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) 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.

III) 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 ?

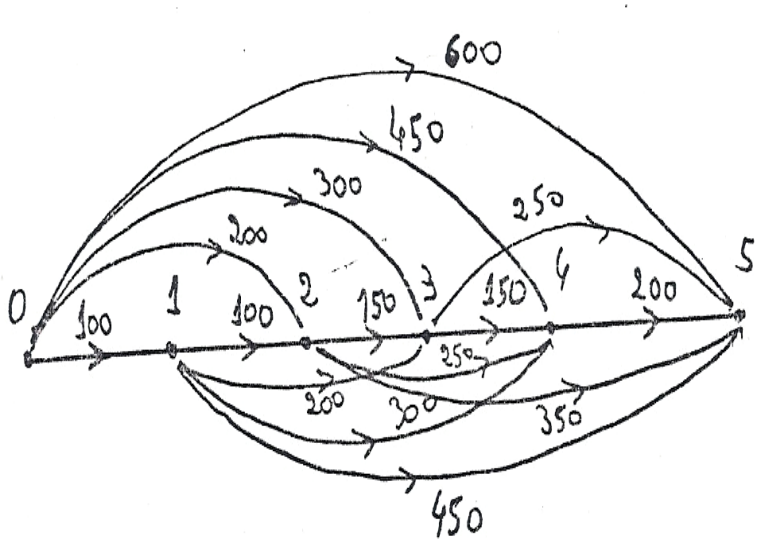
IV) Modélisation

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, poour avoir pendant cinq années une machine en fonctionnement sera donnée par un chemin de valeur minimale allant de 1 à 5.



V) Plus courts chemins

On considère un graphe $G = (X, U)$ orienté et l'on suppose les arcs munis de coût, i.e. une valuation $\omega : U \to R$. Sauf mention contraire, la valuation d'une marche (reps. chemin) est la somme des valuations des arcs de cette marche (reps. chemin). **On remarquera qu'il n'y a aucune hypothèse sur le signe de cette valuation.** On notera $d_G(x, y)$ la longueur d'une plus courte marche entre x et y dans le graphe G. S'il n'existe pas de chemin $d_G(x, y) = \infty$ et s'il n'existe pas de plus courte marche (phénomène lié à l'existence de circuits négatifs) $d_G(x, y) = -\infty$
Une arborescence des plus courts chemins $T = (X, V)$ issue d'un sommet $s \in X$ est une arborescence recouvrante (i.e. $V \subseteq U$) de G de racine s vérifiant : $\forall x \in X$, $d_T(s, x) = d_G(s, x)$
Montrer que :

1. Si G a des plus courtes marches finies de s à tous les autres sommets ssi il existe une arborescences des plus courts chemins de racine s.
2. Une arborescence T de racine s est une arborescence des plus courts chemins ssi $\forall xy \in U$, $d_T(s, x) + \omega(xy) \geqslant d_T(s, y)$.
3. En déduire la complexité de la vérification qu'une arborescence est une arborescence des plus courts chemins.
4. En déduire un algorithme général de calcul de plus courtes marches issues de s, tenant compte des circuits négatifs. Complexité.

VI) Un système de vote basé sur les graphes

Markus Schulze a proposé un système de vote original en 1997, qui a séduit surtout les informaticiens. Ainsi ce système est utilisé par Wikimedia, Free Software Foundation, GNU Privacy Guard, Debian, The Pirate Party et beaucoup d'autres... Ces organismes élisent leur leaders ou prennent des décisions importantes à l'aide de ce système de vote, supposé dur à manipuler. Ces votes étant généralement électroniques il est possible d'organiser plusieurs tours de scrutin successifs.
Le principe est le suivant, on suppose que les candidats sont $c_1, \ldots, c_k$
Chaque votant $x_1, \ldots x_m$ exprime un ordre total sur les candidats. **Un vote égale un ordre total.** On construit un graphe $G = (C, U)$ dont les sommets sont les c_i. Considérons deux candidats a, b. On notera $\#(a, b) =$ le nombre d'ordres totaux (i.e. votes) dans lesquels $a < b$.
Si $\#(a, b) \geqslant \#(b, a)$, on ajoute l'arc $ab \in U$ valué par $\#(a, b) - \#(b, a)$.
et l'on ajoute l'arc $ba \in U$ valué par $\#(b, a) - \#(a, b)$ dans le cas contraire.

Pour un chemin $\mu[a,b]$ dans G, on appelle $Score(\mu)$ la valuation minimale d'un arc de μ et $Score(a,b) = max_{\mu[a,b]}\{Score(\mu)\}$.
L'ensembles de gagnants du vote est alors :
$Gagnants = \{x \in C | \forall y \in C\text{-}x, Score(x,y) \geqslant Score(y,x)\}$.

1. Etudier ce protocole sur un petit graphe (5 candidats).
2. Montrer que l'on peut ignorer les arcs valués par 0.
3. Proposer un algorithme qui calcule les gagnants du vote.
4. * Montrer qu'il existe toujours un ensemble de gagnants.
5. Comment expliquer le succès de ce système ?
6. * Comment manipuler ce système de vote ? Par exemple pour un votant donné, comment choisir les modifications de son vote (i.e. une modification de son classement) afin de changer le résultat final ? Proposer une méthode de calcul.
7. Peut-on proposer un autre système de vote utilisant des algorithmes de graphes, avec des flots par exemple ?

VII) 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

VIII) Isomorphismes

1. Montrer que les trois problèmes suivants sont équivalents :
   Isomorphismes de graphes non orientés
   Isomorphismes de graphes bipartis
   Isomorphismes de graphes d'ordres partiels
2. Que dire de l'isomorphismes d'hypergraphes ?