Le Pèlerin optimiste

Gestion de la concurrence dans les collecticiels : vers des protocoles optimistes

BA Tafsir Mamour*, GARCIA Eric**, HENRIET Julien**, LAPAYRE Jean Christophe**

Département de Génie Informatique - Ecole Supérieure Polytechnique
Dakar - Sénégal - tafba@ucad.sn* ,
Laboratoire d'Informatique de l'université de Franche-Comté - FRE CNRS 2661
Besançon - France - [garcia,henriet,lapayre]@lifc.univ-fcomte.fr**

RÉSUMÉ. L'une des contraintes des collecticiels est de permettre aux utilisateurs d'accéder de manière concurrente à certaines données. Malheureusement, les protocoles classiques de contrôle de concurrence ne conviennent généralement pas aux collecticiels. Notre équipe de recherche dispose d'une plate-forme pour applications coopératives dans laquelle le contrôle de concurrence est géré par un protocole pessimiste. L'inadéquation de ce dernier à certaines catégories de collecticiels nous a amenés à mettre au point un protocole optimiste de gestion de la concurrence en utilisant les notions d'atomisation et de multiversion. Nous définissons ce protocole à l'aide d'un automate d'états finis, et nous le comparons d'un point de vue probabiliste à sa version pessimiste initiale.

ABSTRACT. In some groupwares, users access concurrently to data. Unfortunately, various classical concurrency control protocols do not fit groupwares. Our research team owns a platform for groupware in which concurrency control is managed by a pessimistic protocol. As the latter did not fit to some groupware categories, we achieved a concurrency management optimistic protocol by using notions of atomization and multiversion. We define this protocol with a finite state automaton and we compare these versions on a probabilist point of view.

MOTS-CLÉS : atomisation, automate d'états finis, contrôle de concurrence, multiversion, protocole optimiste

KEYWORDS : atomisation, finite state automaton, concurrency control, multiversion, optimist protocol

1. Introduction

Les applications coopératives doivent pouvoir permettre aux utilisateurs d'effectuer des actions simultanées dans un espace commun appelé espace de production [7] qui peut être considéré comme un ensemble d'objets. Il existe différentes manières de gérer la concurrence : Amoeba [12] propose un ordonnancement total des messages en utilisant l'horloge de Lamport [10]. L'approche est basée sur l'utilisation d'un séquenseur agissant comme un goulot d'étranglement et diminuant les performances. Lansis [1] est un protocole chargé de la cohérence des données utilisant les outils d'Isis [4] et notamment le concept de *virtual synchrony* repris ensuite dans les projets Horus et Ensemble [5]. Lansis permet de respecter une cohérence causale entre les évènements en minismisant le nombre de messages échangés. Si le processeur A décide d'envoyer un message $A2$ dépendant causalement du message $B3$ envoyé précédemment par le processeur B, la syntaxe sera $b3A2$. Si un processeur reçoit $b3A2$ sans avoir reçu précédemment le message $B3$, il envoie un message de type $NACK$ sur $B3$. $B3$ lui sera alors retourné. Un processeur ne peut envoyer de message tant qu'il est dans une situation incohérente. Totem [3], basé lui aussi sur Isis [4], propose un ordonnancement total en faisant circuler un jeton sur un anneau logique. Seul le processeur du domaine possédant le jeton peut diffuser ses modifications en y incluant une estampille. Ces protocoles sont adaptés aux groupes d'utilisateurs pour lesquels les membres peuvent faire du broadcast ou du multicast. L'anneau sur lequel circule le jeton est un anneau logique, et un processeur ayant une vitesse de calcul plus élevée que tous les autres peut recevoir le jeton plusieurs fois pendant un tour d'anneau. Un mécanisme de contrôle des flux est aussi implémenté. Notre approche [8] est quant à elle basée sur la circulation d'un jeton sur un anneau logique contenant en plus toutes les modifications apportées par les différents sites, évitant ainsi au site d'avoir à diffuser lui-même ses modifications à tous les membres du groupe. Un protocole de contrôle de concurrence peut être pessimiste ou optimiste [9]. Avec un protocole pessimiste, seul le propriétaire d'un objet peut le modifier en le verrouillant. Les autres sont obligés de demander la propriété au propriétaire et doivent donc attendre d'acquérir cette dernière avant de pouvoir écrire comme dans le cas du protocole du Pèlerin. Cependant une optimisation du délai d'attente avant de pouvoir écrire est nécessaire à certaines applications. Dans les protocoles dits optimistes [9, 11], il n'y a pas de délai d'attente pour pouvoir modifier un objet.

La première section de ce papier présente la version *classique*, pessimiste, du protocole de gestion de la concurrence nommé le Pèlerin. Dans la deuxième section, nous montrons comment par une *atomisation des objets* combinée à une approche multiversion [6], nous avons conçu un protocole optimiste : le Pèlerin Optimiste. Nous décrivons son fonctionnement à l'aide d'un automate d'états finis. Dans la troisième section, nous comparons les deux versions du protocole du Pèlerin, et nous discutons de leurs avantages et inconvénients.

2. Présentation du Pèlerin Classique

Le Pèlerin [8] est un protocole de gestion de cohérence des données d'une mémoire partagée répartie de type mémoire répliquée à partage d'objets. Il fonctionne sur la base d'un jeton circulant sur un anneau logique unidirectionnel. Mais le jeton est ici un ensemble de données : écritures et actions nécessaires à la gestion des propriétés des variables partagées. Chaque site dispose d'une mémoire locale (mémoire répliquée) qui comporte les différents objets de l'espace de production décrits par des propriétés et regroupés par sites. Lorsqu'une action est effectuée sur un site (création, suppression, modification, demande de propriété, changement de propriété), cette dernière est temporairement stockée dans un jeton local puis écrite sur le jeton à sa réception.

Pour décrire cet algorithme, nous utilisons une partie de l'automate d'états finis définissant la version optimiste de la figure 1 : les états *non-grisés*. Le Pèlerin est un protocole de concurrence pessimiste, dans le sens où un site Non Propriétaire d'un objet doit, s'il veut le modifier, en demander la propriété. Il devient alors Non-Propiétaire Demandeur Potentiel. Sa demande de propriété est placée sur le jeton. Dans le cas de plusieurs demandes, c'est le site qui reçoit un jeton sans commande sur l'objet qui devient le Non-Propriétaire Demandeur. Les autres sites sont déboutés car il n'y a pas de file d'attente. Lorsque le propriétaire reçoit un jeton avec une demande, il le refuse s'il est Propriétaire Actif Bloqué car il est en train d'écrire. S'il n'écrit pas et qu'il reçoit une demande ou un jeton sans demande, il devient Propriétaire Actif Débloqué. La nouvelle réception d'un rjd le rend Propriétaire Inactif. Seule une opération d'écriture peut placer à nouveau le site dans l'état PAB. Un PAD ou PI qui reçoit une demande est obligé de céder sa propriété et donc d'accepter la demande. Il devient alors Propriétaire Perdant. Parallèlement, le site demandeur reçoit une acceptation, et passe dans l'état Non Propriétaire Gangnant. Il émet alors un jeton de changement de propriété. Le site PP devient Non Propriétaire. Après un tour d'anneau, le site dans l'état Non Propriétaire Gagnant la propriété reçoit son jeton de changement et devient PAB. Si le NPD reçoit un refus il redevient NP, sans avoir pu écrire. Ce protocole est pessimiste dans la mesure où un site ne peut pas systématiquement écrire, il lui faut tout d'abord devenir propriétaire.

3. Etude pour rendre le protocole optimiste

L'atomisation d'un objet revient à décomposer l'objet en plusieurs paramètres élémentaires. Sur le plan sémantique, les paramètres sont liés. En revanche, d'un point de vue algorithmique, les paramètres sont alors considérés comme des objets indépendants. A la création de l'objet, le créateur de celui-ci est le propriétaire unique de tous les paramètres. Pour rendre le Pèlerin optimiste nous introduisons la notion de version ([11], [6]). Un site souhaitant modifier un paramètre en demande la propriété. En cas de refus, il bénéficiera

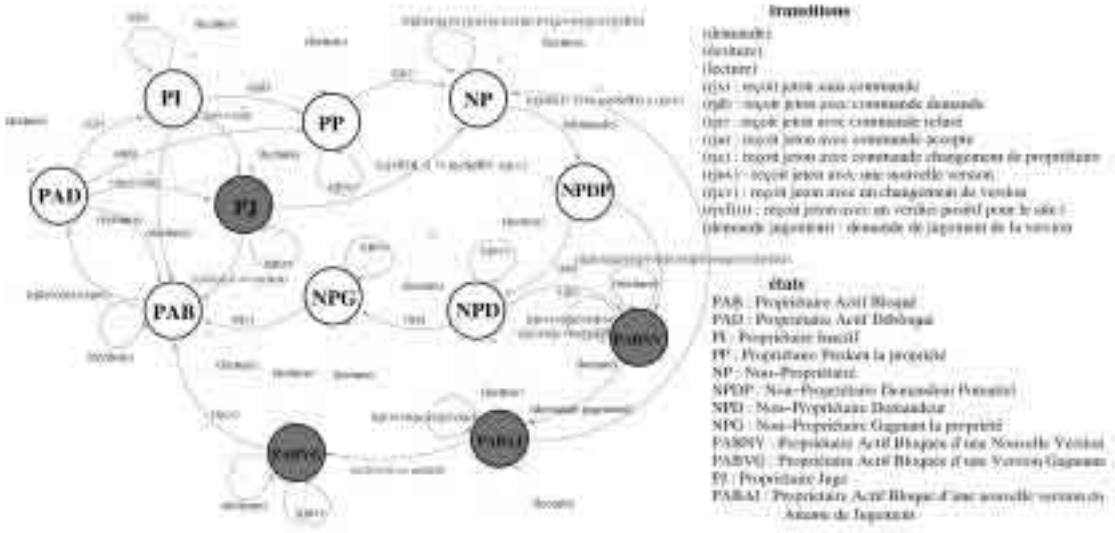

Figure 1. *Automate d'états finis du Pèlerin optimiste*

automatiquement d'une nouvelle version du paramètre qu'il pourra modifier. A un instant t, plusieurs versions, dont celle du propriétaire courant, peuvent exister et une seule version sera choisie par jugement dans notre protocole. Elle remplacera alors la version de référence, le propriétaire de la version gagnante devenant le nouveau propriétaire.

L'automate (figure 1) contient les différents états que peut prendre un site. Les événements aboutissant à un état impossible n'y sont pas mentionnés. L'ensemble du travail est disponible dans [2]. Si un site NPDP reçoit un refus il devient Propriétaire Actif Bloqué d'une Nouvelle Version et peut commencer sa modification sans attendre. S'il n'écrit plus, il émet une demande de jugement et devient Propriétaire Actif Bloqué d'une nouvelle version en Attente de Jugement et émet un avis de nouvelle version qu'il ne peut plus modifier. S'il reçoit un verdict favorable, il devient Propriétaire Actif Bloqué d'une Version Gagnante et émet alors un avis de changement de version. Le verdict rendu, il peut à nouveau écrire. Après un tour, il recevra son propre avis et il passera alors dans l'état PAB. En revanche, si le PABAJ reçoit un verdict défavorable ou un rjcv, il retourne à l'état NP. Comme dans la version pessimiste, un propriétaire reste actif bloqué s'il écrit. Lorsqu'il devient inactif (PAD ou PI), si une nouvelle version est en attente de jugement, il devient Propriétaire Juge (notons qu'une demande est prioritaire sur une nouvelle version) et émet alors un verdict favorable au site vainqueur. La version du juge est elle-même en compétition avec celles des PABAJ. Le verdict peut ou bien être personnel et dépendre du PJ, ou bien peut être issue d'un vote ou encore d'une politique de choix systématique

telle que le Moins Fréquemment Vainqueur. S'il n'est pas choisi, le PJ deviendra PP à la réception de son propre verdict et du rjcv émis par le vainqueur.

4. Discussion

Pour modifier l'un des paramètres d'un objet, il faut en être propriétaire. Il est donc intéressant de calculer la probabilité P qu'un site S NP devienne PAB, dans les deux protocoles quand il n'y a aucune demande sur le jeton à l'instant où il devient NPDP.

4.1. Cas du protocole pessimiste

Soit P_{NPDP} la probabilité qu'un NP devienne NPDP. Ce site doit ensuite passer par différents états pour devenir PAB. Le site passe tout d'abord NPD. Pour cela, il faut qu'aucun des j sites situés entre le jeton et S (dans le sens de l'anneau logique) ne fasse une demande. Soit X la variable aléatoire correspondant au nombre de demandes. La réception du jeton par chacun des j sites constitue une répétition d'évènements indépendants car la demande d'un site n'influe pas sur la possibilité de demande des suivants. De plus, sur chaque site, il y a deux résultats possibles : demande ou non. X peut ainsi être caractérisée par une loi binomiale : $X = Bi(j; P_{NPDP})$ donc $P(X = x) = C_x^j.(P_{NPDP})^x.(1 - P_{NPDP})^{(j-x)}$. Dans notre cas, x vaut 0 car il ne faut aucune demande sur les j sites pour que notre NPDP devienne NPD. Soit P_{NPD} la probabilité qu'un NPDP devienne NPD. On a alors : $P_{NPD} = P(X = 0) = C_0^j.(P_{NPDP})^0.(1 - P_{NPDP})^{(j-0)} = (1 - P_{NPDP})^j$

Pour écrire, le site NPD doit devenir NPG. A la reception de la demande, si le propriétaire est bloqué, il refuse la demande et la probabilité est alors nulle. En revanche si le propriétaire est PAD ou PI, la demande est obligatoirement acceptée. Il ne faut pas non plus que le PAB ou PI écrive avant de recevoir la demande car il redeviendrait alors PAB. Considérons qu'un site écrive en moyenne w fois/tour de jeton sur une propriété donnée. Soit Y la variable aléatoire discrète correspondant à ces écritures. Les actions utilisateurs étant relativement lentes par rapport au temps d'un tour de jeton, les évènements d'écritures sur une propriété sont relativement rares sur un tour de jeton. Par conséquent, Y peut être caractérisée par une loi de Poisson de paramètre w, avec : $P(Y = y) = \frac{(e^{-w}.w^y)}{y!}$. Donc $P_{NPG} = P(Y = 0) = \frac{(e^{-w}.w^0)}{0!} = e^{-w}$. Une fois le site NPD devenu NPG, il émet un rjc qu'il reçoit après un tour. Il deviendra alors PAB. La probabilité correspondante P_{PAB} est égale à 1.

Ainsi un NPDP devient propriétaire s'il devient NPD (évènement A) puis s'il devient NPG (évènement B) et enfin s'il devient PAB (évènement C). D'après le théorème des probabilités composées : $P = P(A).P(B|A).P(C|AB) = P_{NPD}.P_{NPG}.P_{PAB} = (1 - P_{NPDP})^j.e^{-w}$

4.2. Cas du protocole optimiste

Dans le cas optimiste, pour devenir PAB, un site NP doit d'abord effectuer une demande et devenir NPDP. Ensuite, il peut devenir PAB par trois voies (cf. figure 1).

Par la voie classique : soit $Q1$ la probabilité de devenir PAB par cette voie. Elle est similaire au cas pessimiste car la demande est prioritaire sur les versions : $Q1 = Q_{NPD}.Q_{NPG}.P_{PAB1} = P_{NPD}.P_{NPG}$

Par la voie du NPDP débouté : un NPDP qui n'est pas devenu NPD a une nouvelle version et se transforme en PABNV. La demande de jugement étant certaine, il deviendra PABAJ. S'il reçoit un verdict favorable, il deviendra PAB. Soit Q_{PABNV1} la probabilité qu'un NPDP devienne PABNV. Un NPDP devient soit NPD, soit PABNV. Donc $Q_{PABNV1} = 1 - P_{NPD}$. La probabilité Q_{PABAJ} de devenir PABAJ vaut 1. Soit Q_{PAB2} la probabilité qu'un PABNV devienne PAB, soit v le nombre de versions en concurrence. La version du propriétaire est également en compétition. Une version doit être nécessairement choisie à l'issue du jugement. Si l'on considère qu'il y a équiprobabilité, on a alors : $Q_{PAB2} = 1/v$, avec $v > 0$. Soit $Q2$ la probabilité de devenir PAB par cette voie : $Q2 = Q_{PABNV1}.Q_{PABAJ}.Q_{PAB2} = \frac{(1-P_{NPD})}{v}$

Par la voie du NPD recevant un refus. Le site passe de NPDP à NPD mais sa demande est refusée. Il devient alors PABNV puis PABAJ et attend le verdict pour éventuellement devenir PABVG. Un NPD est soit NPG, soit PABNV. Donc $Q_{PABNV2} = 1 - P_{NPG}$. Ensuite, il doit devenir PABAJ puis PAB. Soit $Q3$ la probabilité de devenir PAB par cette voie : $Q3 = Q_{NPD}.Q_{PABNV2}.Q_{PABAJ}.Q_{PAB2} = \frac{P_{NPD}.(1-P_{NPG})}{v}$

Soit Q la probabilité de devenir PAB par l'une des trois voies. D'après le théorème des probabilités totales, nous avons par conséquent : $Q = Q1 + Q2 + Q3 = P_{NPD}.P_{NPG} + \frac{(1-P_{NPD})}{v} + \frac{P_{NPD}.(1-P_{NPG})}{v}$. Il est possible d'exprimer le nombre de versions v en fonction de w. En effet, chaque site écrit en moyenne w fois/tour d'anneau, ainsi, il y aura en moyenne $(n-1)w$ versions si n est le nombre total de sites sur l'anneau. Il suffit d'ajouter la version du site étudié et nous obtenons : $v = (n-1).w + 1$.

4.3. Comparaison

Il s'agit de comparer P (cas pessimiste) et Q (cas optimiste). Nous avons vu que : $Q = P_{NPD}.P_{NPG} + Q2 + Q3 = P + Q2 + Q3$ donc $Q > P \forall_{w,j,P_{NPDP}}$. La figure 2 montre l'évolution de ces probabilités pour 10 sites, j=3 sites et P_{NPDP}=0,5 en fonction de la vitesse d'écriture w d'un site. Cette probabilité est toujours supérieure à 0,6 pour le Pèlerin optimiste et inférieure à 0,2 pour le Pèlerin quelle que soit w. Ces probabilités sont des fonctions décroissantes mais relativement constante dans le cas du Pèlerin (une variation d'environ 3%) tandis qu'elle varie de près de 10% pour la version optimiste. La probabilité du protocole optimiste est en moyenne 8 fois supérieure à celle du protocole pessimiste. Par ailleurs, le nombre de messages échangés pour qu'un site accède à la

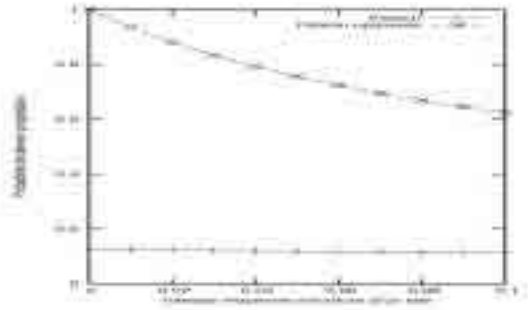

Figure 2. *Probabilités de passer PAB avec le Pèlerin optimiste et le Pèlerin.*

propriété est identique dans les deux protocoles. Dans le cas d'une demande, il y aura au total 3 messages, et 4 messages s'il y a d'autres demandes. En cas de nouvelle version, il y aura 3 messages. Quant au nombre de tours nécessaires pour acquérir la propriété, il est de 2 au meilleur des cas, comme pour le protocole du Pèlerin pessimiste. Néanmoins dans le cas optimiste, la taille du jeton sera supérieure, celui devant contenir toutes les versions concurrentes éventuelles.

5. Hommage

L'acceptation de ce papier nous permet de rendre hommage à Tafsir Mamour Ba, co-auteur, qui est décédé pendant la phase de sélection du CARI.

6. Conclusion

Dans cet article, nous présentons une nouvelle version du protocole du Pèlerin assurant le contrôle de la concurrence sur une plate-forme distribuée. Nous le rendons optimiste en ajoutant la possibilité d'écrire des versions concurrentes des paramètres d'un objet. Nous avons mesuré le gain du protocole optimiste sur la version pessimiste par une étude des probabilités d'écriture sur le paramètre d'un objet partagé. En effet, dans ce nouveau protocole, au niveau applicatif le délai d'attente avant écriture est inférieur en moyenne au cas pessimiste. L'étude réalisée montre également que la complexité du Pèlerin classique en terme de nombre de messages et de tours d'anneaux minimum du jeton nécessaires avant écriture est concervée. Néanmoins, le jeton du Pèlerin optimiste devant contenir toutes les versions concurrentes éventuelles, les temps de transfert de ce dernier peuvent s'en trouver dégradés.

Les travaux à venir se situent sur trois axes de recherche. Nous comptons tout d'abord valider cette étude en implémentant et en comparant les deux protocoles en terme de temps. Nous étudions aussi une amélioration du Pèlerin optimiste en prenant en compte la tolérance aux pannes et en rendant l'atomisation des objets indépendante de la nature de l'application. Par ailleurs, nous étudions un autre protocole optimiste dans lequel le nombre minimal de tours avant d'être propriétaire est diminué. Il en découlera une optimisation du délai d'attente avant écriture au niveau applicatif.

7. Bibliographie

[1] KRAMER S., AMIR Y., DOLEV D., MALKI D., « Transis : A Communication Subsystem for High Availability », *FTCS-22 : 22nd International Symposium on Fault Tolerant Computing, IEEE Computer Society Press, PP 76-84*, 1992.

[2] BA T., HENRIET J., LAPAYRE JC., « Pèlerin Optimiste : preuves et validation », *rapport technique 2004.02 du LIFC, http ://lifc.univ-fcomte.fr*, 2004.

[3] CHEVASSUT O., BERKET K., AGARWAL D. A., « A Practical Approach to the InterGroup Protocols », *Future Generation Computer Systems, vol 18, num 5, PP 709-719*, 2002.

[4] BIRMAN K. P., « The Process Group Approach to Reliable Distributed Computing », *Communications of the ACM, vol 36, num 12, PP 37-53*, 1993.

[5] BIRMAN K. P., HAYDEN M., HICKEY J., KREITZ C., VAN RENESSE R., RODEH O., CONSTABLE B., VOGELS W., « The Horus and Ensemble Projects : Accomplishments and Limitations », *Proceedings of DARPA Information Survivability Conference and Exposition (DISCEX'00), vol 1, PP 149-161*, 2000.

[6] DOMMEL H.P., GARCIA-LUNES J.J., « Floor control for multimedia conferencing and collaboration », *Multimedia Systems, Vol 5, PP 23-38*, 1997.

[7] ELLIS C., WAINER C., « A conceptuel model of Groupware », *Proceedings of CSCW'94, PP 79-88, ACM Press*, 1994.

[8] GARCIA E., LAPAYRE J-C., DAVID G., « Pilgrim Performance over a New CAliF Communication Layer », *IEEE Proceedings of the ICPADS'00, PP 203-210, Japan*, 2000.

[9] GREENBERG S. , MARWOOD D., « Real Time Groupware as a Distributed System :Concurrency Control and its effect on the Interface », *Proceedings of CSCW'94, PP 207-217, ACM Press*, 1994.

[10] LAMPORT L., « Time, clocks and the ordering of events in a distributed system », *Communications of the ACM, 21(7), PP 558-565*, 1978.

[11] SUN C., CHEN D., « A Multi-version Approach to Conflict Resolution in Distributed Groupware Systems », *Proceedings of the ICDCS'2000 International Conference, PP 316-325*, 2000.

[12] KAASHOEK M. F., TANENBAUM A. S., « An Evaluation of the Amoeba Group Communication System », *International Conference on Distributed Computing Systems, PP 436-448*, 1996.