UE Graphes – Épreuve 4 du contrôle continu 2012-2013
Éléments de correction

Les énoncés sont rappelés en italique

Exercice 1 (*4 points*)
Avant que le télégraphe n'existe (voire d'autres moyens encore plus modernes), les informations à délivrer se faisait via un réseau de relais entre lesquels des cavaliers véhiculaient les missives. Le temps mis par un cavalier pour relier deux relais voisins était connu et recensé. Pour l'organisation du 100ème anniversaire du système, le responsable parisien a voulu inviter tous les responsables de relais à Paris. Expliquez comment l'usage des graphes aurait pu l'aider à déterminer le temps nécessaire pour que toutes les invitations, envoyées via le réseau, arrivent à leurs destinataires.

N.B. : toute ressemblance avec un événement réel est fortuit.

Le problème de cet exercice peut se ramener à un problème d'algorithmique des graphes. Les graphes à considérer peuvent être valués ou non. L'algorithme que nous utiliserons peut s'adapter à ces deux types de graphe. Le choix peut se faire en fonction d'informations plus précises notamment sur le fait que le temps pour relier deux relais est le même ou non (par exemple, en milieu montagneux, les durées de trajets peuvent s'avérer différentes selon le sens du trajet). Nous travaillons dans la suite de cette réponse avec des graphes orientés permettant une modélisation plus générale (les graphes non orientés pouvant être modélisés par des graphes non orientés symétriques).

Le graphe que nous considérons ici modélise le réel de la manière suivante :
- sommets = relais et siège de la société à Paris,
- arcs = liaison d'un relais à un relais voisin,
- valuation d'un arc= temps de référence associé à une liaison.

Le problème considéré peut se résoudre en deux temps.

1. Calcul des plus courts chemins de Paris aux différents relais : cette partie peut être résolu grâce à l'algorithme de Dikskstra.
2. Conservation de la plus grande distance pour tous ces courts plus chemins : il s'agit simplement de récupérer la plus grande valeur dans une liste, le résultat de ce calcul fournissant le temps pour que toutes les missives arrivent dans tous les relais.

Remarque 1 : l'algorithme de Dijsktra permet en plus d'avoir un arbre des plus courts chemins permettant de déterminer les routes des missives.

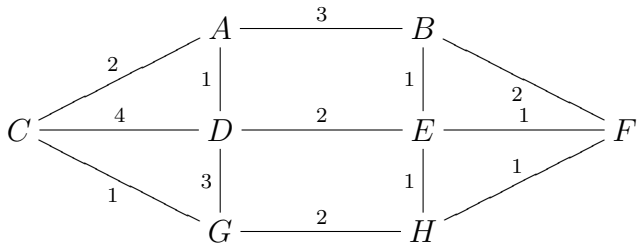
Remarque 2 : si on savait que le graphe du réseau était sans circuit, on pourrait utiliser l'algorithme PERT. Mais rien dans l'énoncé ne permet de faire cette hypothèse.

Exercice 2 (*4 points*)
Rappelez, pour chacun des algorithmes de Dijkstra, Prim, Kruskal et PERT, les hypothèses ou caractéristiques que doivent vérifier un graphe sur lequel l'algorithme est appliqué.

Il s'agit ici d'une question de cours. Voir les supports (notamment ceux qui étaient autorisés lors de l'examen) pour la réponse.

Exercice 3 (*12 points*)

Pour le graphe suivant, déterminez un arbre couvrant de poids minimal et un arbre des plus courts chemins issus de A.



Commençons par calculer un arbre couvrant de poids minimal. Cela peut être réalisé grâce à l'algorithme de Kruskal ou grâce à l'algorithme de Prim. Nous donnons les deux résolutions dans cette correction, mais l'examen n'en attendait qu'une (mais avec une claire identification de l'algorithme utilisé ainsi que des étapes de son application).

Application de l'algorithme de Kruskal

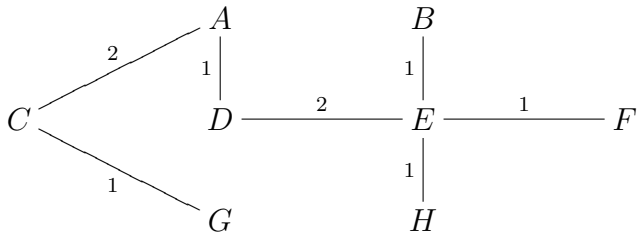
Étape 1 : on ordonne les différents arcs valués par ordre croissant de valuation (et, conformément aux directives du cours, par ordre alphabétique des arcs). : $(A, D, 1)$, $(B, E, 1)$, $(C, G, 1)$, $(E, F, 1)$, $(E, H, 1)$, $(F, H, 1)$, $(A, C, 2)$, $(B, F, 2)$, $(D, E, 2)$, $(G, H, 2)$, $(A, B, 3)$, $(D, G, 3)$, $(C, D, 4)$.

Étape 2 : initialisation du graphe en prenant comme ensemble des sommets, l'ensemble des sommets du graphe de travail : $\{A, B, C, D, E, F, G\}$.

Étape 3 : détermination des arcs du graphe résultat par parcours de la liste des arcs constituée à l'étape 1. Un arc est conservé s'il ne crée pas de cycle dans le graphe résultat. On s'arrête dès que le graphe résultat est connexe.

Liste des arcs du graphe dans l'ordre d'insertion : $(A, D, 1)$, $(B, E, 1)$, $(C, G, 1)$, $(E, F, 1)$, $(E, H, 1)$, $(A, C, 2)$, $(D, E, 2)$ (les arcs $(F, H, 1)$ et $(B, F, 2)$ sont considérés mais non conservés). L'algorithme s'arrête après insertion de $(D, E, 2)$.

Le graphe obtenu est :



Application de l'algorithme de Prim

Il faut choisir un sommet de départ à ajouter dans le graphe résultat. Prenons le sommet A. L'ensemble des sommets du graphe initial est alors partitionné en deux. On considère alors les arcs issus des sommets du graphe résultat de poids minimaux jusqu'à ce que tous les sommets soient considérés.

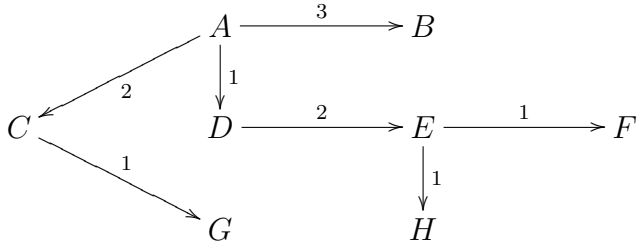
Les arcs (et sommets extrémités) sont ajoutés dans le graphe résultat dans l'ordre : $(A, D, 1)$, $(A, C, 2)$, $(C, G, 1)$, $(D, E, 2)$, $(E, B, 1)$, $(E, F, 1)$, $(E, H, 1)$.

On obtient le même arbre couvrant qu'avec Kruskal.

Considérons à présent l'algorithme de Dijkstra pour calculer un arbre des plus courts chemins à partir de A. Dans le tableau suivant, les "-" signifient que le sommet intitulé de la colonne est déjà marqué.

	A	B	C	D	E	F	G	H
init	∞	∞	∞	∞	∞	∞	∞	∞
marquage de A	0	3	2	1				
marquage de D	-		2	-	3		4	
marquage de C	-		-	-			3	
marquage de B	-	-	-	-	3	5		
marquage de E	-	-	-	-	-	4		4
marquage de G	-	-	-	-	-		-	4
marquage de F	-	-	-	-	-	-	-	4
marquage de H	-	-	-	-	-	-	-	-
distance à partir de A	0	3	2	1	3	4	3	4
Infos sur arbre couvrant	racine	A → B	D → C	A → D	D → E	E → F	C → G	E → H

On obtient l'arbre couvrant



qui peut aussi être dessiné