Deuxième contrôle continu (durée 1h30)

Documents autorisés : 2 feuilles A4 recto-verso.
Barème donné à titre indicatif. *Les réponses non justifiées ne seront pas notées.*

1 Chemins optimums (6 points)

Soit G un graphe à 7 sommets dont les arcs (x, y) sont pondérés par p_{xy} décrit ci-dessous :

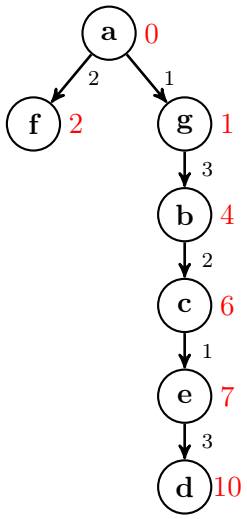
sommet x	a	b	c	d	e	f	g
$y \in \Gamma^+(x)$	b,f,g	c	d,e		d	e	b,c,e,f
p_{xy}	5,2,1	2	5,1		3	7	3,7,9,3

1. (**3 pts**) Calculez les chemins ai-minimaux dans le graphe G ci-dessous. Vous décrirez le déroulement de l'algorithme dans un tableau puis dessinerez l'arborescence des chemins ai-minimaux sur laquelle vous mentionnerez les distances de a à chaque sommet.

Corrigé

sommet choisi	a	b	c	d	e	f	g
init	0	∞	∞	∞	∞	∞	∞
a	/	5	∞	∞	∞	②	①
g	/	④	8	∞	10	2	/
f	/	4	8	∞	9	/	/
b	/	/	⑥	∞	9	/	/
c	/	/	/	11	⑦	/	/
e	/	/	/	⑩	/	/	/
d	/	/	/	/	/	/	/

On a entouré les poids des chemins ai-minimaux quand ils ont été obtenu pour la première fois. ça permet de savoir quel est le sommet qui a permis cette mise à jour. On obtient donc l'arborescence ci-contre.



2. (**1 pt**) Donnez le plus long chemin en termes de nombre d'arcs qui a une origine différente de a et qui est garanti minimal par le résultat de Dijkstra et grâce au principe de Bellman (tout chemin extrait d'un chemin minimal est minimal), donnez son poids.

Corrigé

Le plus long chemin *ai*-minimal en nombre d'arcs est *agbced*, tout chemin extrait de ce chemin est minimal, comme il ne doit pas avoir pour origine a c'est *gbced* qui est le plus long chemin en nombre d'arcs garanti minimal, il est de poids 9.

3. (**2 pts**) L'arborescence obtenue est-elle un arbre couvrant de poids minimum ?

Corrigé

Pour obtenir un acpm on utilise l'algo de Kruskal : on sélectionne successivement les arcs (par ordre croissant des poids à condition qu'ils ne créent pas de cycle) :

arcs	(a,g)	(c,e)	(a,f)	(b,c)	(e,d)	(g,b)	STOP $n-1$ acs
poids	1	1	2	2	3	3	

L'acpm a pour poids 12, c'est le même poids que l'arborescence précédente qui est connexe et sans cycle, donc l'arborescence précédente est bien un acpm. (Notons que cette arborescence est même identique à l'acpm, notons qu'elle est unique car l'ajout de gf de poids 3 est impossible puisqu'il créérait un cycle avec deux autres arcs de poids inférieurs)).

2 Dijkstra et poids quelconques (3 pts)

Nous savons que l'algorithme de Dijkstra ne peut fonctionner que sur un graphe valué positivement alors que Bellman-Kalaba peut lui, être utilisé avec des arcs valués de façon quelconque.

Soit $G = (X, U)$ un graphe valué par une pondération p quelconque sur chaque arc, on note p_{xy} la pondération associée à l'arc (x, y). On décide de faire la procédure suivante :

— On pose $\quad m = \min_{(x,y)\in U} p_{xy}$

— On construit $G' = (X, U)$ identique au graphe G mais pondéré par $p'_{xy} = p_{xy} - m$

Pour les 3 questions suivantes, si la réponse est oui faîtes la démonstration sinon donnez un contre-exemple.

1. (**1 pt**) Peut-on montrer que G' est pondéré avec une pondération positive ou nulle sur chaque arc ?

Corrigé

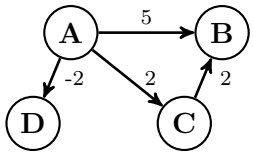
on note que $\forall (x, y) \in U, p_{xy} \geq m$ donc $\forall (x, y) \in U, p'_{xy} = p_{xy} - m \geq 0$. Donc G' est pondéré avec une pondération positive ou nulle sur chaque arc.

2. (**1 pt**) Peut-on montrer que tout chemin minimal de G' est un chemin minimal de G ?

Corrigé

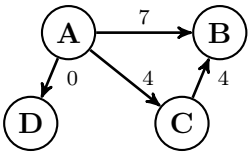
Non car un chemin de n arcs aura un poids augmenté (quand m est négatif) de $-n \times m$. Donc si le chemin minimal possède beaucoup d'arcs il sera beaucoup augmenté. Voici un contre-exemple :

G



plus court chemin de A à B : ACB

G' $(m = -2)$



plus court chemin de A à B : AB

3. (**1 pt**) Peut-on montrer que Dijkstra appliqué à G' donnera une arborescence de chemins qui sont minimaux aussi dans G ?

Corrigé

Non, Dijkstra calcule les plus court chemins dans un graphe pondéré positivement, donc il trouvera bien les plus court chemins de G', mais d'après la question précédente un plus court chemin de G' n'est pas forcément un plus court chemin de G.

3 Problème d'ordonnancement (5 points)

Lors de la construction d'une maison, on distingue 12 travaux distincts :

Tâche	Libellé de la tâche	Durée	Tâches à terminer avant
A	gros oeuvre	8	/
B	charpentes	2	A
C	toiture	1	B, A
D	plomberie	4	A
E	électricité	2	A
F	ravalement	3	A,B, C,D
G	fenêtre	1	A,B
H	aménagements extérieurs	1	C,D,E
I	plâtres	2	A, C,D,E,G
J	sols	2	D,E,G,I
K	peintures	3	I
L	emménagement	1	toutes les tâches

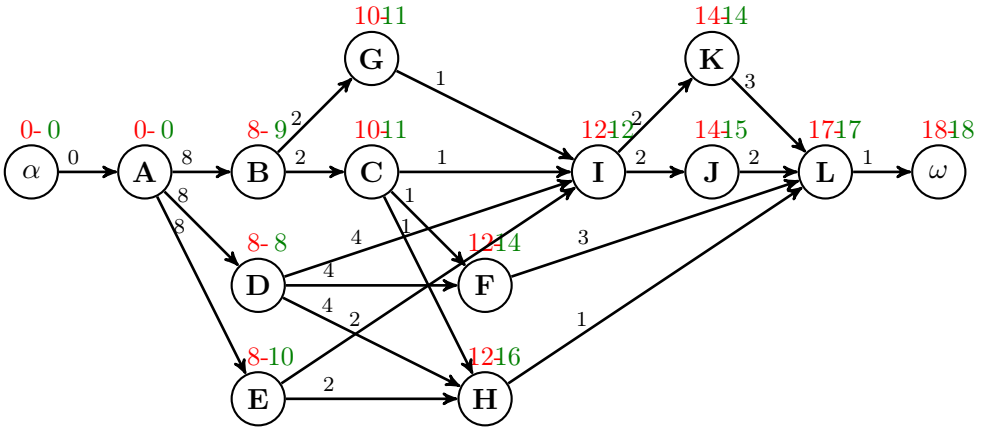
1. (**1 pt**) Y a-t-il des contraintes de précédence (données dans la dernière colonne du tableau) qui sont redondantes ? si oui donnez un exemple sinon prouvez-le.

Corrigé

oui par exemple C est après B et A or B doit être après A, il suffit donc de dire C après B.

2. (**2 pts**) Modélisez la situation à l'aide d'un graphe et déterminer la durée minimale du projet.

Corrigé



Durée minimale du projet : 18.

3. (**2 pts**) Indiquez sur votre graphe les dates de début au plus tôt et au plus tard de chaque tâche permettant de garantir cette durée optimale.

Corrigé

Voir dessin.

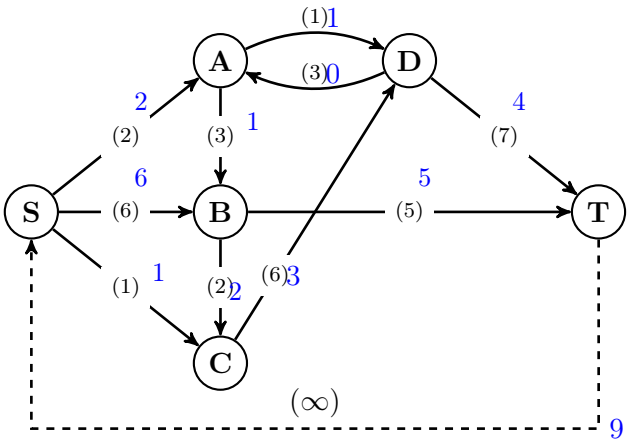
4 Problème de flots (6 points)

Le serveur S est connecté à la machine T par un réseau avec les noeuds A, B, C et D. Les capacités des débits maximums des connexions entre les noeuds sont données dans le tableau suivant (en Mbit/s). Les connexions sont orientées des sommets en ligne vers les sommets en colonne. Les cases vides signifient qu'il n'y a pas de connexion directe entre les noeuds. Les temps de passage dans les noeuds sont négligeables.

	A	B	C	D	T
S	2	6	1		
A		3		1	
B			2		5
C				6	
D	3				7

1. (**1 pt**) Dessinez le réseau. Est-ce un réseau de transport ?

Corrigé



oui en rajoutant l'arc de retour de (T) vers (S).

2. (**1 pt**) L'utilisateur de la machine T télécharge un très grand fichier du serveur S. On veut trouver quelle répartition des paquets sur les connexions permettra de maximiser le débit. Expliquer le lien avec la recherche d'un flot dans ce réseau. Quel type de flot recherche-t-on ?

Corrigé

On cherche à faire passer des informations de S à T de façon à ce qui entre en un noeud en sorte (pas de perte de paquet), on cherche donc un flot (vecteur qui vérifie la loi de Kirchhoff). D'autre part on ne peut pas avoir des paquets qui circulent à l'envers d'une liaison, ni un débit qui excède les capacités : on veut donc un flot compatible. Finalement on veut maximiser le débit c'est-à-dire qu'on veut qu'il arrive un maximum de paquets par seconde, on veut donc un vecteur qui maximise le nombre de données arrivant en T, donc un flot compatible de valeur maximale.

3. (**3 pts**) Trouvez un flot maximum en décrivant les chaînes augmentantes utilisées ainsi que les augmentations réalisées à chaque étape. Vous préciserez le nom de l'algorithme utilisé, vous prouverez que le flot est maximal.

Corrigé

On utilise l'algorithme de Ford-Fulkerson, on part du flot nul,

- on trouve une première chaîne augmentante SBT on peut augmenter de 5 sur cette chaine : $\varphi_0 = 5.[SBTS]$, flot de valeur 5.
- $SABCDT$ on peut augmenter de 2 : $\varphi_1 = \varphi_0 + 2.[SADTS]$ flot de valeur 7
- $SCDT$ on peut augmenter de 1 : $\varphi_2 = \varphi_1 + [SCDTS]$ flot de valeur 8.
- $SBADT$ on peut augmenter de 1 : $\varphi_3 = \varphi_2 + [SBADTS]$ flot de valeur 9
- on ne peut plus marquer que s le flot est donc max car on ne peut pas marquer t

4. (**1 pt**) Proposez une connexion dont l'augmentation de capacité permettrait d'augmenter le débit.

Corrigé

la connexion (sC)