Influence De La Taille Du Jeton Sur Les Performances De L'algorithme D'exclusion Mutuelle De Ricart - Agrawala

Sarah Benkouider[1], Souhila Labgaa[2], Mohamed Yagoubi[3]
Université Amar Telidji
Département d'informatique,
BP 37G, Laghouat 03000.Algérie

[1] sarah_benkouider@yahoo.fr,[2]souhila_lab @yahoo.fr, [3]m.yagoubi@mail.lagh -univ.dz

***Résumé*- Dans le contexte des systèmes répartis on trouve plusieurs solutions pour résoudre le problème de partage de ressources (problème de l'exclusion mutuelle) tels que les algorithmes d'exclusion mutuelle qui sont classés en deux familles : les algorithmes basés sur le principe des permissions et les algorithmes basés sur le principe du jeton.**

Dans cet article, on va utiliser le simulateur NS-2 pour comparer les performances des deux algorithmes de Ricart -Agrawala.

Nous avons axé notre étude comparative par simulation sur les paramètres nombre moyen de messages échangés NMM et temps d'attente moyen TAM.

Les résultats ont montré que l'algorithme basé sur le jeton ayant une meilleure complexité en NMM se retrouve moins performant si le critère de comparaison est le TAM,

Mots clés
Systèmes répartis, exclusion mutuelle, simulateur NS-2.

I. INTRODUCTION

L'apparition des systèmes répartis fait l'objet d'un fort développement depuis une vingtaine d'années. Cet essor s'est accentué avec les progrès technologiques des réseaux de télécommunication. En effet, les systèmes répartis doivent leur naissance en tout premier lieu au mariage de l'informatique et des télécommunications.

Dans tels systèmes, on trouve plusieurs problèmes tels que la détection de la terminaison, la cohérence de données et la concurrence des processus lors de l'accès à des ressources partagées (*le problème de l'exclusion mutuelle).*

Dans cet article nous avons simulé les deux algorithmes d'exclusion mutuelle de Ricart - Agrawala afin de comparer leurs performances. L'implémentation de ces algorithmes sera faite sur un simulateur de réseau appelé Network Simulator NS-2 [1].

II. LES ALGORITHMES SIMULES

Plusieurs algorithmes ont été proposer pour résoudre le problème de l'exclusion mutuelle comme l'algorithme de Lamport [2]qui est basé sur le principe des permission et qu'il demande 3(n-1) messages pour réaliser une exclusion mutuelle (où n est le nombre de sites dans le réseau).

Pour réduire le nombre de messages échangés, Ricart – agrawala ont proposé deux algorithmes, le premier en 1981 basé sur les permissions où il nécessite 2(n-1) messages, et l'autre en 1983 basé sur le principe du jeton, il nécessite « n » messages par entrée en section critique.

Dans cet article, nous avons étudié et comparé l'algorithme de Ricart - Agrawala (RIA 81) [5] basé sur le principe des permissions et l'algorithme de Ricart - Agrawala (RIA 83) [6] basé sur le principe du jeton.

II.1. ALGORITHME DE RICART - AGRAWALA 1 (RIA 81)

Il est basé sur le principe des permissions, il cherche à minimiser le nombre de messages échangés par entrée en section critique et élimine les messages de type libération (**Lib**). Il nécessite **2(n-1)** (où **n** est le nombre de sites dans le système) messages pour réaliser une exclusion mutuelle : n-1 pour indiquer aux autres processus l'intention de Pi de pénétrer en section critique et n-1 autres pour véhiculer les réponses favorables à cette demande qui lui en autorise l'accès.

Lorsqu'un processus Pi désire entrer en section critique il diffuse à tous les autres processus un message de type **Req** doté d'une estampille. Lorsqu'un processus Pj reçoit un tel message il peut répondre favorablement par un message de type **Ack** doté aussi d'une estampille, ou différer sa réponse. S'il ne désire pas entrer en section critique il envoie un message **Ack**.

Par contre, si Pj désire accéder à la section critique, il compare l'estampille de sa requête à celle reçue. Si la requête reçue est plus récente, Pj retarde sa réponse, autrement Pj envoie immédiatement un message **Ack**. Un processus qui a reçu un message **Ack** de tous les autres processus peut entrer en section critique. Lorsqu'il en sort, le processus envoie un message **Ack** à tous les processus pour lesquels il a différé sa réponse.

II.2. ALGORITHME DE RICART - AGRAWALA 2 (RIA 83) [1] [2]

Cet algorithme est basé sur le principe du jeton où le processus qui est en section critique possède un privilège que l'on matérialise par un jeton. Tant qu'un processus garde le jeton, il peut rentrer en section critique sans consulter les autres.

Le jeton est constitué d'un tableau dont le k ième élément mémorise l'estampille de la dernière visite qu'il a effectuée au processus Pk. Initialement, il est affecté à un processus quelconque. Le jeton est demandé par un processus Pi à l'aide d'un message de type « **requête** » estampillé et diffusé à tous les autres processus.

Lorsque le processus Pj qui possède le jeton ne désire plus rentrer en section critique, il cherche dans le tableau qui matérialise le jeton, le premier processus Pr choisi dans l'ordre (j+1,…, n, 1,…,j-1) tel que l'estampille de la dernière requête de Pr soit supérieure à l'estampille mémorisée par le jeton lors de sa dernière visite à Pr . Pj envoie le jeton à Pr.

Cette manière de parcourir le tableau est choisie pour éviter le problème de famine.

Le nombre de messages moyen échangés par entrée en section critique est égal à « 0 » si le processus détient le jeton, si non égal à « n ».

III. SIMULATION

Plusieurs articles [3][4][7][8] se sont intéressés aux simulations des algorithmes d'exclusion mutuelle.

Dans notre simulation, on a utilisé le simulateur NS-2 qui a été crée en 1989 chez UC Berkeley (l'université de Californie à Berkeley). Il est actuellement à sa version 2.30.

Cet outil de simulation est principalement bâti avec les idées de la conception par objets, de réutilisation du code et de modularité.

III.1 LES ETAPES DE REALISATION DE LA SIMULATION

Afin d'implementer les deux algorithmes de Ricart-Agrawala dans NS-2, nous avons créé pour chacun deux fichiers de source écrits en langage C++ ({ }.h , { }.cc).

Le premier est un fichier d'en-tête, qui contient la structure du message échangé entre les sites, le deuxième contient les fonctions nécessaires de l'algorithme (envoyer (message), recevoir (message),…).

En suite il faut modifier le fichier « Makefile » afin de recompiler le simulateur NS-2.

Afin de tester ces algorithmes,il faut créer les fichiers { }.tcl pour visualiser la simulation.
On peut résumer les étapes de réalisation de chaque algorithme par la figure suivant :

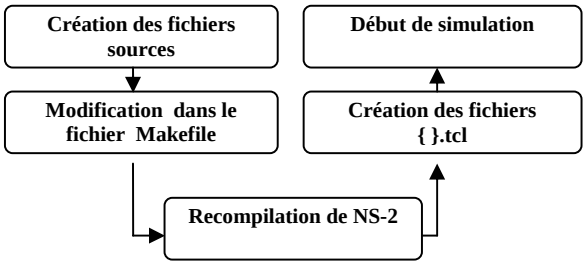

Fig. 1. Les étapes de réalisation

III.3 LES PARAMETRES CHOISIS POUR L'EVALUATION

Les scénarios que nous avons réalisés, nous ont permis de mettre en évidence des paramètres importants qui vont faire la différence dans l'évaluation et la comparaison des algorithmes d'exclusion mutuelle et qui sont :

- *Le nombre de messages moyen (NMM) :*

Le nombre de messages échangés est un paramètre déterminant dans la comparaison des performances des algorithmes d'exclusion mutuelle.

Tous les nouveaux algorithmes à l'étape théorique font apparaître ce paramètre comme élément clé pour évaluer leurs performances.

A l'aide de NS-2, on a voulu vérifier les valeurs de ces performances publiées dans les différents articles de recherches.

Pour chaque algorithme, l'entrée en section critique nécessite le transfert d'un ensemble de messages, l'application qu'on a développée permet de compter le nombre de messages moyen échangés par entrée en section critique qui est calculé par la formule suivante :

$$NMM = \frac{\text{Nombre de messages total}}{\text{Nombre d'entrées en SC}}$$

- *Le temps d'attente moyen (TAM) :*

Dans les algorithmes répartis d'exclusion mutuelle, les auteurs parlent et utilisent parfois le temps d'attente moyen pour évaluer les performances des différents algorithmes.

Avec NS-2, on s'est intéressé de prés à ce paramètre qu'on a appliqué aux algorithmes étudiés.

Un site demandeur doit attendre un intervalle de temps avant la satisfaction de sa demande. Cet intervalle correspond à l'attente de l'arrivée des permissions ou du jeton, ou à l'attente de la sortie d'un autre site qui utilise la section critique. A l'aide de NS-2, on peut calculer le temps d'attente de chaque site ainsi que le temps d'attente moyen qui est calculé par la formule suivante :

$$TAM = \frac{\sum_i T\ attente\ (i)}{\text{Nombre d'entrées en SC}}$$

Où T attente (i) : est le temps d'attente du site.

III.4 RESULTATS ET DESCUSSION

On a varié le nombre de sites de 10 à 70 afin de voir l'influence de ce dernier sur le temps d'attente moyen (TAM) et le nombre de messages moyen (NMM).

Après la simulation des deux algorithmes de Ricart - Agrawala, on a obtenu les résultats ci-dessous :

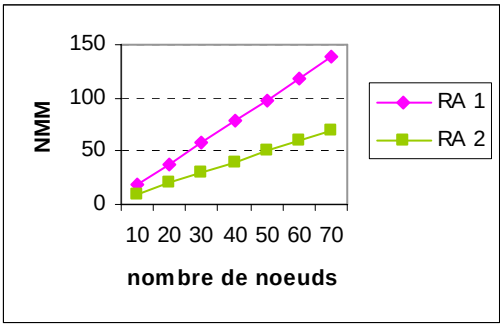

Fig. 2. Influence de nombre de sites sur le NMM.

La figure2 illustre pour les deux algorithmes de Ricart - Agrawala que le nombre de messages moyen (NMM) augmente d'une manière progressive dû à l'augmentation du nombre de sites, ce qui confirme les complexités théoriques énoncées dans la littérature. Pour les deux algorithmes étudiés, l'interprétation de ces complexités est comme suit :

Pour l'algorithme de Ricart - Agrawala avec permissions : NMM = 2(n-1) où « n » est le nombre de sites.

En remarquant qu'à chaque ajout d'un site le nombre de messages moyen sera incrémenté de deux, le premier pour la demande et le deuxième pour la réponse.

On remarque aussi que le nombre de messages moyen (NMM) de l'algorithme de Ricart - Agrawala avec jeton augmente avec l'augmentation du nombre de sites, ce qui est conforme avec la théorie où le nombre de messages moyen est égal à n. Alors l'ajout d'un site va faire augmenter le nombre de messages moyen par un message (pour la demande).

On peut donc conclure que le nombre de sites a une grande influence sur le nombre de messages moyen (NMM) donc l'algorithme de Ricart - Agrawala avec jeton est le plus économique en terme de NMM.

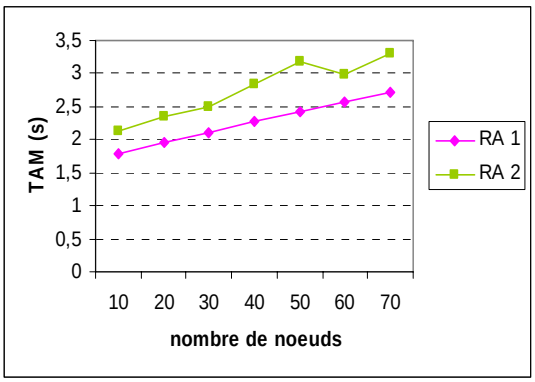

Fig. 3. Influence de nombre de sites sur le TAM.

Dans la figure3, on remarque pour les deux algorithmes que le temps d'attente moyen (TAM) augmente avec l'augmentation du nombre de sites.

Pour l'algorithme de Ricart - Agrawala avec permissions (RA1), si un site parmi les « n » demande d'entrer en section critique, il doit attendre « n-1 » ack, donc à chaque fois qu'on augmente le nombre de sites, le nombre d'ack attendus s'augmente et cela rendre le TAM plus grand.

Dans l'algorithme de Ricart - Agrawala avec jeton (RA2), pour qu'un site puisse accéder à la section critique il faut que les « n-1 » demandes arrivent à destination, et il faut attendre la réception du jeton.

Le paramètre TAM pour l'algorithme de Ricart - Agrawala avec jeton est nettement supérieur au celui avec permission, alors qu'en nombre de messages il est le meilleur. Ceci s'explique par l'influence du nombre de sites sur la taille du message (jeton) qui est présenté par un tableau de taille « n » (où n est le nombre de sites) et aussi sur le nombre de messages de demande.

D'après cette étude comparative, on peut affirmer qu'un nouveau paramètre influe sur le temps d'attente, ce paramètre est la taille du message. Cette influence apparaît dans le cas de l'algorithme de Ricart - Agrawala avec jeton où le jeton a une taille variable (selon le nombre de sites).

Pour mieux voir l'influence de la taille du jeton sur le paramètre (TAM) de l'algorithme (RA2), on a varié la taille du message jeton de 400 à 51200 octets.

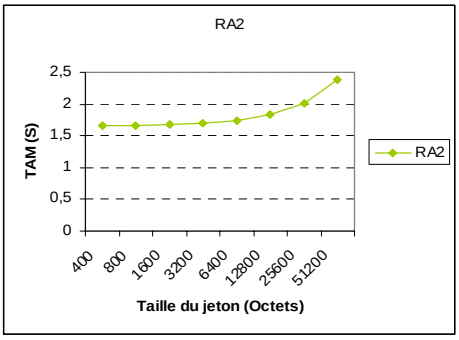

Fig. 4. Influence de la taille du message jeton sur le TAM

La figure4 montre bien comment la courbe continue à croître avec la taille de message (jeton) transporté par le jeton.

On remarque que chaque fois qu'on augmente la taille du jeton, le temps d'attente moyen augmente aussi puisque le jeton prend plus de temps pour arriver à sa destination.

Donc on peut dire que le nombre de sites dans l'algorithme de Ricart - Agrawala avec jeton à un impact sur la taille du jeton qui est présenté par un tableau de dimension « n » (où n est le nombre de sites) donc l'ajout d'un site va changer la taille du tableau qui modélise le jeton (de n à n+1), alors la transmission du jeton prend plus de temps ce qui fait croître le temps d'attente moyen (TAM).

IV. CONCLUSION

Cette étude comparative des différentes complexités nous a permis d'une part de vérifier les complexités théoriques énoncées dans les algorithmes répartis d'exclusion mutuelle et d'autre part de bien prendre en considération tous les facteurs qui peuvent avoir une influence sur ces complexités. Comme la taille du message qui a joué ici un rôle important dans le paramètre TAM.

REFERENCES

[1]. A. Bouzoualegh**, ''**Etude et proposition d'un réseau local acoustique aquatique'', mémoire de thèse Doctorat de l'université de Toulouse II, *Laboratoire de recherches* ICARE EA 3050, juillet 2006.

[2]. L.Lamport. Time, clocks, and the ordering of events in a distributed system. *Communications of the* ACM,21(7):558-564,1978.

[3]. M.B.Yagoubi. L'Exclusion mutuelle dans les systèmes inforùatique répartis. Université d'Evry Val d'Essonne.1997.

[4]. M.Yoke Hean Low and D.M.Nicol. Consistent modeling of distributed mutual exclusion protocol using optimistic simulation. Parallel and Distributed simulation,2001. Proceedings.15th workship on, 15-18 May 2001 pp 137-144.

[5]. G. Ricart and A.K. Agrawala. An optimal algorithm for mutual exclusion in computer networks. *Communications of the* ACM,24(1):9-17,1981.

[6]. G. Ricart and A.K. Agrawala. On mutual exclusion in computer networks. *Communications of the* ACM,26(2):147-148,1983.

[7]. S.S.FU. A Comparison of mutual Exclusion Algorithms for Distributed memory Systems. *News letter of the technical committee of Distributed processing, IEEE 1997.*

[8]. X.Zhang, Y.Yan, R.Castaneda. Evaluating and Disigning Software Mutual Exclusion Algorithms on Shared Memory Multiprocessors. Parallel and Distributed Technology: systems and Aplications, IEEE, V 4, issue:1, spring 1996 pp 25-42.