

# Partiel sur le cours: Algorithmique de graphes

Pierre Fraigniaud, Michel Habib

22 novembre 2006

## 1 Etude d'une relation sur les arêtes (indicatif :15 points)

On considère un graphe  $G = (X, E)$  non orienté, fini, simple (i.e. sans boucle ni arête multiple). On définit la relation  $R$  sur  $E$  comme suit :

Pour  $ab, ac \in E$ , avec  $b \neq c$ ,  $abRac$  si  $bc \notin E$ . On notera  $R^t$  la fermeture transitive de la relation  $R$ .  $R^t$  partitionne donc les arêtes de  $G$  en  $R$ -classes.

1. Proposer un algorithme de calcul de ces  $R$ -classes, l'évaluer en fonction de  $n = |X|$  et  $m = |E|$ .
2. On notera pour une  $R$ -classe  $C$ ,  $X(C)$ , l'ensemble des sommets du graphe adjacents à au moins une arête de  $C$ . Montrer que si une arête d'une classe  $C$  est incluse dans un module  $M$ , alors  $X(C) \subseteq M$ .
3. Montrer que tout module connexe de  $G$  est une union de  $X(C)$ .
4. En déduire qu'un graphe premier n'admet qu'une  $R$ -classe.
5. Réciproquement que peut-on dire des modules d'un graphe connexe n'ayant qu'une  $R$ -classe ?
6. \* Montrer qu'un graphe admet au plus une  $R$ -classe  $C$  telle que  $X(C) = X$ .
7. \* Montrer qu'un graphe  $G$  connexe et donc le complémentaire est connexe admet une  $R$ -classe  $C$  telle que  $X(C) = X$ .
8. En déduire un algorithme de construction de l'arbre de décomposition modulaire basé sur les  $R$ -classes.
9. En fait ces classes ont été introduites pour étudier les orientations transitives d'un graphe de comparabilité. Lorsque  $abR^tcd$  les orientations possibles de ces arêtes dans une orientation transitive du graphe sont liées. Comment engendrer les orientations transitives d'un graphe de comparabilité à l'aide des  $R$ -classes ?
10. Comment engendrer les orientations transitives d'un graphe de comparabilité à l'aide de l'arbre de décomposition modulaire ?
11. Peut-on déduire un algorithme de calcul des  $R$ -classes à partir de l'arbre de décomposition modulaire ? Comparer à votre réponse de la première question.
12. \* Ecrire un algorithme de reconnaissance des graphes de comparabilité basé sur les  $R$ -classes.

## 2 Exercice pair-à-pair et petits mondes (indicatif :5 points)

Cet exercice traite de la conception d'une table de hachage distribuée (DHT) pour systèmes pair-à-pair (P2P), basée sur des concepts issus de la modélisation du phénomène "petit monde" par J. Kleinberg.

Nous considérons un système dans lequel  $k$  clés  $c_0, c_1, \dots, c_{k-1}$  sont réparties en anneau de  $k$  positions consécutives, la clé  $c_i$  occupant la position  $i$  de l'anneau, pour  $i = 0, \dots, k-1$ . Les  $n$  utilisateurs courants sont également répartis dans cet anneau. La position d'un utilisateur  $u$  est notée  $p(u) \in \{0, \dots, k-1\}$ . Les clés gérées par un utilisateur  $u$  forment l'ensemble  $C(u) = \{c_{p(u^-)+1}, \dots, c_{p(u)}\}$  où  $u^-$  est l'utilisateur à la position la plus grande précédent  $p(u)$ . On suppose ici un arrangement cyclique des positions  $0 \prec 1 \prec \dots \prec k-1 \prec 0$ , et toute opération arithmétique est effectuée modulo  $k$ . L'utilisateur  $u^-$  est appelé le prédécesseur de  $u$ , et  $u$  est appelé le successeur de  $u^-$ . Dans tout l'exercice, on suppose que chaque utilisateur  $u$  est connecté à son successeur  $u^+$  (i.e., possède son adresse IP dans sa table de routage). Le routage s'effectue dans une unique direction, d'un utilisateur à son successeur.

**Question 1.** Montrer qu'en l'absence d'autres liens connectant les utilisateurs, la recherche d'une clé  $c$  par un utilisateur  $u$  nécessite au plus  $n-1$  étapes de communication d'utilisateurs à leurs successeurs. Donner un exemple pour lequel cette borne est atteinte.

Pour accélérer la recherche des clés, on rajoute à chaque utilisateur un lien de la façon suivante. Soit  $H$  la fonction harmonique définie par : pour tout entier  $k \geq 1$ ,  $H(k) = \sum_{i=1}^k \frac{1}{i}$ . Chaque utilisateur sélectionne un unique index  $i \in \{1, \dots, k\}$  où la probabilité de choisir l'index  $i$  est  $p_i = \frac{1}{i \cdot H(k)}$ . Soit  $i$  l'index choisit par l'utilisateur  $u$  et soit  $v$  l'utilisateur tel que  $c_{p(u)+i} \in C(v)$ . L'utilisateur  $u$  se connecte alors à  $v$ , appelé le contact de  $u$ .

**Question 2.** Montrer que si la clé  $c$  recherchée vérifie  $c = c_i$  avec  $i = p(u) + d$  pour  $k > d \geq 0$ , alors la probabilité  $p$  que l'utilisateur  $u$  ait son contact  $v$  qui vérifie

$$c \in C(v) \text{ ou } i \leq p(v) + d/2$$

est au moins  $\frac{1}{2H(k)}$ .

**Question 3.** Dédurre de la question précédente que la recherche d'une clé  $c$  par un utilisateur  $u$  procédant suivant le routage glouton nécessite au plus  $O(\log^2 k)$  étapes de communication en moyenne (moyenne prise sur les différents choix des contacts des utilisateurs).

Si le temps moyen de recherche est devenu polylogarithmique grâce à l'ajout d'un unique contact à chaque utilisateur, ce temps reste fonction du nombre de clés  $k$  qui peut être bien supérieur au nombre  $n$  d'utilisateurs. Une des difficultés pour obtenir un temps de recherche polylogarithmique en fonction du nombre d'utilisateurs est que ce nombre varie en permanence et que les utilisateurs n'en n'ont a priori pas connaissance.

**Question 4.** Supposons pour simplifier que  $n$  est stable et connu de chaque utilisateur. Supposons également, pour des raisons de facilité d'écriture, que  $n$  divise  $k$ . Proposez alors un choix du contact de chaque utilisateur permettant d'obtenir un temps de recherche polylogarithmique en fonction du nombre d'utilisateurs.

**Question subsidiaire.** Proposer une méthode permettant à chaque utilisateur d'estimer le nombre courant d'utilisateurs dans le système.