Université Paris Diderot
M1 informatique
Protocoles réseaux 2010-2011

Examen Protocoles réseaux

Tous les exercices sont indépendants.

Exercice 1.—

1. Internet utilise comme principe de communication la communication par commutation de paquets, quels en sont les principes essentiels?
2. Un autre mode de communication est la communication par commutation de circuits (circuits virtuels), quels en sont les principes essentiels? Donner un exemple de protocole utilisant la commutation de circuits?

Exercice 2.—

1. Pour quelle(s) raion(s) les protocoles de "streaming" utilisent plutôt UDP que TCP?
2. Donner deux protocoles de la couche réseau d'Internet.
3. Donner deux protocoles de la couche transport.

Exercice 3.—

1. Quelle est la différence entre le contrôle de flux et le contrôle de congestion?
2. Le protocole TCP assure-t-il le contrôle de flux? Si oui expliquez brièvement de quelle façon?
3. Le protocole TCP assure-t-il le contrôle de la congestion? Si oui expliquez brièvement de quelle façon?
4. Le protocole IP fournit-il explicitement des informations sur la congestion du réseau à la couche transport?
5. Le protocole ICMP fournit-il explicitement des informations sur la congestion du réseau à la couche transport?

Exercice 4.— On considère un alphabet fini A et un alphabet fini B. Soit C une application qui associe à chaque lettre de A un mot sur B. On étend C naturellement aux mots sur A. (Si $w = a_1 \dots a_n$ et pour tout i $(1 \leq i \leq n)$ $a_i \in A$ alors $C(w) = c(a_1) \dots C(a_n)$.)

1. Soit m un mot sur B. Si pour toute lettre x de A, $C(x)$ est de longueur 3, est-il possible de trouver deux mots différents α et β sur A tels que $C(\alpha) = C(\beta) = m$?
2. Soit $E = \{w \in B^* | \text{la longueur de } w = 3\}$. Montrer que E est un code.
3. Soit E un ensemble de mots de A tel que si pour tout $u \in E$ et tout $v \in E$ s'il existe $w \in A^*$ tel que $uw = v$ alors w est le mot vide. Montrer que E est un code.

Exercice 5.— On considère un réseau représenté par le graphe $N =< S, A >$, S étant l'ensemble des noeuds du réseau et A l'ensemble des arêtes ($(x, y) \in A$ si et seulement si les noeuds x et y sont voisins).

Soit $c(x, y)$ le coût associé à l'arête (x, y) dans le réseau. Par hypothèse on suppose que (a) si x et y ne sont pas voisins, $c(x, y) = \infty$ (b) si $x \neq y$ alors $c(x, y) > 0$ et (c) $c(x, y) = c(y, x)$.

Soit un chemin C de x_0 à x_n: $C = (x_0, x_1)(x_1, x_2) \dots (x_{n-1}, x_n)$ (C est un chemin d'arêtes successives (x_i, x_{i+1})), le coût du chemin C est $c(C) = \Sigma_{i=0}^{i=n-1} c(x_i, x_{i+1})$.

On note $d(x, y) = \min\{c(C) | C \text{ chemin de } x \text{ à } y\}$ la distance de x à y. On rappelle que l'on a (Bellman-Ford):

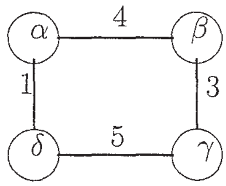
$$d(x, y) = \min\{c(x, z) + d(z, y) | z \text{ voisin de } x\} \qquad (1)$$

L'algorithme suivant calcule les valeurs de $d(x, y)$ en utilisant l'équation (1). Dans cet algorithme, chaque noeud x connaît $c(x, y)$ et maintient une variable (tableau) D_x telle que $D_x(y)$ contient la valeur estimée de $d(x, y)$. Dans cet algorithme chaque noeud x exécute:

- Initialement, $D_x(y) = c(x, y)$ et envoie D_x à tous ses voisins.
- à chaque changement de D_x, chaque x envoie à tous ses voisins la nouvelle valeur de D_x.
- quand x reçoit un tableau D de son voisin z, x met à jour D_x comme suit:

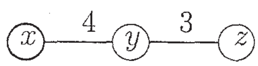
$$\forall y : D_x(y) := \min D_x(y), c(x, z) + D(y)$$

1. Soit le réseau:



   $c(\alpha, \beta) = 4$, $c(\beta, \gamma) = 3$,$c(\delta, \gamma) = 5$ et $c(\delta, \alpha) = 1$. Décrire un comportement possible de l'algorithme jusqu'à ce qu'il se stabilise (plus aucune variable n'est modifiée).

2. Montrer que l'on a toujours pour tout x et tout y, $D_x(y) \geq d(x, y)$.
3. Montrer qu'à chaque modification de $D_x(y)$ pour tout y la nouvelle valeur de $D_x(y)$ est inférieure ou égale à l'ancienne.
4. Montrer que si plus aucun D_x n'est modifié, alors pour tout y $D_x(y) = d(x, y)$
5. Montrer que si $D_x(y) > d(x, y)$ alors pour au moins un z un t et un u voisin de z on a $D_z(t) > D_u(t) + c(z, u)$.
6. Déduire des questions précédentes que l'algorithme stabilise (plus aucune variable n'est modifiée) et pour tout x et tout y $D_x(y) = d(x, y)$
7. On veut transformer l'algorithme précédent en un algorithme qui construit une table de routage. Dans une table de routage, chaque noeud x doit construire une variable (tableau) R_x telle que il existe un chemin C de x à y tel que $d(x, y) = c(C)$ et dont la première arête est $(x, R_x(y))$ ($R_x(y)$ est un voisin de x par lequel passe un plus court chemin de x à y). Comment modifier l'algorithme précédent de façon à ce qu'il construise une table de routage?
8. En reprenant le premier exemple ci dessus, si après la stabilisation, le coût $c(\alpha, \delta)$ change et $c(\alpha, \delta) = 40$, (et $D_\alpha(\delta) = D_\delta(\alpha) = 40$), est ce que l'algorithme se re-stabilise vers les bonnes valeurs? Combien d'itérations cela peut-il prendre?
9. Soit le réseau suivant:



   Quand l'algorithme stabilise quels sont les valeurs de $D_x(z)$ et $D_y(z)$? Supposons que z soit ensuite déconnecté, y le détectant trouve que $c(y, z) = D_y(z) = \infty$, en appliquant l'algorithme que se passe-t-il?