Université de Strasbourg
UFR Mathématique et Informatique

Licence 3 Informatique
Année 2013/2014

Systèmes Distribués

Correction du CC convoqué

9 avril 2014

Durée : 2h00
Aucun document autorisé. Chaque partie est indépendante. L'examen est noté sur 23.

Partie I : Questions en vrac (8 points)

Justifiez vos réponses en utilisant des exemples si nécessaire (1 point par question).

a- Pourquoi les horloges vectorielles ne suffisent pas à assurer un ordre causal dans le cas général ?
Car elles ne suffisent pas à rapporter l'activité de tous les sites contrairement aux horloges matricielles ou dans la cas particulier de la diffusion.

b- Donnez un exemple simple où l'utilisation d'une horloge logique pourrait conclure à tort à une dépendance ?
L'exemple du premier CM est suffisant : deux evts sont ordonnés sans qu'il y est de dépendance causal.

c- Dans une base de données répartie, quelles sont les différents niveaux de cohérence possible ?
Illustrez sur un petit exemple. Faible (aucune), Forte (horloges logiques), Causal (horloges vectorielles ou matricielles) : l'exemple du premier CM est valable. On constate que dans le premier cas, on compte sur la commutativité des evts, dans le second tous les evts seront traités dans le même ordre arbitraire, et dans le dernier cas, la seule contrainte sur l'ordre est causal (donc pas nécessairement le même ordre strict).

d- Comment assurer un ordonnancement efficace lorsque le système de tâches n'est pas déterminé à l'avance ?
Donnez une méthodologie possible. Il faut utiliser les techniques de migration dans le contexte d'une grappe de processeurs. Pour cela, il faut utiliser des méthodes de sondage pour déterminer le niveau d'activité. Par exemple, on migre les processus sdes processeurs les plus chargés vers les processeurs les moins chargés.

e- Comment définir formellement un inter-blocage ?
Citez trois façons de traiter le problème et leurs avantages/inconvénients respectifs. Via la formule donnée dans le CM sur le sujet : il doit exister un ensemble de taches bloqués ET non débloquables si on relache les ressources utilisées par l'ensemble des processus non bloqués. Prévention, Évitement, & Détection. Dans le premier cas, on ne laisse pas les inter-blocages se produirent mais il faut définir des niveaux de priorités arbitraires. Dans le second, on peut contourner le problème mais il faut connaitre le système à l'avance et il n'existe pas de méthodes à coût polynomial pour résoudre le problème. Pour finir, on laisse le problème se produire et on le traite au cas par cas mais qui décide ? et qu'est-ce qu'on débloque ?

f- Dans le contexte de la détection d'inter-blocage, expliquez les différents modes possibles pour initier la détection. Listez toutes les formes d'inter-blocages à capturer. Chaque processus, chaque contrôleur ou un proc. spécifique. Les inter-blocages peuvent etre locaux ou multi-sites avec ou sans la participation du site "initiateur" (cf. les illustrations du CM).

g- Dans un système de clés asymétriques, où doit-on copier la clé publique et la clé privée pour initier une connexion sécurisée (dans le cas d'un accès `ssh` par exemple) ? La clé publique doit etre copiée sur la machine du récepteur/serveur (passif) qui pourra l'utiliser pour chiffrer les données vers l'émetteur et la clé privée reste sur la machine de l'émetteur/client (l'initiateur de la connexion).

h- Que doit faire l'homme du milieu pour mettre en faillite une connexion sécurisée à la Diffie Hellman ou RSA ?
Comment contourner ce problème en pratique ? Le MIM ne peut rien faire s'il reste passif, il doit donc être actif, i.e., il doit se faire passer pour Bob du coté d'Alice et réciproquement. Il s'agit d'un problème d'authentification qui peut être contourner par l'utilisation d'un système de confiance transitif, e.g., un système à la Needham-Schroeder/Kerberos.

Partie II : Diffusion (5 points)

a-(1 pts) Quel est la différence entre uniformité et ordre causal dans le cadre d'une diffusion atomique ? Quelles sont les solutions associées lorsqu'on lève l'hypothèse FIFO ? Uniformité = cohérence forte alors que l'ordre causal est plus relaché (l'ordre n'est pas nécessairement le même sur tous les sites.) ABCAST et CBCAST sont respectivement les deux solutions associées.

b-(2 pts) Qu'est qu'une date de validation ? Pour justifiez l'emploi de telles dates, montrez, sur un exemple simple, que les dates d'émission ne sont pas suffisantes lorsque les canaux de communications ne sont pas FIFO. Citez une autre limitation inhérente aux simples dates d'émission justifiant d'employer des dates de validation même lorsque le système est FIFO. Une date de validation correspond à l'application d'uen fonction max sur les dates d'acquittements. Voir l'exemple du CM pour se faire une idée. L'autre limitation est la lenteur du système dans tous les cas.

c-(2 pts) Considérons l'algorithme de diffusion distribué ABCAST :
- quelles sont les règles et la signalisation utilisées pour délivrer les messages diffusés ?
- dans quel cas un message dit utilisable (i.e. sa date de validation est reçue) est mis en attente avant délivrance ?
On utilise des dates de validation avec l'utilisation d'horloges logiques. On ne peut considérer un message utilisable que lorsque celui-ci dispose d'une date de validation. Avant de traiter les messages utilisables selon leurs dates de validation, il faut s'assurer que les messages pour lesquels on a proposé une date d'acquittement inférieur soient validés à leur tour par les autres sites.

Partie III : Inter-blocages (5 points)

a-(1 pt) Donnez un exemple d'inter-blocage impliquant trois processus et deux ressources (représentez l'inter-blocage sous la forme d'un graphe). Il suffit de représenter un cycle dans ce graphe biparti avec trois carrés et deux cercles.

b-(2 pts) Expliquez les deux sous-approches possibles pour prévenir le problème. Sur quels critères les comparer (ré-utilisez votre exemple précédent pour illustrer) ? Pourquoi ce type d'approches est limité en pratique ? L'approche attendre ou mourrir et l'approche blesser ou attendre. Le temps total d'exécution est un critère mais également le temps total d'utilisation CPU. Ces approches sont limités car arbitraires : comment définir finement un niveau de priorité ?

c-(2 pts) Dans quel contexte se situe l'algorithme du banquier ? Est-il satisfaisant dans tous les cas de figure (justifiez) ? quelle est la méthodologie de validation des transitions ? Dans le contexte des méthodes d'évitement. Non il n'est pas satisfaisant lorsqu'il répond non car une solution peut exister (pas de faux positifs mais des faux négatifs.) On valide les transitions de libération et on associé un état virtuel au pire pour les transitions de demande : si cet état virtuel est valide, c'est bon sinon ça casse !

Partie IV : Ordonnancement (5 points)

a-(1 pts) Quelle famille de graphe permet d'assurer l'optimalité d'un ordonnancement quelque soit le nombre de processeurs ? Comment faire et pourquoi ?Les anti-arborescences car chaque tâche n'a qu'un seul sucesseur. Il suffit d'ordonner les taches selon leurs dates au plus tard.

b-(2 pts) Donnez le graphe de précédence correspondant au tableau 1. Calculez la durée du chemin critique et les dates au plus tôt et au plus tard de chaque tâche. Proposez un ordonnacement sans limite sur le nombre de processeurs.TODO : facile !

c-(2 pts) Vous disposez maintenant de deux processeurs, donnez le meilleur ordonnancement possible pour résoudre le graphe de précédence de la question précédente (quel est l'algorithme sous-jacent ?). Est-il minimal ? Optimal ? Justifiez. Est-ce que cet algorithme conserve ses propriétés dans le cas où le coûts de communications inter-tâches sont pris en compte (et comment l'adapter) ?TODO (T_6 est prioritaire sur T_5 car inclusion des successeurs). L'algorithme à utiliser est celui de Coman et Graham. Il est minimal si on obtient le temps théorique du chemin critique, optimal sinon. Très bonne question (a priori, la réponse est non...) ! On peut calculer les

dates au plus tard (pour l'application directe de Coman & Graham) sur base de l'algo de recherche des arbres critiques mais celui-ci fait l'hypothèse d'un nb infini de CPU, donc les dates pourraient être biaisées.

Numéro de tâche	temps d'exécution	dépendances (au sens sucesseurs)
T_1	6	T_2, T_3
T_2	3	T_4, T_5
T_3	2	T_5, T_6
T_4	7	T_{10}
T_5	4	T_7, T_8
T_6	3	T_7, T_8, T_9
T_7	1	T_{10}
T_8	1	T_{10}
T_9	5	T_{10}
T_{10}	3	$\emptyset$

TABLE 1 – Un ensemble de tâches T_i et leurs dépendances