Algorithmique
TD n° 3 : Parcours

Michel Habib habib@irif.fr

I) Premières applications des parcours de graphes

1. Algorithme de vérification qu'un graphe est biparti.
2. Résolution de 2-sat. Système de clauses en logique des propositions à exactement deux variables par clause.
3. Calcul des composantes 2-arêtes connexes à l'aide d'un parcours.
4. Calcul du centre d'un arbre.

II) Arborescences

On considère une arborescence A finie. Les feuilles de A sont étiquetées par des éléments d'un ensemble X. A est représentée par la fonction père (tout élément sauf la racine admet un père unique), et l'on suppose en outre que chaque noeud de l'arborescence possède une variable qui contient le nombre de ses enfants. On suppose qu'un pointeur permet d'associer à chaque élément de X la feuille qui lui correspond dans A.
Etant donné $S \subseteq X$ écrire un algorithme qui vérifie qu'il existe un noeud $a_s \in A$ tel que S corresponde à l'ensemble des feuilles de la sous-arborescence $A(S)$ de A engendrée par a_s. Peut-on écrire un algorithme en $O(|A(S)|)$ lorsque l'arbre ne possède pas de noeuds internes n'ayant qu'un fils ?
Corrigé :
On fait un parcours en mettant au début les éléments de A dans OUVERTS.
A chaque sommet on associe t une variable Descendant qui compte l'ensemble des descendants de t qui appartiennent à A.
Au départ Descendant(x)=0 si $x \notin A$ et 0 sinon.
A la fin de l'algo, lorsque l'ensemble des OUVERTS est vide, on considère le dernier sommet y exploré.
Si Descendant(y)=|A| alors la réponse est OUI et NON dans le cas contraire.

III) 2-connexité

Montrer que pour un graphe non orienté G, les quatre conditions suivantes sont équivalentes :

1. G est 2-connexe
2. G n'admet pas de point d'articulation (i.e., x tel que $G - x$ non connexe)
3. $\forall x, y, z \in G$, il existe une chaîne de x à y passant par z
4. $\forall x, y, z \in G$, il existe une chaîne de x à y évitant z

IV) Parcours en largeur étendu

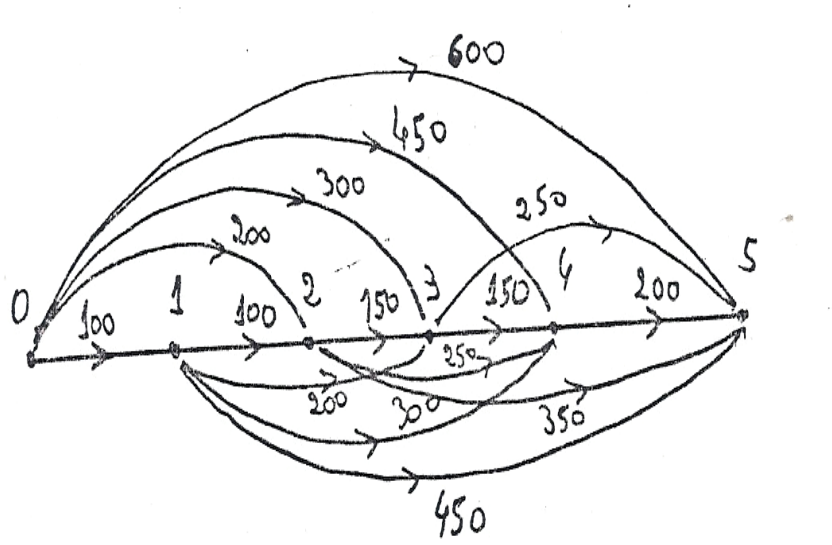
On considère un graphe $G = (X, E)$ connexe et à partir d'un sommet initial s on construit les niveaux $L_0 = \{s\}$, $L_1 = N(s)$, ... $L_i = \{$sommets à distance exactement i de $s\}$, ... Il est facile d'obtenir ces ensembles de sommets L_i à l'aide d'un simple parcours en largeur de G. Nous aimerions calculer $\forall i$ les sous-ensembles maximaux C_j de L_i vérifiant :
$\forall x, y \in C_j$ il existe une chaîne de x à y n'utilisant que des sommets des niveaux L_k, avec $k \geqslant i$.
Montrer que ces ensembles permettent de définir un arbre et proposer un algorithme efficace pour les calculer.

V) Modélisation

Une entreprise doit acquérir une nouvelle machine pour remplir un contrat de 5 ans. Le coût estimé d'une machine neuve est de 300 K euros aux années 0 et 1, de 350 K euros aux années 2 et 3, de 400 K euros à l'année 4. Les prix de revente et les coûts de maintenance (cumulés) sont décrits dans le tableau suivant :

Nombre d'années d'utilisation	Prix de revente	Coût maintenance
1	200	0
2	150	50
3	100	100
4	50	200
5	0	300

Déterminer la politique optimale d'acquisition de cette machine sur les 5 ans.



Dans ce modèle, dont la pertinence économique ne sera pas discutée ici, une politique optimale au sens de moindre coût, pour avoir pendant cinq années une machine en fonctionnement sera donnée par un chemin de valeur minimale allant de 1 à 5.