Travaux Dirigés d'algorithmique n°6

Cours d'Informatique de Deuxième Année

—DEUG Sciences S4-MI—

Chemin critique, ordonnancement

Lors de cette séance, nous cherchons à éditer les différents chemins critiques d'un sommet à un autre dans un graphe orienté. Cet algorithme permet de résoudre certains problèmes d'ordonnancement.

▸ **Exercice 1.** ***(Problème d'ordonnancement)*** *On considère les tâches suivantes intervenant dans la construction d'une maison :*

Nom	*Contenu*	*Durée (en semaines)*	*Tâches antérieurs*
A	*Travaux de maçonnerie*	*7*	-
B	*Charpente de la toiture*	*4*	*A*
C	*Choix des finitions*	*2*	-
D	*Installation sanitaire, électrique*	*6*	*A, C*
E	*Toiture*	*2*	*B*
F	*Fenêtres*	*1*	*C, D, E*
G	*Aménagement du jardin*	*1*	*D, E*
H	*Travaux de plafonnage*	*1*	*F*
I	*Façade*	*4*	*D, E*
J	*Mise en peinture, finitions*	*2*	*H*
K	*Emménagement*	*1*	*G, I, J*

Construire le graphe correspondant à l'ordonnancement de ces tâches, en ajoutant deux tâches fictives DT et FT représentant respectivement le début et la fin des travaux. Quelles sont les chemins les plus longs (en nombre de semaines) allant de DT à FT ? Quelle est la durée minimale des travaux ?

▸ **Exercice 2.** ***(Chemin critique)*** *On cherche la longueur d'un chemin critique entre un sommet X et un sommet Y notée $l(X,Y)$. On note Y_i les prédécesseurs de Y. On a la relation suivante :*

$$l(X,Y) = max_i(l(X,Y_i) + val(Y_i,Y))$$

où val est la valuation de l'arête entre les deux sommets arguments.

- *Cette remarque implique que l'on doit avoir traité tous les prédécesseurs d'un sommet S avant de traiter S. Écrire un algorithme qui numérote les sommets pour traiter les sommets "dans le bon ordre".*
- *Les sommets sont maintenant "correctement" numérotés. Écrire un algorithme formel de recherche d'un chemin critique entre le sommet numéro 1 et le sommet numéro nb_{sommet}.*

▸ **Exercice 3.**

- *Comment ajouter la valuation d'une arête à notre représentation d'un graphe par **FS** et **APS** ?*
- *Construire la file des prédécesseurs **FP** et l'adresse des premiers prédécesseurs **APP** à partir de **FS** et **APS**. Écrire la fonction correspondante de prototype* `void FSAPS2FPAPP(int *fs, int *aps, int tfs, int taps, int **fp, int **app)`. *Évaluer la complexité de cette fonction.*
- *Nous avons désormais accès facilement aux prédécesseurs d'un sommet. Écrire une fonction permettant d'éditer tous les chemins critiques d'un sommet à un autre.*