Chloé DESDOUITS
Guillerme DUVILLIE
Swan ROCHER

M1 Informatique MOCA

TP d'algorithmes distribués

2012

UM2

Table des matières

Chapitre 1

Introduction

Afin de résoudre des problèmes complexes sur des données de grandes tailles, il existe plusieurs possibilités : utiliser un algorithme avec une bonne complexité, augmenter la puissance de l'ordinateur utilisé ou utiliser plusieurs ordinateurs interconnectés. Lorsque les deux premières possibilités ont été poussées à leur maximum, il est nécessaire d'utiliser des algorithmes dits distribués afin de répartir les calculs sur plusieurs machines.

Les algorithmes distribués sont divisés en plusieurs catégories. Il existe tout d'abord des algorithmes d'élection qui permettent de donner la priorité temporaire à un site par rapport aux autres. Il y a également des algorithmes d'exclusion mutuelle qui permettent à tous les sites de travailler sur la même ressource (section critique) et qui garantissent le fait que deux sites ne peuvent pas accéder en même temps à cette ressource. Enfin les algorithmes de terminaison permettent de détecter la fin du calcul distribué pour tous les sites et ainsi de terminer l'application.

Nous allons nous intéresser aux algorithmes distribués d'exclusion mutuelle. Ceux-ci sont jugés sur les critères suivants :

- la vivacité : le fait qu'un site qui demande à entrer en section critique puisse y entrer au bout d'un temps fini,
- la sureté : la capacité de l'algorithme à faire en sorte qu'un seul site à la fois soit en section critique,
- le respect de l'ordre des demandes,
- la tolérance aux pannes,
- la complexité en nombre de messages échangés.

Nous allons tout d'abord étudier l'algorithme de Naïmi-Trehel [1] qui fait partie de la catégorie des algorithmes distribués d'exclusion mutuelle. Nous allons ensuite nous intéresser à une extension tolérante aux pannes de cet algorithme [2]. Nous expliquerons alors les choix effectués lors de l'implémentation de ces deux algorithmes. Enfin, nous exposerons les résultats de nos phases de test.

Chapitre 2

Algorithme d'exclusion mutuelle de Naimï-Trehel

2.1 Description

L'algorithme de Naimi-Trehel [1] est basé sur le fait de conserver deux structures de données distribuées et un jeton. Le jeton est présent sur un des sites à la fois et matérialise la permission d'entrer en section critique. Les structures de données distribuées sont un arbre logique dynamique et une file d'attente.

La file d'attente stocke les sites qui ont demandé à entrer en section critique. La file d'attente étant distribuée, chaque site stocke uniquement son suivant (s'il y en a un) dans la file d'attente. La tête de la file est le site actuellement en section critique et la queue de la file est le dernier site à avoir demandé à y entrer. Quand un site demande à entrer en section critique, il est ajouté à la fin de la file. Quand un site quitte la section critique, il envoie le jeton à son suivant dans la file (s'il y en a un).

La seconde structure de données est l'arbre logique dynamique. Cet arbre a pour but d'indiquer aux sites lequel d'entre eux est le dernier dans la file d'attente. La structure étant distribuée, chaque site stocke uniquement son père dans l'arbre. Lorsqu'un site veut entrer en section critique, il transmet la requête à son père qui (s'il n'est pas racine) transmet la requête à son père jusqu'à ce que la requête atteigne la racine de l'arbre. Le site racine dans l'arbre étant également le dernier dans la file d'attente, il fixe son suivant au site qui a émis la requête. Puis tous les sites qui ont transmis la requête fixent leur père au site qui a émis la requête (désormais le dernier dans la file d'attente).

2.2 Exemple

Dans cet exemple, initialement, le nœud a possède le jeton (figure 2.2).

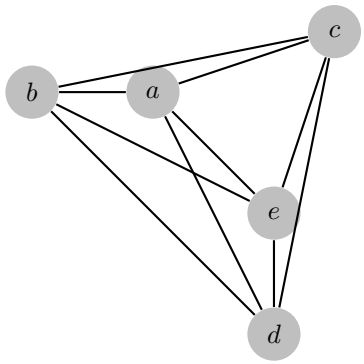

FIGURE 2.1 – Réseau physique

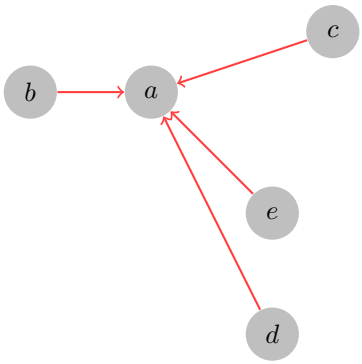

FIGURE 2.2 – Arbre logique

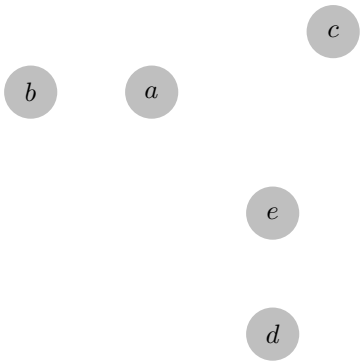

FIGURE 2.3 – File d'attente

Puis le nœud b fait une requête pour obtenir la section critique.

2.3 Propriétés

Chapitre 3

Extension tolérante aux pannes

Chapitre 4

Implémentation

Chapitre 5

Tests et résulats

Bibliographie

[1] M. Naimi, M. Trehel, and A. Arnold. A log (n) distributed mutual exclusion algorithm based on path reversal. *Journal of Parallel and Distributed Computing*, 34(1) :1–13, 1996.

[2] J. Sopena, L. Arantes, M. Bertier, and P. Sens. A fault-tolerant token-based mutual exclusion algorithm using a dynamic tree. *Euro-Par 2005 Parallel Processing*, pages 644–644, 2005.

Réalisé avec LaTeX