

CC 2017 - CORRIGÉ


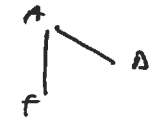
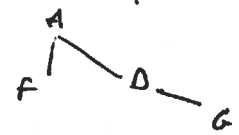
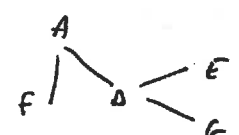
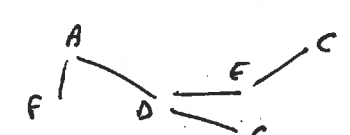
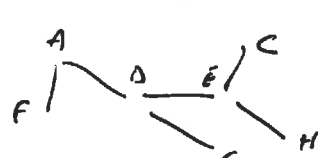
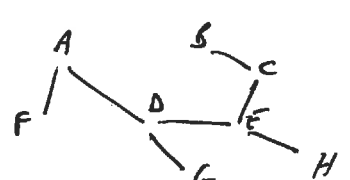
Exercice 1

$$1. \begin{matrix} & A & B & C & D & \dots \\ \begin{matrix} A \\ B \\ C \\ D \\ \vdots \end{matrix} & \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots \end{pmatrix} \end{matrix}$$

$$\text{ou} \begin{matrix} & A & B & C & D & \dots \\ \begin{matrix} A \\ B \\ C \\ D \\ \vdots \end{matrix} & \begin{pmatrix} 0 & 0 & 0 & 2 \\ 0 & 0 & 3 & 4 \\ 0 & 3 & 0 & 0 \\ 2 & 4 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots \end{pmatrix} \end{matrix}$$

(à compléter sur votre copie)

2. On part de T restreint à A

Étape	Arêtes considérées	Arête choisie	Structure courante
1.	(AD) (AF)	(AF)	
2.	(AD) (FD)	(AD)	
3.	(DB) (DE) (DG)	(DG)	
4.	(DB) (DE) (GE)	(DE)	
5.	(DB) (BE) (EC) (EH)	(EC)	
6.	(DB) (BE) (BC) (EH)	(EH)	
7.	(DB) (BE) (BC)	(BC)	

On obtient un poids total de 12.

Vous pouvez avoir trouvé un autre arbre, mais de même poids

Exercice 2

1.

<u>Etape</u>	<u>Propositions faites</u>	<u>Distance validée</u>
1.	$(a, 8) (b, 5)$	$(b, 5)$
2.	$(a, 8) (c, 11) (e, 8)$	$(a, 8), (e, 8)$
3.	$(p_1, 17) (c, 11) (p_2, 13) (p_3, 14)$	$(c, 11)$
4.	$(p_1, 17) (p_2, 13) (p_3, 14) (d, 15)$	$(p_2, 13)$
5.	$(p_1, 17) (p_3, 14) (d, 15)$	$(p_3, 14)$
6.	$(d, 15) (p_1, 17)$	$(d, 15)$
7.	$(p_1, 17)$	$(p_1, 17)$

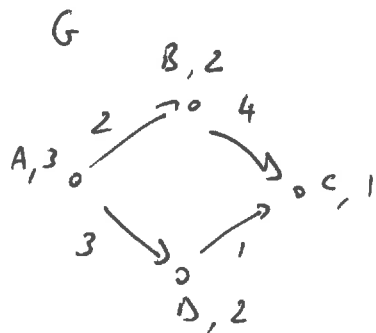
On obtient un coût minimal de 17 pour p_1 , 13 pour p_2 , 14 pour p_3 .

2. On considère la modification suivante de l'instance G avec des poids sur les sommets :

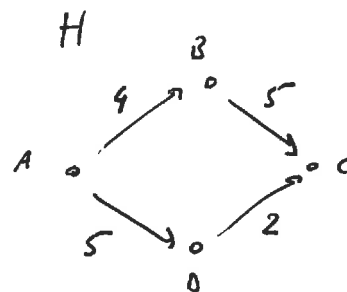
On définit H avec les mêmes arêtes que G et tel que $w_H(u, v) = w_G(u, v) + c_G(v)$

où w_H et w_G sont les poids des arêtes dans H et G et c_G est le poids du sommet dans G .

Exemple :



devient



Chaque chemin de G correspond à exactement un chemin de même poids dans H . Le résultat de Dijkstra dans H donne donc le meilleur chemin dans G .

Exercice 3

1. Considérons que la racine est bleue (sinon, il suffit d'échanger toutes les couleurs).

les sommets de niveau 1 sont voisins de la racine, donc rouges.

les sommets de niveau 2 sont voisins de sommets de niveau 1, donc bleus.

Par récurrence, on obtient que tous les sommets de niveau pair sont bleus et tous les sommets de niveau impair sont rouges.

2. On propose l'algorithme suivant :

- 1) On lance un DFS et on colore les sommets de niveau pair en bleu et ceux de niveau impair en rouge.

- 2) On regarde les arêtes une à une. Si elles relient toutes un sommet bleu à un sommet rouge, on répond OUI. Si au moins une arête relie deux rouges ou deux bleus, on répond NON.

Alors, on répond OUI si et seulement si G est 2-colorable

En effet, si on répond OUI, la coloration proposée est bien une coloration qui satisfait les contraintes.

Inversement, si G est 2-colorable, il existe une coloration qui satisfait les contraintes. Or, d'après 1., cette coloration peut être choisie bleue sur les nœuds pairs et rouge sur les autres. Cette coloration correspond donc à celle construite par l'algorithme, qui répondra bien OUI.

Les deux étapes sont de complexité $O(m)$, l'algorithme donc aussi.

NB : On peut améliorer l'algorithme en lançant le DFS

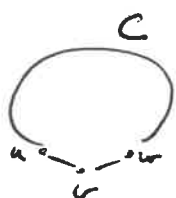
- et en le modifiant ainsi :
- on colore les sommets quand on les découvre (couleur inverse du père)
 - si à un moment, le sommet de la pile a un voisin déjà visité de la même couleur, on arrête et on répond NON
 - si le DFS va au bout sans s'interrompre, on répond OUI

Exercice 4

1. Trouver un circuit qui passe au moins une fois par chaque sommet et revient au point de départ, et ayant un poids minimal.

2. Il s'agit de déterminer les distances entre sommets, ce qui peut être fait avec l'algorithme de Dijkstra.

3. Soit C_v le cycle C privé de v



Comme C est un cycle hamiltonien, C_v passe par tous les sommets de H_v . Il s'agit donc d'un arbre couvrant

Par conséquent, $w(C_v) \geq w(T_v)$

Soit (u, v) et (v, w) les deux arêtes de C adjacentes à v .

Par définition de $c(v)$, $c(v) \leq w(u, v) + w(v, w)$

Finalement
$$w(C) = w(C_v) + w(u, v) + w(v, w) \geq w(T_v) + c(v)$$

4. Il faut lancer Dijkstra sur chaque sommet, ce qui coûte $\mathcal{O}(nm + m^2 \log n)$, puis chercher ~~les~~ arbres couvrants de poids minimale, ce qui coûte $\mathcal{O}(nm \log m)$, et finalement déterminer les $c(v)$, ce qui coûte $\mathcal{O}(nm)$.

Finalement le coût total est $\mathcal{O}(nm \log m)$