ISTG - RICM2
Examen d'algorithmique distribuée
Durée 2h

Tous documents autorisés – Le barême est indicatif

23 avril 2002

1) Anneau (12 pts)

On se donne un réseau où les noeuds sont identifiés uniquement et sont reliés en anneau via des liaisons unidirectionnelles fiables. Il n'y a pas de relation entre l'identité d'un noeud et sa position dans l'anneau. Tout site peut émettre un message de données sans contrainte sur sa liaison de sortie (pas de jeton d'émission). Tout message porte le numéro du site destinataire. Chaque site est responsable de l'acheminement d'un message jusqu'à son destinataire selon le schéma suivant. Un message de donnée (étiquette `MSG`) reçu sur la liaison entrante `in` d'un site est traité localement si le site est le destinataire sinon est propagé sur la liaison de sortie `out`.

```
repeter alt
      besoin d'émettre(dst,data)        ==> out!<MSG,dst,data>
      in?<MSG, dest,data> && dest!=ICI  ==> out!<MSG, dest, data>
      in?<MSG, dest,data> && dest==ICI  ==> traitement_local(data)
```

Figure 1: Routage en anneau

1.1) Vague sur anneau (2 pts) Rappeler l'algorithme permettant à tout noeud d'émettre une vague (étiquette `TOK`) sur l'anneau indépendamment des vagues issues des autres noeuds.

```
repeter alt
      (decision_vague==true) ==> out!<TOK, ICI,...>
      in?<TOK, I,....> && I!=ICI  ==> .....
      in?<TOK, I,....> && I==ICI  ==> ......
```

1.2) Découverte des noeuds(2 pts) Proposer une spécialisation de vague qui permette aux noeuds de découvrir les identités des membres du réseau. N'importe quel noeud peut initier une vague de découverte (étiquette `DISC`) qui enregistre les identités des noeuds traversés. Le résultat de la vague est la liste des noeuds dans l'ordre de parcours.

```
repeter alt
      (anneau_inconnu==true) ==> out!<DISC, ICI,...>
      in?<DISC, I,....> && I!=ICI  ==> .....
      in?<DISC, I,....> && I==ICI  ==> ......
```

1.3) Elimination de vagues(2 pts) Dans le cas où des noeuds initient concurremment des vagues ayant le même rôle, on surcharge le réseau inutilement. Une solution est de ne conserver qu'une vague (élection) parmi les vagues en cours. Proposer une vague à extinction (étiquette `ELEC`)qui en cas de plusieurs vagues issues d'initiateurs différents[1] ne garde qu'une vague (par exemple celle de l'initiateur d'identité la plus élevée).

```
repeter alt
      (decision_vague==true) ==> out!<ELEC, ICI,...>
      in?<ELEC, I,....> && I!=ICI  ==> .....
      in?<ELEC, I,....> && I==ICI  ==> ......
```

1.4)Découverte des noeuds(2 pts) Modifier l'algorithme de découverte du réseau par de multiple vagues indépendantes pour utiliser une vague à extinction. A la fin de l'exécution de cet algorithme, tous les noeuds du réseau doivent connaitre la constitution de l'anneau.

* donner brièvement l'idée générale de ce nouvel algorithme ? En combien de tours d'anneau, tous les noeuds sont ils informés de la constitution de l'anneau?

* donner l'algorithme

```
repeter alt
      (anneau_inconnu==true) ==> out!<DISC, ICI,...>
      in?<DISC, I,....> && I!=ICI  ==> .....
      in?<DISC, I,....> && I==ICI  ==> ......
      ......................................
```

1.5)Snapshot sur anneau FIFO (4 pts) On se pose une variante du problème du cliché (snapshot) sur un anneau à liaisons monodirectionnelles. On se limite au cas où les liaisons sont FIFO fiables. On considère qu'un seul site peut initier un cliché. Le cliché doit être pris en un tour d'anneau. Aucune autre demande de cliché ne peut être émise avant fin du cliché en cours. A son retour à l'initiateur, le message de cliché (préfixé par `SHOT`) contient le cliché complet.

* que doit enregistrer l'initiateur après émission initiale du message de cliché `SHOT`? et pourquoi?

* que doit enregistrer un site recevant le message `SHOT` ? au moment de la réception ? après ? et pourquoi?

* que contient le message de cliché ?

* Compléter le comportement des sites pour les demandes de cliché issues de l'initiateur 0 .

```
repeter alt
      besoin d'émettre(dst,data)        ==> out!<MSG,dst,data>
      in?<MSG, dest, data> && dest!=ICI ==> .....  out!<MSG, dest, data>
      in?<MSG, dest,data> && dest==ICI ==>  .....  traitement_local(data)
      decision_de_cliché && !cliché_en_cours  ==>cliché_en_cours=true;............
      in?<SHOT,.....>  && ...  ==>  ........
```

2) Anneau cordal(10 pts)

L'inconvénient d'un anneau est que les délais de transmission peuvent être importants pour les noeuds les plus éloignés. Une façon d'accélérer les transmissions est de relier quelques noeuds distants par des liaisons supplémentaires : des cordes. Dans un schéma de ce type (cf. figure 2), il y des noeuds normaux, des noeuds routeurs et des noeuds concentrateurs. Les routeurs doivent choisir la route la plus courte pour atteindre le noeud destinataire : la voie normale ou une corde. Un noeud peut être routeur et concentrateur. On notera `Cout` l'ensemble des cordes sortantes d'un noeud et `Cin` l'ensemble des cordes entrantes. Ces ensembles peuvent être vides selon le type du noeud.

2.1) Découverte des noeuds concentrateurs(3 pts) A la réception d'un message, ces noeuds devront donc décider de reémettre sur une corde ou sur la liaison normale. Pour cela, un routeur doit découvrir les identités des noeuds concentrateurs au bout de ses cordes. Connaissant la composition de l'anneau, il pourra donc décider de prendre une corde ou non pour atteindre un destinataire donné.

[1] Un initiateur n'émet qu'une vague à la fois.

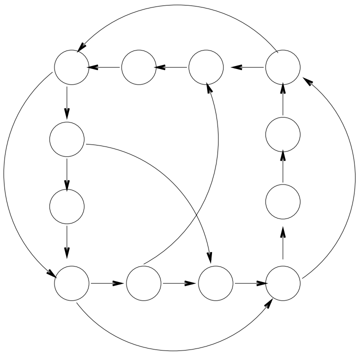

Figure 2: Un anneau cordal

- Donner brièvement l'idée générale de la façon dont un noeud routeur concentrateur peut découvrir l'identité du concentrateur qui est au bout d'une corde.
- Modifier le programme de découverte de l'anneau pour qu'une fois l'anneau connu, les noeuds routeurs enchaine par la recherche des concentrateurs de bout de cordes.

2.2) Routage(3 pts)

* Modifier l'algorithme de routage initial (cf algorithme 1)en supposant donnée la fonction de routage `getRoute` qui à partir d'une identité de noeud donne une liaison de sortie appartenant à l'ensemble `Cout U {out}` .

```
repeter alt
      besoin d'émettre(dst,data)        ==> ........
      exist l dans (Cin U {in})  l?<MSG, dest, data> && dest!=ICI ==> .....
      exist l dans (Cin U {in}) l?<MSG, dest,data> && dest==ICI ==> traitement_local(data)
```

* Donner brièvement l'idée générale de la façon dont la table de routage d'un noeud routeur peut être fabriquée.

2.3)Snapshot sur anneau cordal (4 pts) On reste sous les mêmes hypothèses simplificatrices qu'en 1.5. Le noeud initiateur peut être routeur-concentrateur. Le cliché doit être pris en un tour d'anneau. Aucune autre demande de cliché ne peut être émise avant fin du cliché en cours.

* que doit enregistrer l'initiateur après émission initiale du message de cliché `SHOT` sur `out` et `Cout` ? et pourquoi?

* que doit enregistrer un site recevant le message `SHOT` sur `in`? sur une corde de `Cin`? au moment de la réception ? après ? et pourquoi?

* que contient le message de cliché ?

* Compléter le comportement des sites pour les demandes de cliché issues de l'initiateur 0 .

```
repeter alt
      exist l dans (Cin U {in})  l?<MSG, dest, data> && dest!=ICI ==> .....
      exist l dans (Cin U {in})
           l?<MSG, dest,data> && dest==ICI ==>  .....  traitement_local(data)
      decision_de_cliché && !cliché_en_cours  ==> cliché_en_cours=true;.............
      exist l dans (Cin U {in})l?<SHOT,.....> && .....   ==>  ........
```