Prática 7

21.3 - 1

for $i = 1$ to 16
  MakeSet(x_i)] I

for $i = 1$ to 15 by 2
  Union(x_i, x_{i+1})] II

for $i = 1$ to 13 by 4
  Union(x_i, x_{i+2})] III

Union(x_1, x_5)
Union(x_{11}, x_{13}) } IV

Union(x_1, x_{10}) V

FindSet(x_2) VI

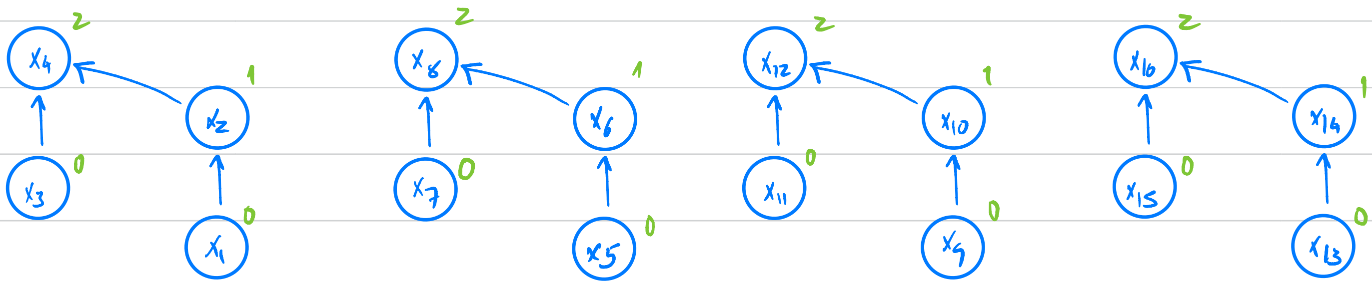
FindSet(x_9) VII

I

II

III

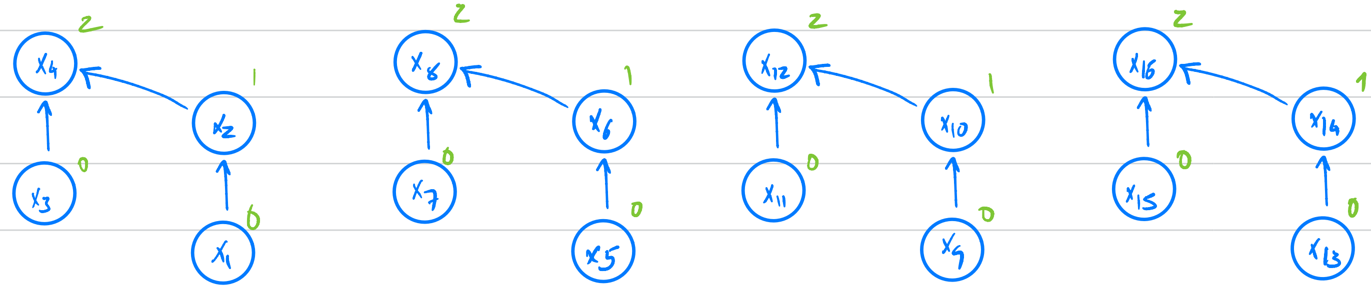
Union(x_1, x_3)
Union(x_5, x_7)
⋮
Union(x_{13}, x_{15})



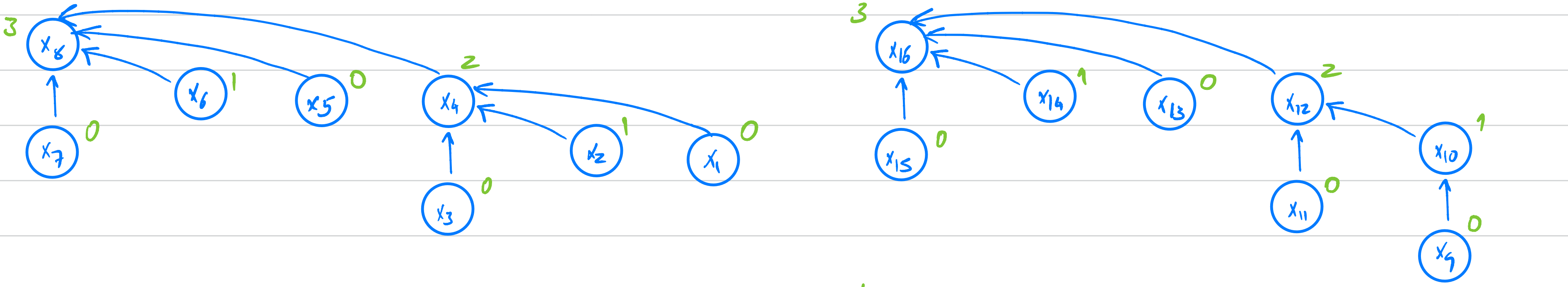
21.3 - 1

Union (x_1, x_5)
Union (x_{11}, x_{13}) } IV
Union (x_9, x_{10}) V
FindSet (x_2) VI
FindSet (x_9) VII

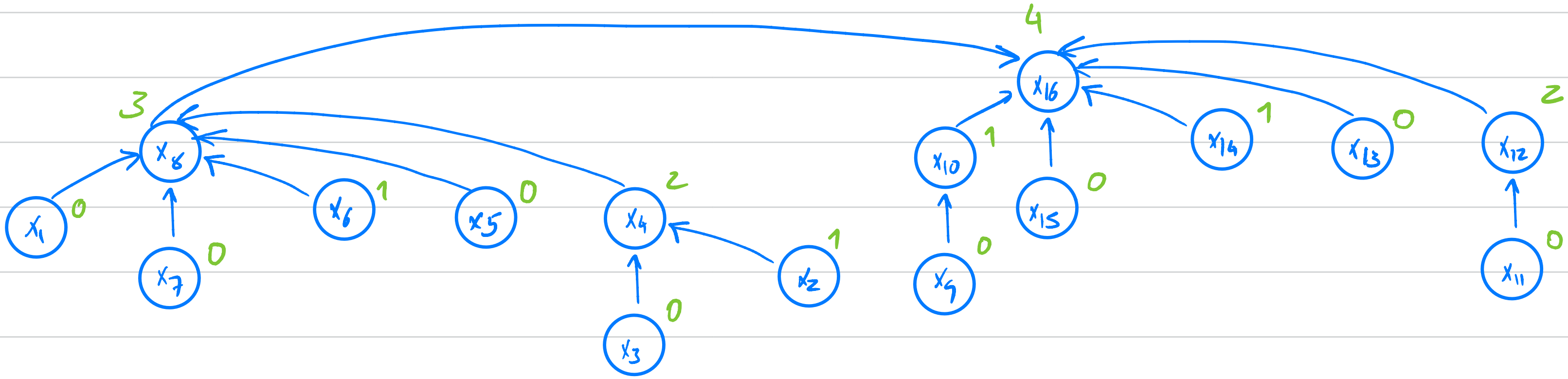
III



IV



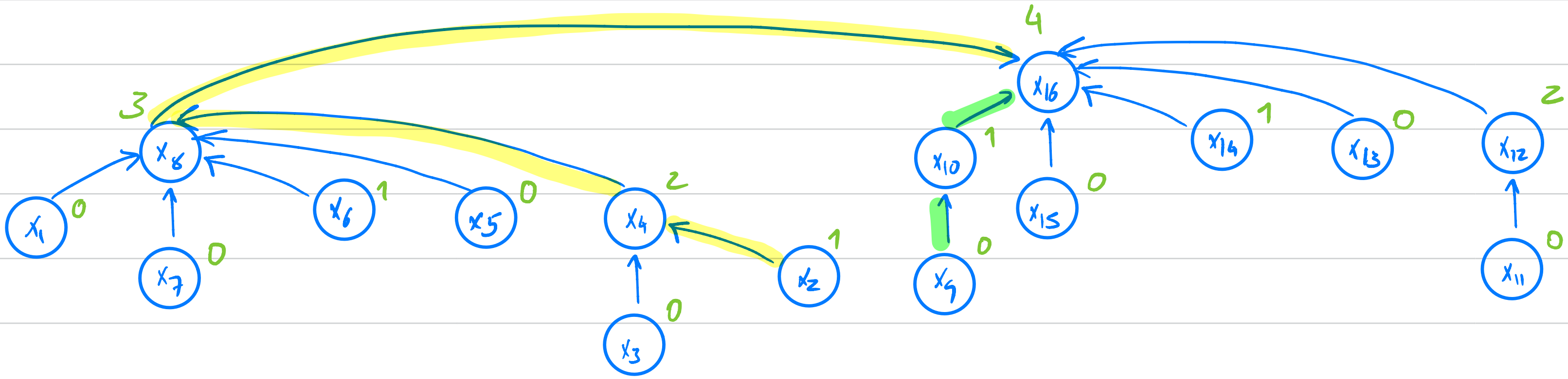
V



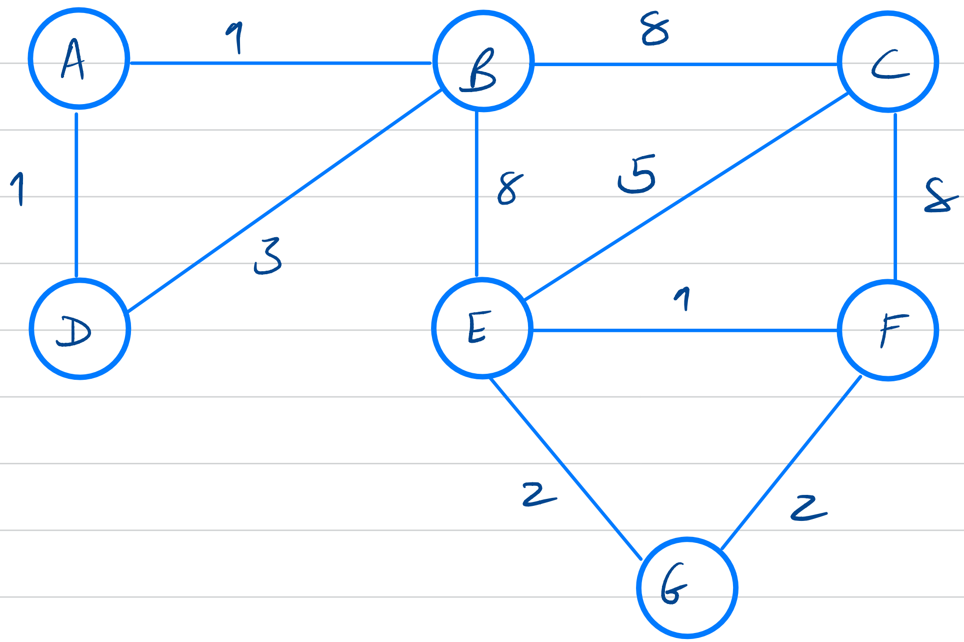
21.3 - 1

FindSet (x_2) VI

FindSet (x_9) VII

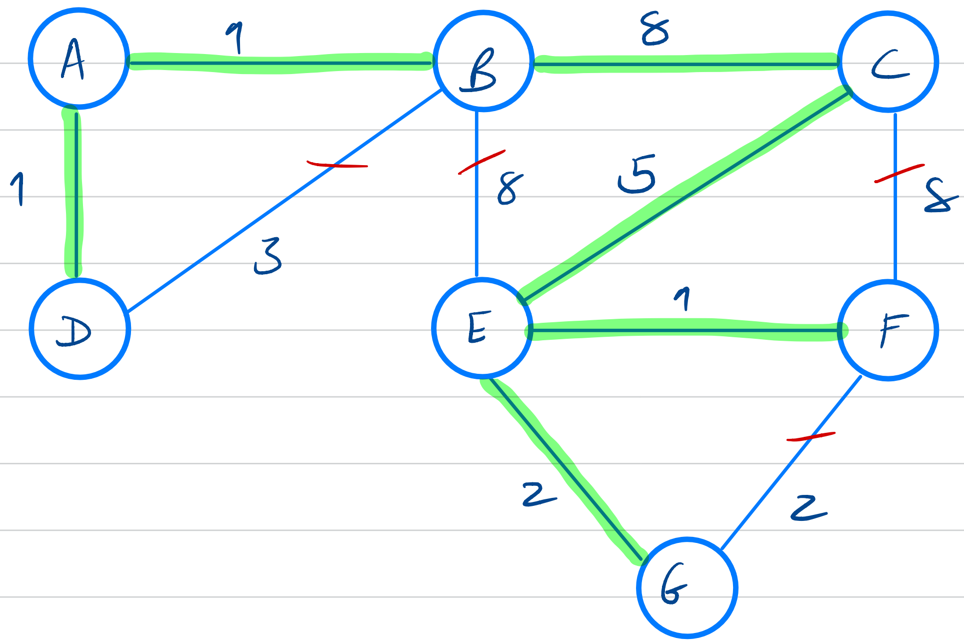


TI 06/07 I.3 -



- Kruskal
- Prim

TI 06/07 I.3 -

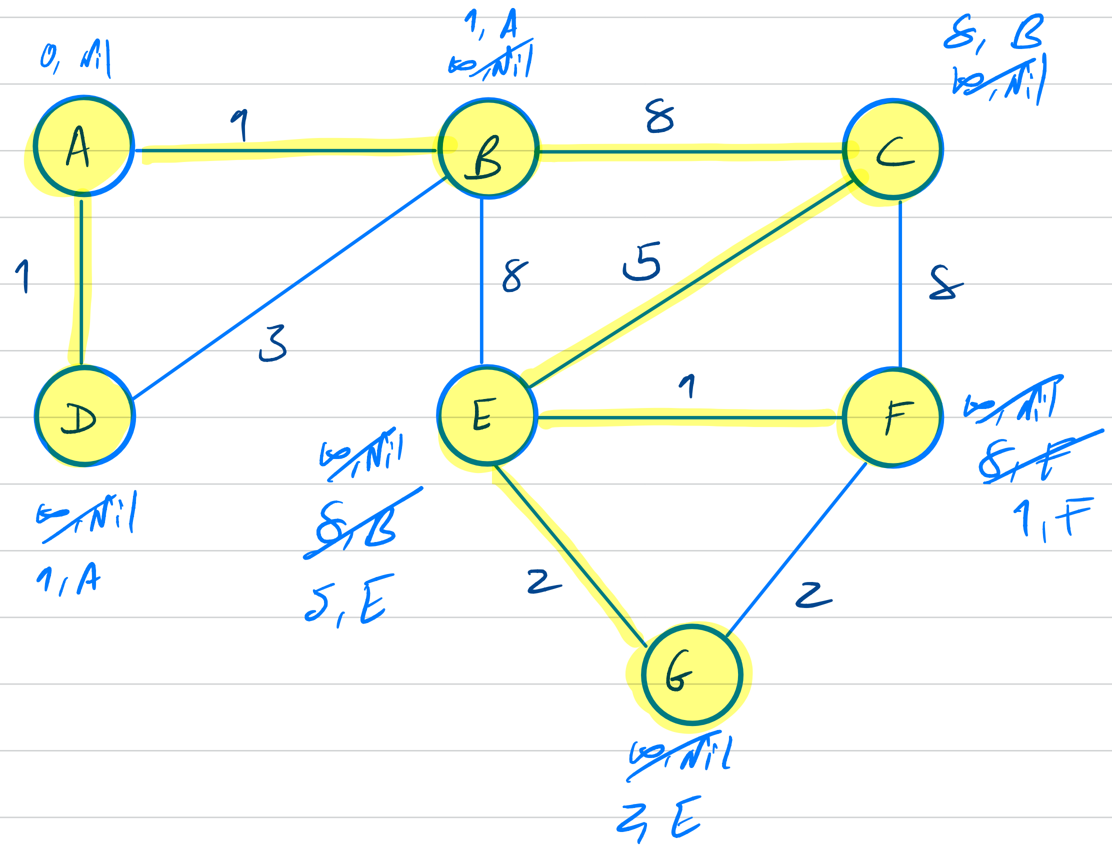


- Kruskal
- Prim

$W(T) = 1 + 1 + 1 + 2 + 5 + 8$

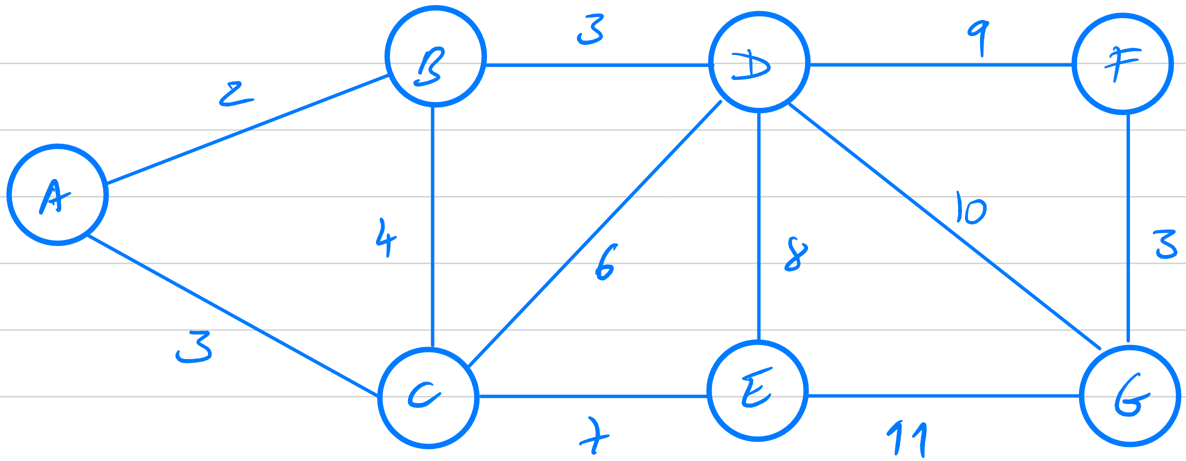
$= 18$

TI 06/07 I.3 -



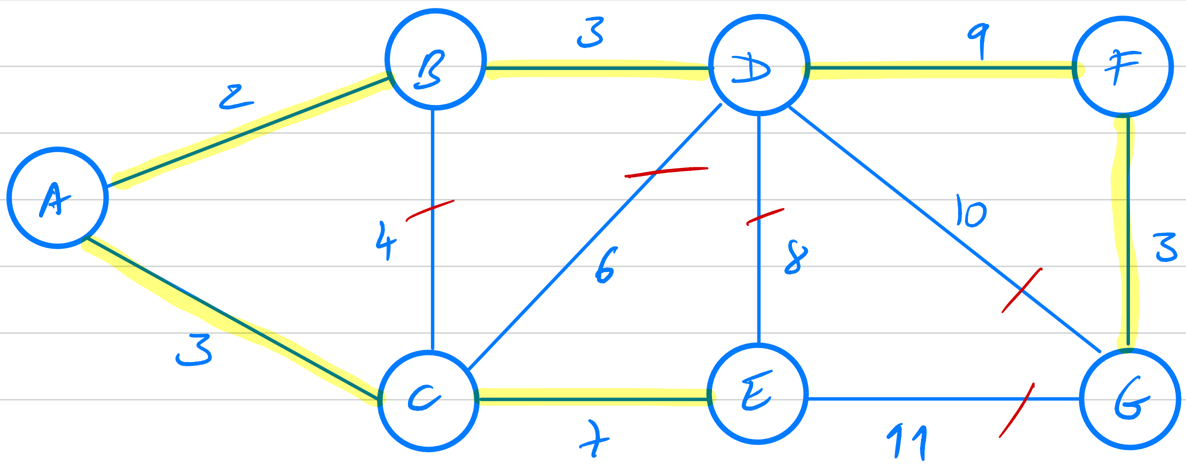
- Kruskal
- Prim

TI 07/08 - II.1



- Uma MST
- Peso da MST
- nº de MSTs

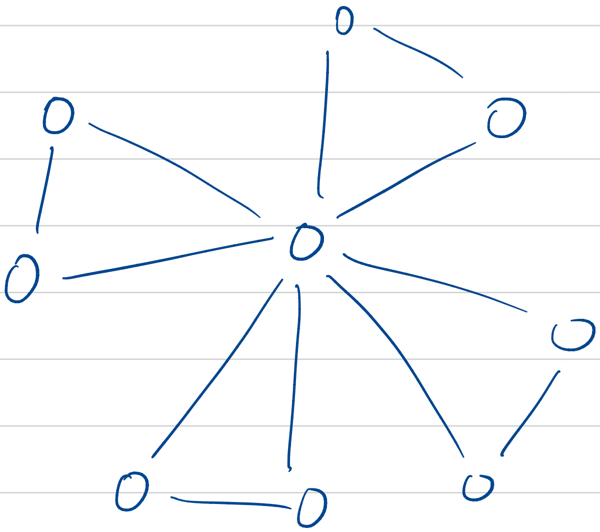
TI 07/08 - II.1



- Uma MST
- Peso da MST
- nº de MSTs ⇒ 1

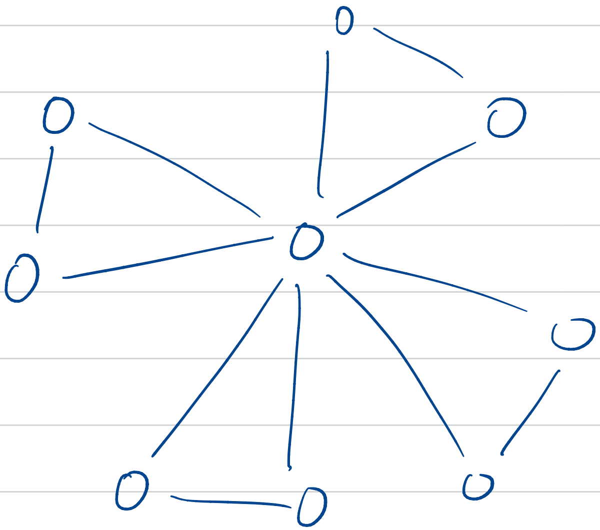
$$W(T) = 2+3+3+3+9+7$$
$$= 2+9+9+7$$
$$= 27$$

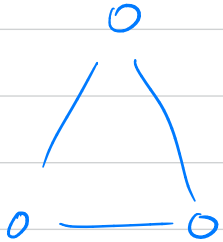
TI 08/09 II.1



- Todos os arcos pesam o mesmo

TI 08/09 II.1





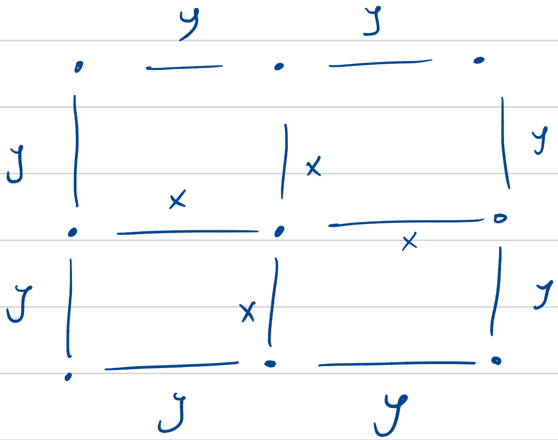
Quantas MSTs?

3

• Todos os arcos pesam o mesmo

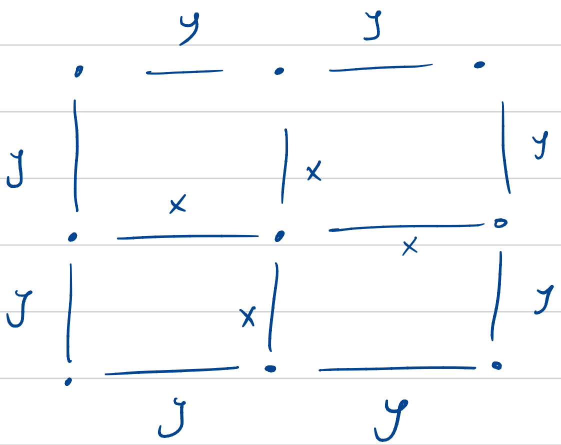
Total: $3^4 = 81$

RI 08/09 II.1

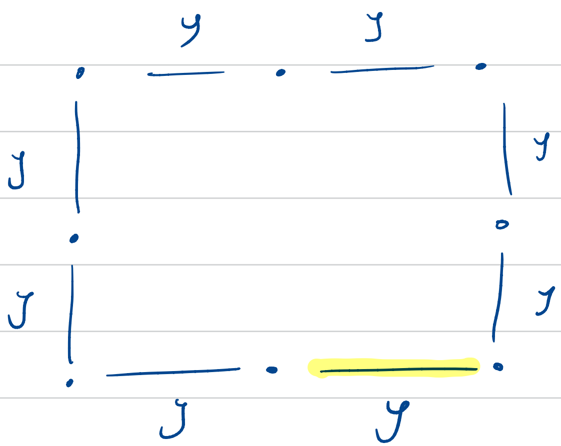


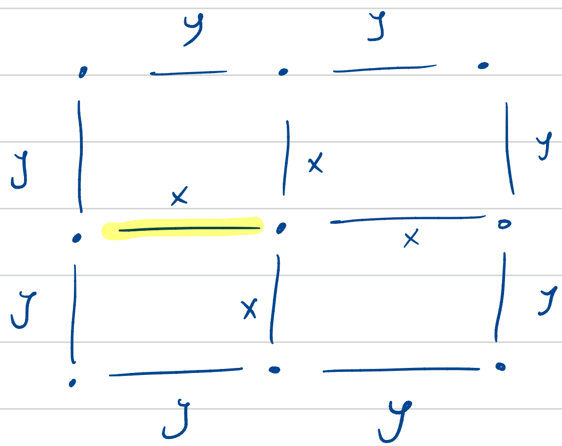
$x > y$

R1 08/09 II.1



$x > y$





8 MSTs possíveis

Qual o arco q pode ser removido?

Nº total de MSTs: $8 \times \underline{4} = 32$

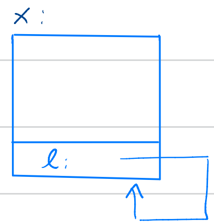
↳ Qual o arco interior q vai ser mantido?

Ex 21.3-4

```
MakeSet(x)
    x.p = x
    x.Rank = 0
```

$\longrightarrow$

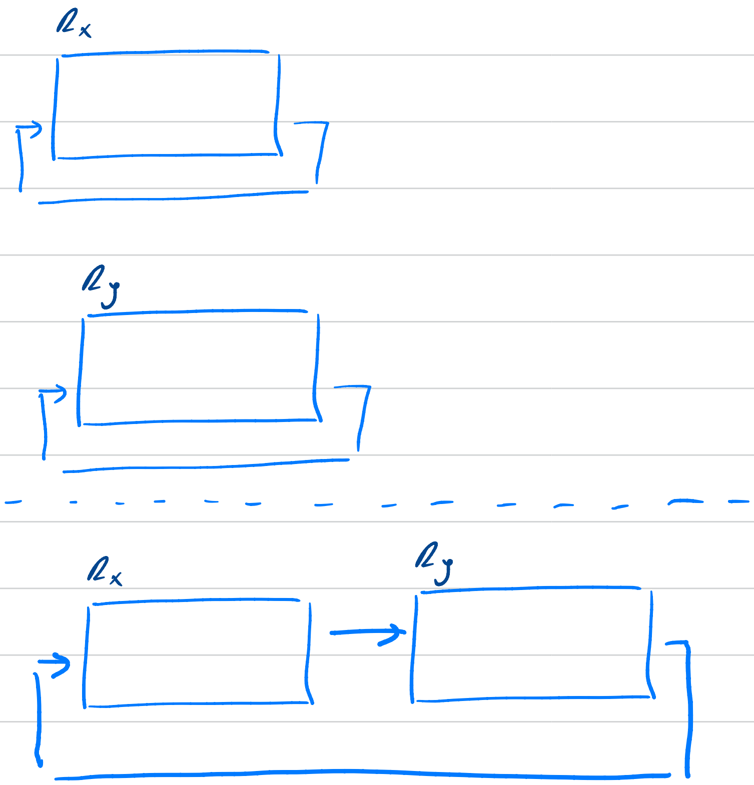
```
MakeSet(x)
    x.p = x
    x.Rank = 0
    x.l = x
```



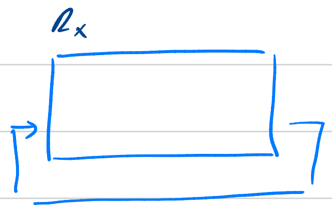

```
Union(x,y)
  let Rx = FindSet(x)
  let Ry = FindSet(y)
  if (Rx == Ry) return
  if (Rx.Rank > Ry.Rank)
      Ry.p := Rx
  else if (Ry.Rank > Rx.Rank)
      Rx.p := Ry
  else
      Ry.p := Rx
      Ry.Rank := Rx.Rank + 1
```

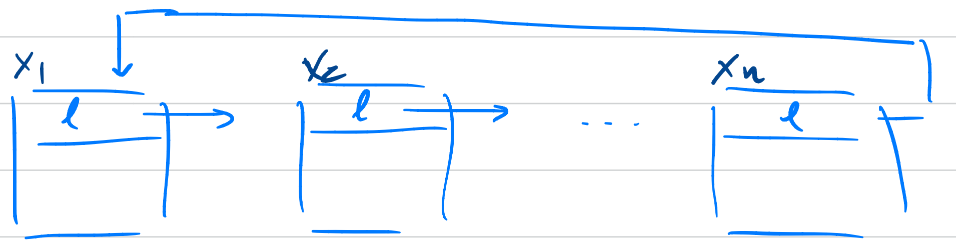
$\longrightarrow$

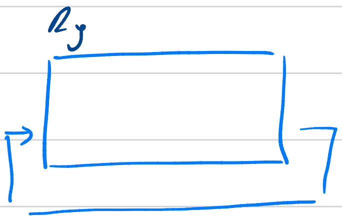
```
Union(x,y)
  let Rx = FindSet(x)
  let Ry = FindSet(y)
  if (Rx == Ry) return
  if (Rx.Rank > Ry.Rank)
     Ry.p := Rx
     Extend(List(Rx, Ry))
  else if (Ry.Rank > Rx.Rank)
     Rx.p := Ry
     Extend(List(Ry, Rx))
  else
     Ry.p := Rx
     Ry.Rank := Rx.Rank + 1
     ExtendList(Rx, Ry)
```

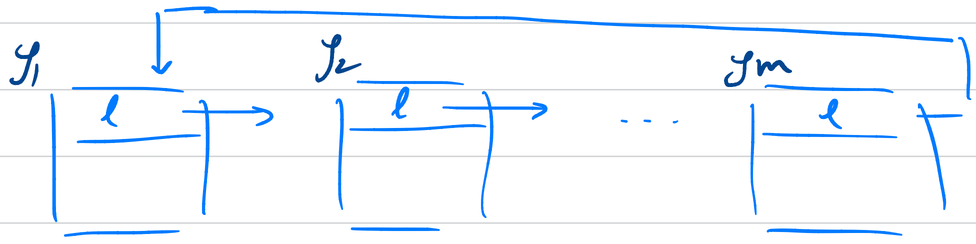


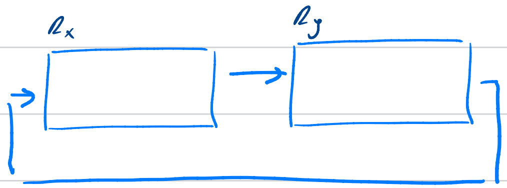
Joining Circular Lists:



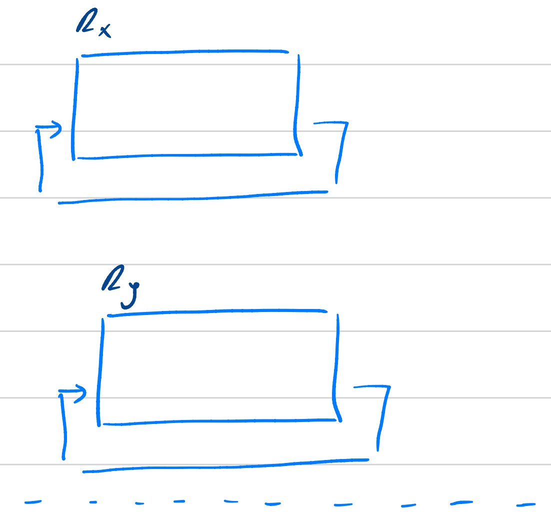


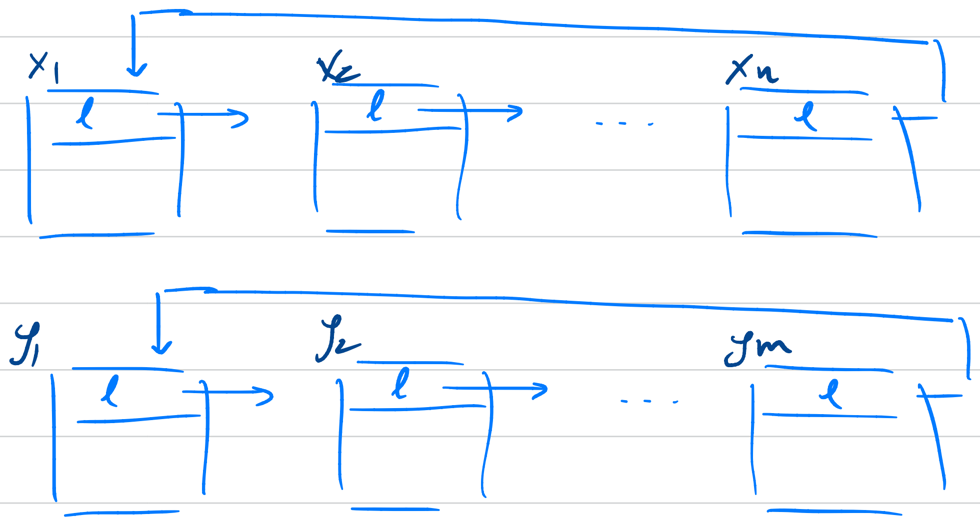


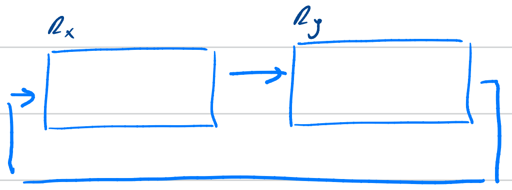




Joining circular lists:

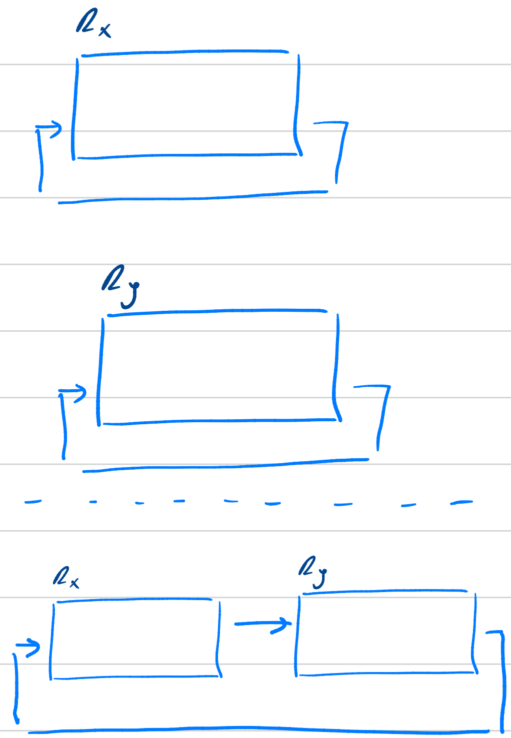


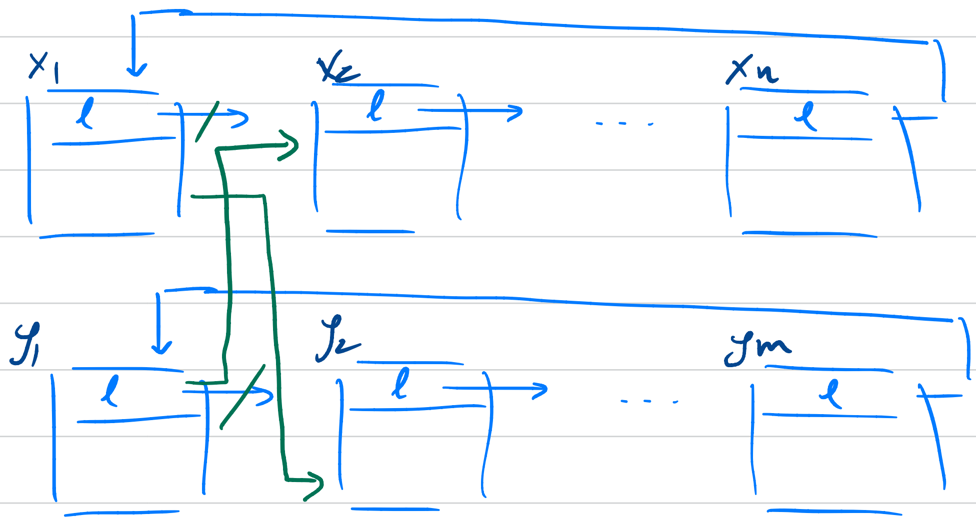


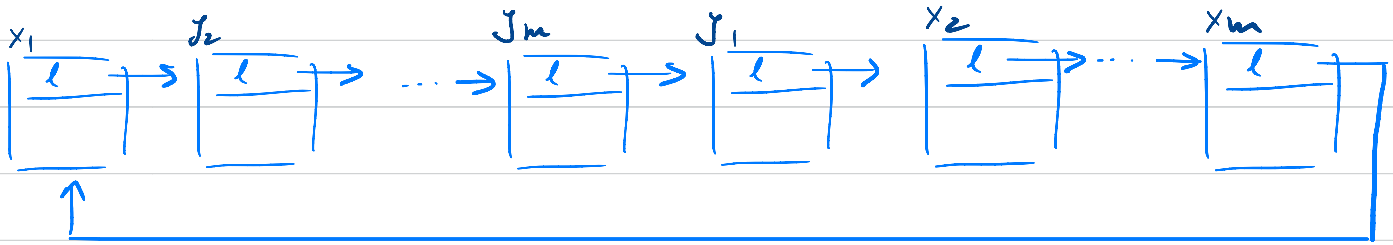



```
PrintSet(x)
    let head = x
    do {
        print(x)
        x = x.l
    } while (x ≠ head)
```

Joining circular lists:







ExtendCList(x, y)

$tmp := x.l$
$x.l := y.l$
$y.l := tmp$

23.2-2

```
Prim(G, w, r)
  for each v ∈ G.V
    v.key = ∞; v.π = Nil
  r.key := 0;
  let Q be a min-priority queue with content G.V
  while Q ≠ []
    let u = ExtractMin(Q)
    for each v ∈ G.Adj[u]
      if (v.key > w(u,v)) && v ∈ Q
        v.key := w(u,v); v.π := u
```

$\Longrightarrow$

W → matriz de pesos
n → nº de vértices
r → vértice de origem

```
Prim(W, n, r)
let key[1..n] be a new array
let π[1..n] be a new array
let done[1..n] be a new array
for i=1 to n
    key[i] := ∞; π[i] := Nil; done[i] := false
key[r] := 0
for j=1 to n-1
   let w* = min{ key[i] | done[i] = false }
   Pick u s.t. key[i] = w*
   done[u] := true
   for i=1 to n
```

23.2-2

```
Prim(G, w, r)
  for each v ∈ G.V
    v.key = ∞; v.π = Nil
  r.key := 0;
  let Q be a min-priority queue with content G.V
  while Q ≠ []
    let u = ExtractMin(Q)
    for each v ∈ G.Adj[u]
      if (v.key > w(u,v)) && v ∈ Q
        v.key := w(u,v); v.π := u
```

⟹

```
Prim(W, n, r)
  let key[1..n] be a new array
  let π[1..n] be a new array
  let done[1..n] be a new array
  for i=1 to n
      key[i] := ∞; π[i] := Nil; done[i] := false
  key[r] := 0
  for j=1 to n-1
    let w* = min{ key[i] | done[i] = false }
    Pick u s.t. key[i] = w*
    done[u] := true
    for i=1 to n
      if (W[u,i] < key[i] && done[i] = false)
        key[i] := W[u,i]
        π[i] := u
```

W → matriz de pesos
n → nº de vértices
r → vértice de origem

23.2-2

Prim(G, w, r)
  for each $v \in G.V$
    $v.key = \infty$; $v.\pi = Nil$
  $r.key := 0$;
  let Q be a min-priority queue with content $G.V$
  while $Q \neq [\,]$
    let u = ExtractMin(Q)
    for each $v \in G.Adj[u]$
      if $(v.key > w(u,v))$ && $v \in Q$
        $v.key := w(u,v)$; $v.\pi := u$

$\Longrightarrow$

Prim(W, n, r)
$W \to$ matriz de pesos
$n \to$ nº de vértices
$r \to$ vértice de origem

let $key[1..n]$ be a new array
let $\pi[1..n]$ be a new array
let $done[1..n]$ be a new array
for $i = 1$ to n
  $key[i] := \infty$; $\pi[i] := Nil$; $done[i] := false$
$key[r] := 0$
for $j = 1$ to $n-1$ — $O(n^2)$
  let $w^* = \min\{ key[i] \mid done[i] = false \}$ — $O(n)$
  Pick u s.t. $key[i] = w^*$
  $done[u] := true$
  for $i = 1$ to n — $O(n)$
    if $(W[u,i] < key[i]$ && $done[i] = false)$
      $key[i] := W[u,i]$
      $\pi[i] := u$

23.1-6

- Se para qualquer corte num grafo pesado existe existe um único arco leve q̄ cruza o corte então o grafo admite uma única MST.

a) Provar a implicação
b) Contra-exemplo para o sentido inverso.

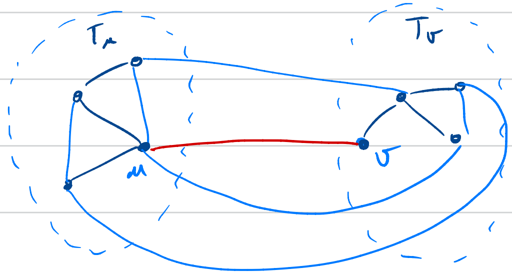
a) $\underbrace{\text{Para todo o corte, existe um único arco leve q̄ cruza o corte}}_{(+)}$ $\Rightarrow$ O grafo admite uma única MST T.

- Sejam T_1 e T_2 duas quaisquer MSTs de G.
  Vamos provar que $T_1 = T_2$ dado (+).

- Provar que $T_1 = T_2$ corresponde a provar que:
  $\forall u, v \in V.\ (u,v) \in T_1 \Leftrightarrow (u,v) \in T_2$

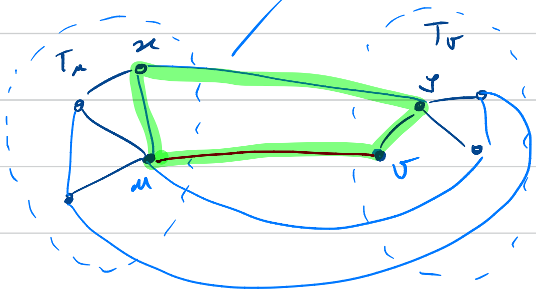
  ↳ Como a prova é simétrica basta provar um sentido da implicação.

23.1-6 Queremos provar que: $\forall u, v \in V.\ (u,v) \in T_1 \Rightarrow (u,v) \in T_2$



- Tome-se qualquer arco (u,v) em T_1.
  Se $(u,v) \in T_2$, não há nada a provar. Admitimos portanto que $(u,v) \notin T_2$.
  A remoção do arco (u,v) induz um corte (T_u, T_v), onde T_u contém os vértices atingíveis a partir de u usando arcos em $T_1 \setminus \{(u,v)\}$ e T_v contém os arcos atingíveis a partir de v.

ciclo q (u,v) introduz em T_2



- Como T_1 é MST concluímos q (u,v) é um arco leve q cruza o corte.

- Como $(u,v) \notin T_2$, a sua inclusão em T_2 dá origem a um caminho circular, p_c, que cruza o corte duas vezes:
  - uma vez pelo arco (u,v) e outra por um arco de T_2; seja (x,y) esse arco

- Como (u,v) é leve, usamos a hipótese p/ concluir que:
  $w(u,v) < w(x,y)$.

Segue q: $T_2' = (T_2 \setminus \{(x,y)\}) \cup \{(u,v)\}$

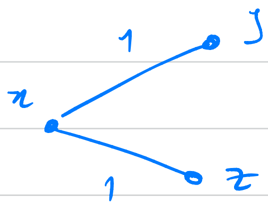
é árvore abrangente e $w(T_2') < w(T_2)$.
Concluímos q T_2 não é MST, estabelecendo a contradição.

23.1 - 6

- Se para qualquer corte num grafo pesado existe existe um único arco leve q̄ cruze o corte então o grafo admite uma única MST.

a) Provar a implicação
b) Contra-exemplo para o sentido inverso.

b)



23.1-8

- Esboço de prova

Suponhamos por contradição que um dado grafo pesado G admite duas MSTs T_1 e T_2 com listas de pesos L_1 e L_2, respectivamente, tais que $L_1 \neq L_2$.

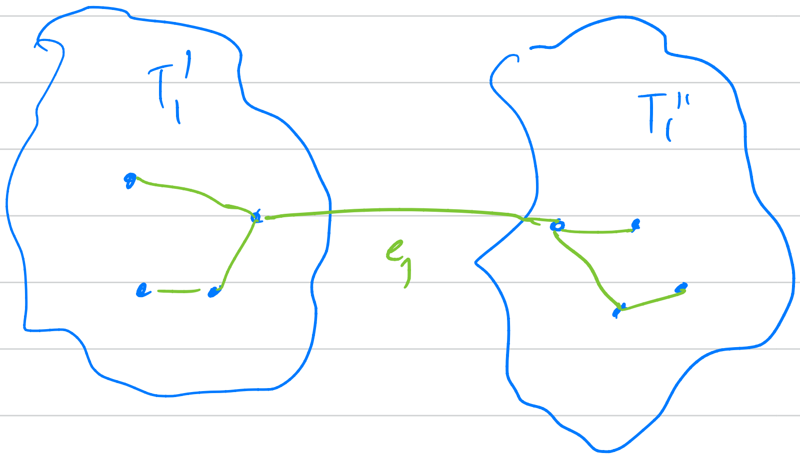
- Para facilitar a prova consideramos os multiconjuntos com os valores de L_1 e os valores de L_2, W_1 e W_2 respectivamente.

- Consideremos o conjunto: $W^* = \overbrace{W_1 \Delta W_2}$ ↳ diferença simétrica

Seja $w^* = \min W^*$

O nº de vezes q w^* ocorre em L_1 é diferente do nº de vezes que w^* ocorre em L_2.
Assumimos sem perda de generalidade q o nº de vezes que w^* ocorre em L_1 é $\geq$ ao nº de vezes que w^* ocorre em L_2. Tem portanto de existir um arco de peso w^* q ocorre em T_1 e não ocorre em T_2.
Seja $\underline{e_1}$ esse arco.

- Consideremos o corte $(S, V \setminus S)$ induzido por T_1 e e_1 em G.



- $T_1 = T_1' \cup T_1'' \cup \{e_1\}$

- Seja $E(S)$ o conjunto dos arcos de G q cruzam o corte.

- Seja e_2 o arco em $T_2 \cap E(S)$ com peso mínimo

- Há 3 casos a considerar:

  (I) $w(e_2) < w^*$

  (II) $w(e_2) > w^*$

  (III) $w(e_2) = w^*$

I $w(e_2) < w^*$

Considere-se a MST:

$$\overline{T_1} = T_1' \cup T_1'' \cup \{e_2\}$$

Temos que: $w(\overline{T_1}) < w(T_1)$ $\Rightarrow$ Contradição!

II $w(e_2) > w^*$

Considere-se a MST: $\overline{T_2} = T_2' \cup T_2'' \cup \{e_1\}$

Temos que: $w(\overline{T_2}) < w(T_2)$ $\Rightarrow$ Contradição!

III $\quad v(e_2) = w^*$

- Considere-se a árvore:

$$\hat{T}_1 = T_1' \cup T_1'' \cup \{e_2\}$$

- $w(\hat{T}_1) = W(T_1)$ e a lista de pesos dos arcos de $\hat{T}_1$ é igual à lista de pesos dos arcos de T_1.

- Repetimos o argumento original para as árvores $\hat{T}_1$ e T_2, observando q as árvores $\hat{T}_1$ e T_2 têm menos arcos de peso w^* diferentes. Como T_2 tem um nº finito de arcos com peso w^* (e inferior a T_1), inevitavelmente vamos acabar por estar no caso I ou II da prova.