

Arbres couvrants

Algorithmique et structures de données, 2024-2025

P. Albuquerque (B410) et O. Malaspinas (A401), ISC, HEPIA
2025-06-06

En partie inspirés des supports de cours de P. Albuquerque

Les arbres couvrants

Trouver un réseau électrique pour

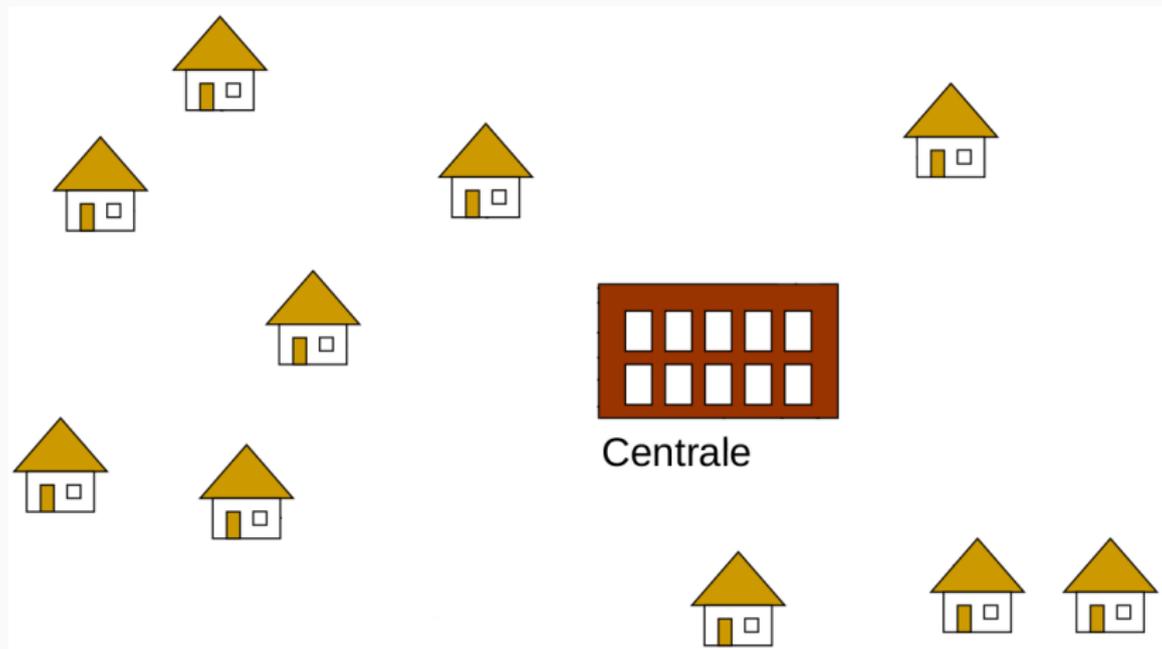


Figure 1: Ces maisons n'ont pas d'électricité.

Solution: pas optimale

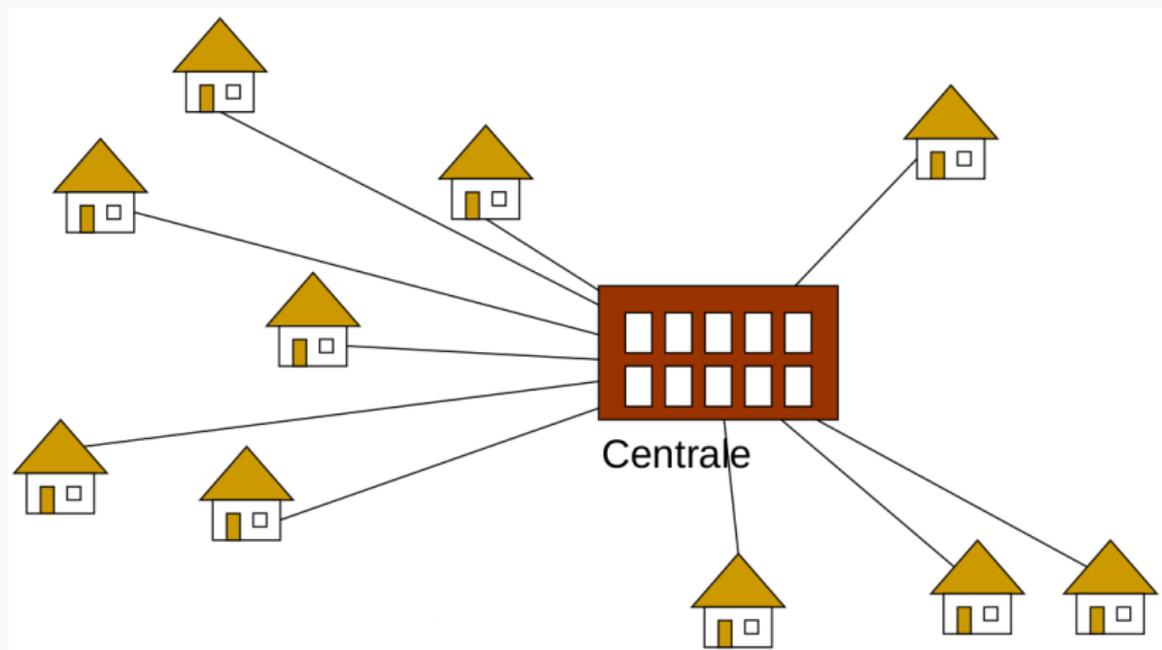


Figure 2: Le réseau simple, mais nul.

- La longueur totale des câbles est super longue!

Solution: optimale

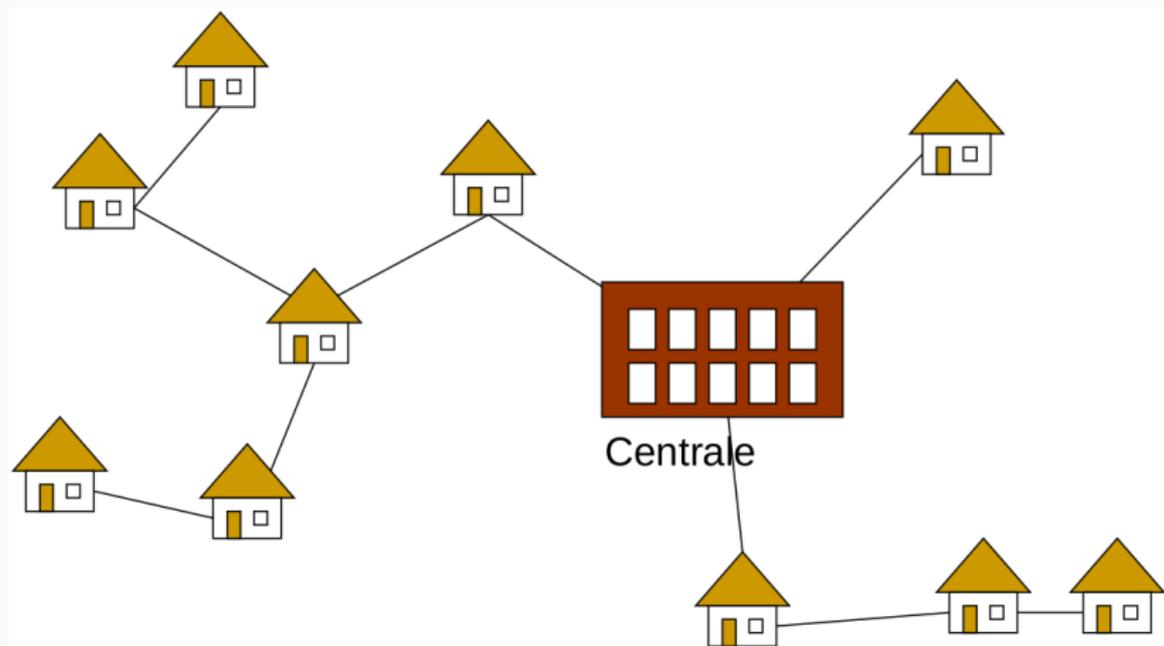


Figure 3: Le meilleur réseau.

Application: minimisation des coûts

- Équipement d'un lotissement avec des lignes électriques/téléphoniques, des canalisations, ...

Application: minimisation des coûts

- Équipement d'un lotissement avec des lignes électriques/téléphoniques, des canalisations, ...
- Pour réduire les coûts, on cherche à minimiser la longueur totale des câbles/tuyaux.

Application: minimisation des coûts

- Équipement d'un lotissement avec des lignes électriques/téléphoniques, des canalisations, ...
- Pour réduire les coûts, on cherche à minimiser la longueur totale des câbles/tuyaux.
- Les lignes/tuyaux forment un *arbre couvrant*.

Application: minimisation des coûts

- Équipement d'un lotissement avec des lignes électriques/téléphoniques, des canalisations, ...
- Pour réduire les coûts, on cherche à minimiser la longueur totale des câbles/tuyaux.
- Les lignes/tuyaux forment un *arbre couvrant*.
- La meilleure option est un *arbre couvrant minimal*.

Formalisation: Les arbres couvrants

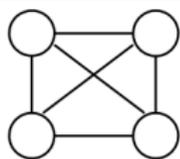
- Qu'est-ce qu'un arbre couvrant? Des idées? De quel objet part-on?
Où va-t-on?

Formalisation: Les arbres couvrants

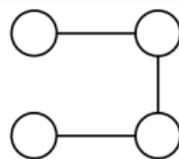
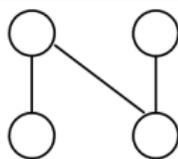
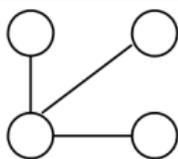
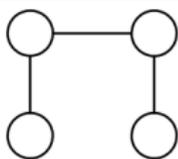
- Qu'est-ce qu'un arbre couvrant? Des idées? De quel objet part-on?
Où va-t-on?
- Un arbre couvrant d'un graphe non-orienté et connexe est:
 - un arbre inclus dans le graphe qui connecte tous les sommets du graphe.

Formalisation: Les arbres couvrants

- Qu'est-ce qu'un arbre couvrant? Des idées? De quel objet part-on? Où va-t-on?
- Un arbre couvrant d'un graphe non-orienté et connexe est:
 - un arbre inclus dans le graphe qui connecte tous les sommets du graphe.



Graphe non-orienté
connexe



Quatre arbres couvrants de ce graphe

Figure 4: Exemple d'arbres couvrants d'un graphe connexe.

Arbres couvrants

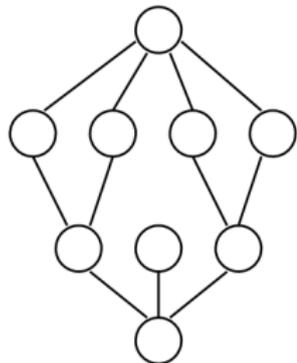
- Quels algorithmes que nous avons déjà vus, permettent de construire des arbres couvrants?

Arbres couvrants

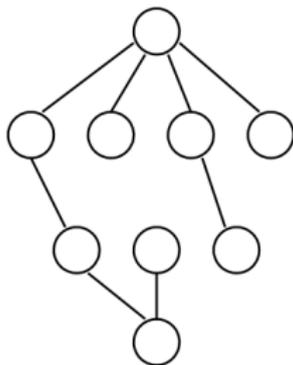
- Quels algorithmes que nous avons déjà vus, permettent de construire des arbres couvrants?
- Les parcours en largeur et en profondeur!

Arbres couvrants

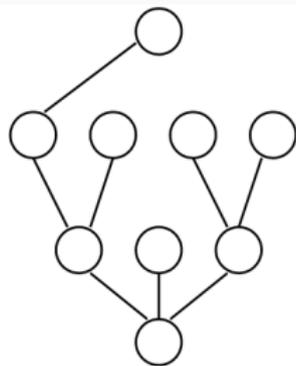
- Quels algorithmes que nous avons déjà vus, permettent de construire des arbres couvrants?
- Les parcours en largeur et en profondeur!



Graphe non-orienté



Arbre couvrant
obtenu via un BFS



Arbre couvrant
obtenu via un DFS

Figure 5: Graphe et parcours comme arbres couvrants.

Arbres couvrants minimaux

- Un *arbre couvrant minimal* est un sous-graphe d'un graphe non-orienté pondéré $G(V, E)$ tel quel:
 - C'est un arbre (graphe acyclique);
 - Il couvre tous les sommets de G et contient $|V| - 1$ arêtes;
 - Le coût total associé aux arêtes de l'arbre est minimum parmi tous les arbres couvrants possibles.

Arbres couvrants minimaux

- Un *arbre couvrant minimal* est un sous-graphe d'un graphe non-orienté pondéré $G(V, E)$ tel quel:
 - C'est un arbre (graphe acyclique);
 - Il couvre tous les sommets de G et contient $|V| - 1$ arêtes;
 - Le coût total associé aux arêtes de l'arbre est minimum parmi tous les arbres couvrants possibles.
- Est-il unique?

Arbres couvrants minimaux

- Un *arbre couvrant minimal* est un sous-graphe d'un graphe non-orienté pondéré $G(V, E)$ tel quel:
 - C'est un arbre (graphe acyclique);
 - Il couvre tous les sommets de G et contient $|V| - 1$ arêtes;
 - Le coût total associé aux arêtes de l'arbre est minimum parmi tous les arbres couvrants possibles.
- Est-il unique?
- Pas forcément.

Arbres couvrants minimaux

- Comment générer un arbre couvrant minimal?

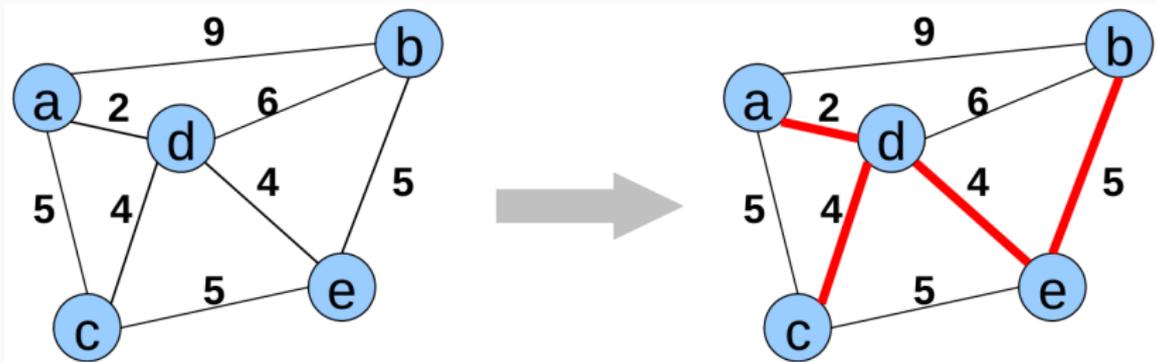


Figure 6: Un graphe, connexe, non-orienté, pondéré, et un arbre couvrant minimal.

Algorithme de Prim

Un exemple

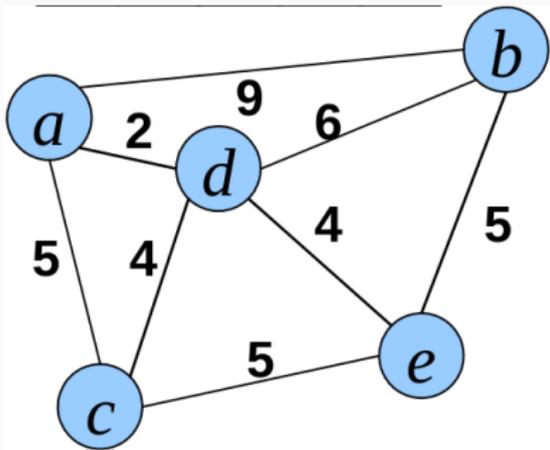


Figure 7: Le graphe de départ.

On part de *e* (au hasard)

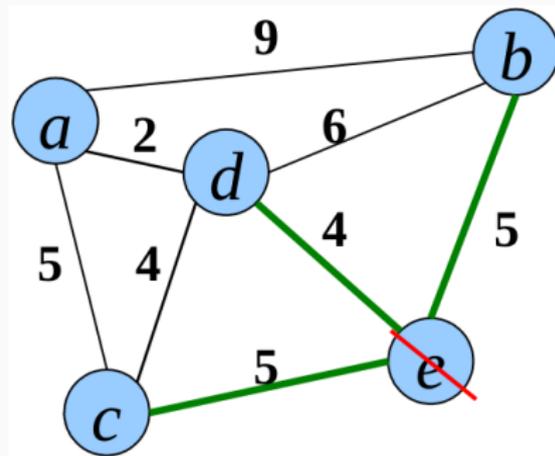


Figure 8: Le sommet *e* est couvert.

Algorithme de Prim

On choisit comment?

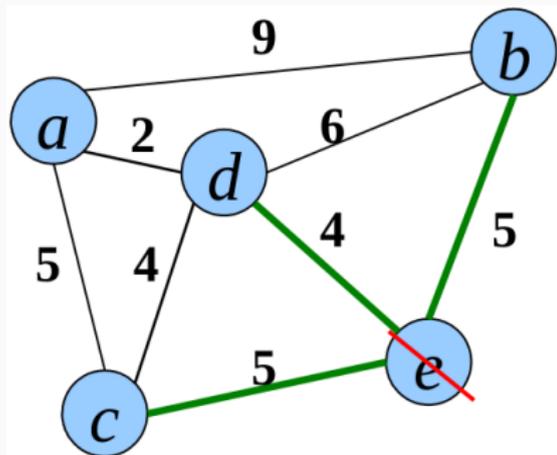


Figure 9: Quelle arête choisir?

Algorithme de Prim

On choisit comment?

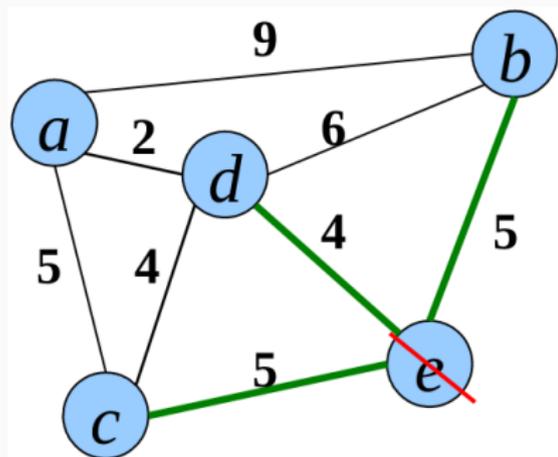


Figure 9: Quelle arête choisir?

- L'arête la plus courte sortant d'un sommet déjà visité, et entrant dans un sommet non-visité.

Algorithme de Prim

On choisit comment?

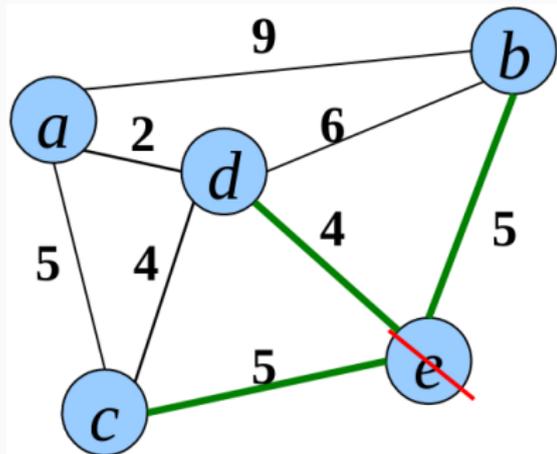


Figure 9: Quelle arête choisir?

- L'arête la plus courte sortant d'un sommet déjà visité, et entrant dans un sommet non-visité.

L'arête $e \rightarrow d$

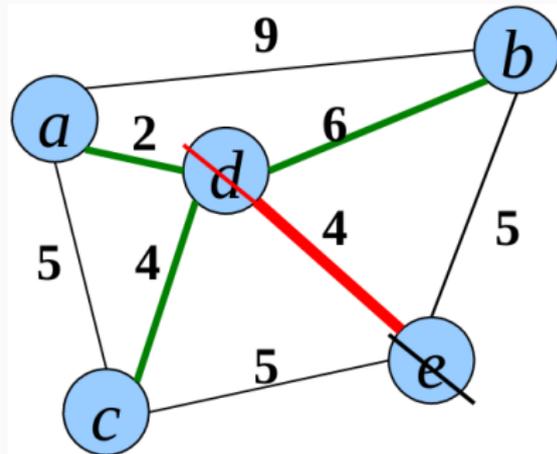


Figure 10: Le sommet d est couvert.

Algorithme de Prim

On choisit comment?

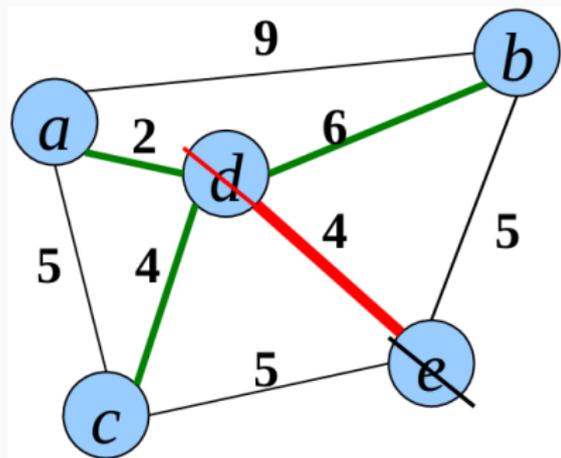


Figure 11: Quelle arête choisir?

Algorithme de Prim

On choisit comment?

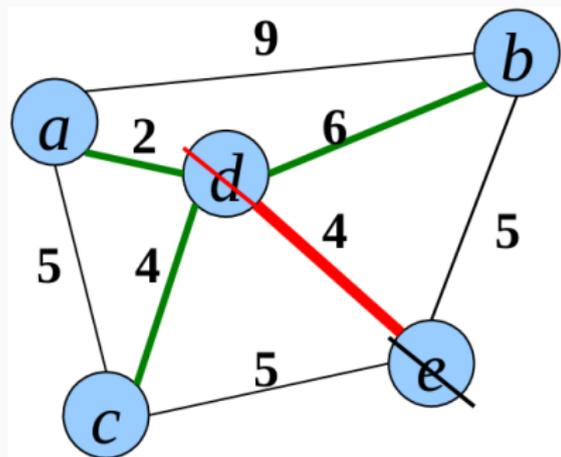


Figure 11: Quelle arête choisir?

- L'arête la plus courte sortant d'un sommet déjà visité, et entrant dans un sommet non-visité.

Algorithme de Prim

On choisit comment?

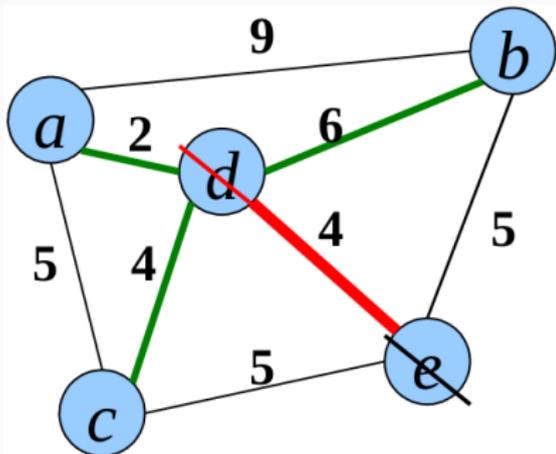


Figure 11: Quelle arête choisir?

- L'arête la plus courte sortant d'un sommet déjà visité, et entrant dans un sommet non-visité.

L'arête $d \rightarrow a$

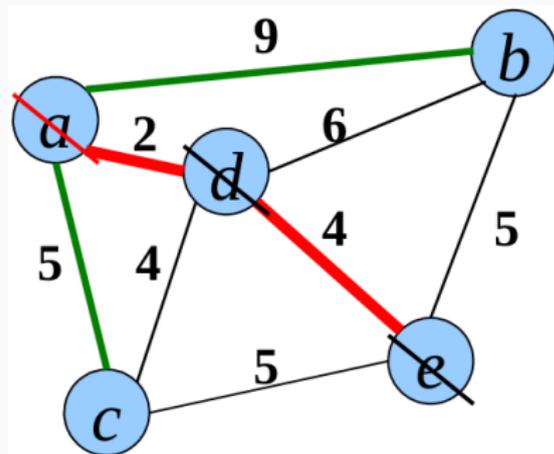


Figure 12: Le sommet a est couvert.

Algorithme de Prim

On choisit comment?

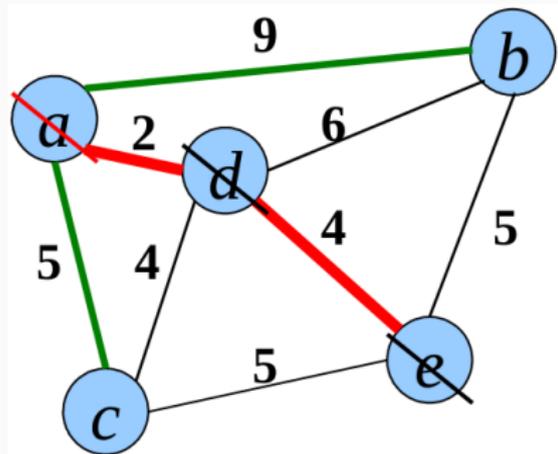


Figure 13: Quelle arête choisir?

Algorithme de Prim

On choisit comment?

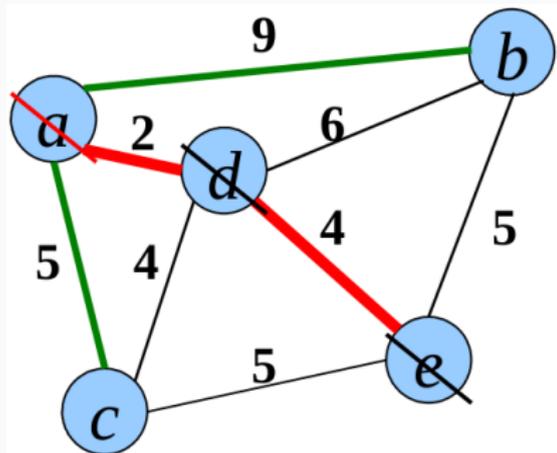


Figure 13: Quelle arête choisir?

- L'arête la plus courte sortant d'un sommet déjà visité, et entrant dans un sommet non-visité.

Algorithme de Prim

On choisit comment?

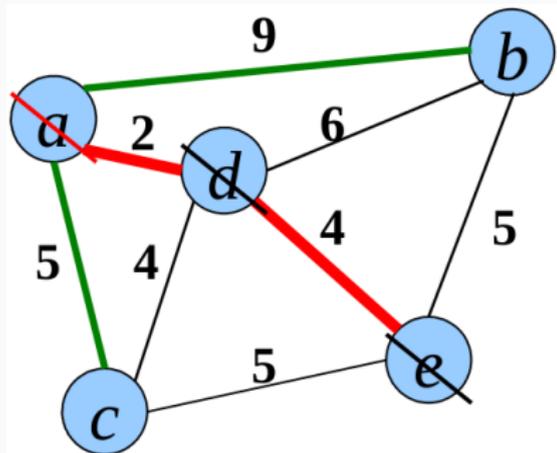


Figure 13: Quelle arête choisir?

- L'arête la plus courte sortant d'un sommet déjà visité, et entrant dans un sommet non-visité.

L'arête $d \rightarrow c$

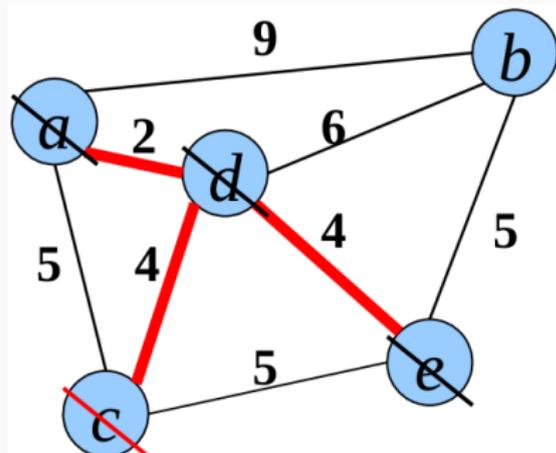


Figure 14: Le sommet c est couvert.

Algorithme de Prim

On choisit comment?

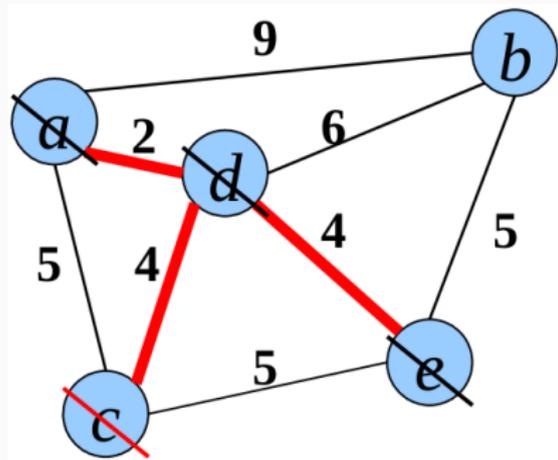


Figure 15: Quelle arête choisir?

Algorithme de Prim

On choisit comment?

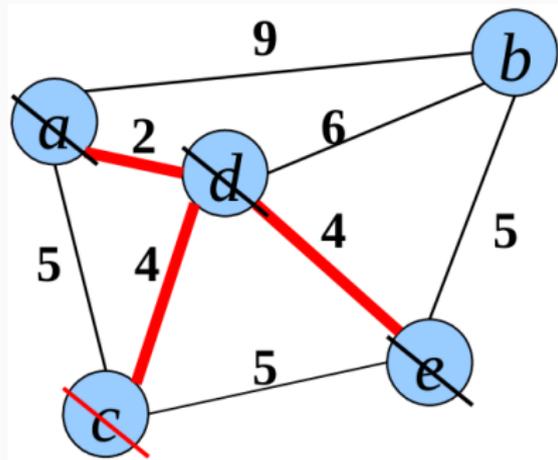


Figure 15: Quelle arête choisir?

- L'arête la plus courte sortant d'un sommet déjà visité, et entrant dans un sommet non-visité.

Algorithme de Prim

On choisit comment?

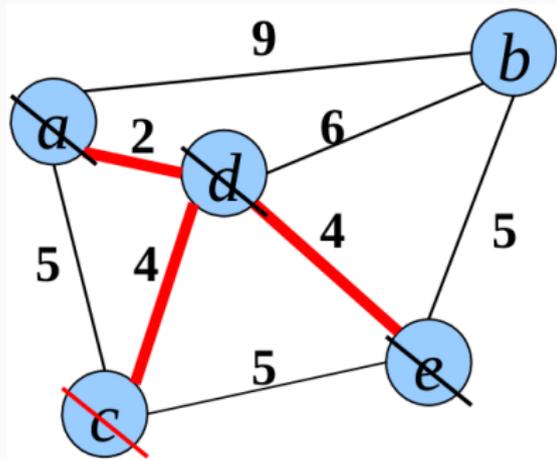


Figure 15: Quelle arête choisir?

- L'arête la plus courte sortant d'un sommet déjà visité, et entrant dans un sommet non-visité.

L'arête e→b

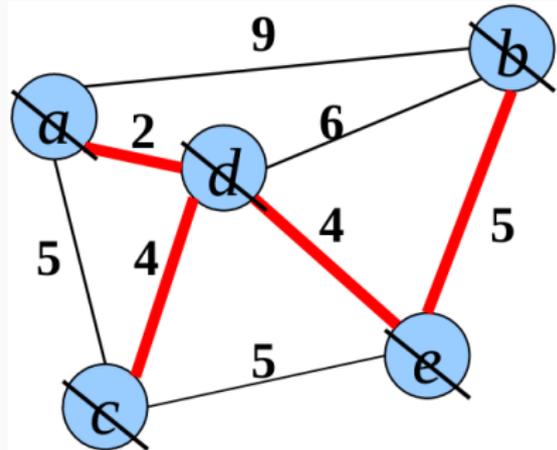


Figure 16: Le sommet b est couvert.

- Game over!

Exemple d'algorithme de Prim

Un exemple

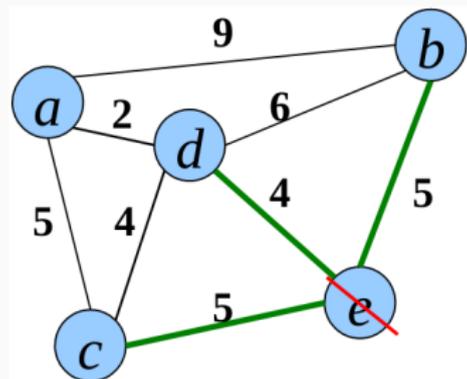


Figure 17: Étape 1.

FP		e		d		b		c		a		

D		0		inf		inf		inf		inf		
			e		d		b		c		a	

P		-		-		-		-		-		

Devient?

Exemple d'algorithme de Prim

Un exemple

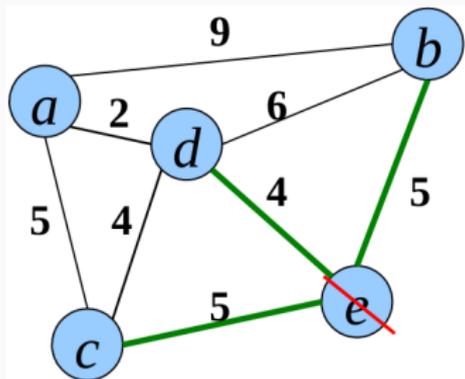


Figure 17: Étape 1.

FP	e	d	b	c	a	
----	---	---	---	---	---	--

D	0	inf	inf	inf	inf	
---	---	-----	-----	-----	-----	--

	e	d	b	c	a	
--	---	---	---	---	---	--

P	-	-	-	-	-	
---	---	---	---	---	---	--

Devient?

FP	d	b	c	a	
----	---	---	---	---	--

D	4	5	5	inf	
---	---	---	---	-----	--

	e	d	b	c	a	
--	---	---	---	---	---	--

P	-	e	e	e	-	
---	---	---	---	---	---	--

Exemple d'algorithme de Prim

Un exemple

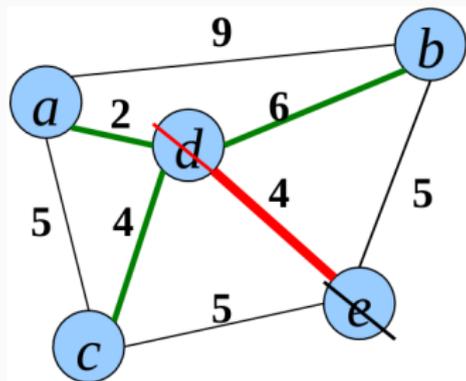


Figure 18: Étape 2.

FP		d		b		c		a				

D		4		5		5		inf				
			e		d		b		c		a	

P		-		e		e		e		-		

Devient?

Exemple d'algorithme de Prim

Un exemple

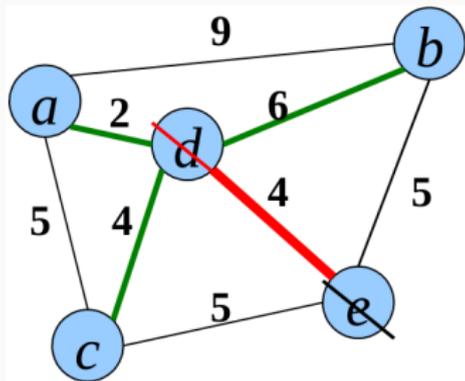


Figure 18: Étape 2.

FP		d		b		c		a	

D		4		5		5		inf	

		e		d		b		c		a	

P		-		e		e		e		-	

Devient?

FP		a		c		b	

D		2		4		5	

		e		d		b		c		a	

P		-		e		e		d		d	

Exemple d'algorithme de Prim

Un exemple

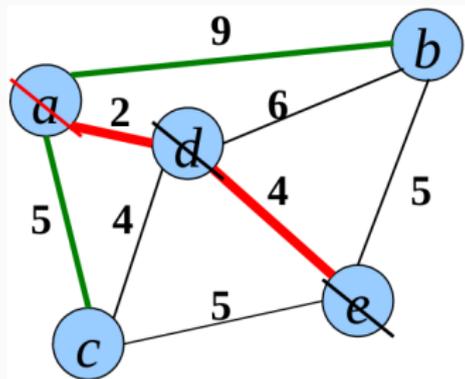


Figure 19: Étape 3.

FP		a		c		b	
----	--	---	--	---	--	---	--

D		2		4		5	
---	--	---	--	---	--	---	--

		e		d		b		c		a	
--	--	---	--	---	--	---	--	---	--	---	--

P		-		e		e		d		d	
---	--	---	--	---	--	---	--	---	--	---	--

Devient?

Exemple d'algorithme de Prim

Un exemple

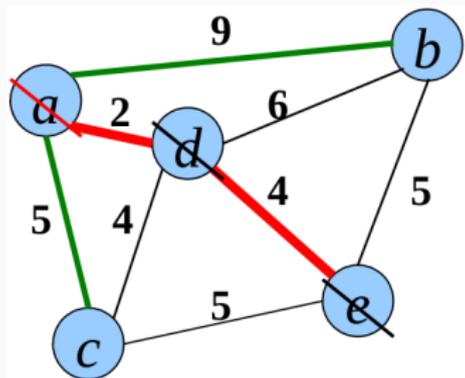


Figure 19: Étape 3.

FP		a		c		b	
----	--	---	--	---	--	---	--

D		2		4		5	
---	--	---	--	---	--	---	--

		e		d		b		c		a	
--	--	---	--	---	--	---	--	---	--	---	--

P		-		e		e		d		d	
---	--	---	--	---	--	---	--	---	--	---	--

Devient?

FP		c		b	
----	--	---	--	---	--

D		4		5	
---	--	---	--	---	--

		e		d		b		c		a	
--	--	---	--	---	--	---	--	---	--	---	--

P		-		e		e		d		d	
---	--	---	--	---	--	---	--	---	--	---	--

Exemple d'algorithme de Prim

Un exemple

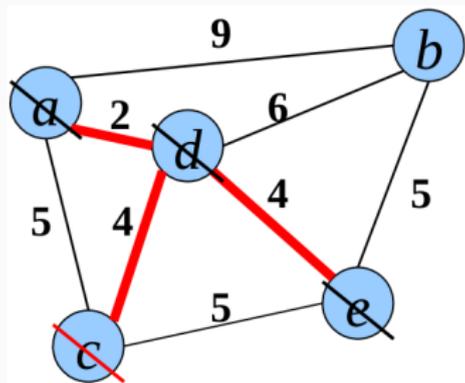


Figure 20: Étape 4.

FP		c		b	

D		4		5	
---	--	---	--	---	--

		e		d		b		c		a	

P		-		e		e		d		d	
---	--	---	--	---	--	---	--	---	--	---	--

Devient?

Exemple d'algorithme de Prim

Un exemple

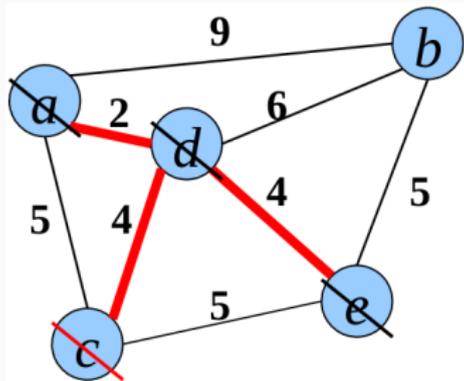


Figure 20: Étape 4.

FP | c | b |

D | 4 | 5 |

| e | d | b | c | a |

P | - | e | e | d | d |

Devient?

FP | b |

D | 5 |

| e | d | b | c | a |

P | - | e | e | d | d |

Exemple d'algorithme de Prim

Un exemple

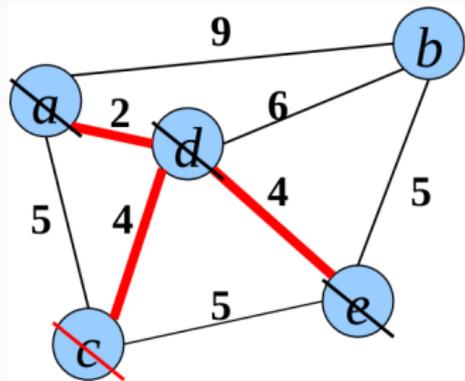


Figure 21: Étape 5.

FP | b |

D | 5 |

| e | d | b | c | a |

P | - | e | e | d | d |

Devient?

Exemple d'algorithme de Prim

Un exemple

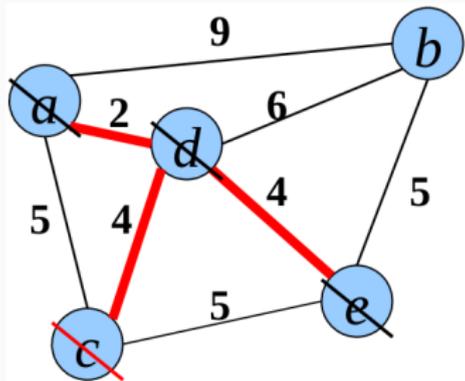


Figure 21: Étape 5.

FP | b |

D | 5 |

| e | d | b | c | a |

P | - | e | e | d | d |

Devient?

FP |

D |

| e | d | b | c | a |

P | - | e | e | d | d |

Structures de données

- Dans quoi allons-nous stocker les sommets?

Structures de données

- Dans quoi allons-nous stocker les sommets?
- File de priorité min.
- Autre chose?

Structures de données

- Dans quoi allons-nous stocker les sommets?
- File de priorité min.
- Autre chose?
- Tableau des distances (comme pour Dijkstra).
- Autre chose?

Structures de données

- Dans quoi allons-nous stocker les sommets?
- File de priorité min.
- Autre chose?
- Tableau des distances (comme pour Dijkstra).
- Autre chose?
- Tableau des parents (presque comme pour Dijkstra).
- Autre chose?

Structures de données

- Dans quoi allons-nous stocker les sommets?
- File de priorité min.
- Autre chose?
- Tableau des distances (comme pour Dijkstra).
- Autre chose?
- Tableau des parents (presque comme pour Dijkstra).
- Autre chose?
- Non.

Algorithme de Prim

Initialisation: Pseudo-code (2min)

Algorithme de Prim

Initialisation: Pseudo-code (2min)

```
file_priorité, distance, parent initialisation(graphe)
  s_initial = aléatoire(graphe)
  distance[s_initial] = 0
  fp = file_p_vide()
  pour s_courant dans sommets(graphe)
    si s_courant != s_initial
      distance[s_courant] = infini
  parent[s_courant] = indéfini
  fp = enfiler(fp, s_courant, distance[s_courant])
retourne fp, distance, parent
```

Algorithme de Prim

Algorithme: Pseudo-code (5min)

Algorithme de Prim

Algorithme: Pseudo-code (5min)

```
distance, parent prim(graphe)
    fp, distance, parent initialisation(graphe)
    sommets = vide
    tant que !est_vide(fp)
        s_courant, fp = défiler(fp)
        sommets = insérer(sommets, s_courant)
        pour s_voisin dans voisinage(s_courant) et pas dans sommets
            // ou dans fp
                si poids(s_courant, s_voisin) < distance[s_voisin]
                    parent[s_voisin] = s_courant
                    distance[s_voisin] = poids(s_courant, s_voisin)
                    fp = changer_priorité(fp, s_voisin,
                                            poids(s_courant, s_voisin))
    retourne distance, parent
```

Exercice: algorithme de Prim

Appliquer l'algorithme de Prim à (15min):

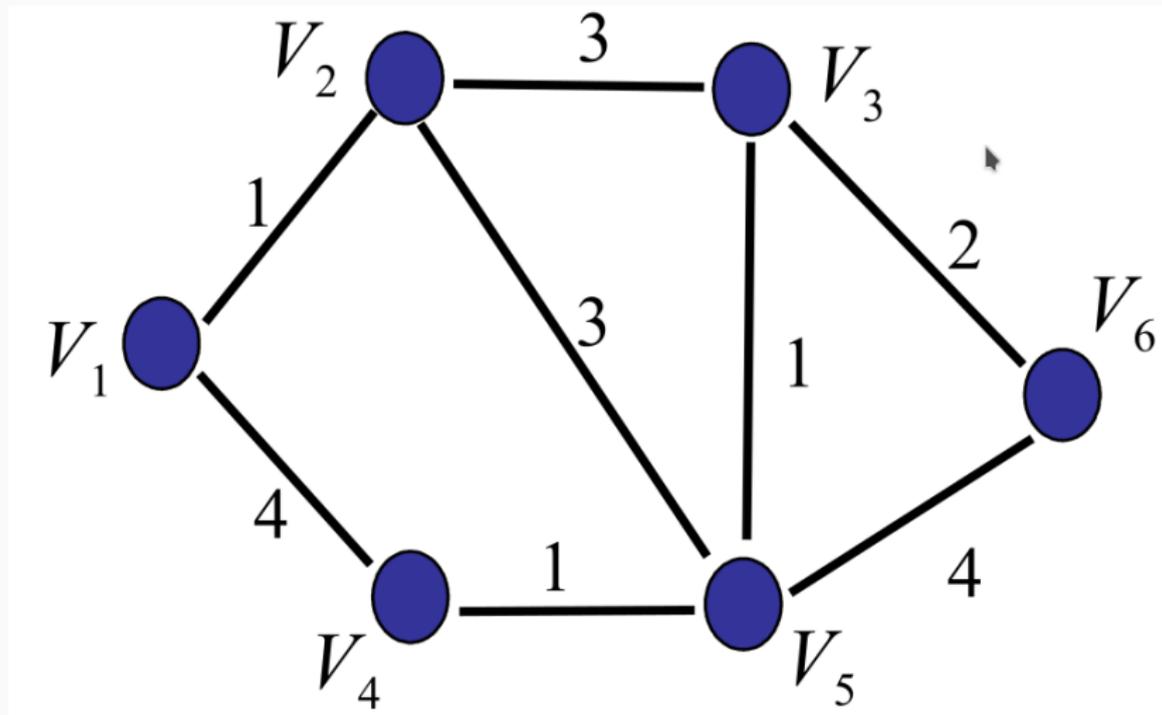
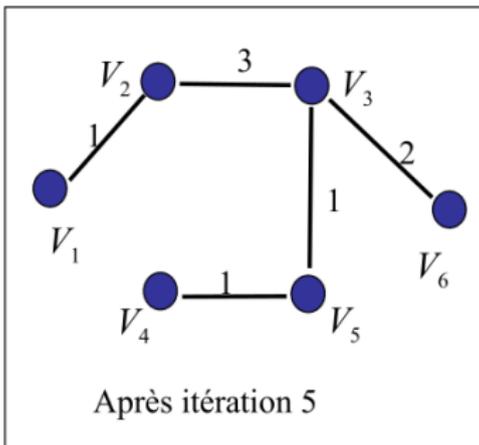
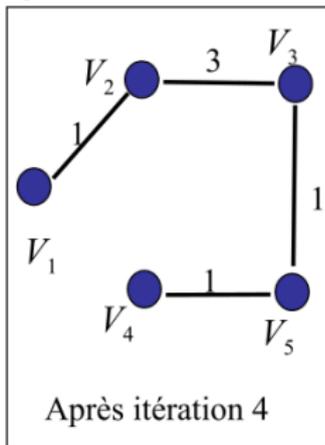
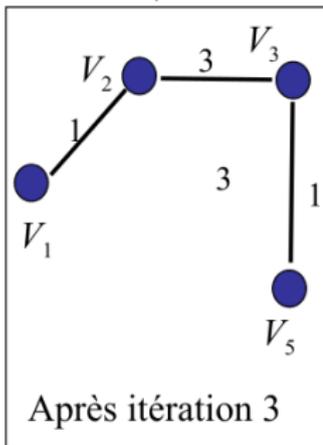
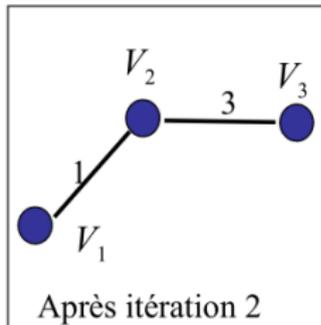
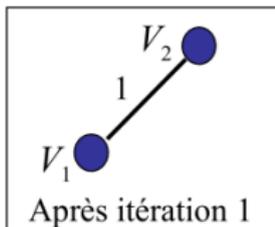
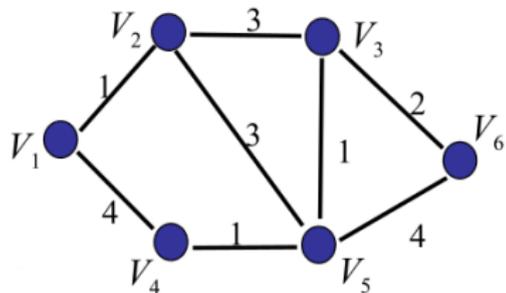


Figure 22: En démarrant du sommet V_1 .

Exercice: algorithme de Prim

Solution



Complexité de l'algorithme de Prim

```
file_priorité, distance, parent initialisation(graphe)
    // choix r et initialisation
    pour v dans sommets(graphe)
        // initialisation distance et parent en  $O(|V|)$ 
        fp = enfiler(fp, v, distance[v])
    retourne fp, distance, parent
distance, parent prim(graphe)
    fp, distance, parent initialisation(graphe) //  $O(|V|)$ 
    sommets = vide
    tant que !est_vide(fp)
        u, fp = défiler(fp) //  $O(|V|)$ 
        sommets = insérer(sommets, u)
        pour v dans voisinage de u et pas dans sommets
            si poids(u, v) < distance[v] //  $O(|E|)$ 
                // māj distance + parent
                fp = changer_priorité(fp, v, poids(u, v)) //  $O(|V|)$ 
    retourne distance, parent
```

- $O(|V|) + O(|E|) + O(|V|^2) = O(|E| + |V|^2)$
- Remarque: $O(|E|)$ n'est pas mutliplié par $O(|V|)$, car les arêtes ne sont traitées qu'une fois en **tout**.

Algorithme de Kruskal

- On ajoute les arêtes de poids minimal:
 - si cela ne crée pas de cycle;
 - on s'arrête quand on a couvert tout le graphe.

Algorithme de Kruskal

- On ajoute les arêtes de poids minimal:
 - si cela ne crée pas de cycle;
 - on s'arrête quand on a couvert tout le graphe.
- Comment on fait ça?

Algorithme de Kruskal

- On ajoute les arêtes de poids minimal:
 - si cela ne crée pas de cycle;
 - on s'arrête quand on a couvert tout le graphe.
- Comment on fait ça?
- Faisons un exemple pour voir.

Algorithme de Kruskal: exemple

Un exemple

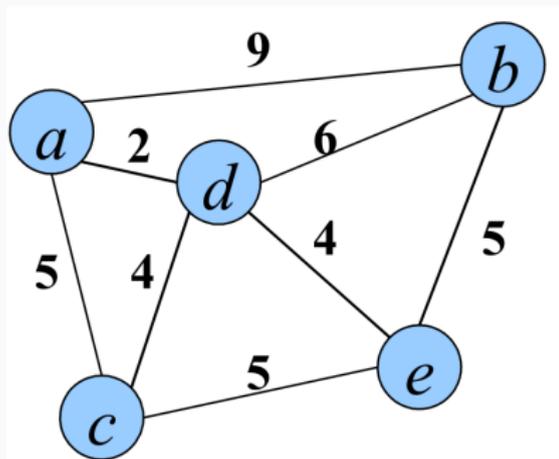


Figure 23: Le graphe de départ.

On part de (*a*, *d*) (poids le plus faible)

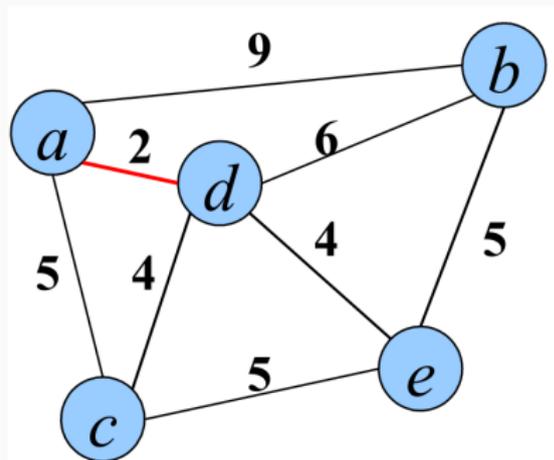


Figure 24: Les sommets *a*, *d* sont couverts.

Algorithme de Kruskal: exemple

On continue avec (c, d)

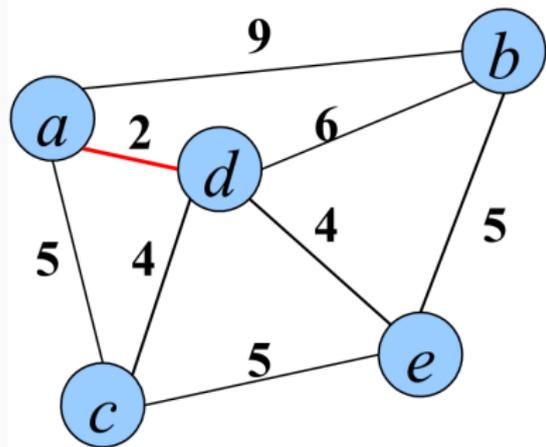


Figure 25: On aurait pu choisir (d, e) aussi.

Résultat

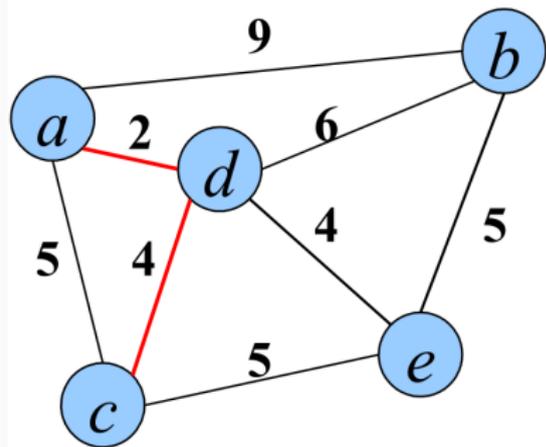


Figure 26: Les sommets a, d, c sont couverts.

Algorithme de Kruskal: exemple

On continue avec (d, e)

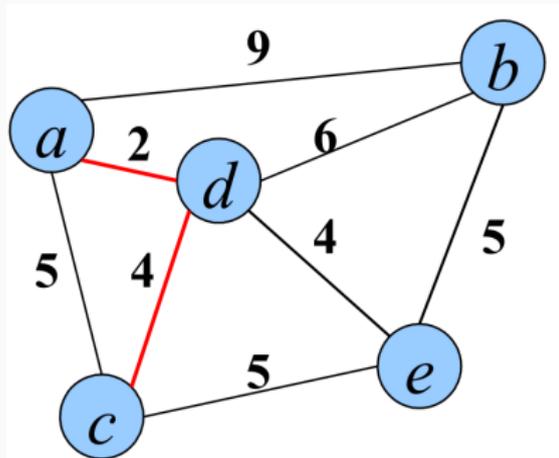


Figure 27: Le poids de (d, e) est le plus bas.

Résultat

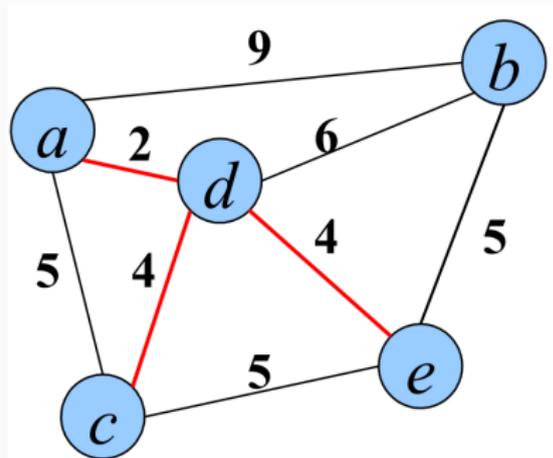


Figure 28: Les sommets a , d , c , e sont couverts.

Algorithme de Kruskal: exemple

On continue avec (b, e)

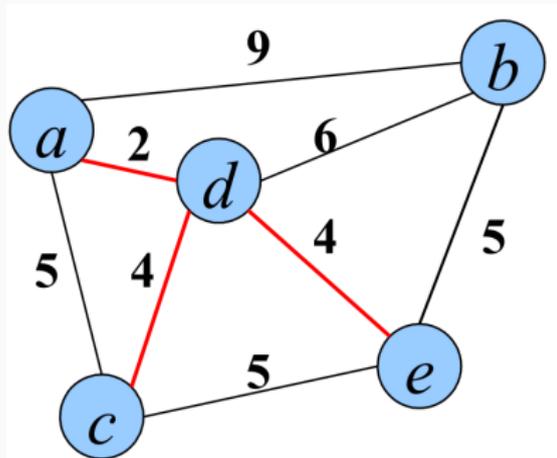


Figure 29: Le poids de (b, e) est le plus bas.

Résultat

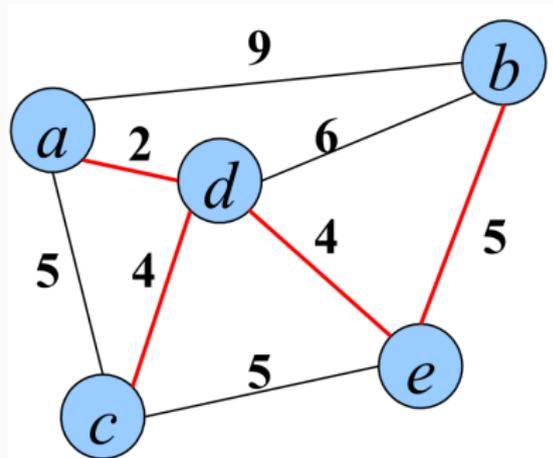


Figure 30: Les sommets a, d, c, e, b sont couverts.

Algorithme de Kruskal: exemple

Mais pourquoi pas (c, e) ?

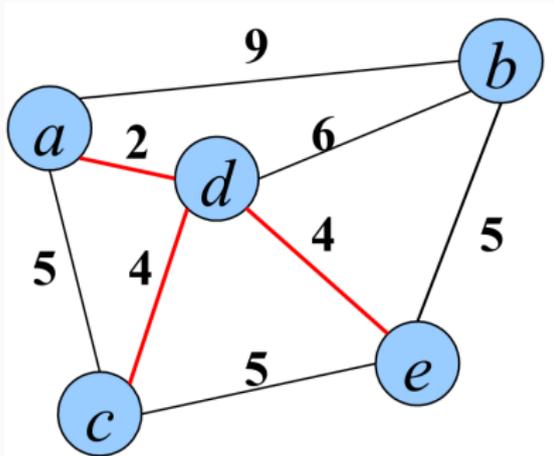


Figure 31: Le poids de (b, e) ou (a, c) est le même.

Résultat: un cycle

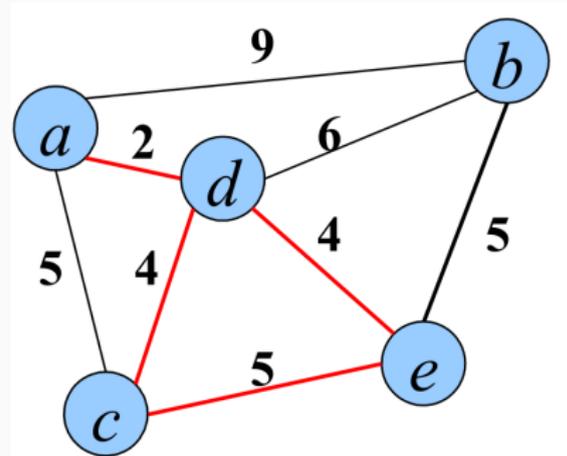


Figure 32: Les sommets a, d, c, e sont couverts.

- Comment faire pour empêcher l'ajout de (c, e) ou (a, c) ?

Algorithme de Kruskal: exemple

Mais pourquoi pas (c, e) ?

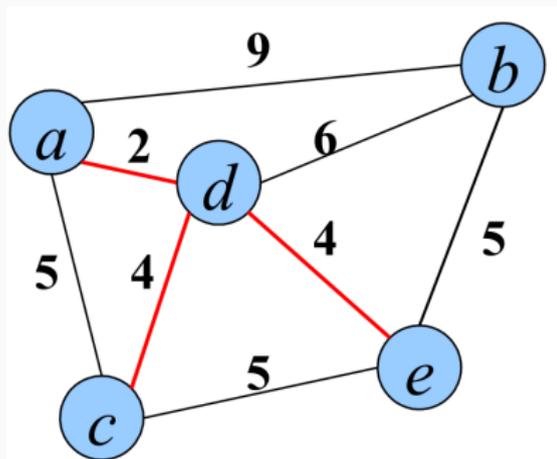


Figure 31: Le poids de (b, e) ou (a, c) est le même.

Résultat: un cycle

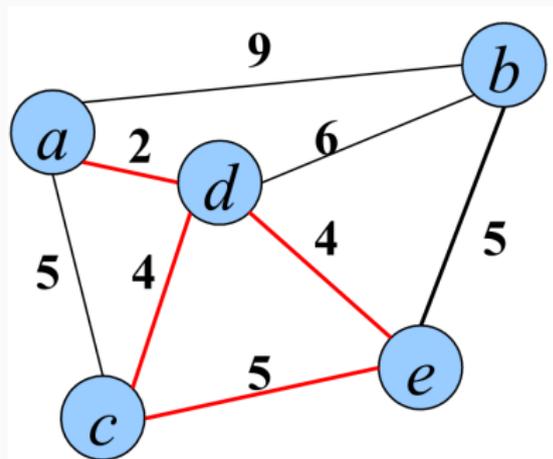


Figure 32: Les sommets a, d, c, e sont couverts.

- Comment faire pour empêcher l'ajout de (c, e) ou (a, c) ?
- Si les deux sommets sont déjà couverts nous sommes sauvés (presque)!

Algorithme de Kruskal

L'initialisation

- Créer un ensemble de sommets pour chaque de sommet du graphe (V_1, V_2, \dots) :
 - $V_1 = \{v_1\}, V_2 = \{v_2\}, \dots$
 - S'il y a n sommets, il y a n V_i .
- Initialiser l'ensemble A des arêtes "sûres" constituant l'arbre couvrant minimal, $A = \emptyset$.
- Initialiser l'ensemble des sommets couverts $F = \emptyset$.
- Trier les arêtes par poids croissant dans l'ensemble E .

Mise à jour

- Tant qu'il reste plus d'un V_i :
 - Pour $(u, v) \in E$ à poids minimal:
 - Retirer (u, v) de E .
 - Si $u \in V_i$ et $v \in V_j$ avec $V_i \cap V_j = \emptyset$:
 - Ajouter (u, v) à A ;
 - Fusionner V_i et V_j dans F .

Algorithme de Kruskal: exemple

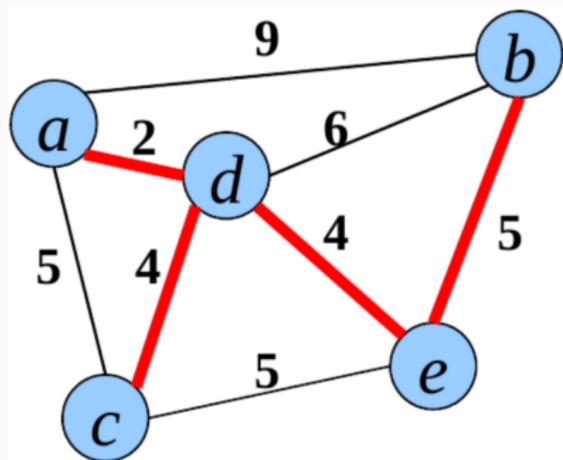
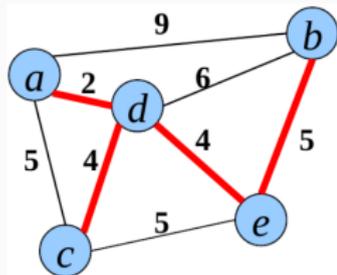


Figure 33: Couvrir cet arbre bon sang!

Algorithme de Kruskal: solution



$$E = \{ \overset{2}{\cancel{(a,d)}}, \overset{4}{\cancel{(c,d)}}, \overset{4}{\cancel{(d,e)}}, \overset{5}{\cancel{(a,c)}}, \overset{5}{\cancel{(b,e)}}, \overset{5}{\cancel{(c,e)}}, \overset{6}{(b,d)}, \overset{9}{(a,b)} \}$$

Forest

$\{\overline{a}, \overline{b}, \overline{c}, \overline{d}, \overline{e}\}$

$\{\overline{a,d}, \overline{b}, \overline{c}, \overline{e}\}$

$\{\overline{a,d,c}, \overline{b}, \overline{e}\}$

$\{a,d,c,e\}, \overline{b}$

$\{a,d,c,e,b\}$

A

\emptyset

$\{\overline{(a,d)}\}$

$\{(a,d), \overline{(c,d)}\}$

$\{(a,d), (c,d), \overline{(d,e)}\}$

$\{(a,d), (c,d), (d,e), (b,e)\}$

Figure 34: La solution!

Algorithme de Kruskal: exercice

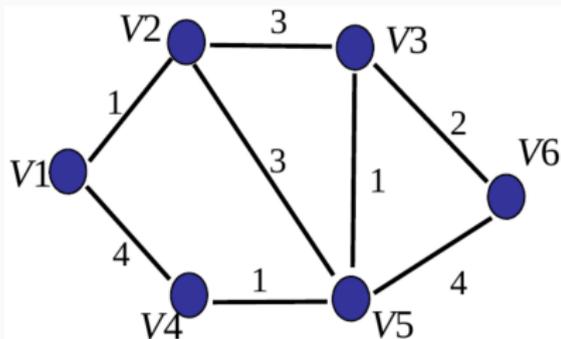


Figure 35: Couvrir cet arbre bon sang!

Algorithme de Kruskal: solution

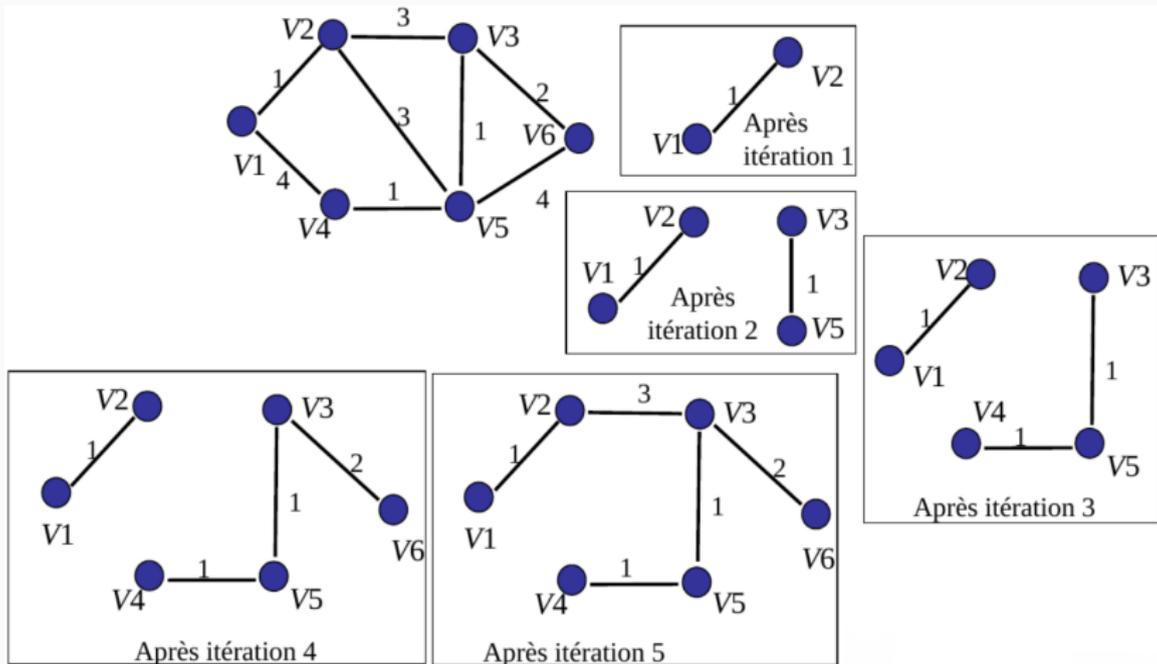


Figure 36: La solution!