TD 6 : 1

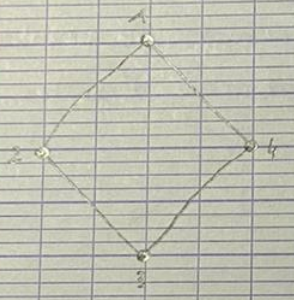
I. Combien d'arbres couvrants ?

Graphe non orienté, peut être pondéré (avec des poids)

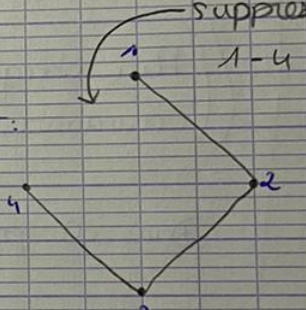
Un arbre est un graphe connexe sans cycle et un arbre couvrant d'un graphe est un sous graphe qui contient tous les sommets du graphe.

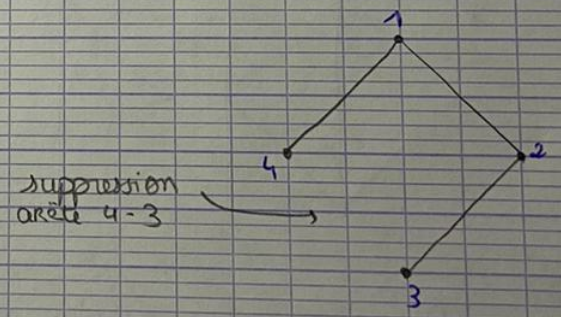
Quand on fait un parcours d'un graphe on induit un sous graphe

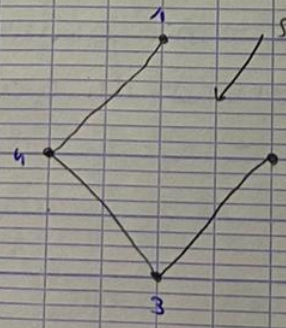
$G = (V, E)$. $|V| = N$, l'arbre couvrant : $(N-1)$ arêtes



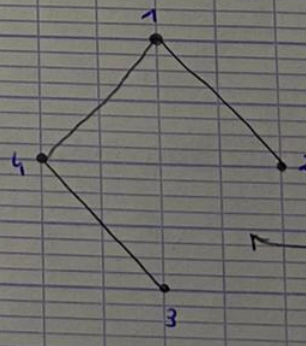
Arbre couvrant :

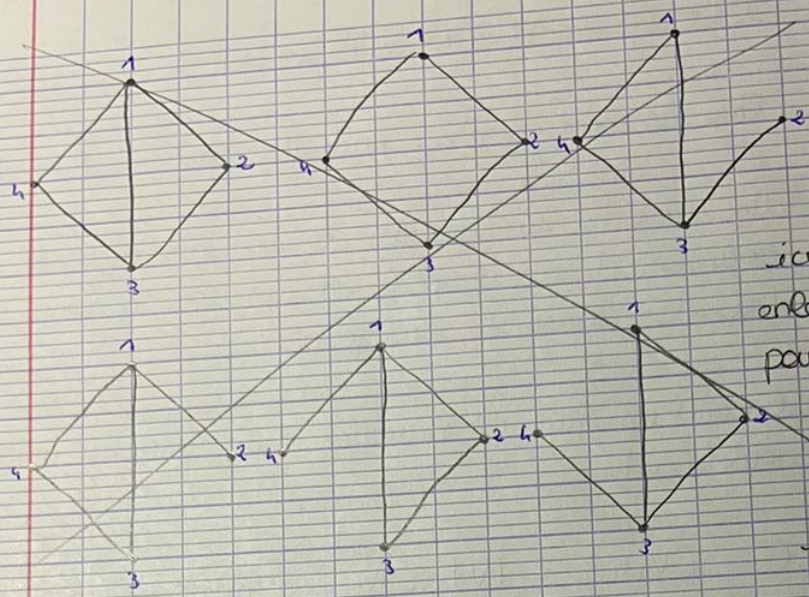






et

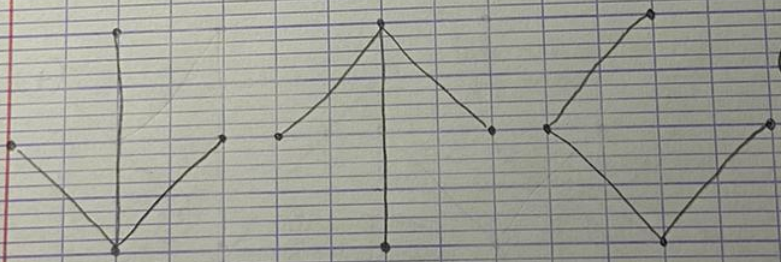




ici, on doit d'abord enlever une arête pour "enlever" le cycle
sinon ce n'est pas un arbre.

((Pour chaque graphe, pour trouver le nombre d'arbre couvrants, on regarde le nombre d'arêtes.))

Dans le premier : C_4^1



(+ cf premier graphe)

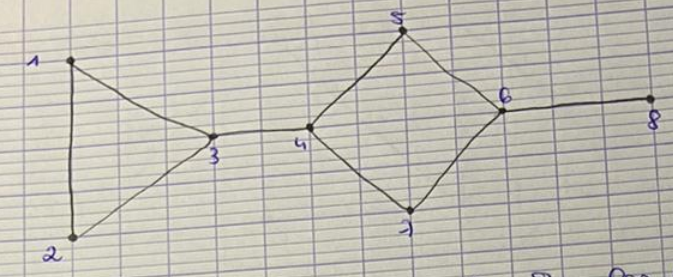
15

Attention : les arrêtes ne doivent pas être déconnectantes
↓
enlevées → ne pas isoler une partie / ou un sommet du graphe.

Deuxième graphe : $C_5^2 - 2$

TD 6 2.

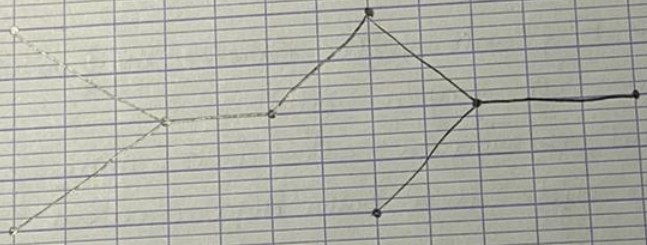
Autre exemple

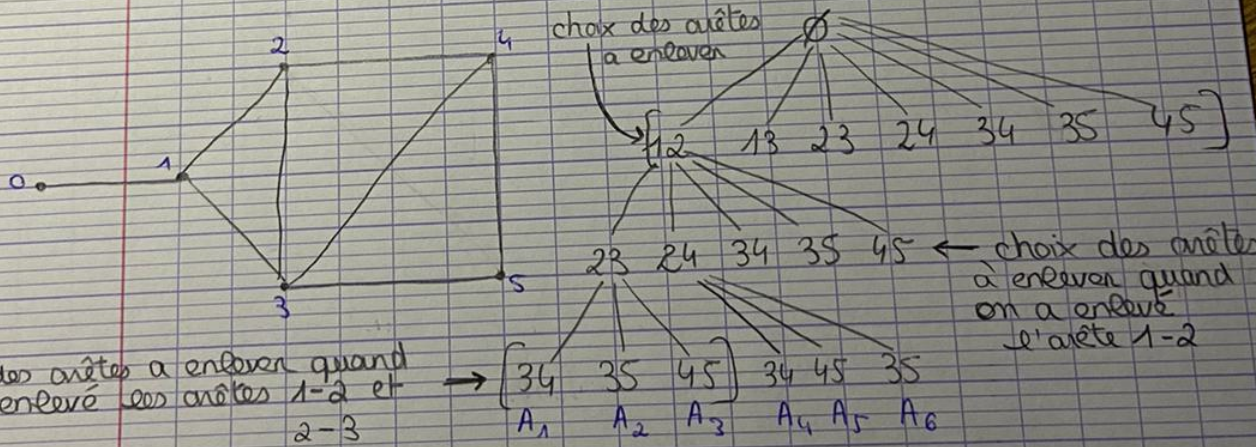


- Si je supprime 3-4 ou 6-8, les deux arêtes sont déconnectantes → on ne peut pas les enlever.

- Si on prend deux arêtes du même cycle, impossible. Donc, on prend une arête de chaque cycle ici.

Par exemple :



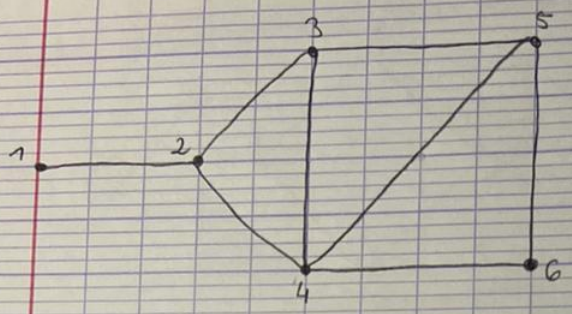


choix des arêtes à enlever quand on a enlevé les arêtes 1-2 et 2-3

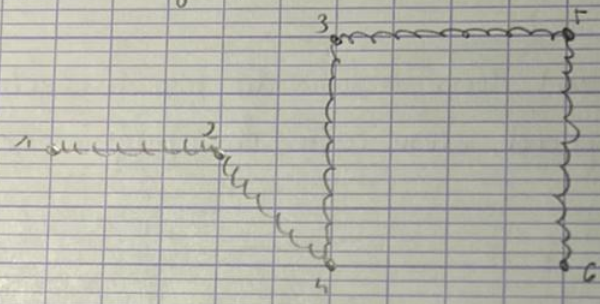
← choix des arêtes à enlever quand on a enlevé l'arête 1-2

A chaque fois que je descends je regarde le nombre "d'isme" et je liste les autres

↑ isme : arête déconnectantes, on ne peut pas les enlever.



je pars du graphe avec que les sommets

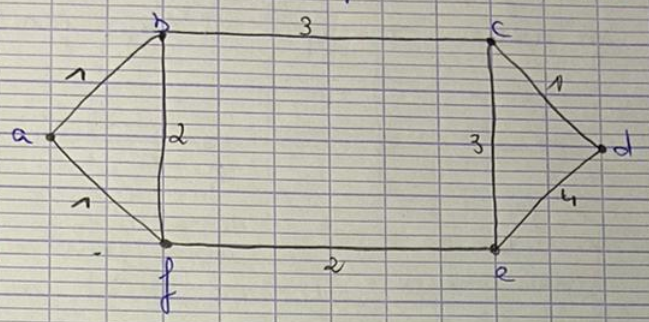


(un exemple d'arbre couvrants)

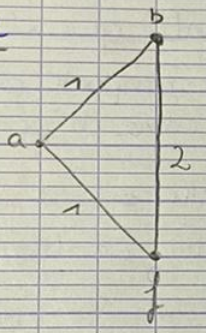
- Soit j'enlève les arêtes sauf si elles déconnectent avec le graphe induit.
- Soit je rajoute les arêtes mais pas celles qui forment un cycle avec celles déjà ajoutées

II Arbre couvrant de poids minimum

Ex :

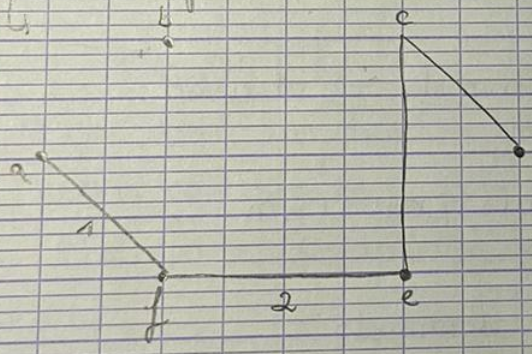


S = a

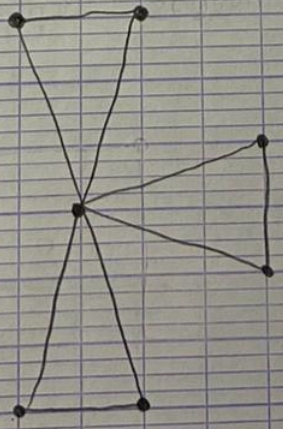


En appliquant l'algo dans ce graphe
→ problème, je tombe sur un cycle

Si je fais un autre choix !

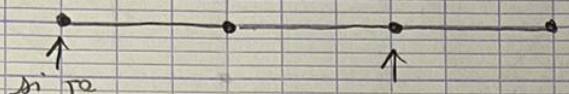


En appliquant l'algo dans ce graphe
Problème : b.



Quelque soit le sommet de départ, impossible de faire fonctionner l'algo
→ cycle

Si mon graphe de départ est une chaine cela ne m'assure pas que l'algo fonctionne



Le seul graphe qui garanti que l'algo marche :
→ cycle à n sommets, peu importe par quel sommet l'on commence, on obtiendra un arbre de "poid min".

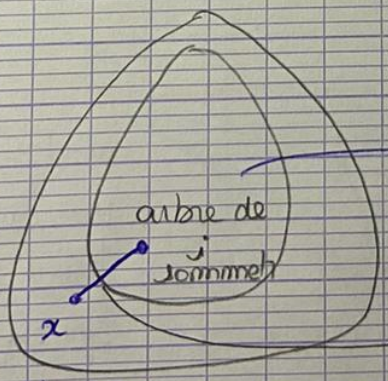
Prouve par récurrence : (vu en cours, H-S)
⇒ arbre de $n \geq 1$ sommets contient $n-1$ arêtes

• 1 sommet → 0 arête

•——• 2 sommets → 1 arête

pour $i \geq 1$, vraie pour $i = n$

arbre de $i+1$ sommets

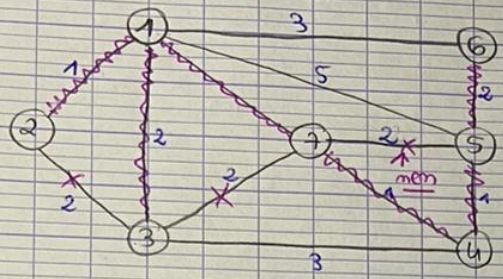


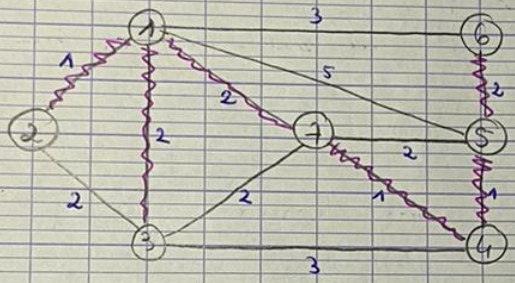
par hypothèse de récurrence
→ je sais que cet arbre contient $i-1$ arêtes

$+1$ arête → i

pour $i+1$ sommets → i arêtes.

III C'est parti pour Prim et Kruskal

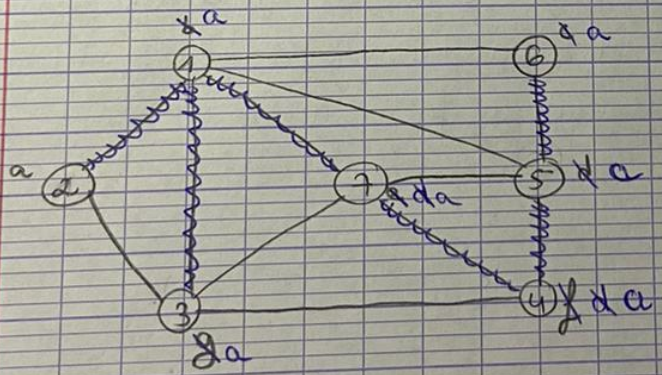




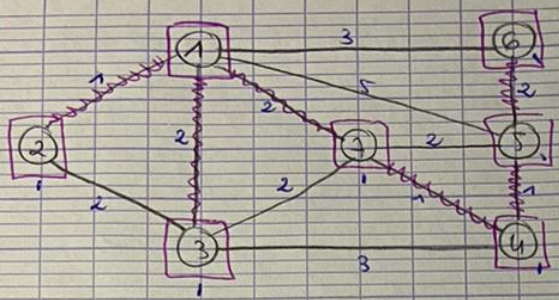
12 ; 54, 74 | 13, 17, 56, 57, 23, 37 | 16, 34 | 15

↳ on fait un tri en fonction du poids

Dans certains cas cela changera le choix du chemin → on ne prends pas les arêtes si elles forment un cycle



On considère chaque sommet comme un arbre ("étiquetage") on place les arêtes qui relient deux arbres diff entre eux et on étiquette avec le nom du plus petit par exemple.



Prim : sommets et arêtes
On prend un sommet s, on marque un sommet s et tant qu'il existe un sommet x non marqué voisin de y marqué
→ $[x, y]$ poid min
→ marque $[x, y)$
→ marque x.

ici, on ne parle pas de "ne pas choisir une arête qui fait un cycle avec celles choisies auparavant" puisque le marquage l'induit : pour faire un cycle on doit revenir sur un sommet marqué.

→ ce qui "coute" c'est le choix des arêtes de même poids, ou trouver le plus petit choix
→ cela revient à faire un tri.