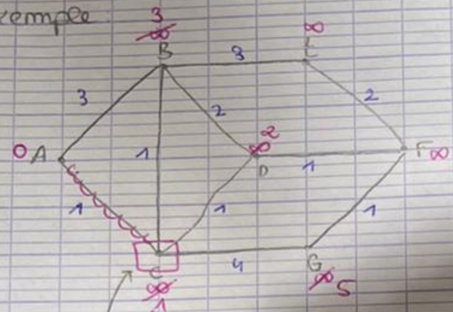
TD 7 : 1

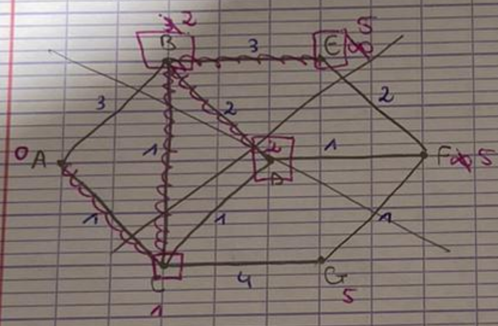
I. Commençons sur un petit graphe

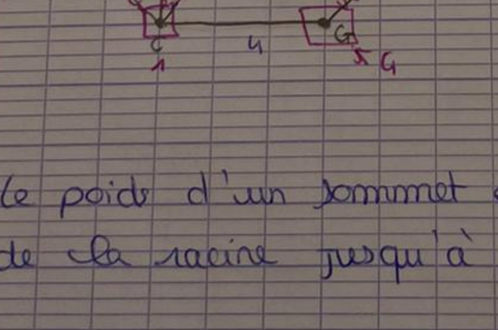
Exemple :



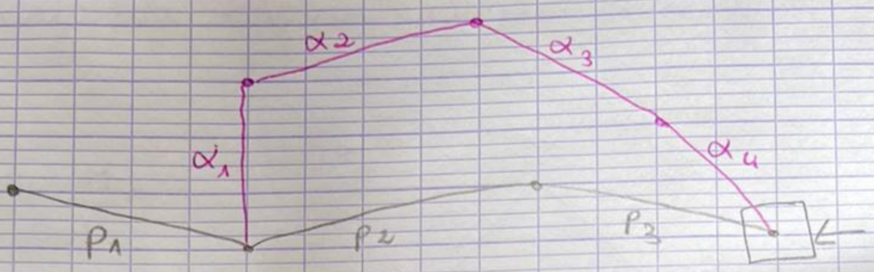
- Marquer la racine
- Arête, sommet avec valeur

choix entre B ou C
poids C < poids B
on choisit de marquer C





le poids d'un sommet est le poids de la chaine de la racine jusqu'à ce sommet

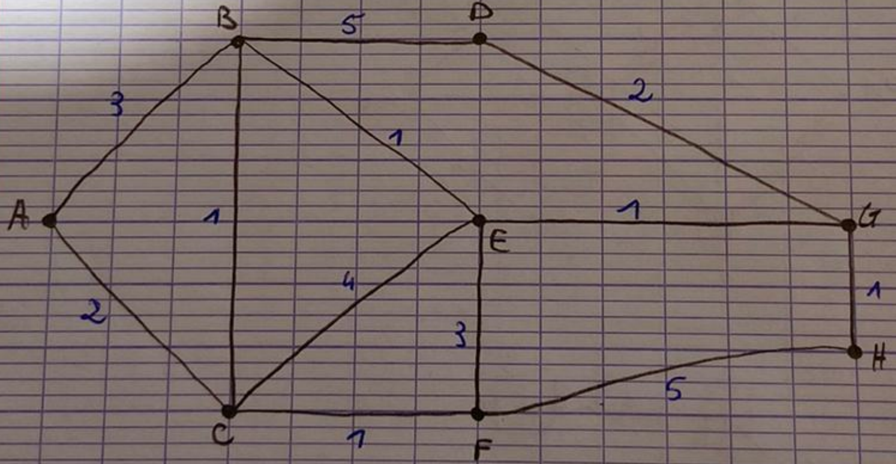


$$P_1 + \alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 < P_1 + P_2 + P_3$$

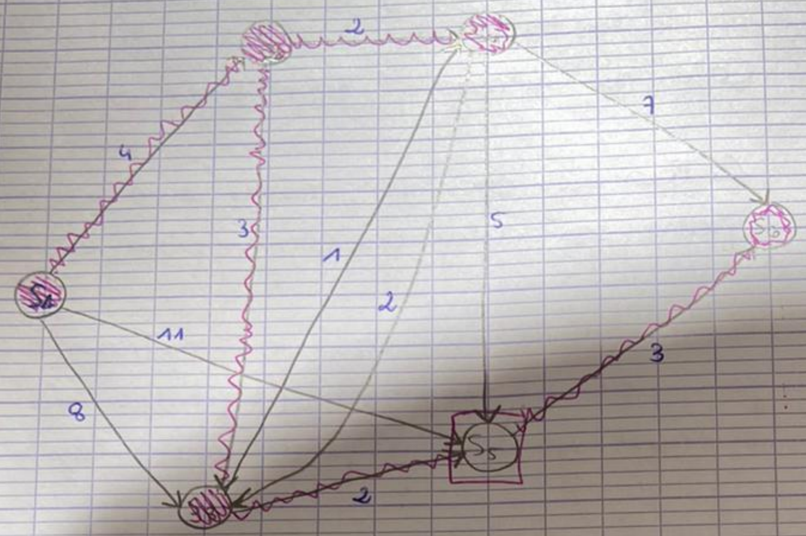
le sommet (←) a le poids le plus minimum

→ si on peut mieux faire on choisit de le faire dès le début.

↳ Si on a une meilleure chaine elle aurait déjà été marqué



I. (suite)



Sommets	S_1	S_2	S_3	S_4	S_5	S_6
d	0	4	8	∞	11	∞
père	-	S_1	S_1	-	S_1	-

$T = \{S_1, S_2\}$

Sommets	S_1	S_2	S_3	S_4	S_5	S_6
d	0	4	7	6	11	∞
père	-	S_1	S_2	S_2	S_1	-

$T = \{S_1, S_2, S_4\}$

Sommets	S_1	S_2	S_3	S_4	S_5	S_6
d	0	4	7	6	11	13
père	-	S_1	S_2	S_2	S_1	S_4

$T = \{S_1, S_2, S_4, S_3\}$

Sommets	S_1	S_2	S_3	S_4	S_5	S_6
d	0	4	7	6	9	12
père	-	S_1	S_2	S_2	S_3	S_5

$T = \{S_1, S_2, S_4, S_3, S_5, S_6\}$

v ; Min poids (w) + poids (w, v)
w tq $w \in \Gamma_G^-(v)$
(w marqué)

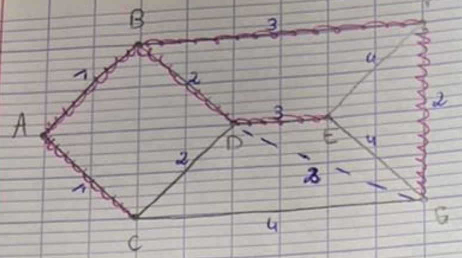
Coût de l'algo : $\Theta(n^2)$ au pire

⚠ Tous les chemins, même sans partir de l'origine passe par l'arborescence pour être minimum.

Pour réduire le coût on ne passe qu'une seule fois par un sommet. (djikstra le prend en compte en appliquant l'algo de marquage)

Est ce que je peux utiliser l'algo pour trouver un chemin de poids max, avec le chemin qui ne passe qu'une seule fois par un sommet.

TD 6 . 5



Tri

AB, AC (1), BD, CD, FG (2), BF, DE (3), CG, EF, EG (4)

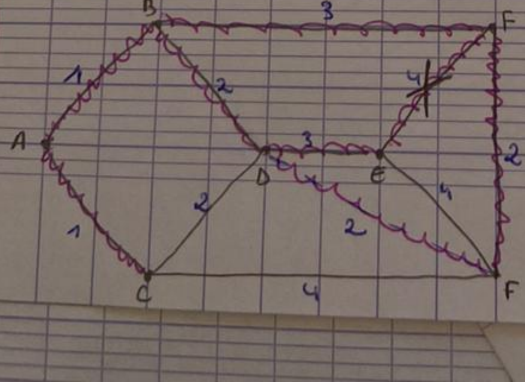
forme un cycle

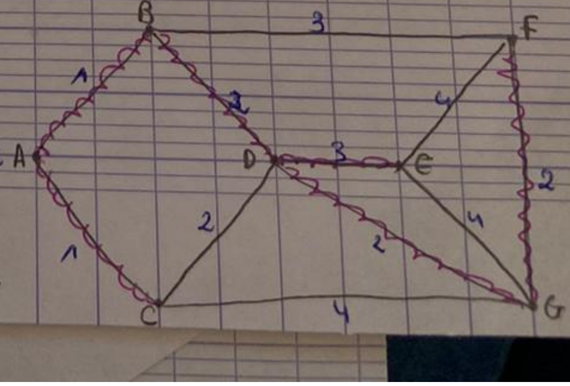
1) On effectue un tri en fonct° du poids des arêtes
2) On parcourt le tri en marquant les arêtes, on peut réétiqueter les sommets qui appartiennent au même sommet (→ permet d'éviter de prendre les arêtes qui forment un cycle)

Si une nouvelle arête est ajoutée : (ici DG)

- On place l'arête dans le tri en fonction de son poids, on supprime les autres arêtes (de poids sup) et on recommence à partir de la nouvelle arête
  → Plus court mais pas le plus optimal

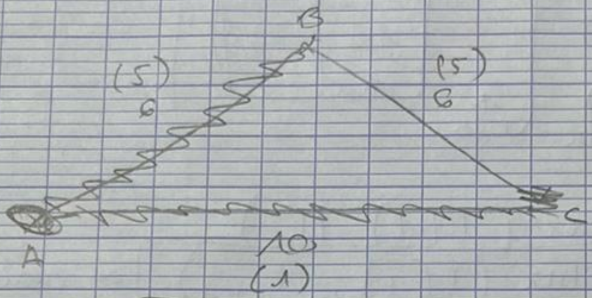
- On considère le même arbre et la nouvelle arête (la nouvelle arête forme un cycle dans l'arbre). On regarde le cycle et on supprime l'arête de poids max.





TD 7.

	S1	S2	S3
d	0	5	1
père	-	S1	S1



III

S	A	B	C	D	E	F	H
0	5	2	3	10	7	7	12
-	B	S	S	A	A	B	E

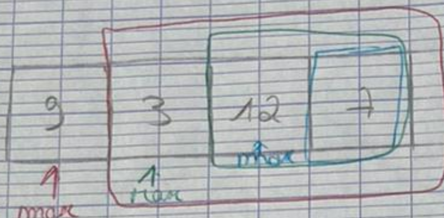
Plusieurs chemins min possible (poids égaux sur les chemins)

A s(5)
B s(2)
C s(3)

A s(5), c(5), B(6) 5 on marque AC et AS
E B(6)
F B(7), C(7)

Arbre obtenu DAG.
On cherche à représenter chaque chemin min dans l'arbre. A chaque fois que deux chemins ont le même poids on marque les deux arêtes et les sommets.

Prendre le chemin à l'envers pour obtenir seulement les chemins min.
Partir de M et faire un parcours en largeur



$M_1 = Max\,(T[1], M_2)$

$\boxed{M_i = Max\,(T[i], M_{i+1})}$ ← vrai quand $i < N$
et $M_N = T[N]$

Resultat final M_1 :

Sac à dos : P : poid Max.

m_i pierres précieuses ($1 \le i \le k$)

V_i : valeur de la pierre

P_i : poids de la pierre.

$$\max \sum_{i=1}^{k} m_i \times v_i \;/\; \sum_{i=1}^{k} m_i \times P_i \le P$$

poids du sac $\xrightarrow{\pi}$ 0, 1, ..., P

I

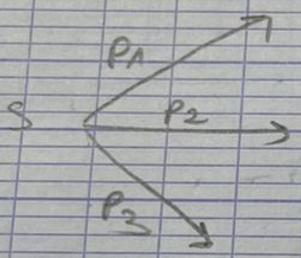
1
1 2
1 2 3
1 2 3 4
⋮
1 ..... k

$$S(P,k) \longrightarrow Max \begin{cases} S(P-1, k) \\ \max\limits_{1<J<H} S(P - J - P_k) \end{cases}$$

$$S(P, k) = \max_{0 \leq J \leq d} J - v_k + S(P - J - P_k, k - 1)$$

$$S(P_1, 1) = v_1$$

$$P_k = \left\lfloor \frac{P}{P_k} \right\rfloor = d$$



3 Le chemin le plus court pour aller d'un point à un autre, c'est encore de ne pas y aller (exercice partiel 2019)

Soit un graphe $G = (V,A)$ orienté et pondéré avec des poids strictement positifs $w(u,v)$ sur chaque arc (u,v) et un noeud source s. L'algorithme de Dijkstra calcule une valeur D_i pour chaque noeud i correspondant à la valeur du plus court chemin de s à i. On souhaite ajouter à cet algorithme de Dijkstra (rappelé ci-dessous) le compte du nombre de plus courts chemins de s vers chaque noeud i. Il s'agit donc de compléter l'algorithme de Dijkstra pour y ajouter le calcul de N_i qui doit contenir le nombre de plus courts chemins de s à i.

- Initialisations :
  - $T = \{s\}$; $D_s = 0$;
  - $\forall i \neq s$, si l'arc (s,i) existe alors $D_i = w(s,i)$ (∞ sinon)
- Boucle principale : Tant que $T \neq V$ faire
  - Trouver un noeud t de $V - T$ tq $D_t = Min(D_i, i \in V - T)$,
  - $T = T \cup \{t\}$
  - $\forall k \in \Gamma_t^+$, si $(D_k > D_t + w(t,k))$ alors $D_k = D_t + w(t,k)$

1. Réécrire l'algorithme de Dijkstra en y insérant le calcul de ces valeurs N_i.
2. L'appliquer sur le graphe ci-dessous. Vous détaillerez bien les différentes itérations et comment D et N évoluent.
3. Finalement, quels sont les différents plus courts chemins pour aller de s à H ?