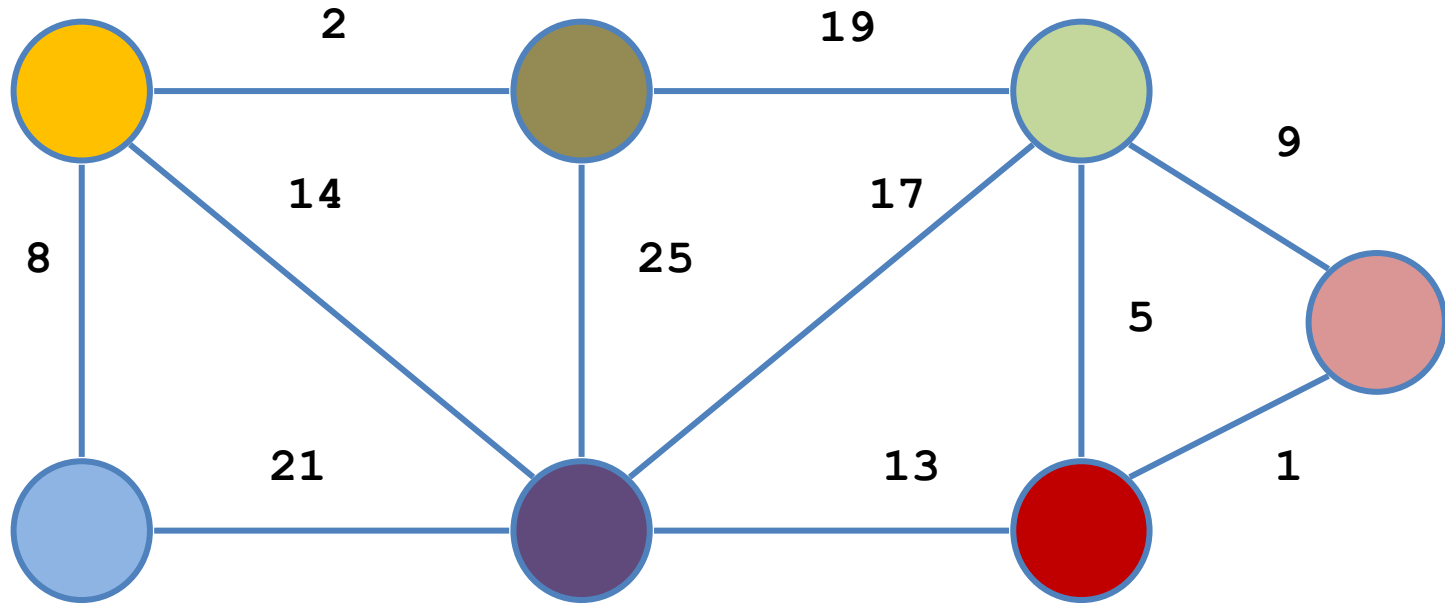
1 – Algoritmo de Kruskal

Exemplo:



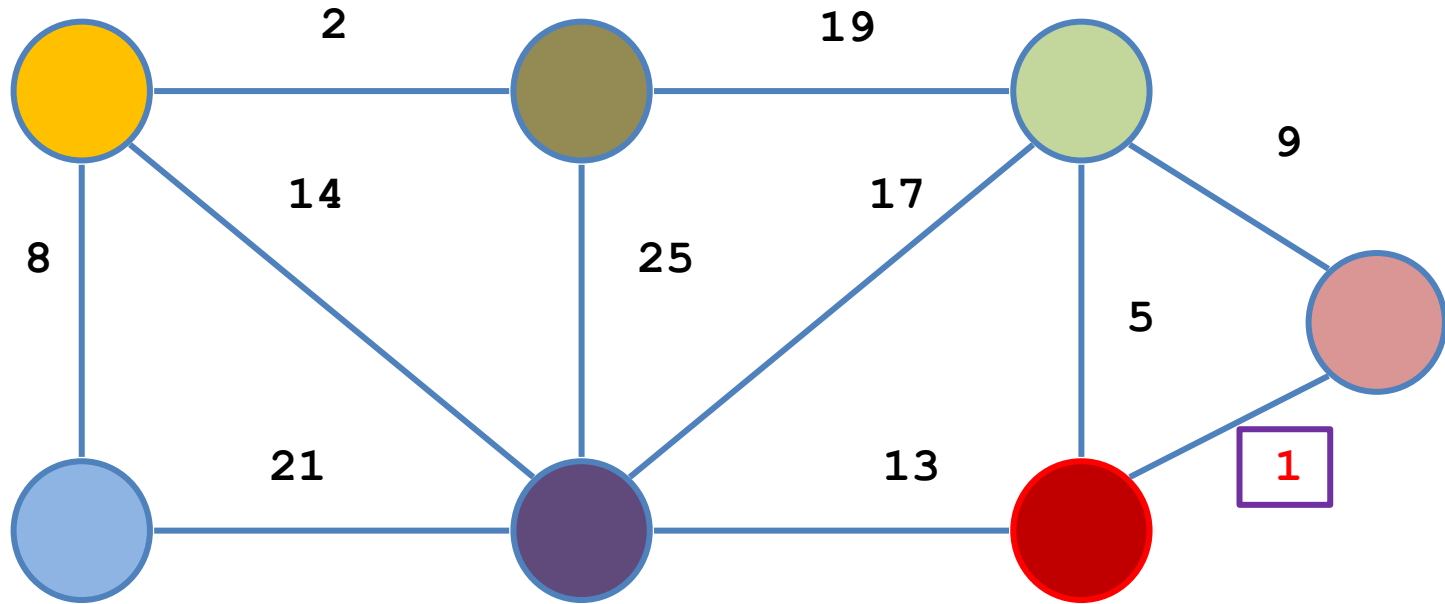
1 – Algoritmo de Kruskal

Exemplo:



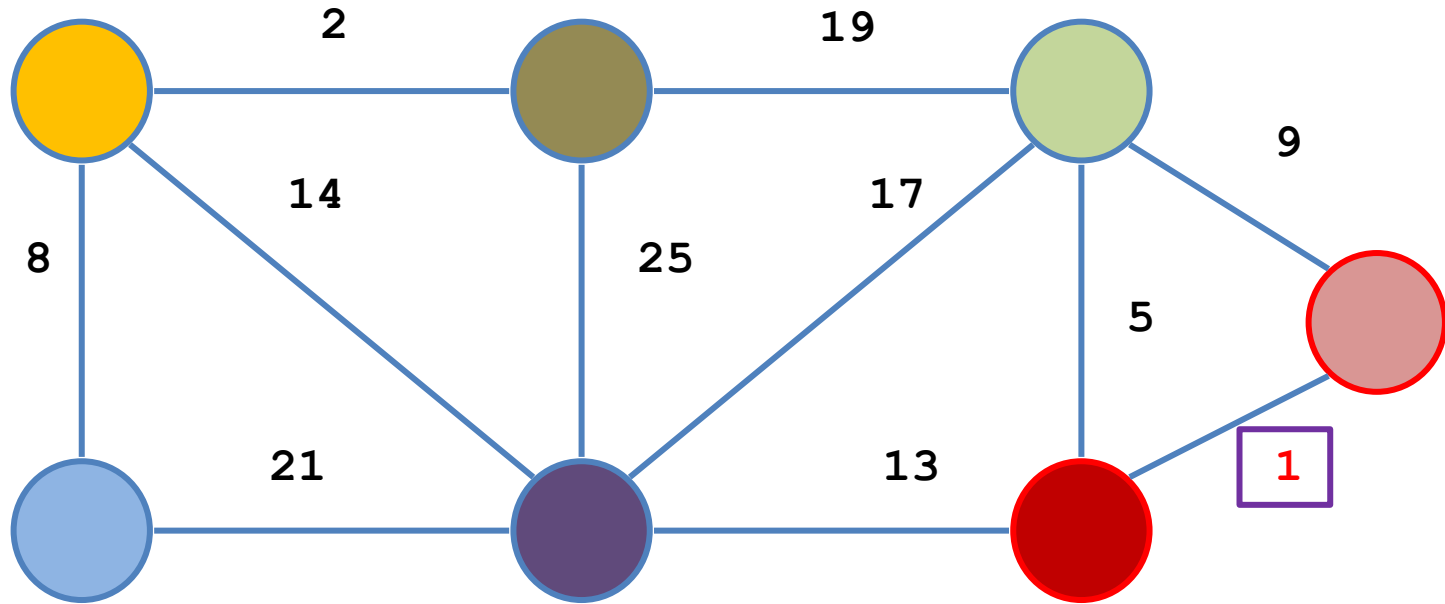
1 – Algoritmo de Kruskal

Exemplo:



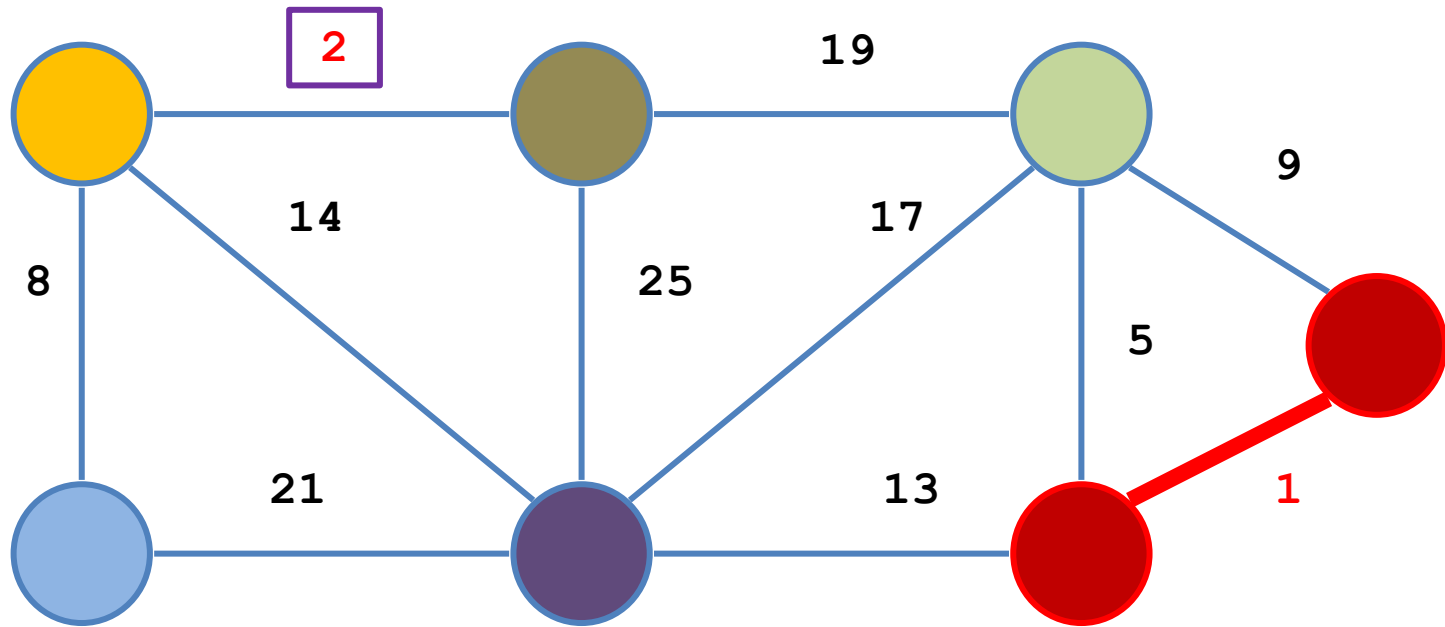
1 – Algoritmo de Kruskal

Exemplo:



1 – Algoritmo de Kruskal

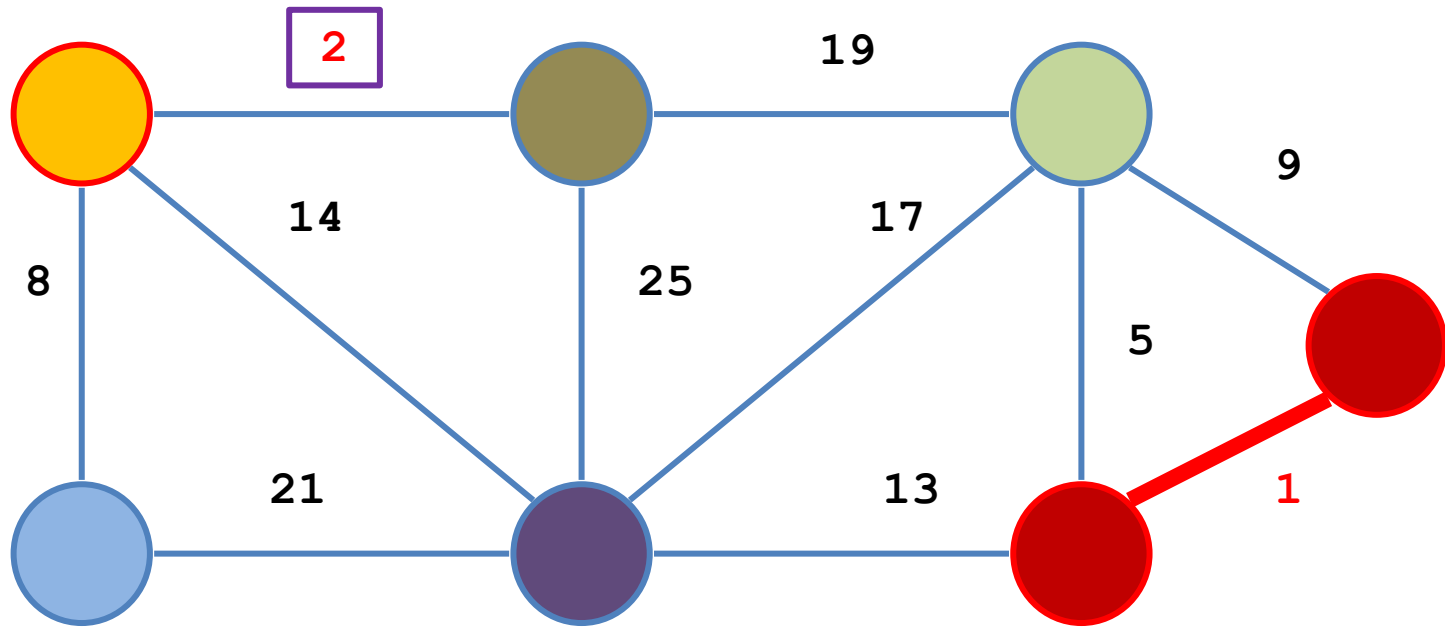
Exemplo:



$$w_{T_1} = 1$$

1 – Algoritmo de Kruskal

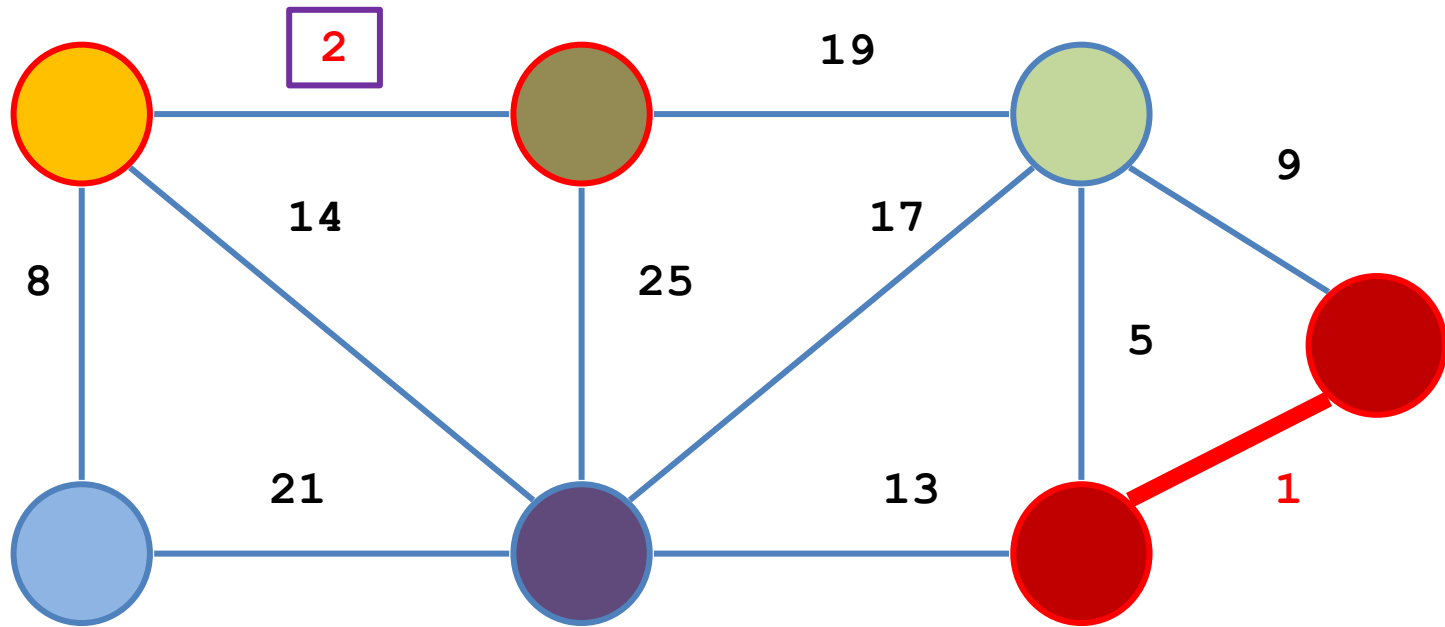
Exemplo:



$$w_{T_1} = 1$$

1 – Algoritmo de Kruskal

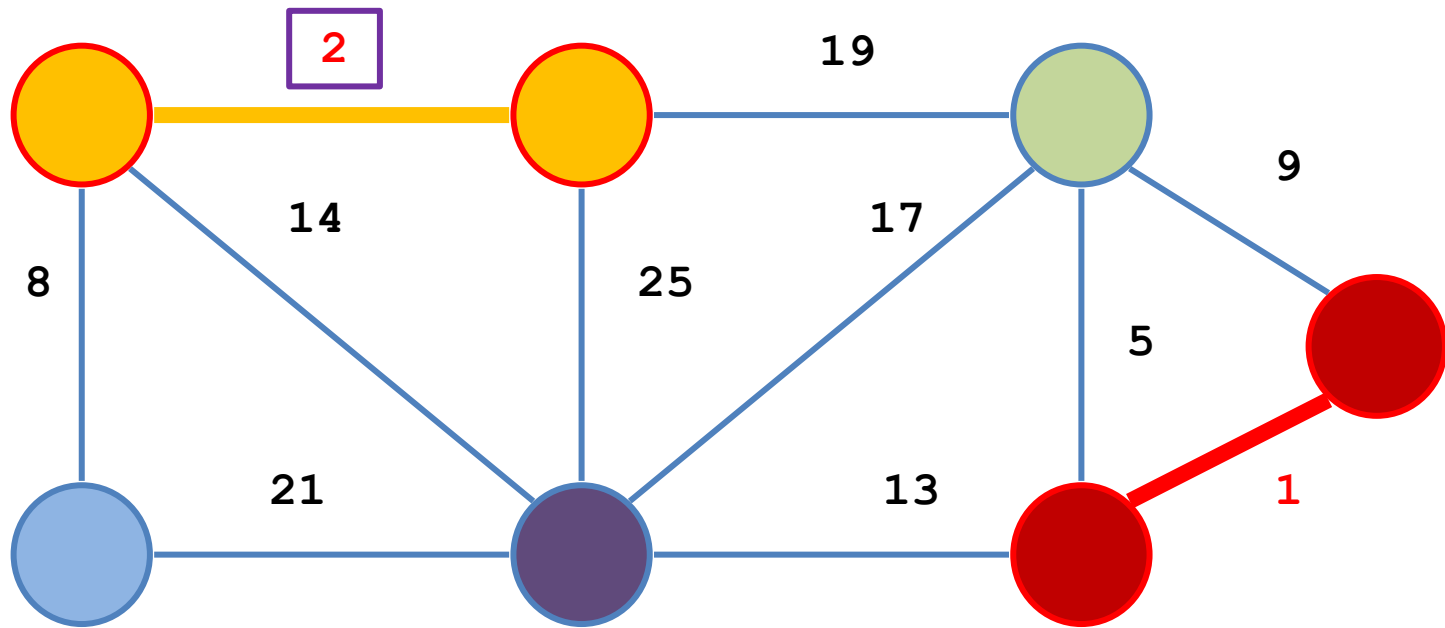
Exemplo:



$$w_{T_1} = 1$$

1 – Algoritmo de Kruskal

Exemplo:

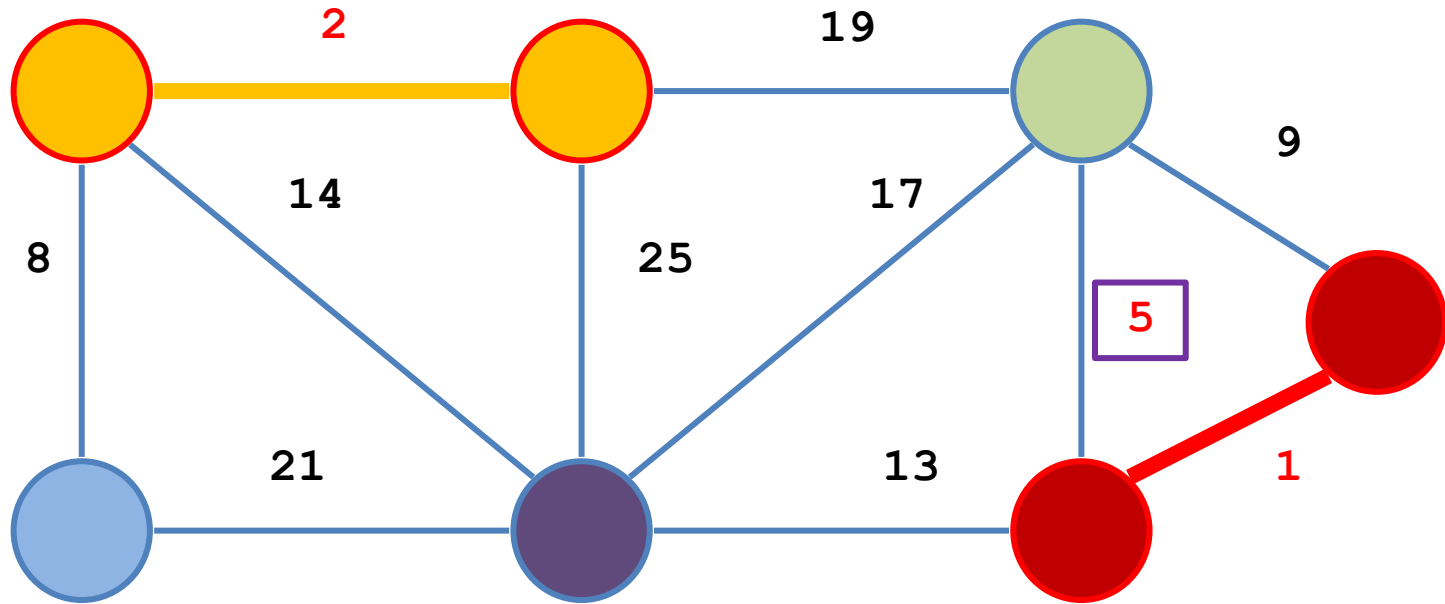


$$w_{T_1} = 1$$

$$w_{T_2} = 2$$

1 – Algoritmo de Kruskal

Exemplo:

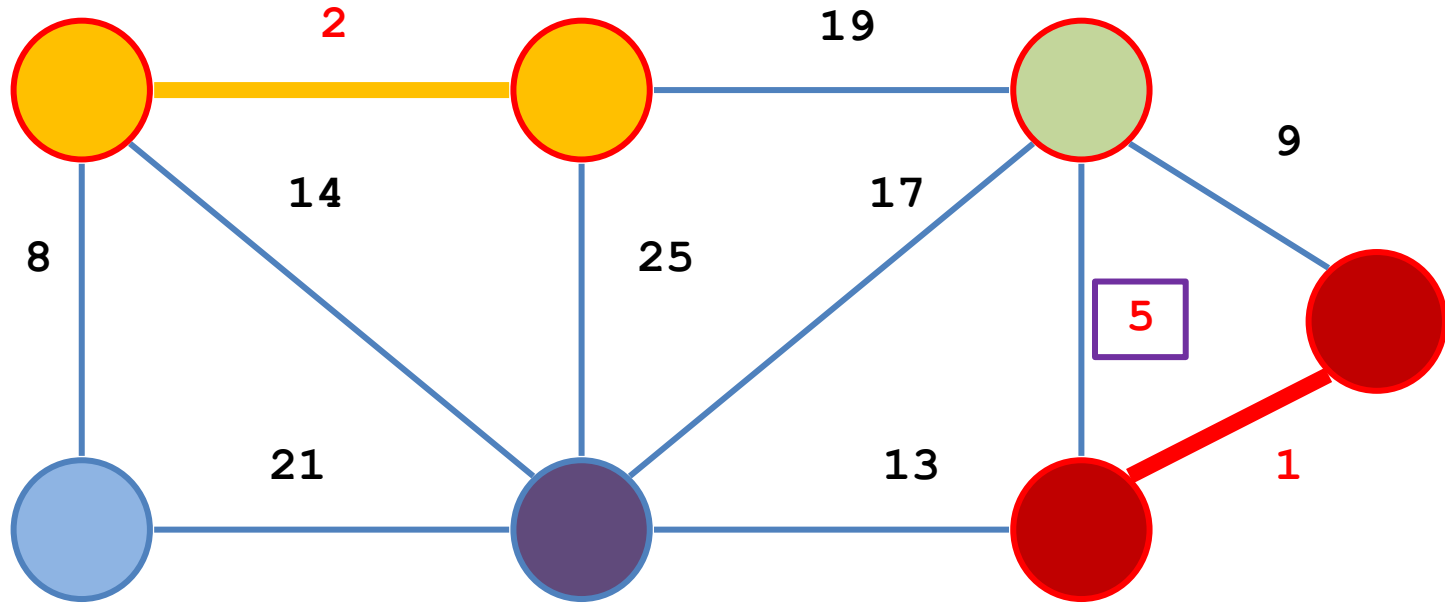


$$w_{T_1} = 1$$

$$w_{T_2} = 2$$

1 – Algoritmo de Kruskal

Exemplo:

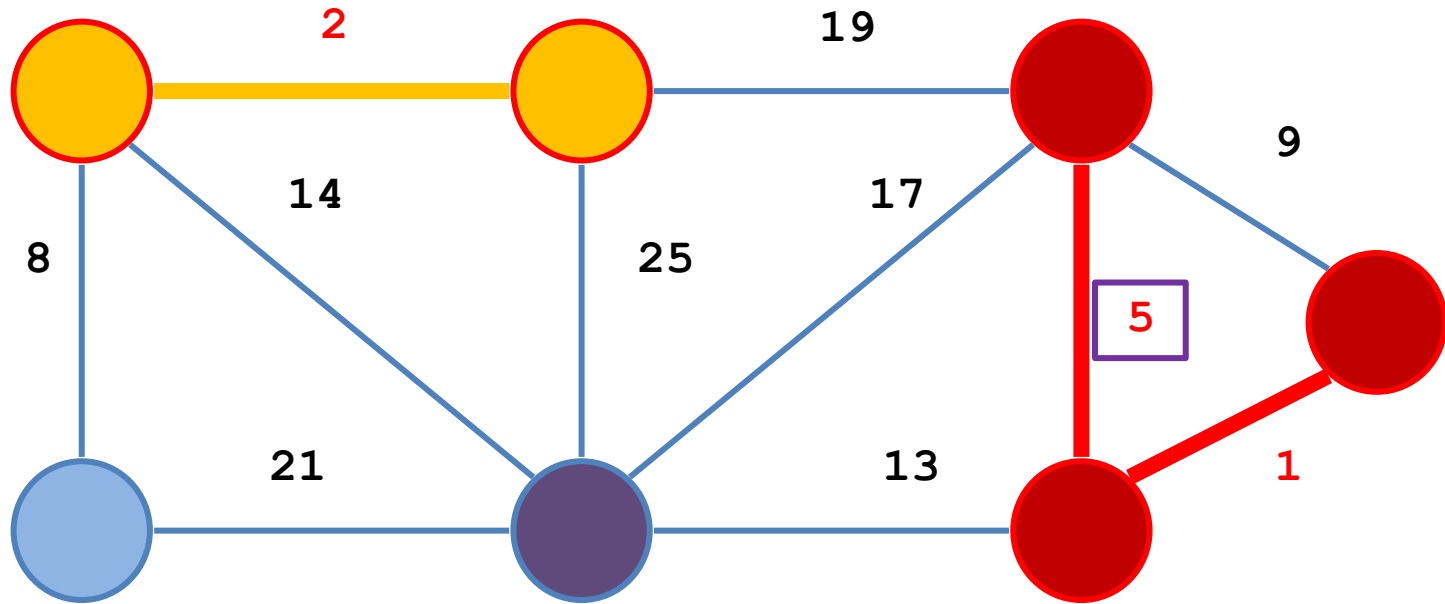


$$w_{T1} = 1$$

$$w_{T2} = 2$$

1 – Algoritmo de Kruskal

Exemplo:

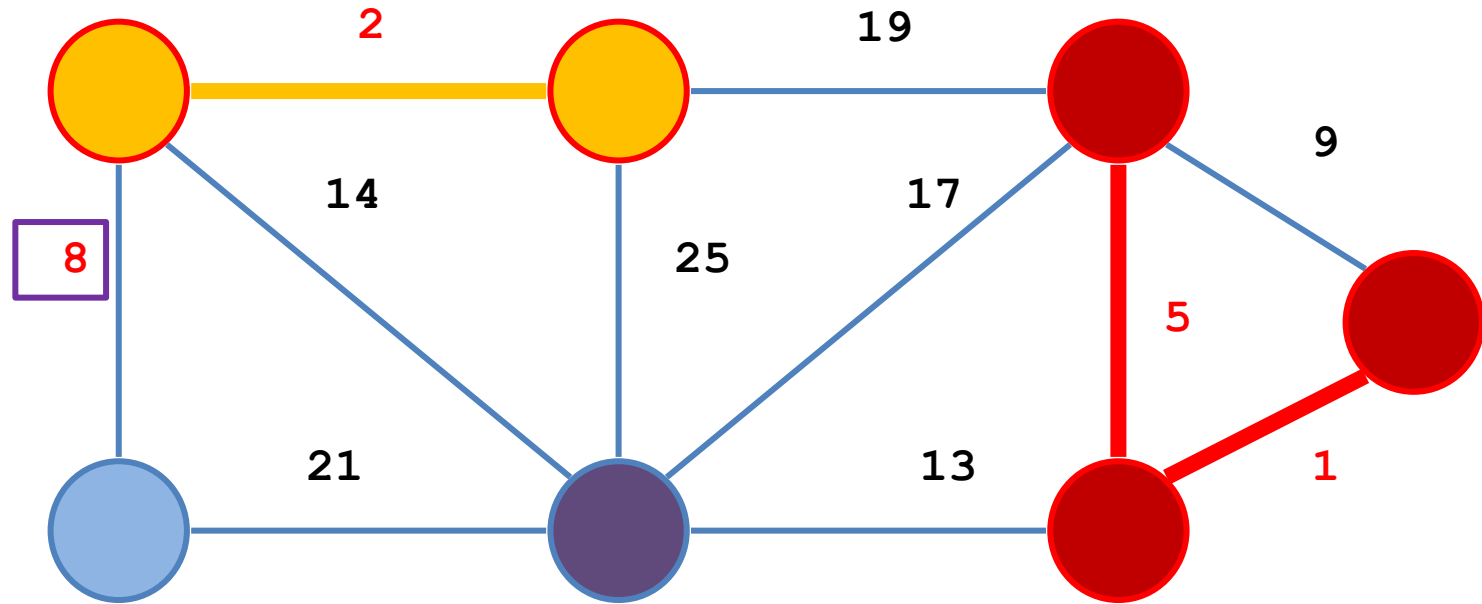


$$w_{T1} = 1 + 5$$

$$w_{T2} = 2$$

1 – Algoritmo de Kruskal

Exemplo:

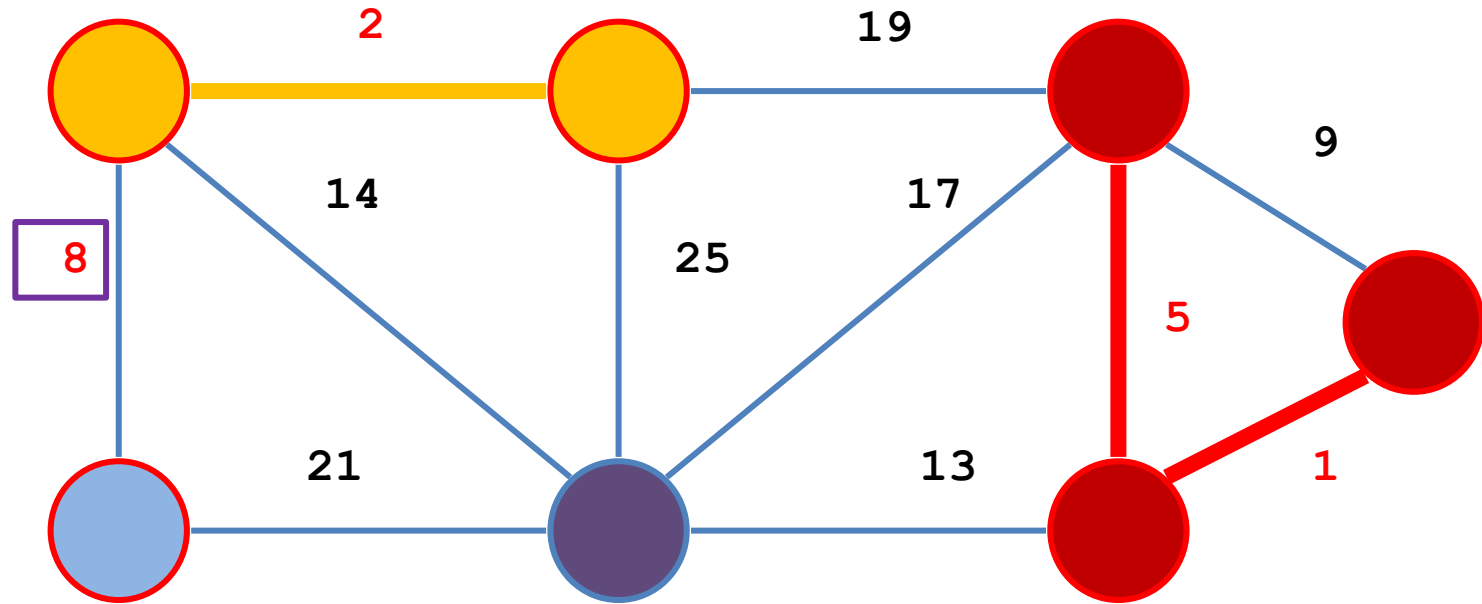


$$w_{T1} = 1 + 5$$

$$w_{T2} = 2$$

1 – Algoritmo de Kruskal

Exemplo:

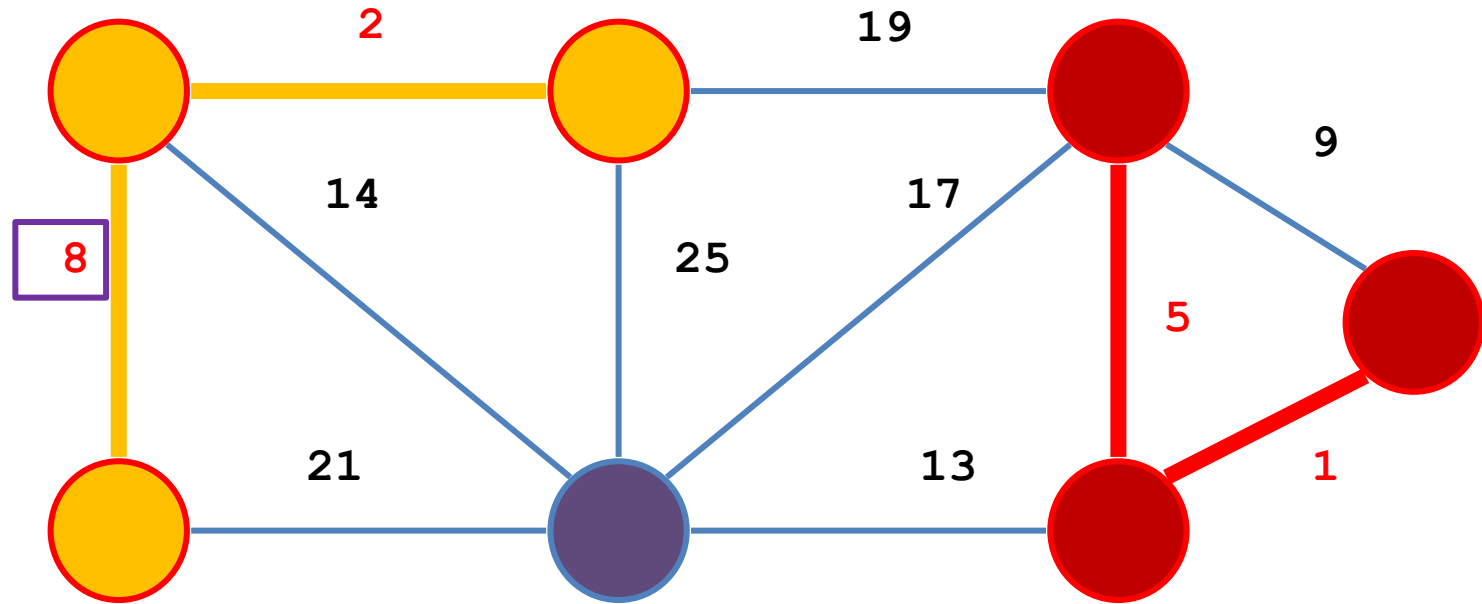


$$w_{T1} = 1 + 5$$

$$w_{T2} = 2$$

1 – Algoritmo de Kruskal

Exemplo:

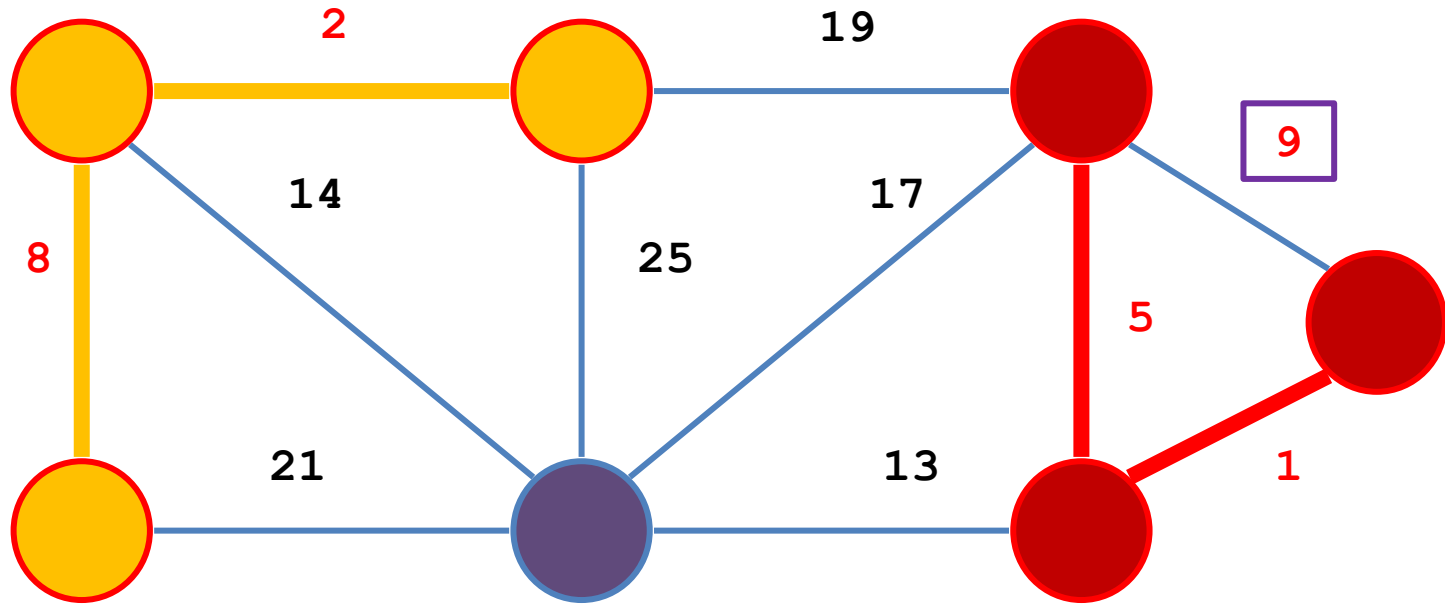


$$w_{T_1} = 1 + 5$$

$$w_{T_2} = 2 + 8$$

1 – Algoritmo de Kruskal

Exemplo:

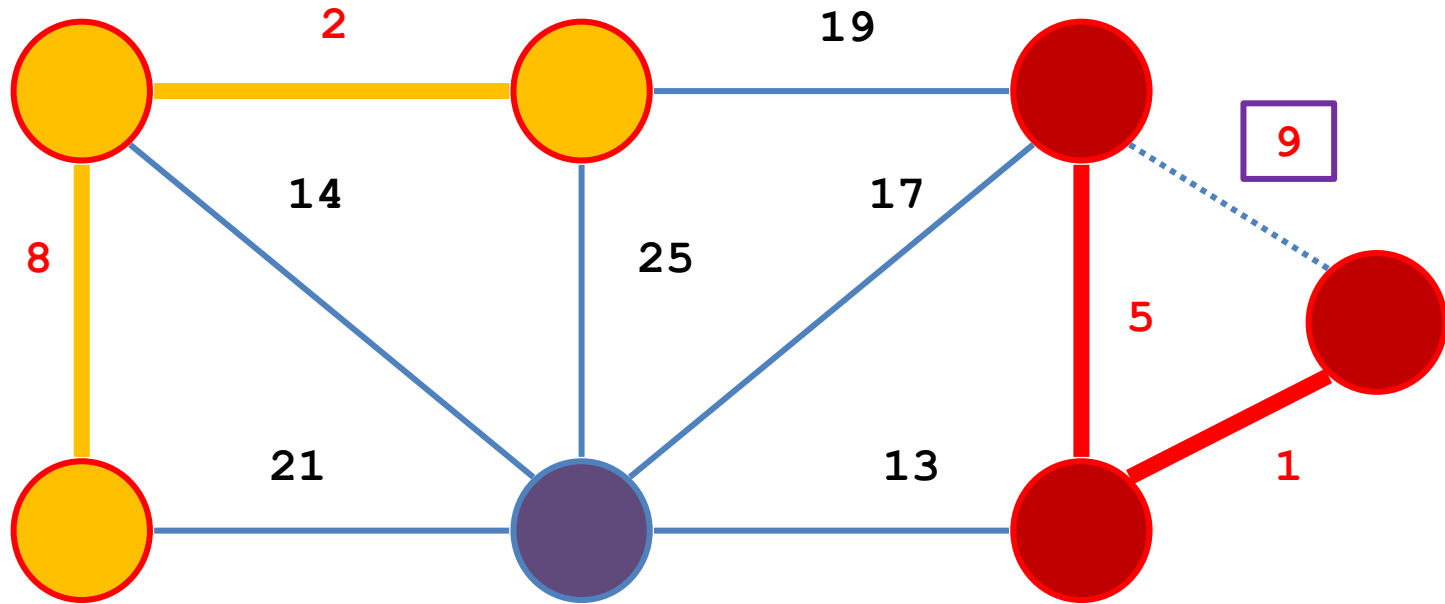


$$w_{T_1} = 1 + 5$$

$$w_{T_2} = 2 + 8$$

1 – Algoritmo de Kruskal

Exemplo:

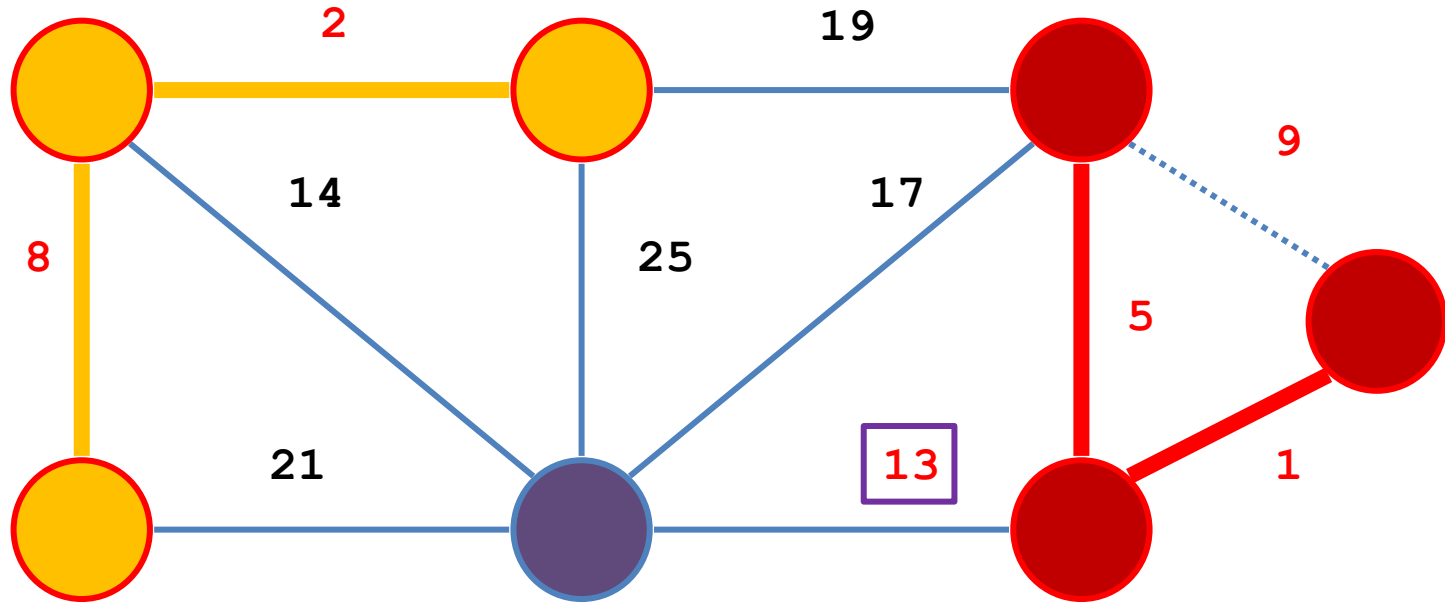


$$w_{T_1} = 1 + 5$$

$$w_{T_2} = 2 + 8$$

1 – Algoritmo de Kruskal

Exemplo:

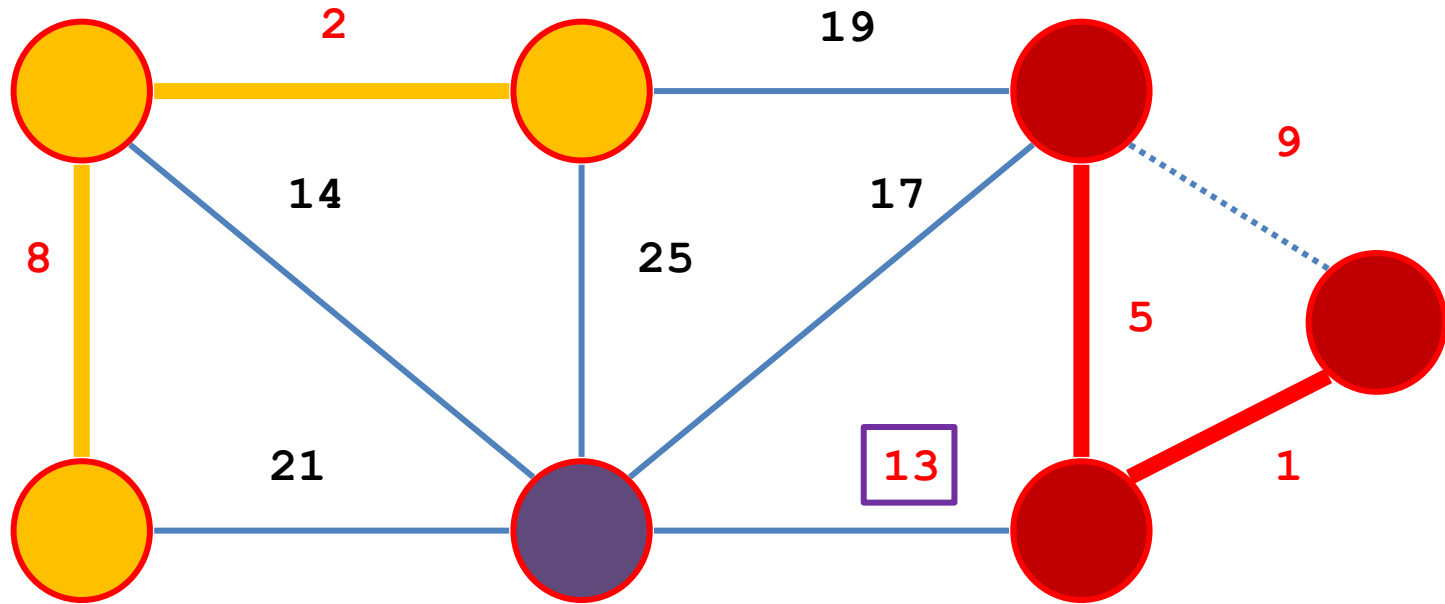


$$w_{T_1} = 1 + 5$$

$$w_{T_2} = 2 + 8$$

1 – Algoritmo de Kruskal

Exemplo:

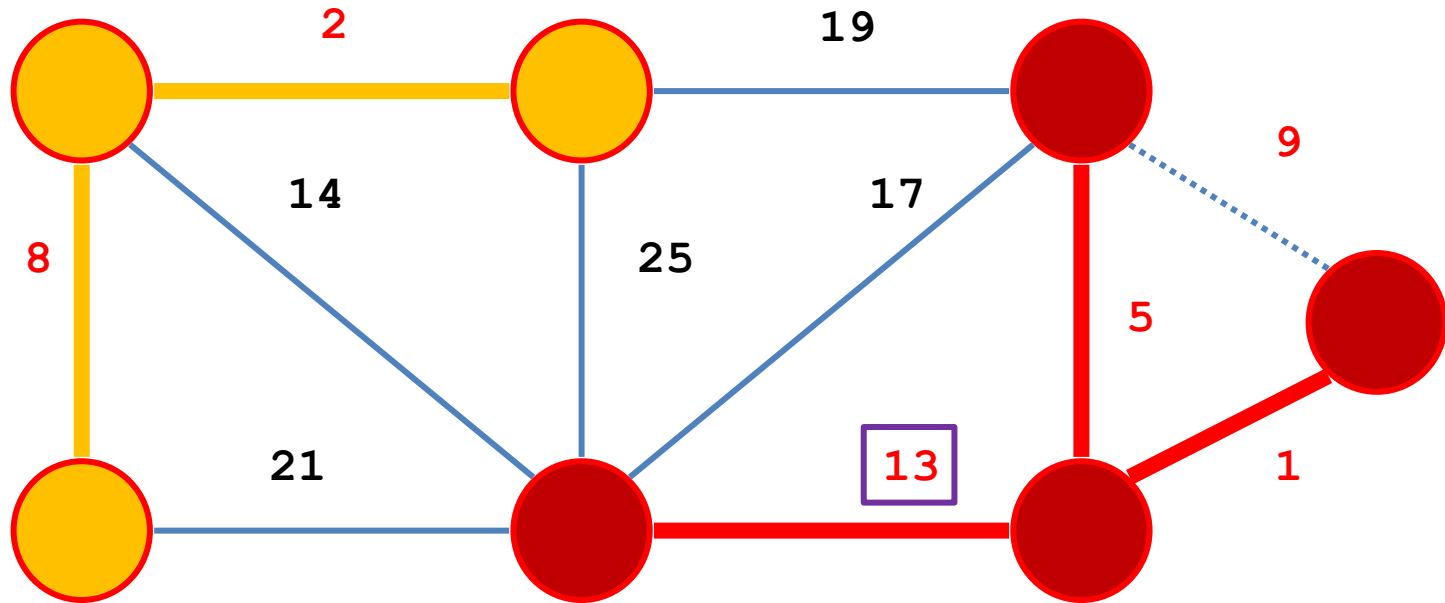


$$w_{T_1} = 1 + 5$$

$$w_{T_2} = 2 + 8$$

1 – Algoritmo de Kruskal

Exemplo:

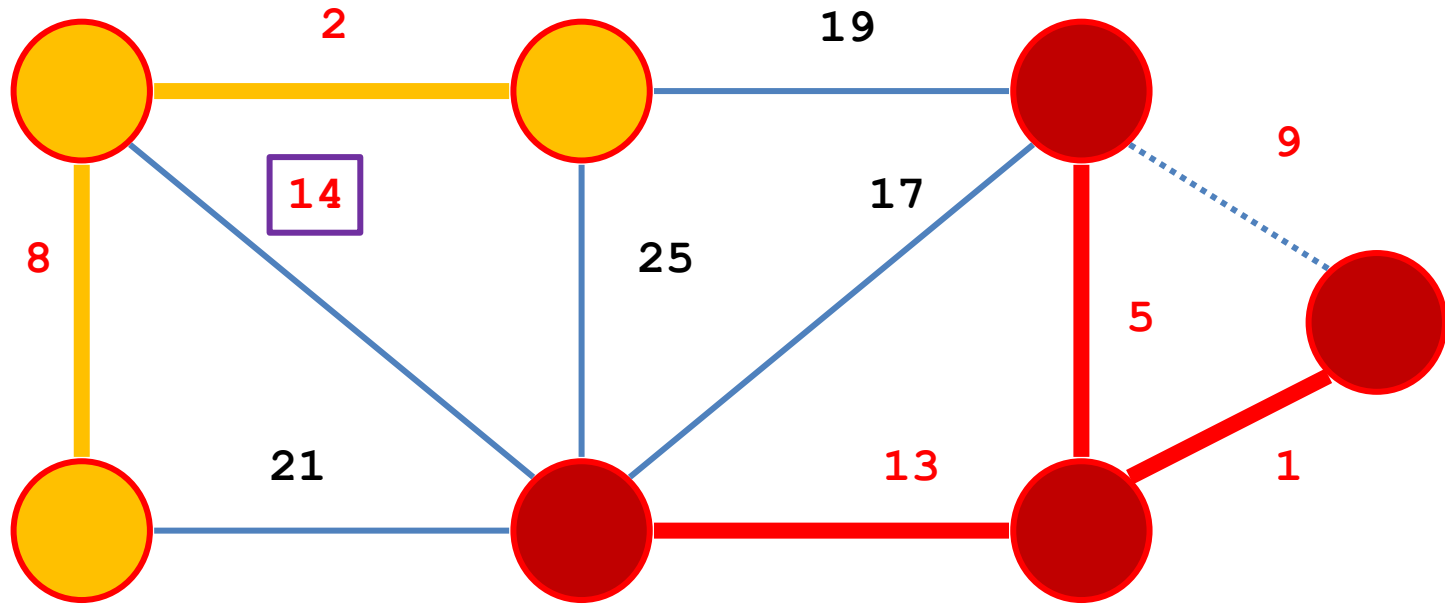


$$w_{T_1} = 1 + 5 + 13$$

$$w_{T_2} = 2 + 8$$

1 – Algoritmo de Kruskal

Exemplo:

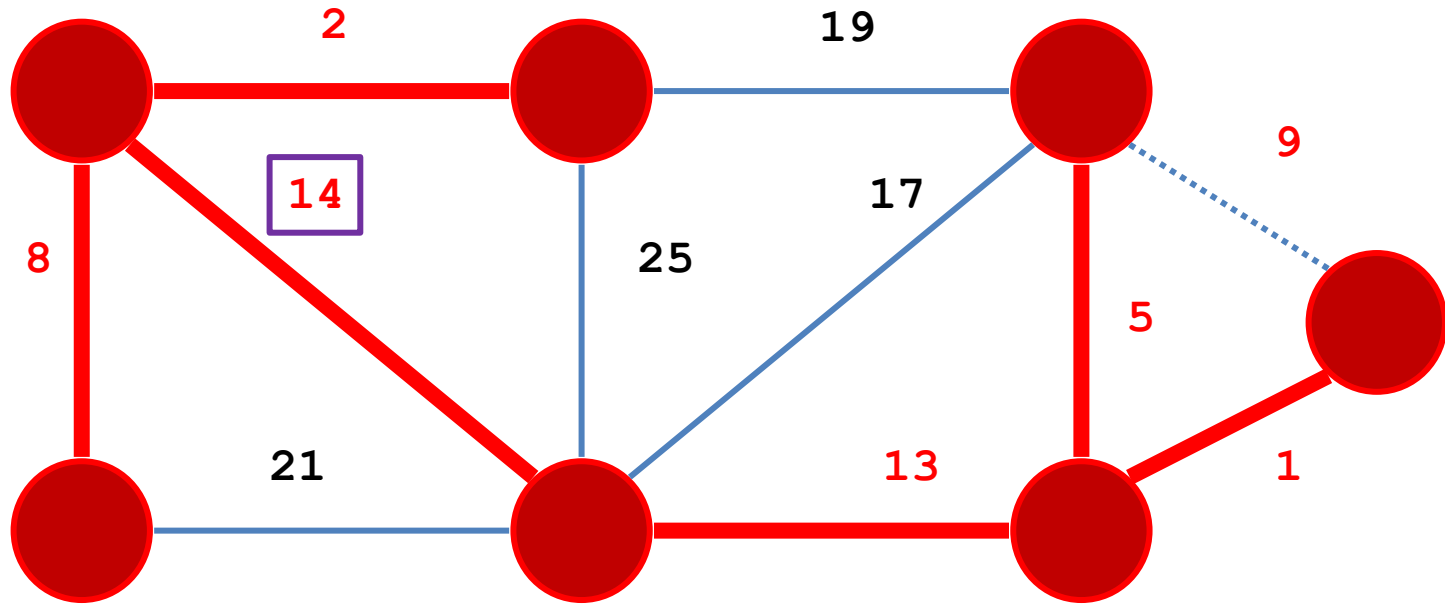


$$w_{T_1} = 1 + 5 + 13$$

$$w_{T_2} = 2 + 8$$

1 – Algoritmo de Kruskal

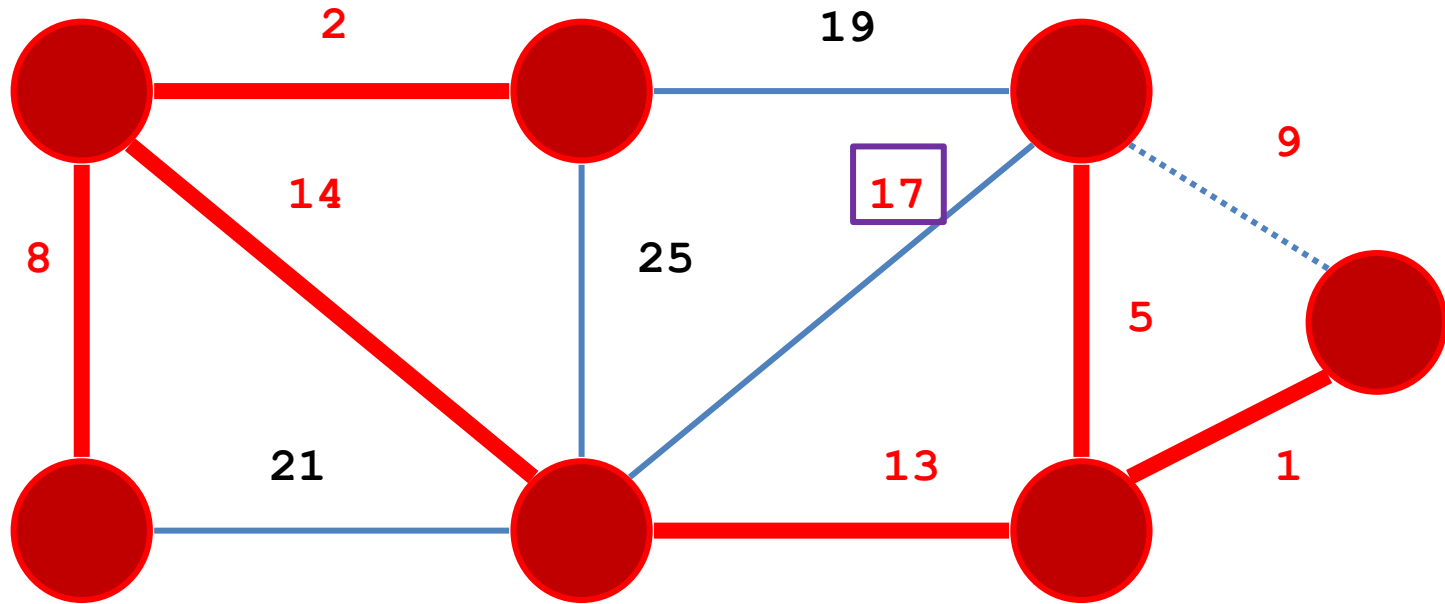
Exemplo:



$$\left. \begin{array}{l} w_{T_1} = 1 + 5 + 13 \\ w_{T_2} = 2 + 8 \end{array} \right\} w_{T_1 + T_2} = 19 + 10 + 14$$

1 – Algoritmo de Kruskal

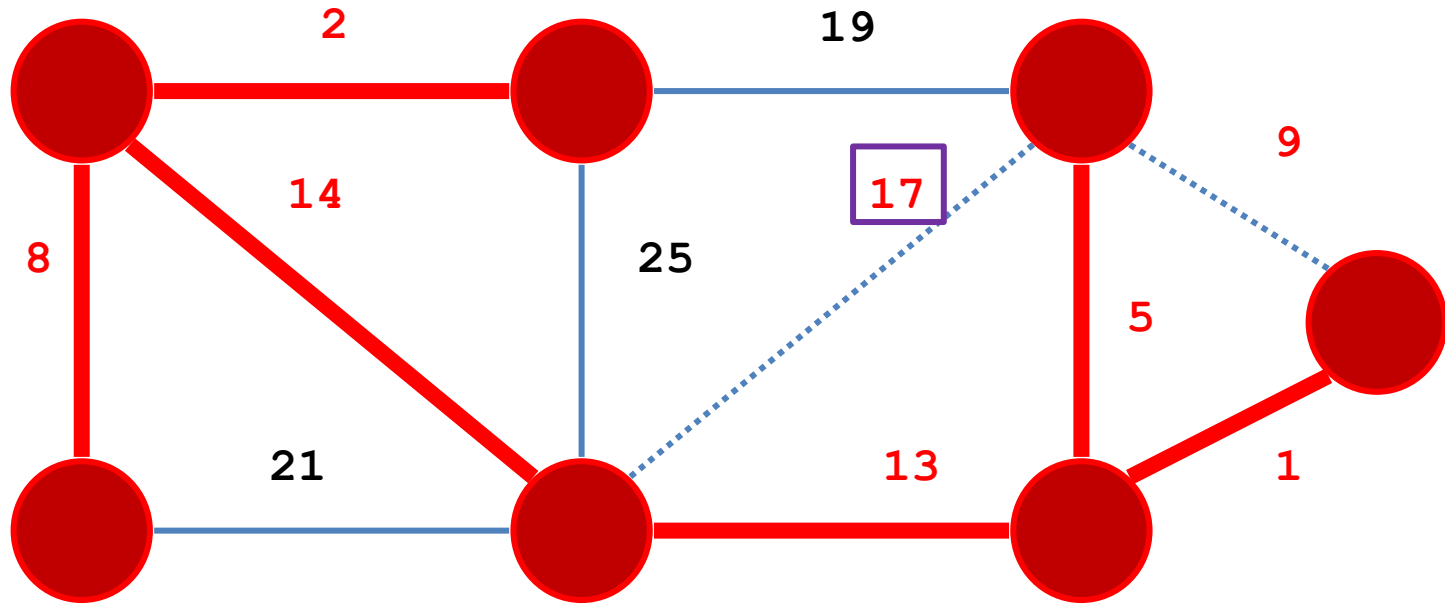
Exemplo:



$$\left. \begin{array}{l} w_{T_1} = 1 + 5 + 13 \\ w_{T_2} = 2 + 8 \end{array} \right\} w_{T_1 + T_2} = 19 + 10 + 14$$

1 – Algoritmo de Kruskal

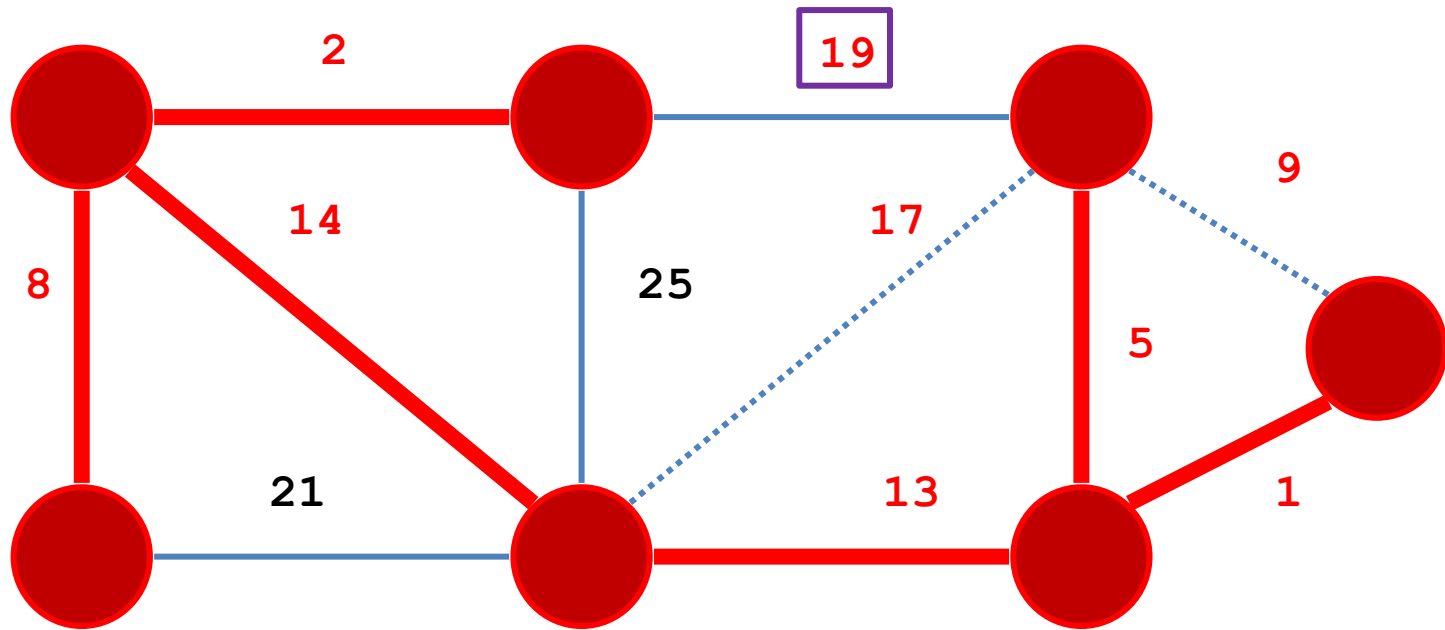
Exemplo:



$$\left. \begin{array}{l} w_{T_1} = 1 + 5 + 13 \\ w_{T_2} = 2 + 8 \end{array} \right\} w_{T_1 + T_2} = 19 + 10 + 14$$

1 – Algoritmo de Kruskal

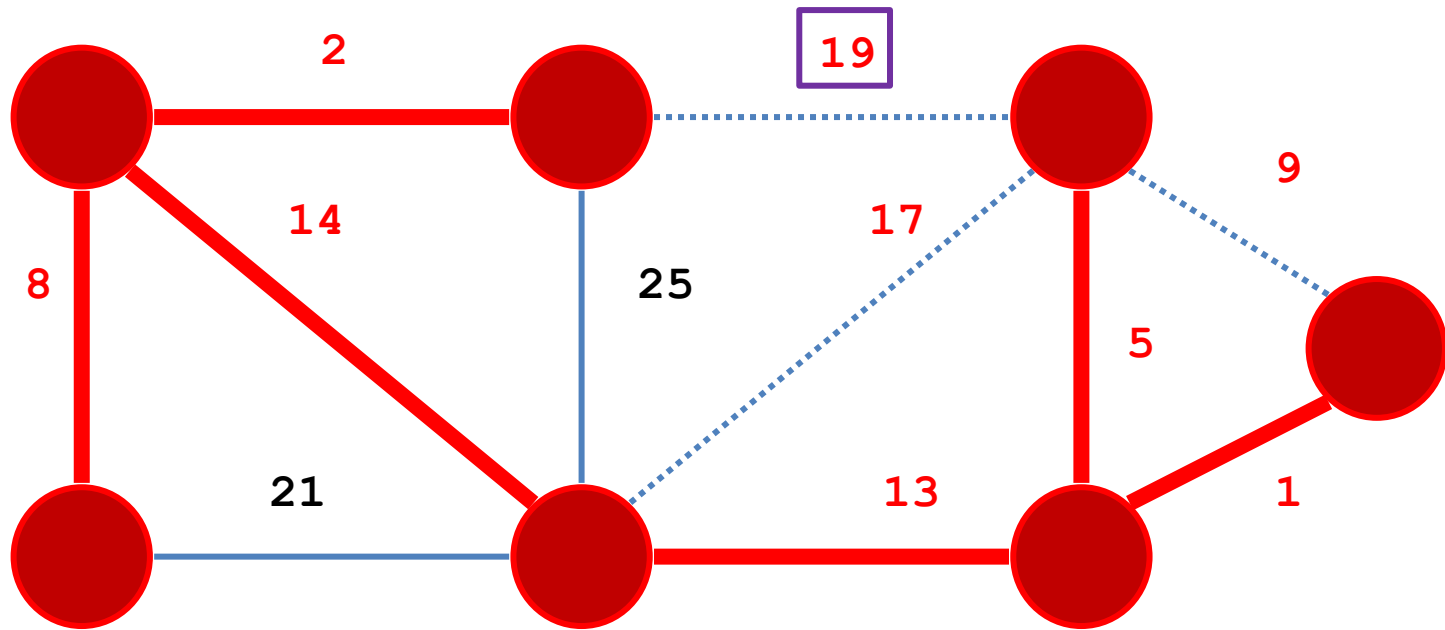
Exemplo:



$$\left. \begin{array}{l} w_{T_1} = 1 + 5 + 13 \\ w_{T_2} = 2 + 8 \end{array} \right\} w_{T_1 + T_2} = 19 + 10 + 14$$

1 – Algoritmo de Kruskal

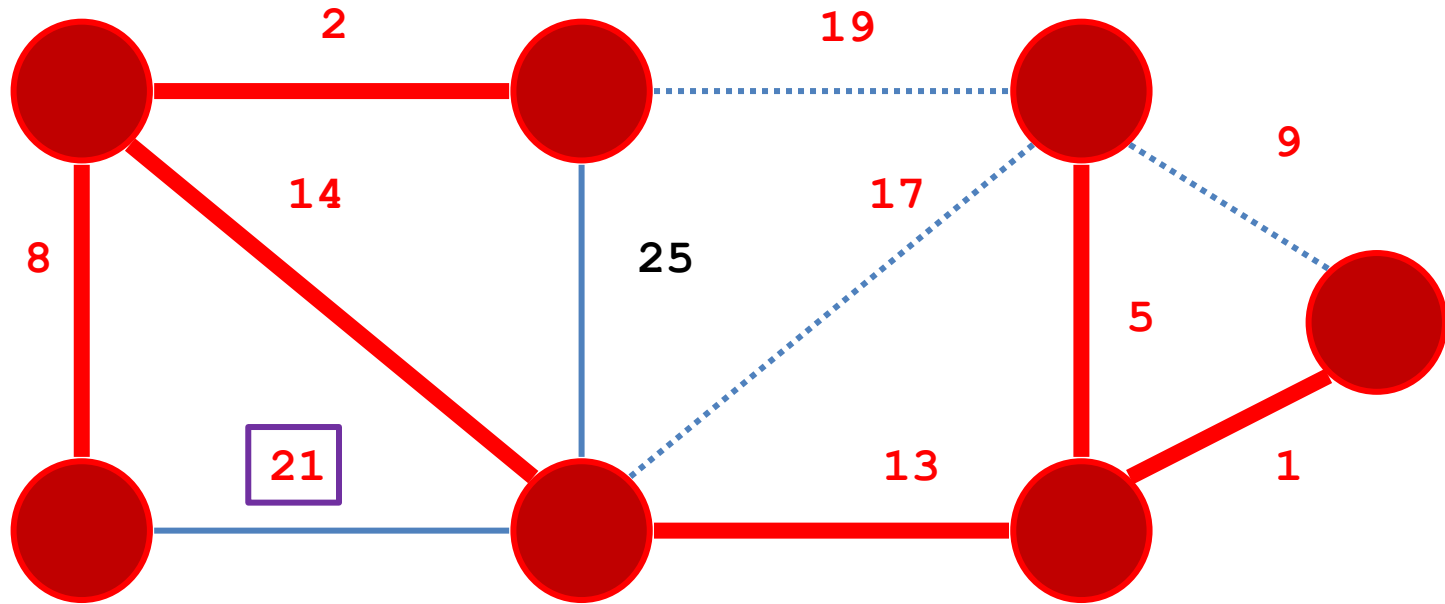
Exemplo:



$$\left. \begin{array}{l} w_{T_1} = 1 + 5 + 13 \\ w_{T_2} = 2 + 8 \end{array} \right\} w_{T_1 + T_2} = 19 + 10 + 14$$

1 – Algoritmo de Kruskal

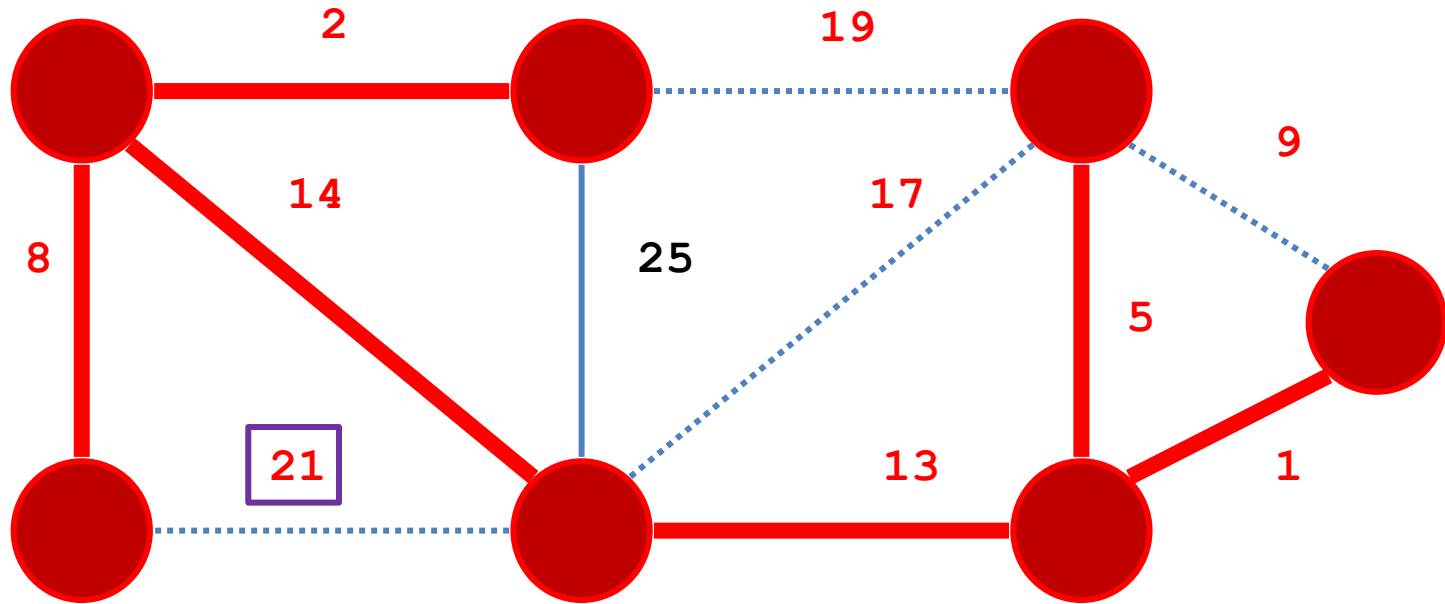
Exemplo:



$$\left. \begin{array}{l} w_{T_1} = 1 + 5 + 13 \\ w_{T_2} = 2 + 8 \end{array} \right\} w_{T_1 + T_2} = 19 + 10 + 14$$

1 – Algoritmo de Kruskal

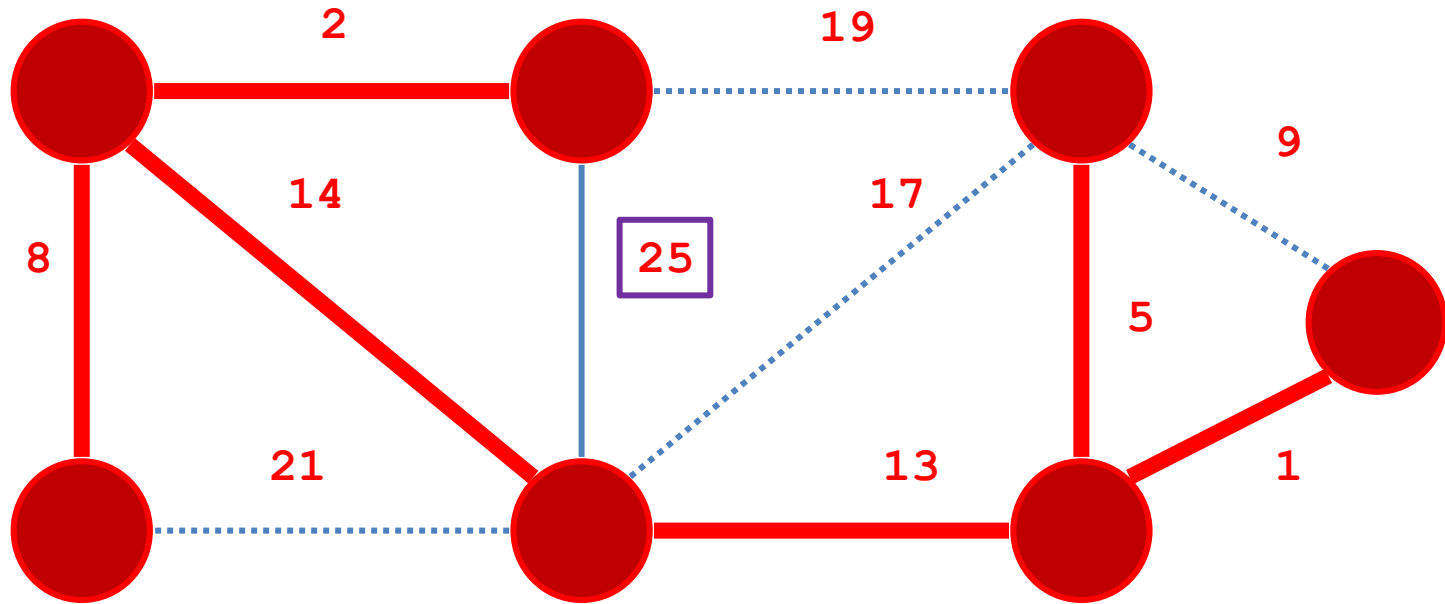
Exemplo:



$$\left. \begin{array}{l} w_{T_1} = 1 + 5 + 13 \\ w_{T_2} = 2 + 8 \end{array} \right\} w_{T_1 + T_2} = 19 + 10 + 14$$

1 – Algoritmo de Kruskal

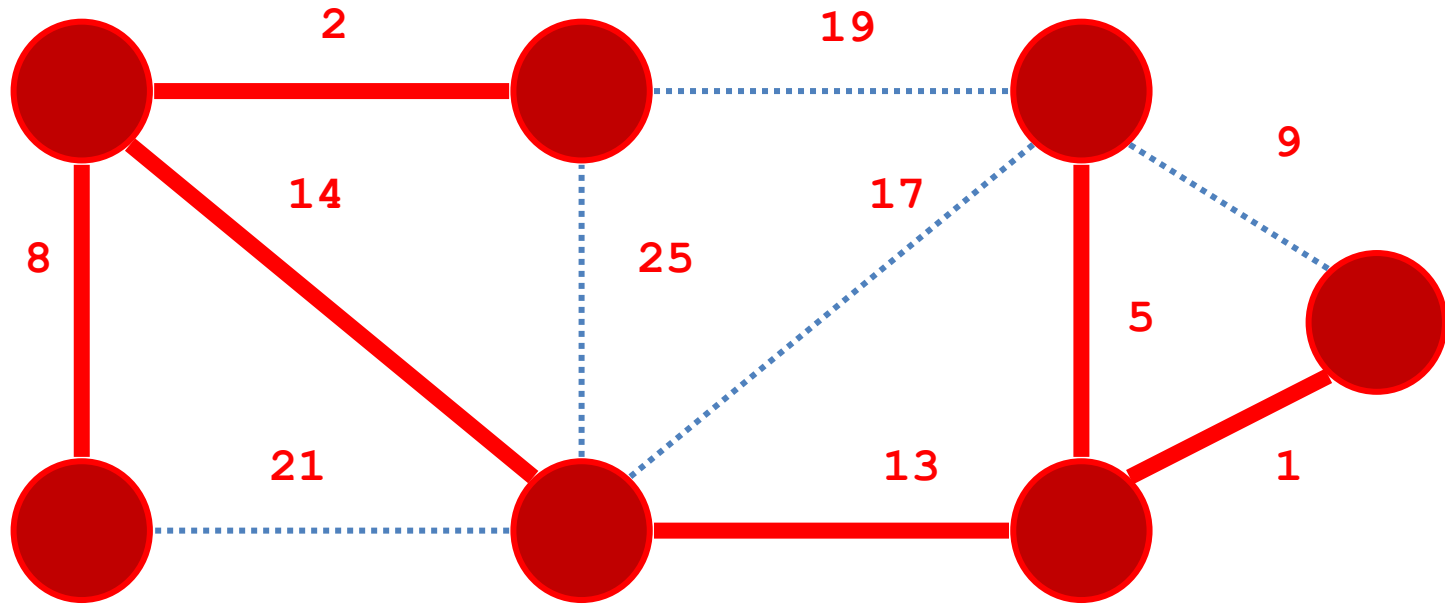
Exemplo:



$$\left. \begin{array}{l} w_{T_1} = 1 + 5 + 13 \\ w_{T_2} = 2 + 8 \end{array} \right\} w_{T_1 + T_2} = 19 + 10 + 14$$

1 – Algoritmo de Kruskal

Exemplo:



$$w_{T_1} = 1 + 5 + 13$$

$$w_{T_2} = 2 + 8$$

$$w_{T_1 + T_2} = 19 + 10 + 14 = 43$$

1 – Algoritmo de Kruskal

Algoritmo:

```
Kruskal() {  
  1  A =  $\emptyset$ ;  
  2  for cada vértice  $v \in V[G]$   
  3      do MakeSet( $v$ ) ;  
  4  Ordene as arestas E de forma crescente ao peso w  
  5  for cada aresta  $(u,v) \in E$  (ordenado)  
  6      do if FindSet( $u$ )  $\neq$  FindSet( $v$ )  
  7          then A = A  $\cup$  {  $(u,v)$  } ;  
  8  Union(FindSet( $u$ ) , FindSet( $v$ )) ;  
  9  return A ;  
}
```

1 – Algoritmo de Kruskal

Algoritmo:

Kruskal () {

1 $A = \emptyset$;

2 for cada vértice $v \in V[G]$

3 do MakeSet(v) ;

4 Ordene as arestas E de forma crescente ao peso w

5 for cada aresta $(u,v) \in E$ (ordenado)

6 do if FindSet(u) \neq FindSet(v)

7 then $A = A \cup \{(u,v)\}$;

8 Union(FindSet(u) , FindSet(v)) ;

9 return A ;

}

Inicializa o conjunto **A** como o conjunto vazio e cria **V** árvores, cada uma contendo 1 vértice.

1 – Algoritmo de Kruskal

Algoritmo:

```
Kruskal() {  
  1  A =  $\emptyset$ ;  
  2  for cada vértice  $v \in V[G]$   As arestas em E são ordenadas em  
  3      do MakeSet(v);          sequência crescente por peso.  
  4  Ordene as arestas E de forma crescente ao peso w  
  5  for cada aresta  $(u,v) \in E$  (ordenado)  
  6      do if FindSet(u)  $\neq$  FindSet(v)  
  7          then A = A  $\cup$  { (u,v) };  
  8  Union(FindSet(u), FindSet(v));  
  9  return A;  
}
```

1 – Algoritmo de Kruskal

Algoritmo:

```
Kruskal() {  
  1  A =  $\emptyset$ ;  
  2  for cada vértice  $v \in V[G]$   
  3      do MakeSet( $v$ ) ;  
  4  Ordene as arestas E de forma crescente ao peso w  
  5  for cada aresta  $(u,v) \in E$  (ordenado)  
  6      do if FindSet( $u$ )  $\neq$  FindSet( $v$ )  
  7          then A = A  $\cup$  {  $(u,v)$  } ;  
  8  Union(FindSet( $u$ ) , FindSet( $v$ ) ) ;  
  9  return A ;  
}
```

Verifica, para cada aresta (u, v) , se os pontos extremos u e v pertencem à mesma árvore. Se sim, a aresta é descartada pois **não é permitido ciclo**. Se não, a aresta é adicionada em A e os vértices são intercalados.

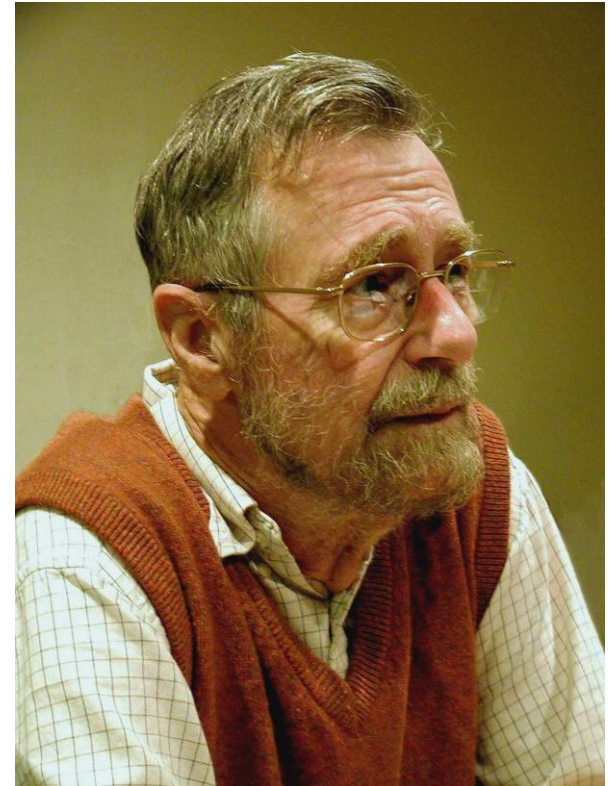
2

Algoritmo de Dijkstra

2 – Algoritmo de Dijkstra

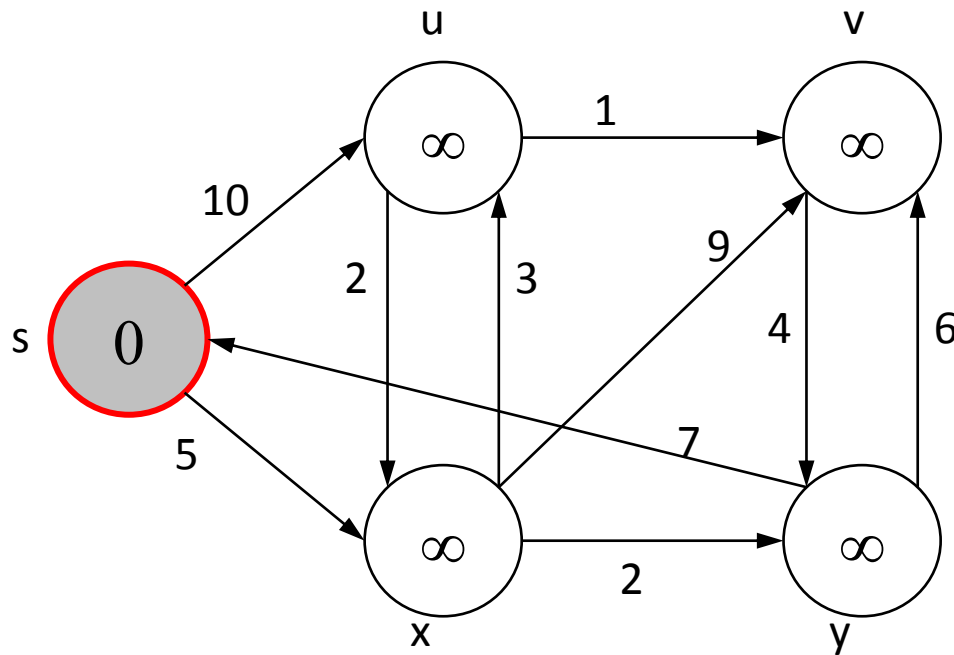
Características:

- O **algoritmo de Dijkstra**, concebido pelo cientista da computação holandês Edsger Dijkstra em 1956 e publicado em 1959.
- É um algoritmo de busca, guloso, que resolve o problema de caminhos mais curtos de única origem em um grafo orientado ponderado.
- As arestas não podem ter pesos negativos.



2 – Algoritmo de Dijkstra

Exemplo:

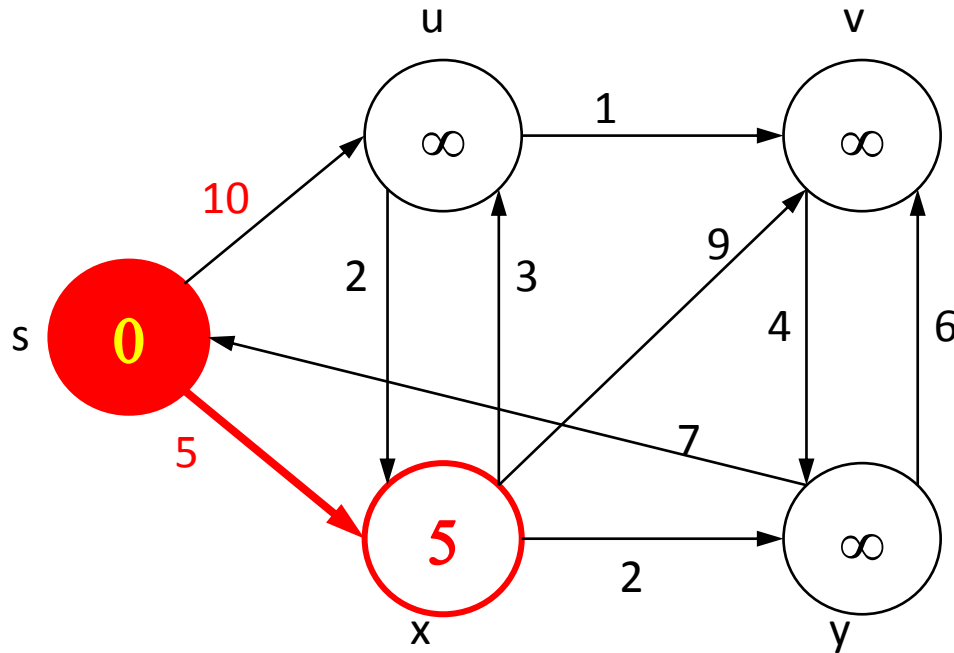


Inicialmente todos os vértices têm um custo infinito, exceto **s** (a raiz da busca) que tem valor 0:

Vértices	s	u	v	x	y
Estimativas	0	∞	∞	∞	∞
Precedentes	-	-	-	-	-
Fechado	Não	-	-	-	-

2 – Algoritmo de Dijkstra

Exemplo



Selecione o vértice aberto de estimativa mínima: **s**

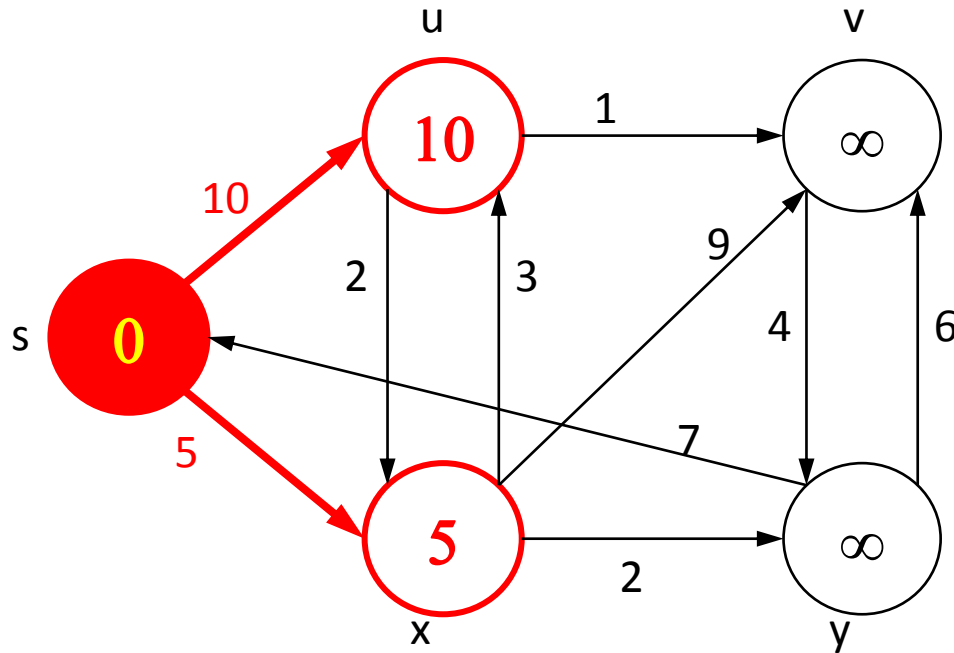
Feche **s**.

Recalcule as estimativas de **u** e **x**.

Vértices	s	u	v	x	y
Estimativas	0	∞	∞	5	∞
Precedentes	-	-	-	s	-
Fechado	Sim	-	-	Não	-

2 – Algoritmo de Dijkstra

Exemplo:



Selecione o vértice aberto de estimativa mínima: **s**

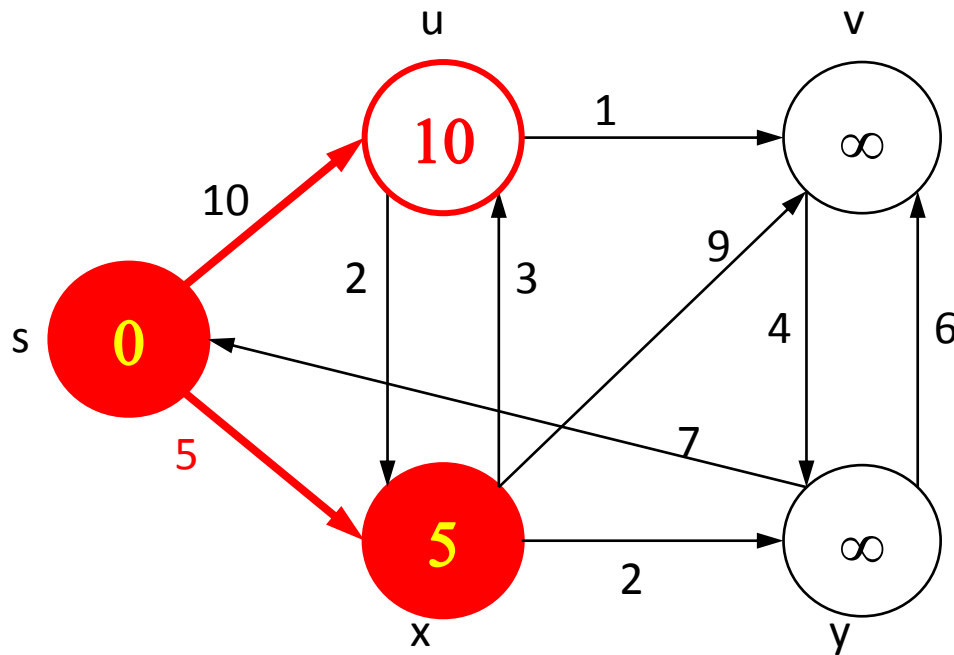
Feche **s**.

Recalcule as estimativas de **u** e **x**.

Vértices	s	u	v	x	y
Estimativas	0	10	∞	5	∞
Precedentes	-	s	-	s	-
Fechado	Sim	Não	-	Não	-

2 – Algoritmo de Dijkstra

Exemplo:



Selecione o vértice aberto de estimativa mínima: **x**

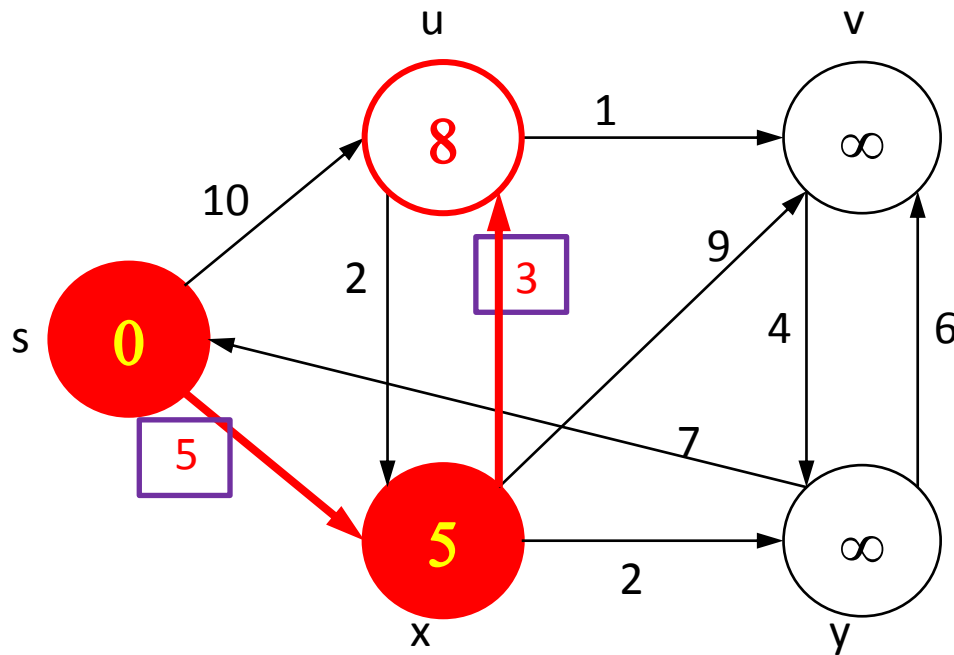
Feche **x**.

Recalcule as estimativas de **u**, **v** e **y**.

Vértices	s	u	v	x	y
Estimativas	0	10	∞	5	∞
Precedentes	-	s	-	s	-
Fechado	Sim	Não	-	Sim	-

2 – Algoritmo de Dijkstra

Exemplo:



Selecione o vértice aberto de estimativa mínima: **x**

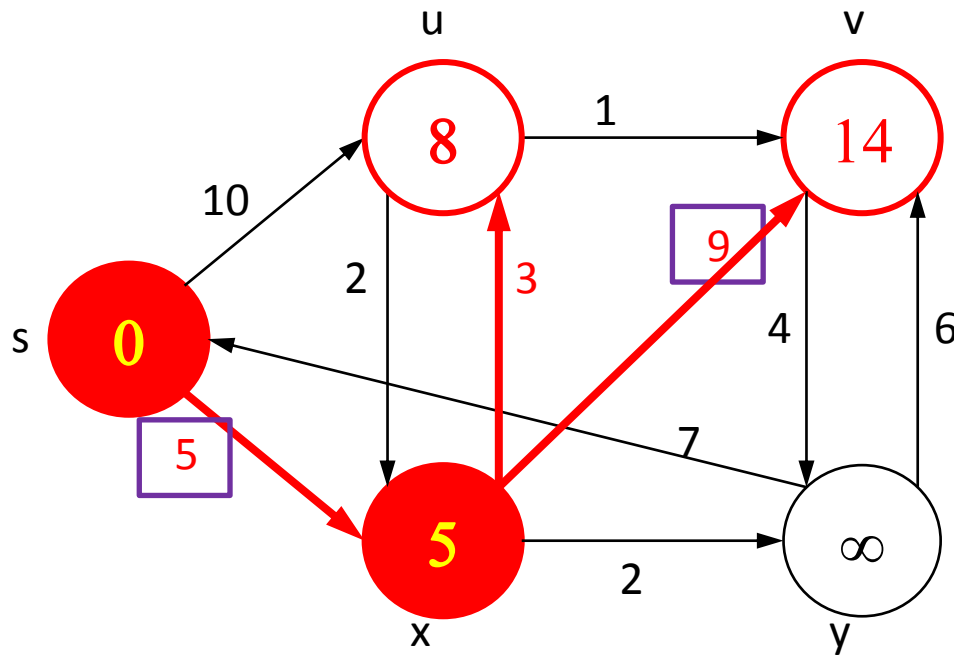
Feche **x**.

Recalcule as estimativas de **u**, **v** e **y**.

Vértices	s	u	v	x	y
Estimativas	0	10 8	∞	5	∞
Precedentes	-	x	-	s	-
Fechado	Sim	Não	-	Sim	-

2 – Algoritmo de Dijkstra

Exemplo:



Selecione o vértice aberto de
estimativa mínima: **x**

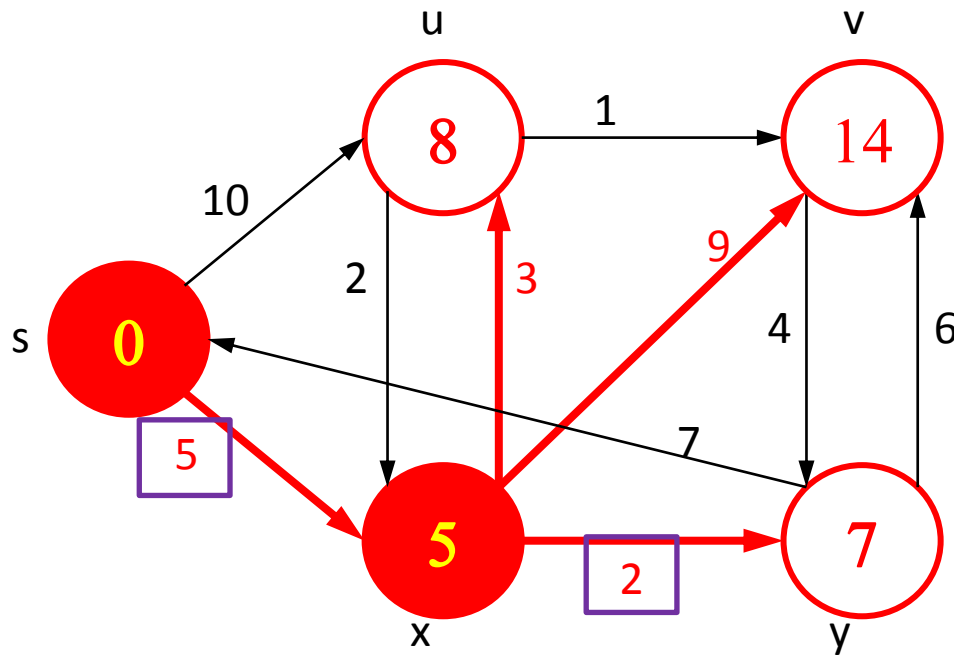
Feche **x**.

Recalcule as estimativas de **u**, **v** e **y**.

Vértices	s	u	v	x	y
Estimativas	0	10 8	14	5	∞
Precedentes	-	x	x	s	-
Fechado	Sim	Não	Não	Sim	-

2 – Algoritmo de Dijkstra

Exemplo:



Selecione o vértice aberto de estimativa mínima: **x**

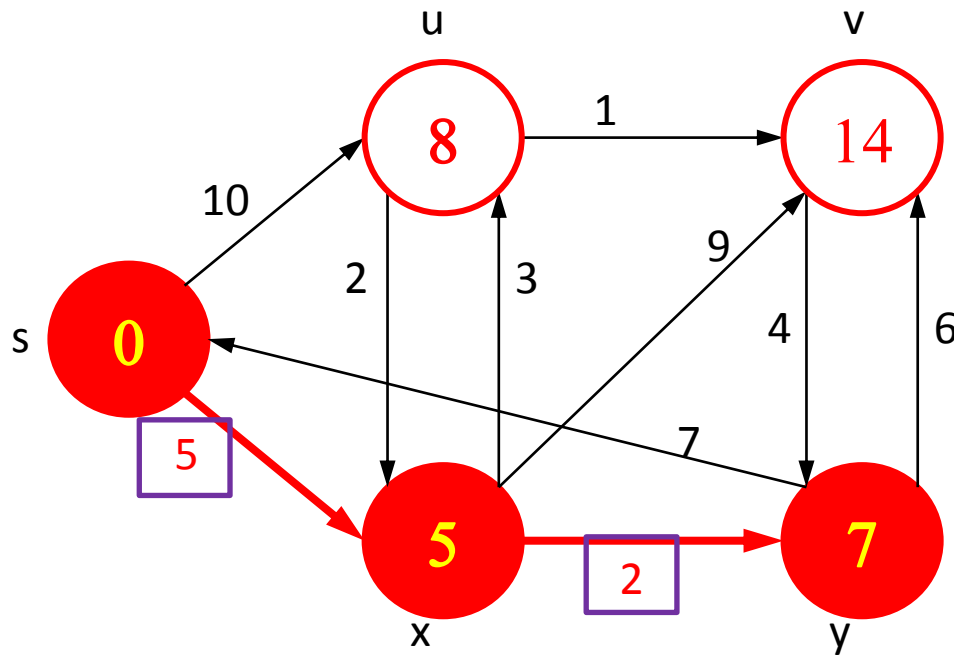
Feche **x**.

Recalcule as estimativas de **u**, **v** e **y**.

Vértices	s	u	v	x	y
Estimativas	0	10 8	14	5	7
Precedentes	-	x	x	s	x
Fechado	Sim	Não	Não	Sim	Não

2 – Algoritmo de Dijkstra

Exemplo:



Selecione o vértice aberto de estimativa mínima: **y**

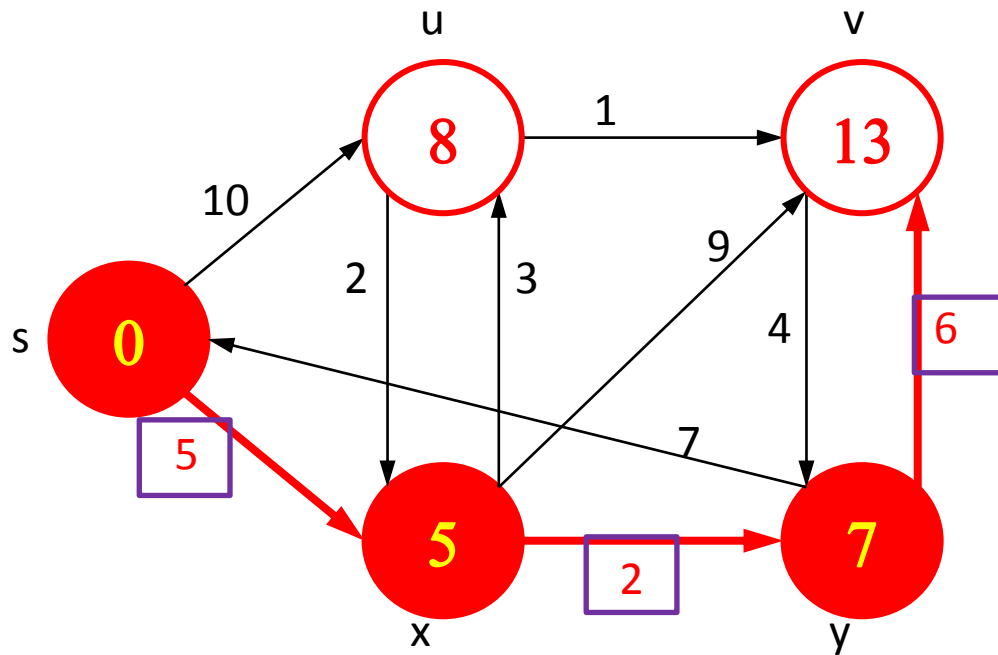
Feche **y**.

Recalcule as estimativas de **v**.

Vértices	s	u	v	x	y
Estimativas	0	10 8	14	5	7
Precedentes	-	x	x	s	x
Fechado	Sim	Não	Não	Sim	Sim

2 – Algoritmo de Dijkstra

Exemplo:



Selecione o vértice aberto de estimativa mínima: **y**

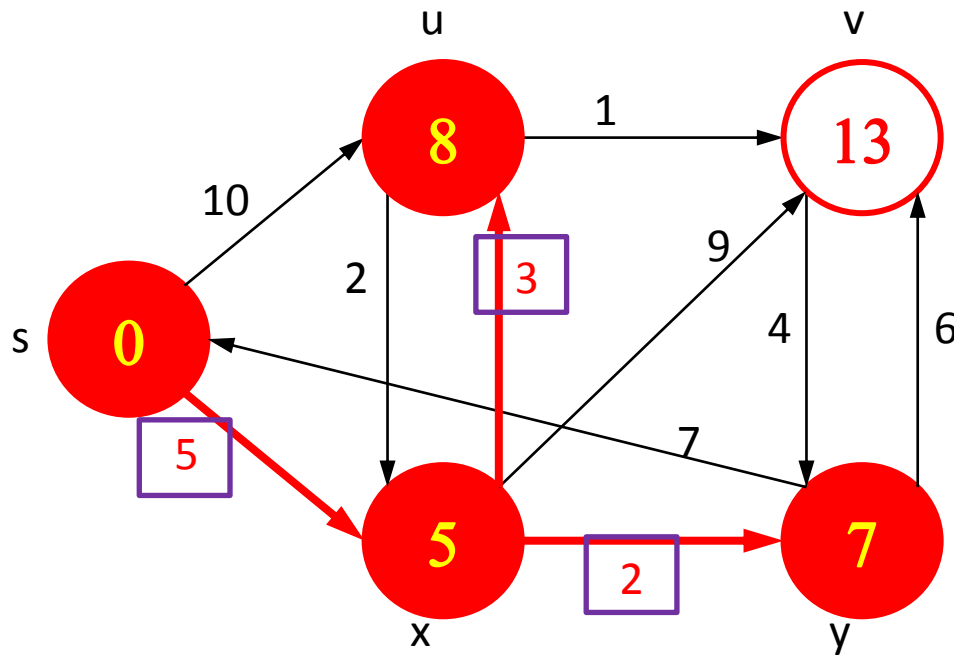
Feche **y**.

Recalcule as estimativas de **v**.

Vértices	s	u	v	x	y
Estimativas	0	10 8	14 13	5	7
Precedentes	-	x	y	s	x
Fechado	Sim	Não	Não	Sim	Sim

2 – Algoritmo de Dijkstra

Exemplo:



Selecione o vértice aberto de estimativa mínima: **u**

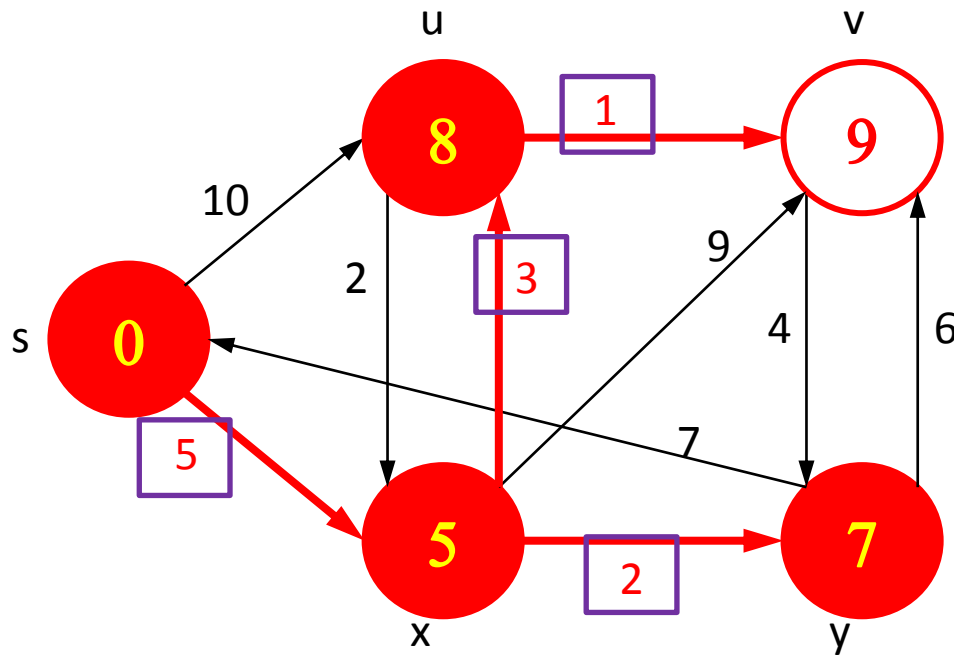
Feche **u**.

Recalcule as estimativas de **v**.

Vértices	s	u	v	x	y
Estimativas	0	8	14 13	5	7
Precedentes	-	x	y	s	x
Fechado	Sim	Sim	Não	Sim	Sim

2 – Algoritmo de Dijkstra

Exemplo:



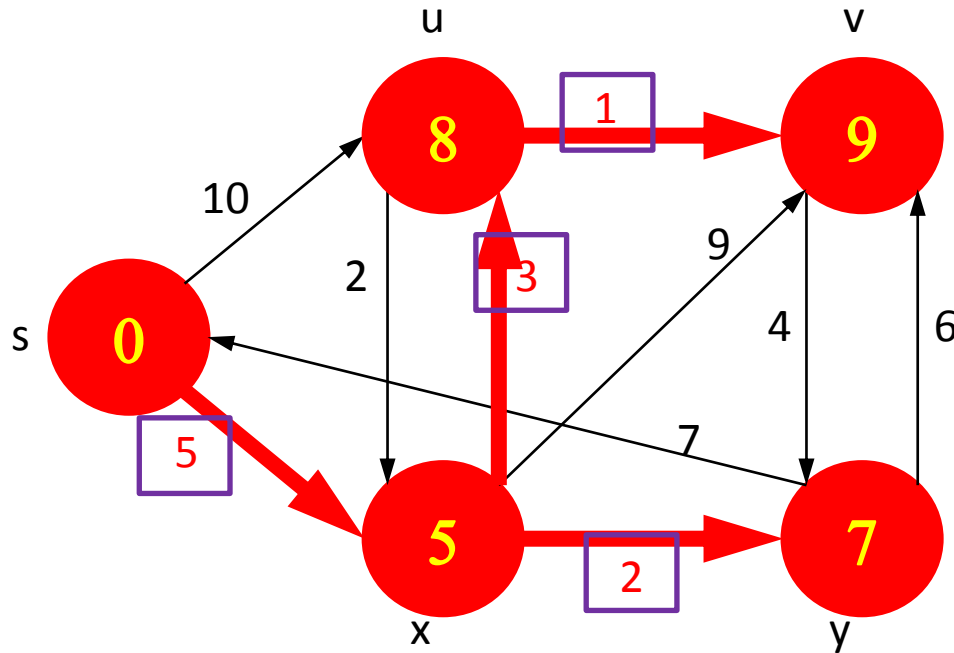
Selecione o vértice aberto de estimativa mínima: **v**

Feche **v**.

Vértices	s	u	v	x	y
Estimativas	0	8	13 9	5	7
Precedentes	-	x	u	s	x
Fechado	Sim	Sim	Não	Sim	Sim

2 – Algoritmo de Dijkstra

Exemplo:



Selecione o vértice aberto de estimativa mínima: **v**

Feche **v**.

Vértices	s	u	v	x	y
Estimativas	0	8	9	5	7
Precedentes	-	x	u	s	x
Fechado	Sim	Sim	Sim	Sim	Sim

2 – Algoritmo de Dijkstra

Algoritmo:

```
Dijkstra( $G, w, s$ )  
1 for cada vertex  $u \in V$   
2    $d(u) = \infty$   
3    $\text{pai}(u) = (\text{NIL})$   
4  $d(s) = 0$   
5  $S \leftarrow \emptyset$   
6  $Q = V[G]$   
7 while  $Q \neq \emptyset$   
8   do  $u \leftarrow \text{extractMin}(Q)$   
9      $S \leftarrow S \cup \{u\}$   
10    for cada  $v \in \text{adjacente}(u)$   
11      Relax( $u, v, w$ )
```

3

Algoritmo de Prim

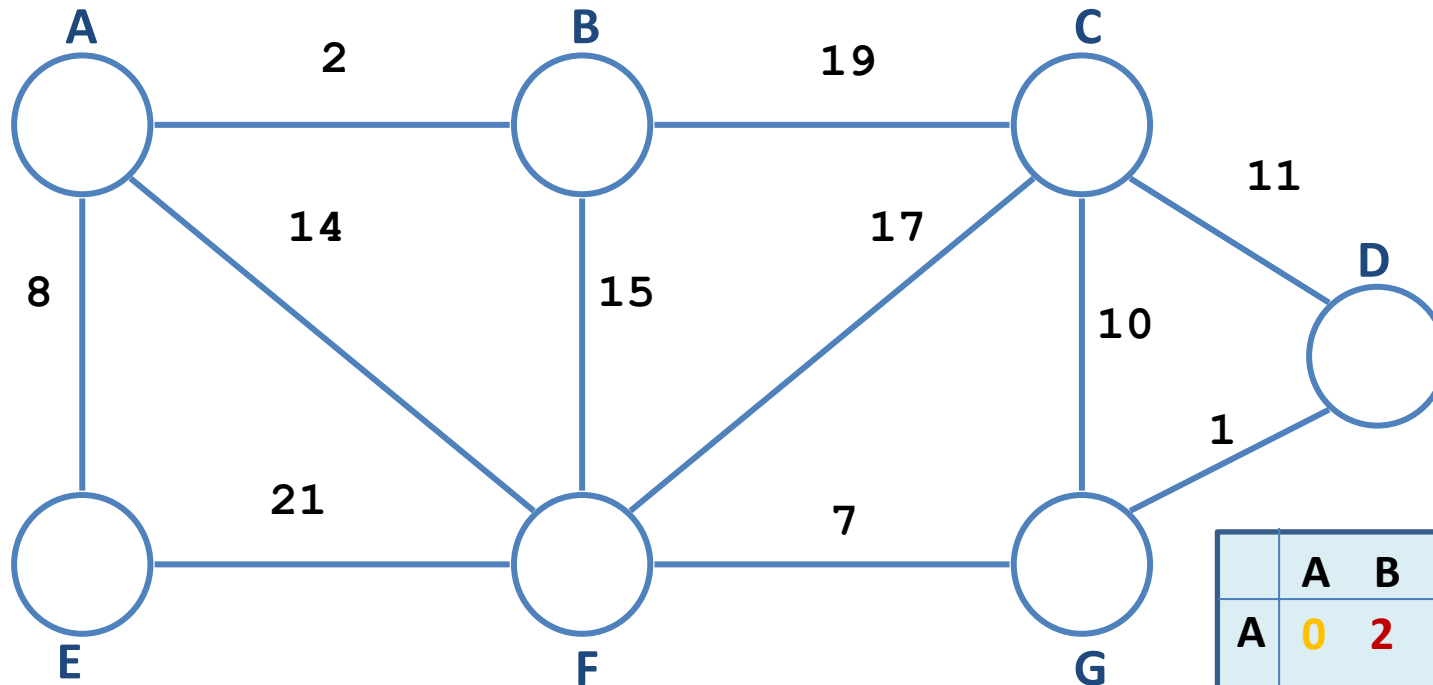
3 – Algoritmo de Prim

Características:

- O **algoritmo de Prim** é um algoritmo guloso, utilizado para encontrar uma árvore geradora mínima em um grafo conectado, ponderado, orientado ou não.
- Foi desenvolvido em 1930 pelo matemático Vojtěch Jarník e depois pelo cientista da computação Robert C. Prim em 1957 e redescoberto por Edsger Dijkstra em 1959.
- Deve-se determinar um vértice de origem.
- Fornecerá uma resposta que não necessariamente é a única.

3 – Algoritmo de Prim

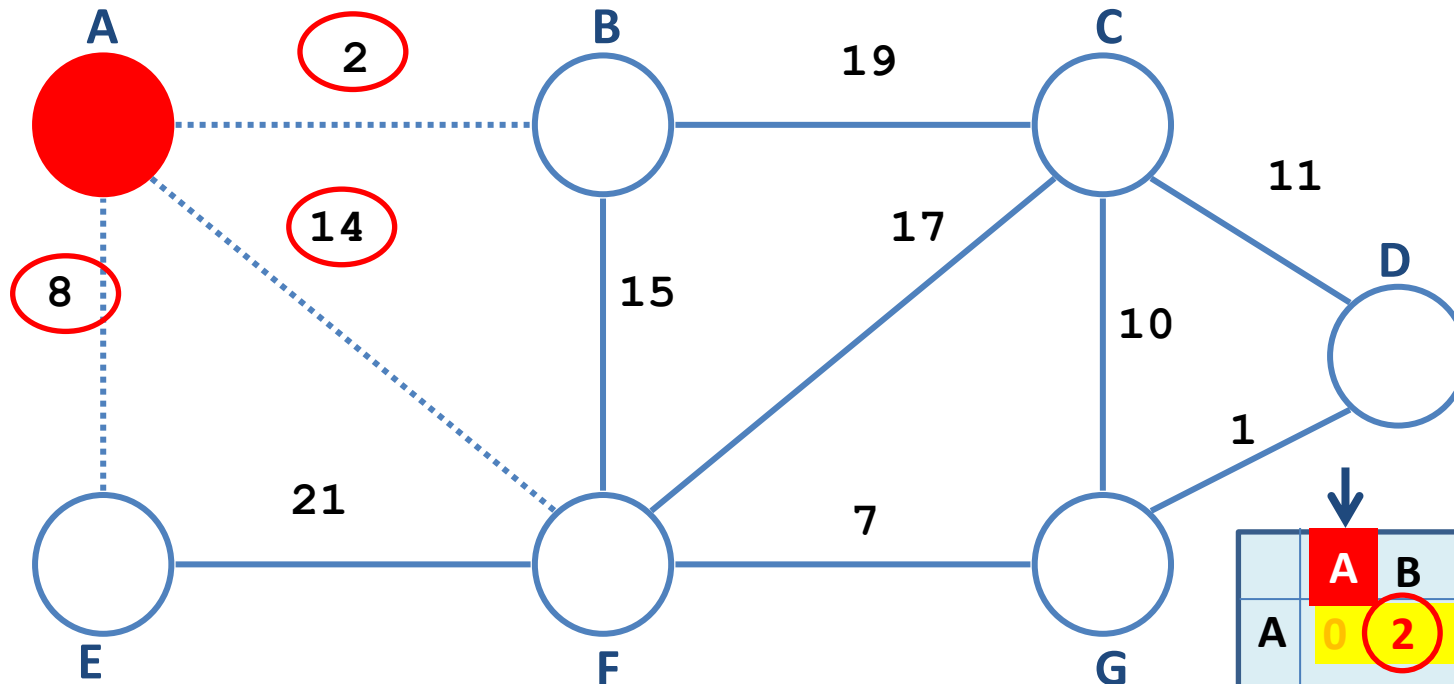
Exemplo: iniciando em A



	A	B	C	D	E	F	G
A	0	2	0	0	8	14	0
B		0	19	0	0	15	0
C			0	11	0	17	10
D				0	0	0	1
E					0	21	0
F						0	7
G							0

3 – Algoritmo de Prim

Exemplo: iniciando em A

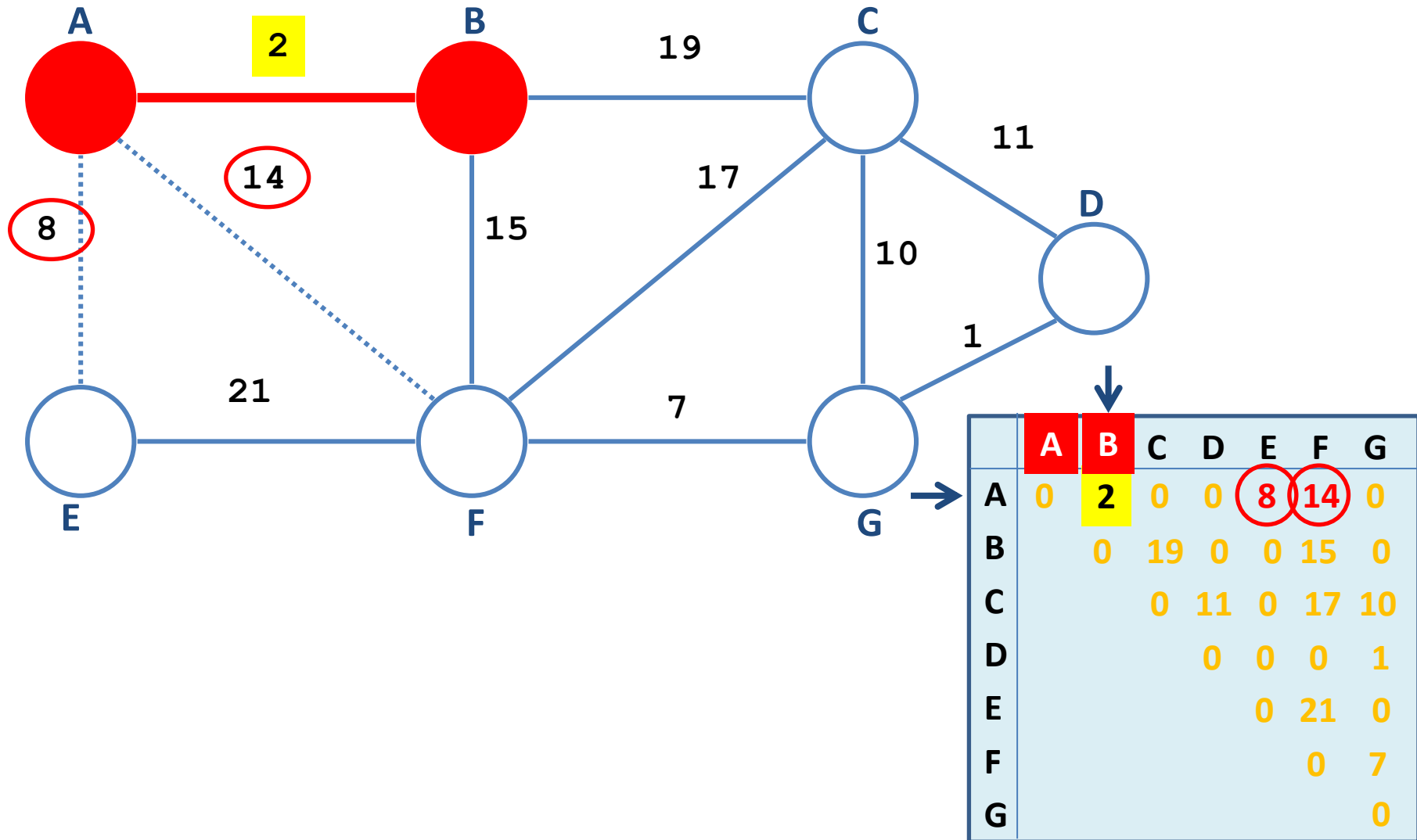


↓

	A	B	C	D	E	F	G
A	0	2	0	0	8	14	0
B		0	19	0	0	15	0
C			0	11	0	17	10
D				0	0	0	1
E					0	21	0
F						0	7
G							0

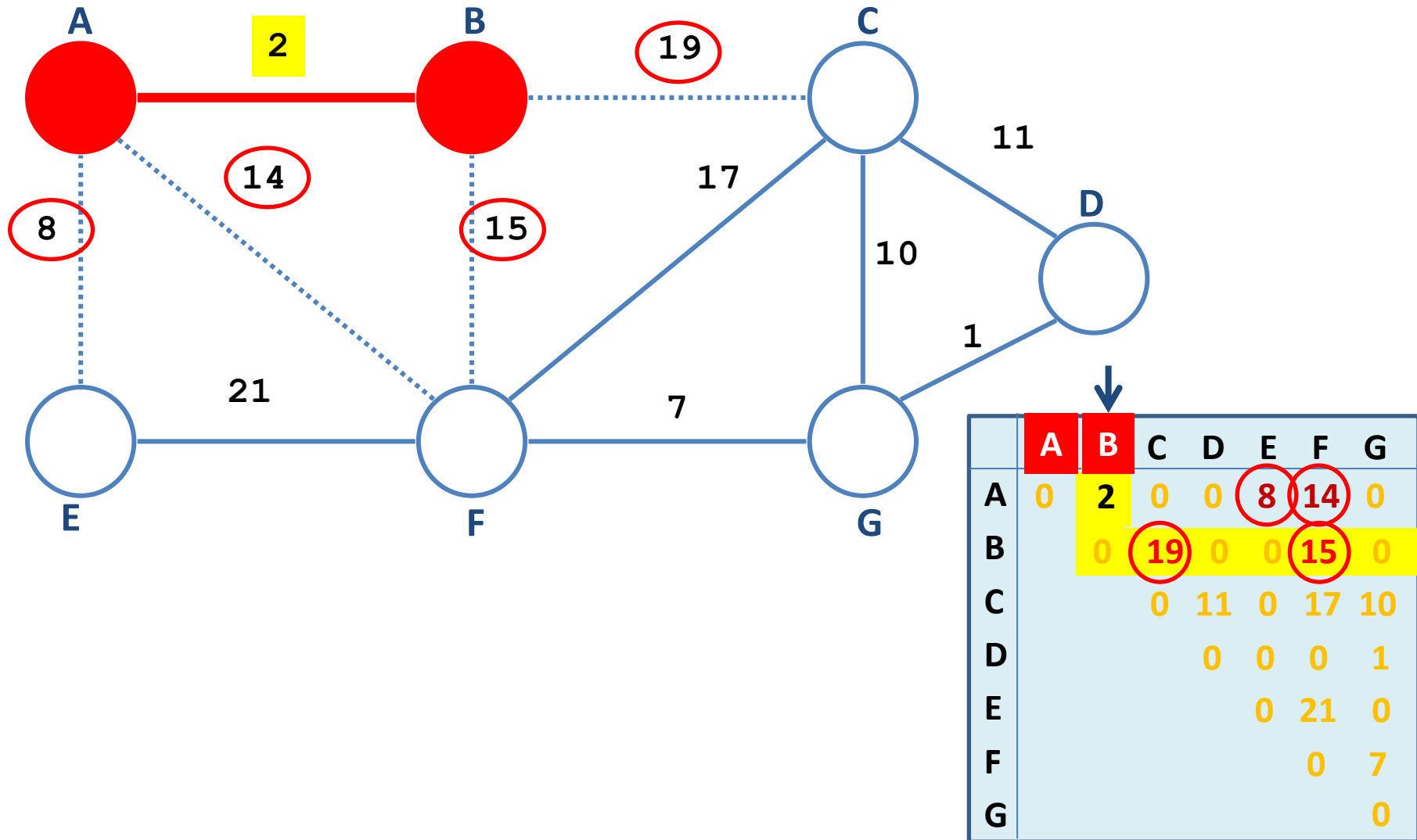
3 – Algoritmo de Prim

Exemplo: iniciando em A



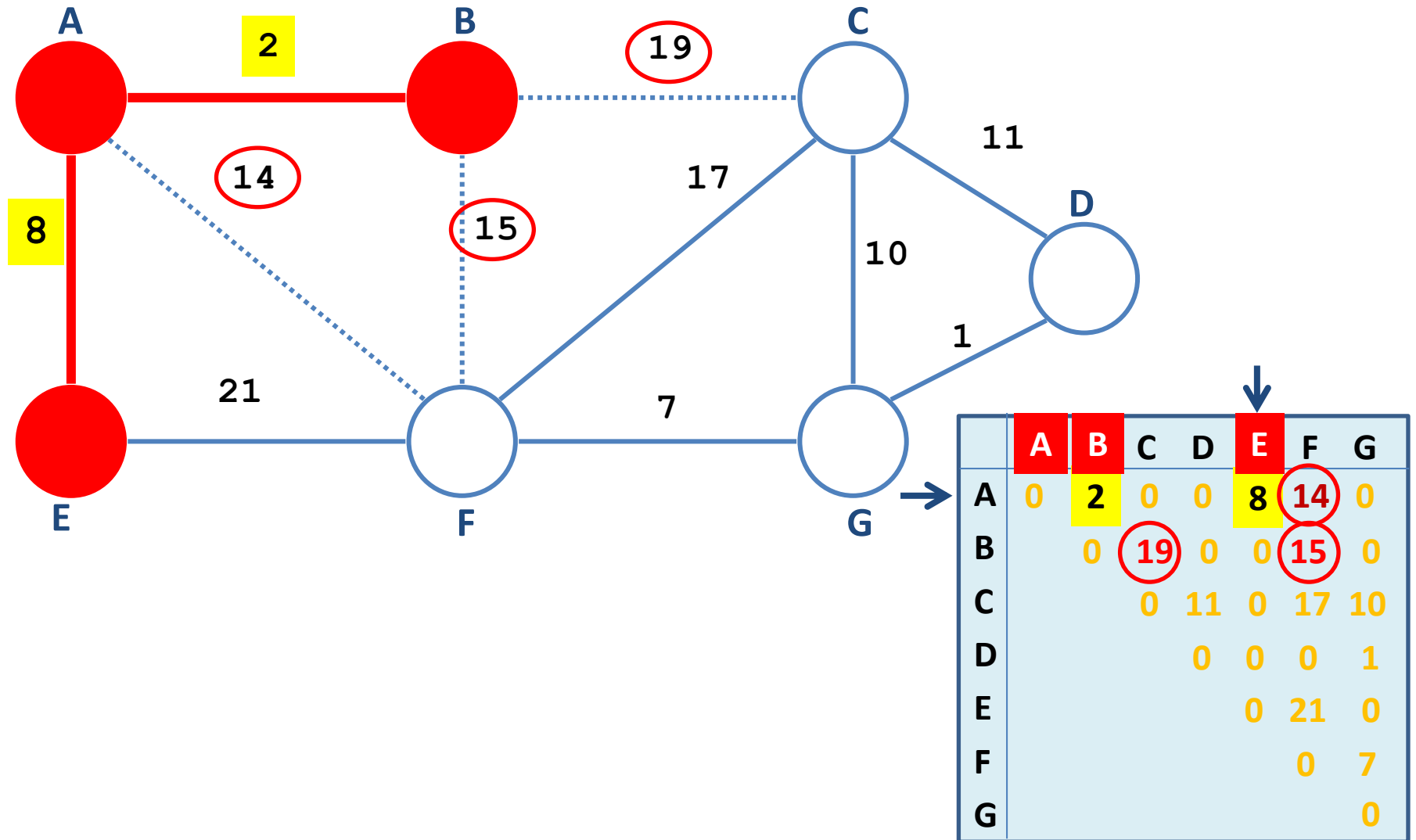
3 – Algoritmo de Prim

Exemplo: iniciando em A



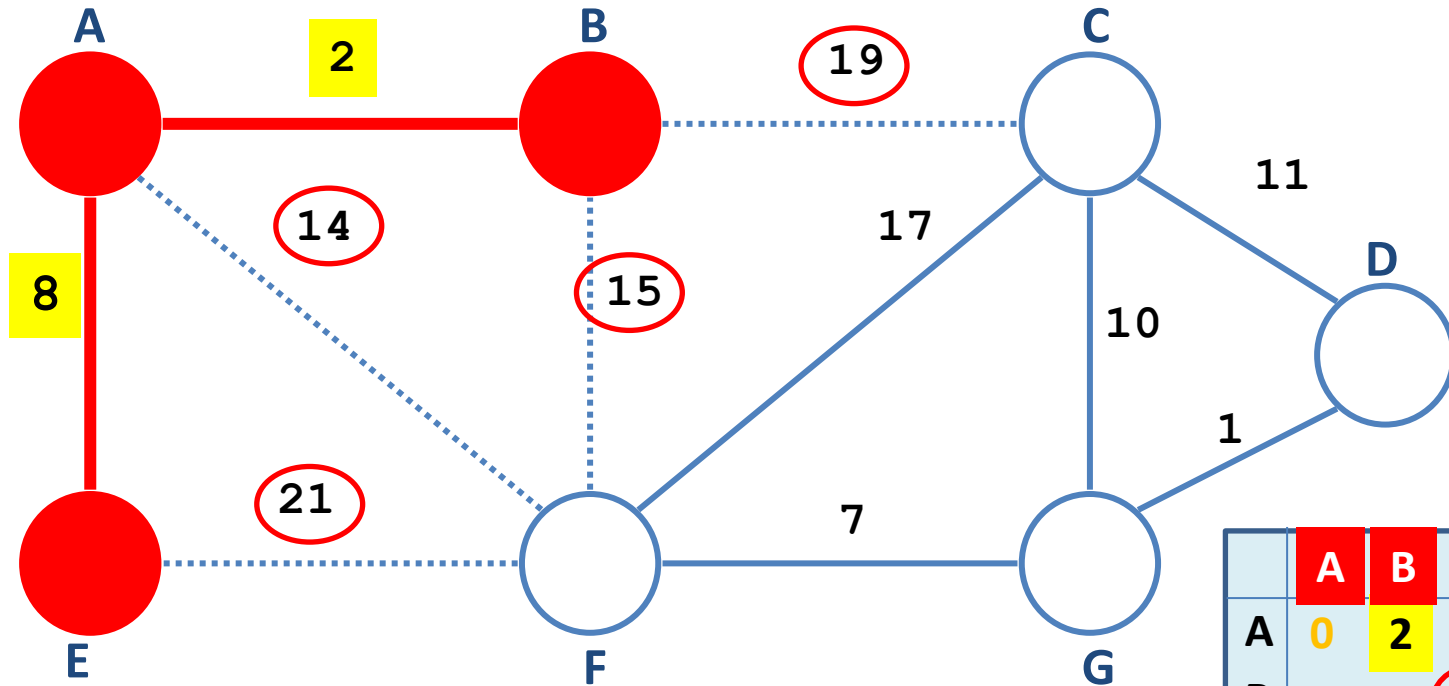
3 – Algoritmo de Prim

Exemplo: iniciando em A



3 – Algoritmo de Prim

Exemplo: iniciando em A

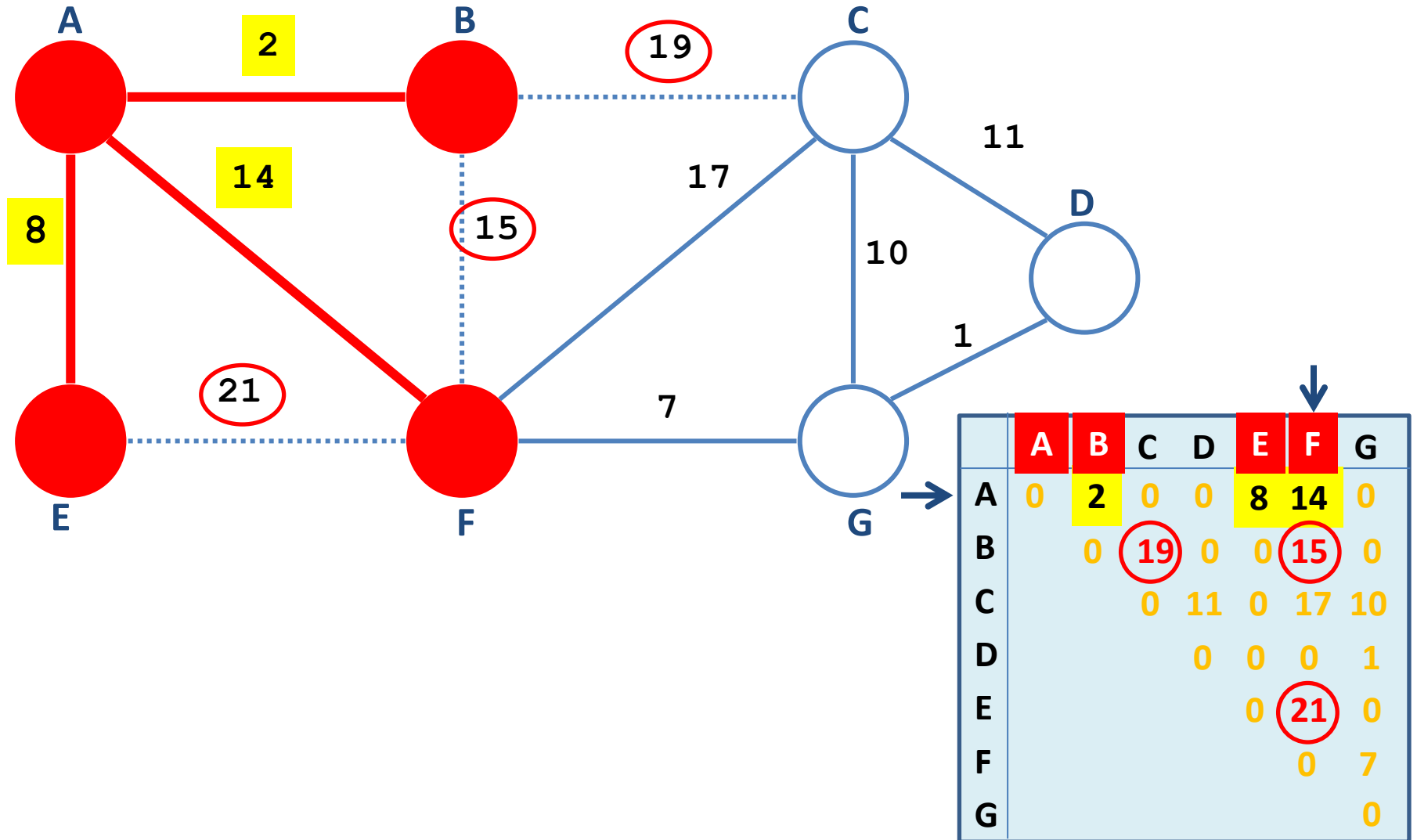


↓

	A	B	C	D	E	F	G
A	0	2	0	0	8	14	0
B		0	19	0	0	15	0
C			0	11	0	17	10
D				0	0	0	1
E					0	21	0
F						0	7
G							0

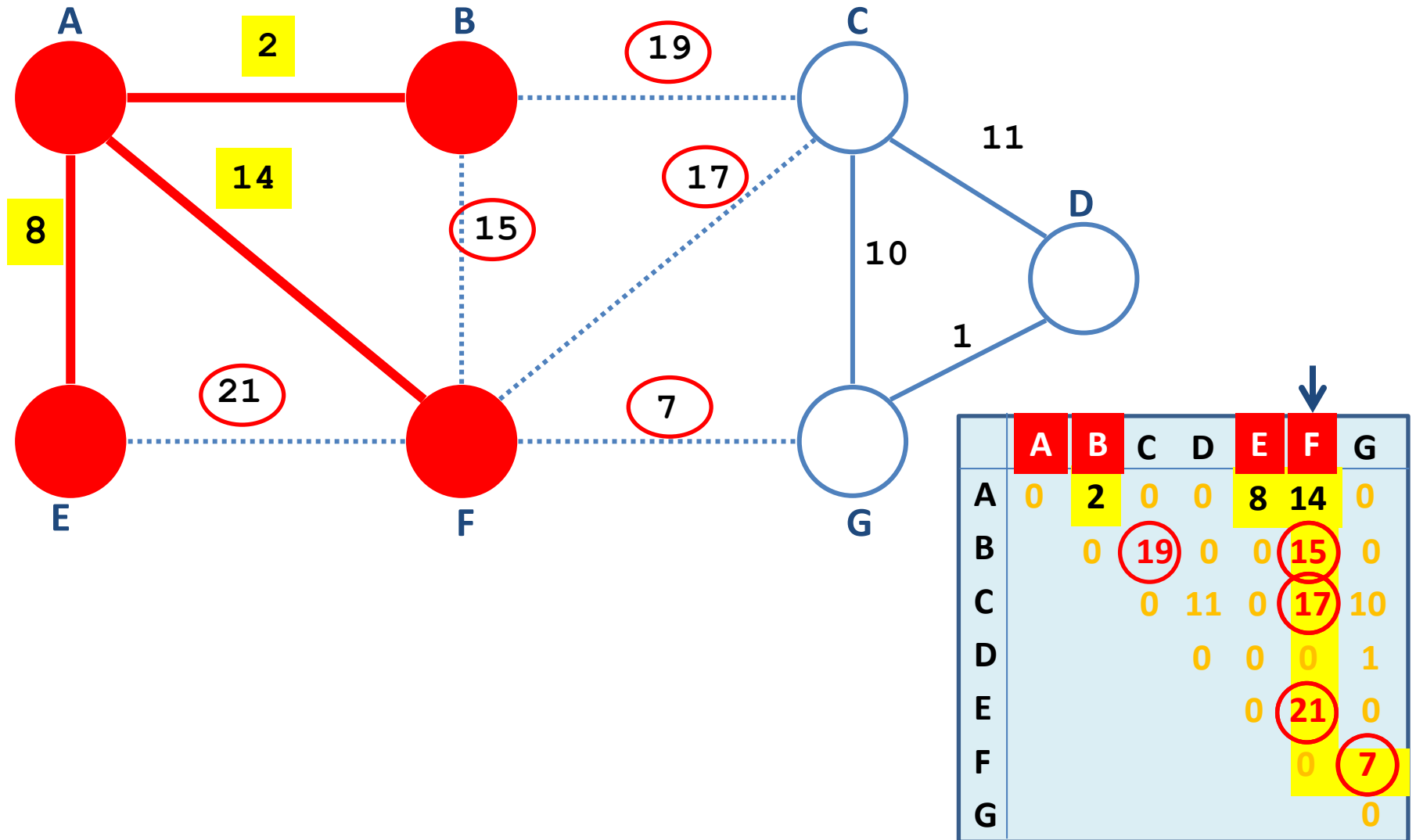
3 – Algoritmo de Prim

Exemplo: iniciando em A



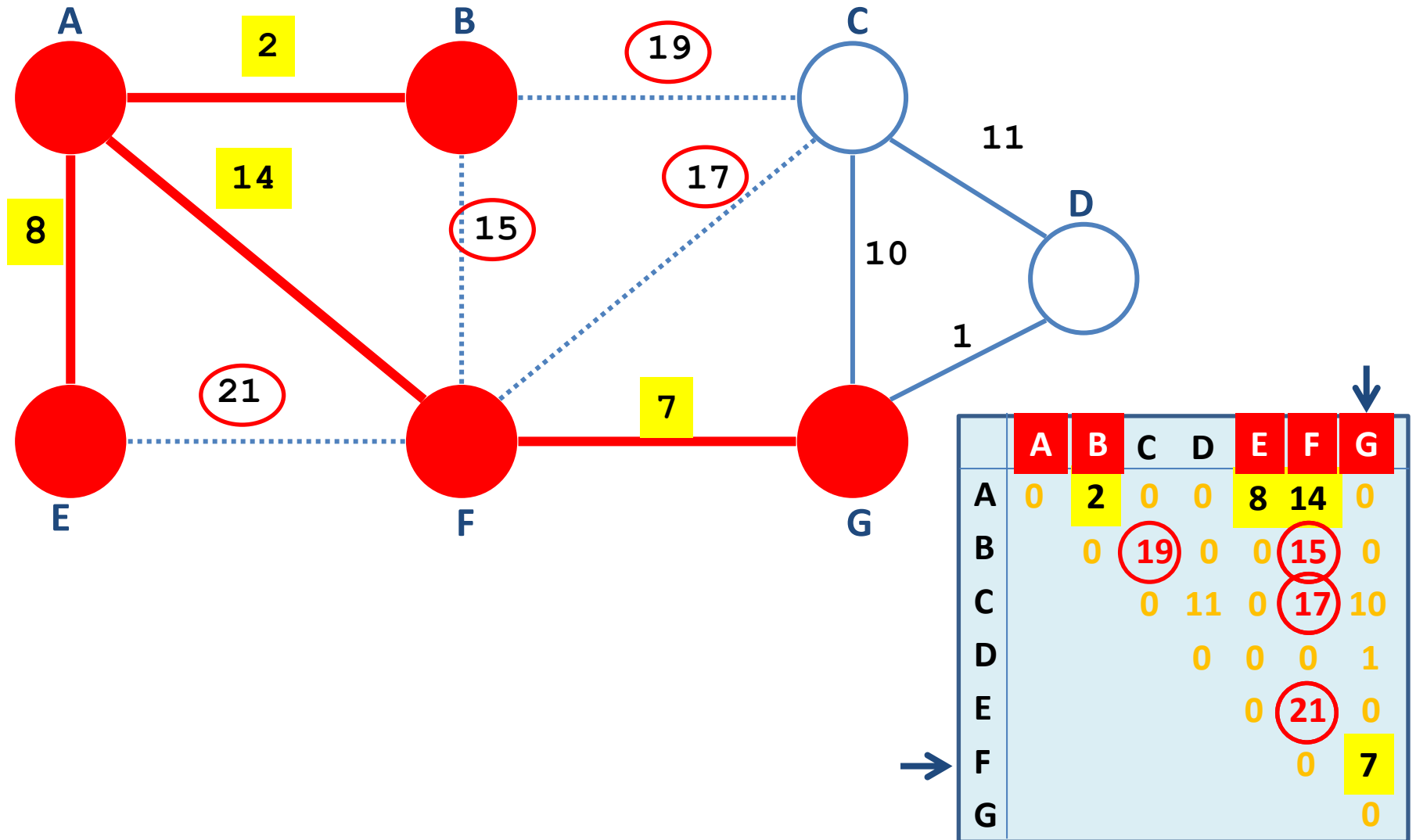
3 – Algoritmo de Prim

Exemplo: iniciando em A



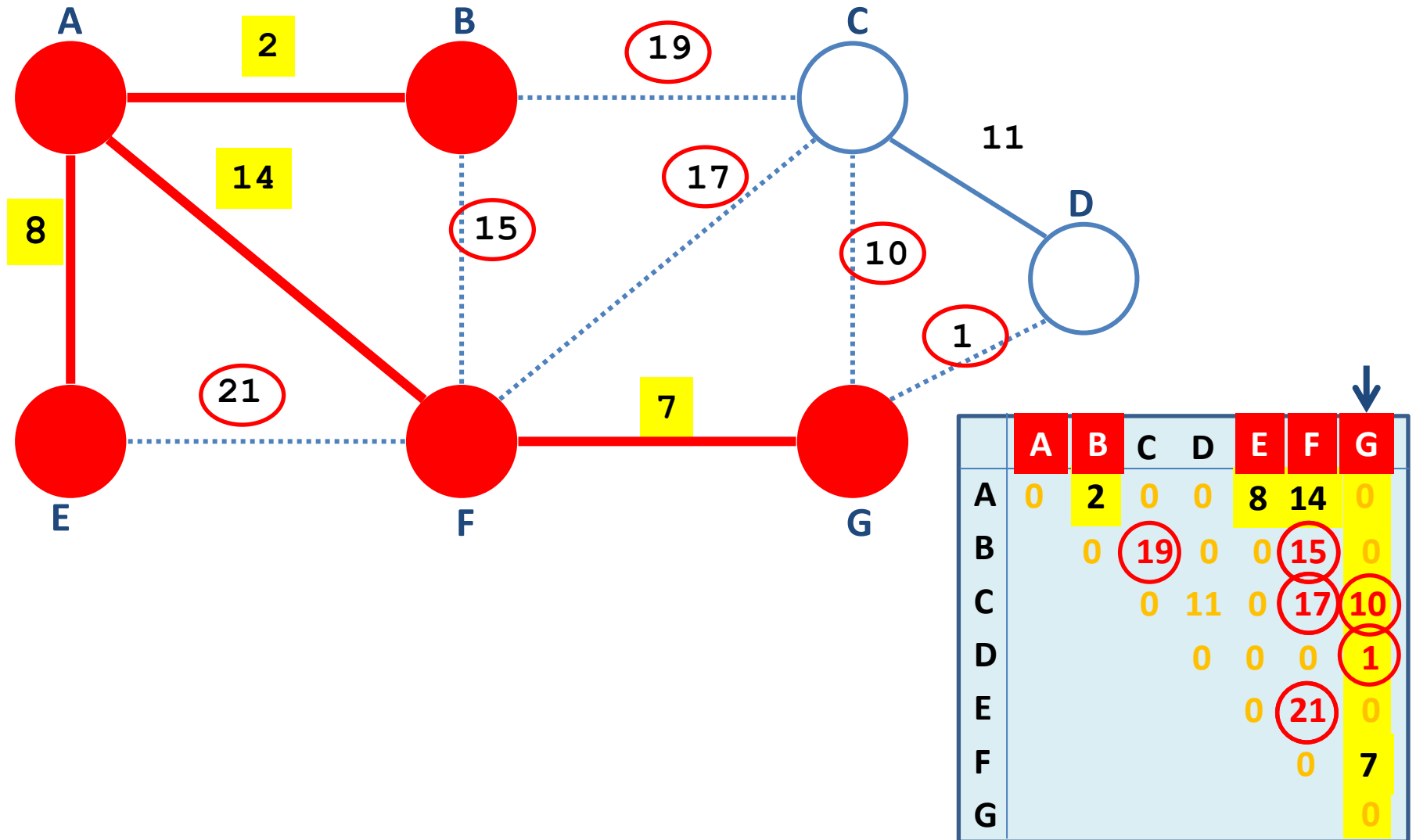
3 – Algoritmo de Prim

Exemplo: iniciando em A



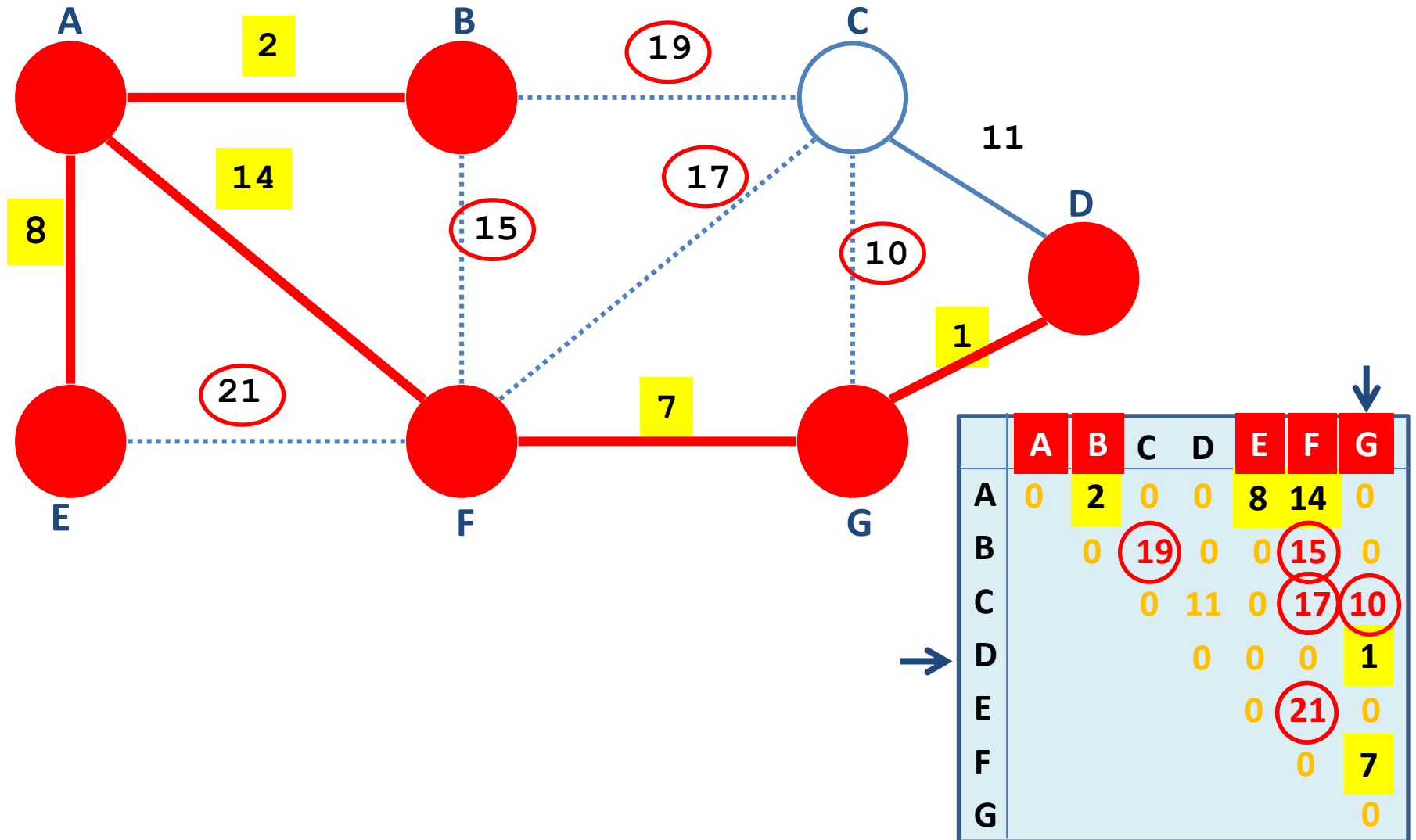
3 – Algoritmo de Prim

Exemplo: iniciando em A



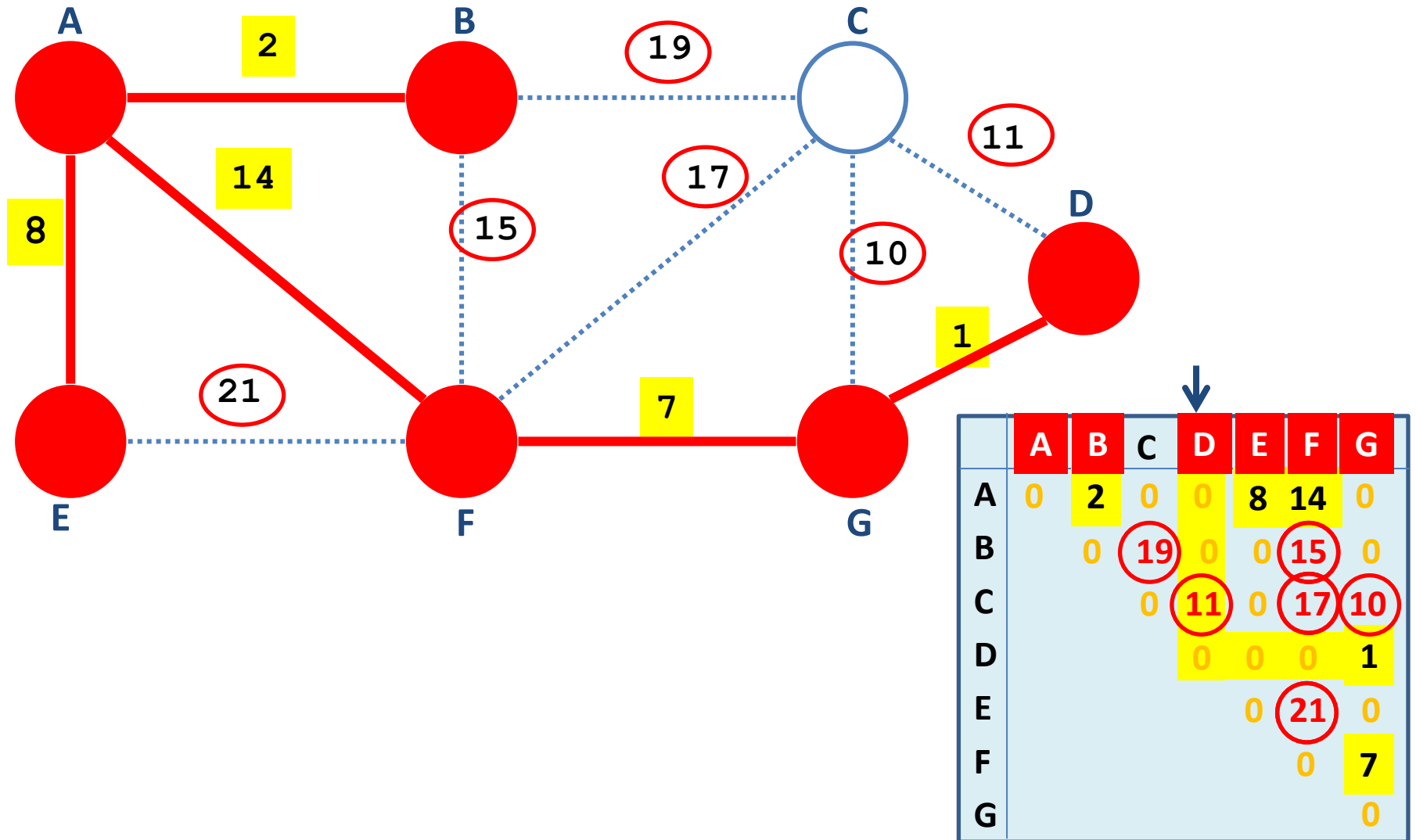
3 – Algoritmo de Prim

Exemplo: iniciando em A



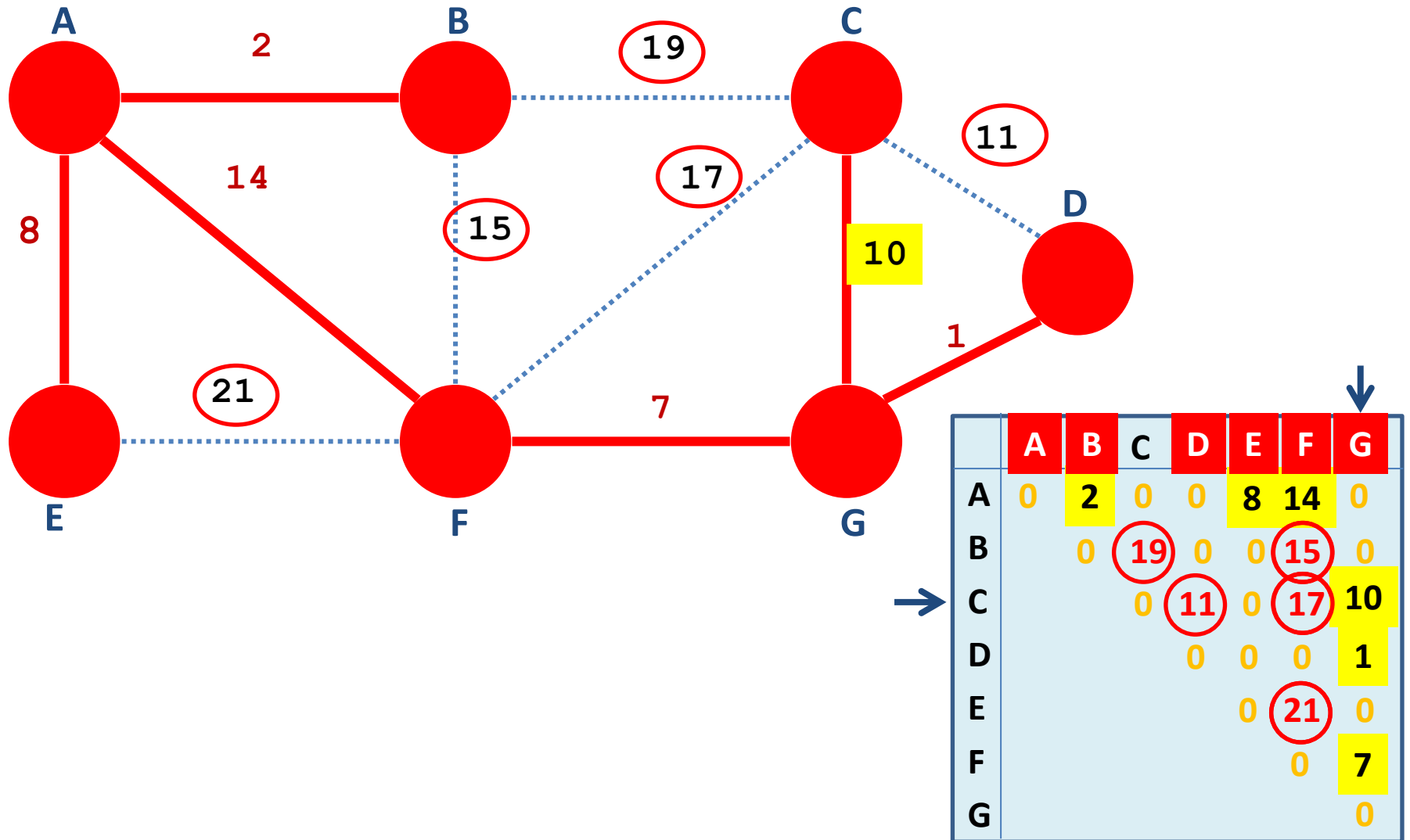
3 – Algoritmo de Prim

Exemplo: iniciando em A



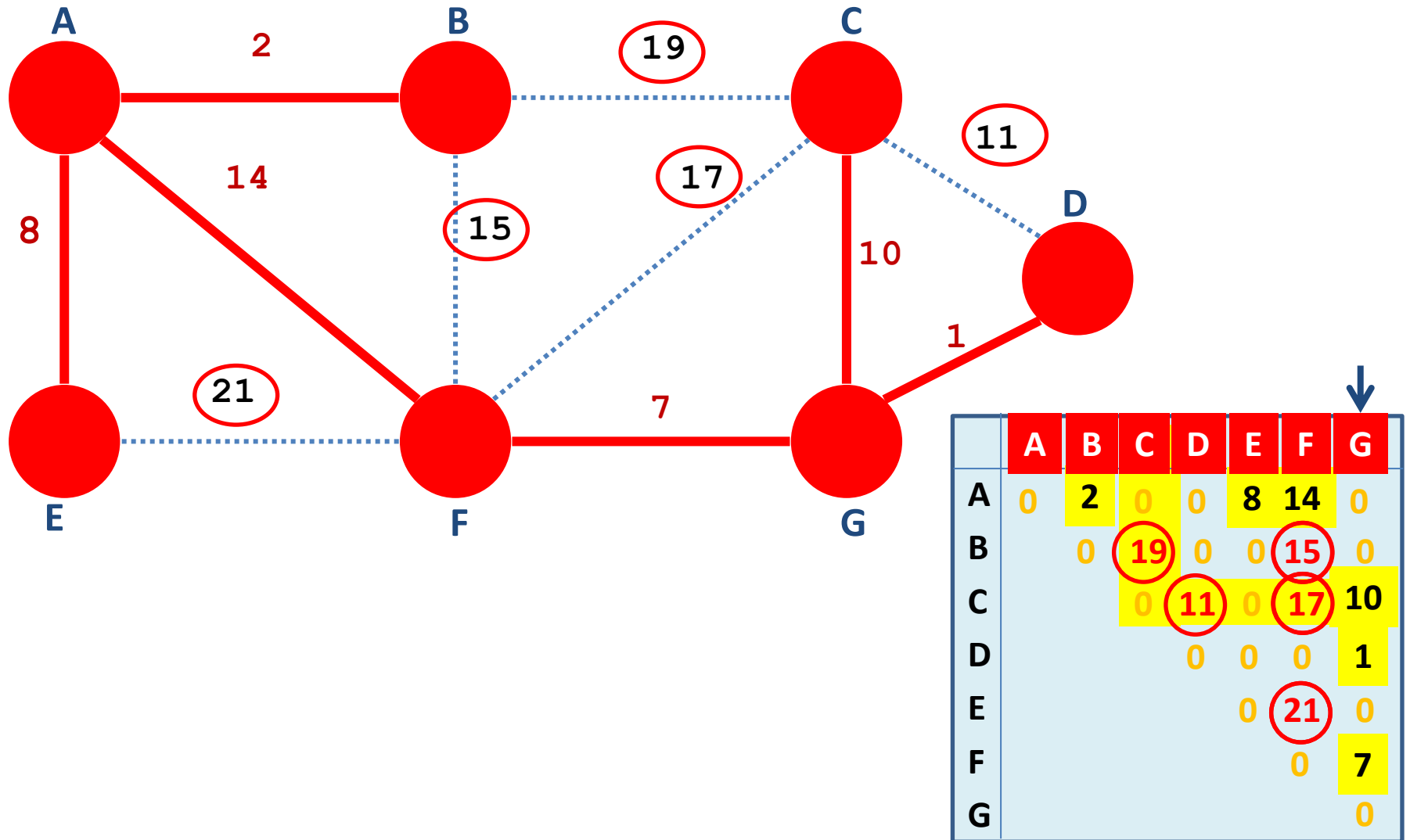
3 – Algoritmo de Prim

Exemplo: iniciando em A



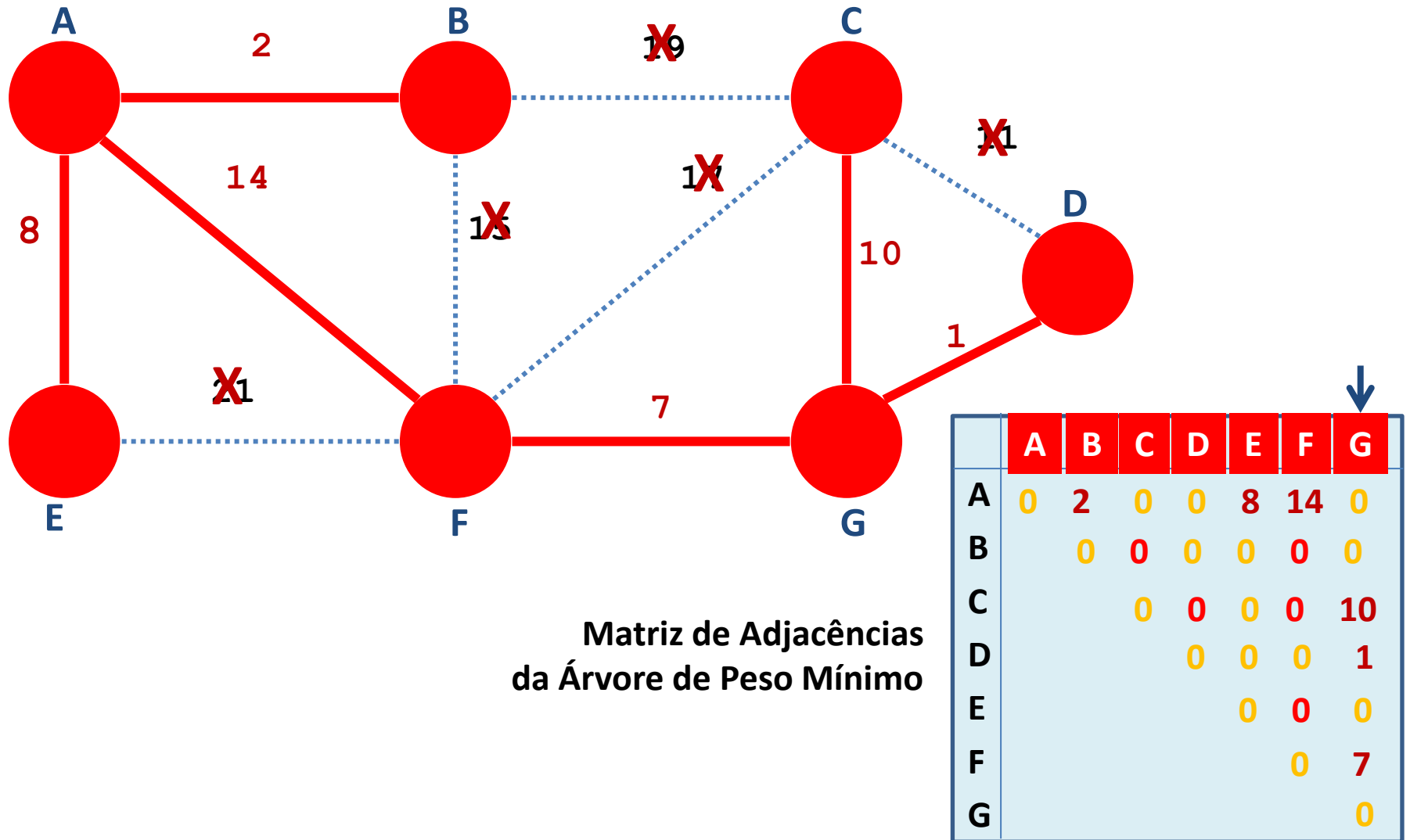
3 – Algoritmo de Prim

Exemplo: iniciando em A



3 – Algoritmo de Prim

Exemplo: iniciando em A



3 – Algoritmo de Prim

Algoritmo:

```
PRIM( $G, w, r$ )
1 for cada  $u$  em  $V[G]$  do
2      $chave[u] \leftarrow \infty$ 
3      $pai[u] \leftarrow NIL$ 
4  $chave[r] \leftarrow 0$ 
5  $Q \leftarrow V[G]$ 
6 while  $Q \neq \{\}$  do
7      $u \leftarrow EXTRACT-MIN(Q)$ 
8     for cada  $v$  em  $Adj[u]$  do
9         if  $v \in Q$  e  $w(u, v) < chave[v]$ 
10             then  $pai[v] \leftarrow u$ 
11              $chave[v] \leftarrow w(u, v)$ 
```

Bibliografias

- i. CORMEN, LEISERSON, RIVEST, STEIN, **Algoritmos: Teoria e Prática**. 2ª ed. Rio de Janeiro: Campus, 2002: Capítulo 23.
- ii. TONIDANDEL, Flávio, Slides das aulas de Algoritmos Computacionais. FEI, 2011.
- iii. <http://www.inf.ufsc.br/grafos/temas/custo-minimo/dijkstra.html>
- iv. http://www.ime.usp.br/~pf/algoritmos_para_grafos/aulas/dijkstra.html
- v. <http://www.lcad.icmc.usp.br/~nonato/ED/Dijkstra/node84.html>
- vi. http://en.wikipedia.org/wiki/Prim%27s_algorithm
- vii. <http://www.ic.unicamp.br/~meidanis/courses/mo417/2003s1/aulas/2003-05-16.html>
- viii. <https://sites.google.com/site/tecprojalgoritmos/problemas/prim-e-kruskal>