**INF8480 – Systèmes répartis et infonuagique**



**Automne 2018**

**TP2 – Services distribués et gestion des pannes**

**[1739632] – [Vincent Leduc]**

**[1584742] – [François-Xavier Dueymes]**

**Remis à Adel Belkhiri**

**[06-11-2018]**

Division des tâches par le répartiteur

Tout d’abord, on tri l’ensemble des serveurs de calcul disponibles en fonction de leur capacité, de la plus grande capacité à la plus petite. Ensuite, on essaie d’envoyer l’ensemble des calculs au premier serveur de calcul. Si ce serveur ne peut accepter l’ensemble d’opérations, on réduit cet ensemble jusqu’à ce que le taux de refus soit inférieur à 20%. Lorsque le nœud accepte la tâche, on met sa capacité à 0. À ce moment, on tri à nouveau les serveurs de calcul et on tente d’envoyer les opérations restantes au serveur en tête de file. Si l’ensemble des nœuds de calcul est occupé, le distributeur redemande l’ensemble des nœuds de calcul au service de noms.

Cette méthode de division des tâches ne permet pas de paralléliser au maximum les calculs, mais s’assure que les calculs seront effectués si au moins un serveur est disponible. Ce choix de division a été fait puisqu’il permet de s’assurer de l’intégrité des données et que le système donnera une réponse.

Gestion des tâches échouées et pannes intempestives

Lorsque le distributeur appelle une méthode sur un serveur de calcul qui a planté, le distributeur affiche la RemoteException et recharge la liste des calculateurs. Si le serveur de calcul plante pendant ses opérations, on envoie la liste d’opérations qui lui était assignée à un autre serveur de calcul selon le procédé de division des tâches.

Cette méthode de gestion d’échec a été implémentée de la sorte puisqu’elle permet de s’assurer que les données seront traitées en cas de défaillance et que le système n’entrera pas dans une boucle d’attente continue.

Mode sécurisé

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Op-588 | Op-1949 | Op-2809 | Op-3216 | Op-1481 |
| 1 serveur | 583 ms | 587 ms | 556 ms | 587 ms | 1281 ms |
| 2 serveurs | 855 ms | 853 ms | 933 ms | 726 ms | 744 ms |
| 3 serveurs | 988 ms | 985 ms | 839 ms | 1050 ms | 980 ms |
| 4 serveurs | 940 ms | 981 ms | 780 ms | 870 ms | 970 ms |

Tableau 1 : Temps d’opérations selon le nombre de serveur

Fig 1 : Temps d’exécution selon le nombre de serveur

De manière générale, on remarque que le temps de calcul augmente avec la quantité de serveurs disponibles. Ceci s’explique de deux manières. D’abord, la gestion d’un plus grand nombre de serveurs par le distributeur augmente le nécessaire à la distribution, ce qui correspond au coût de l’indirection. Ensuite, puisque nous avons implémenté un système qui envoie le maximum possible de calculs au serveur le plus disponible, la parallélisation implémentée est sous-optimale lorsque les calculs nécessaires ne dépassent pas de façon significative les capacités individuelles des nœuds de calcul, puisqu’il est fort probable dans le cas contraire qu’un seul nœud effectue la majeure partie des opérations à lui seul, ce qui ne tire pas avantage de l’ensemble du réseau distribué.

Mode non-sécurisé

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Op-588 | Op-1949 | Op-2809 | Op-3216 | Op-1481 |
| 3 serveurs fiables | 1431 ms | 1506 ms | 1290 ms | 1426 ms | 1526 ms |
| 2 serveurs fiables  1 serveur malicieux 50% | 1810 ms | 1912 ms | 1689 ms | 1937 ms | 1828 ms |
| 2 serveurs fiables  1 serveur malicieux 75% | 2147 ms | 2276 ms | 2001 ms | 2192 ms | 2298 ms |

Fig 2 : Temps d’exécution selon la malice des serveurs en mode sécurisé

On remarque que plus les serveurs de calcul sont malicieux, plus le temps d’exécution est long. Nos données expérimentales nous suggèrent que cette corrélation est croissante et linéaire avec le niveau de malice des serveurs de calcul. Ceci s’explique par la nécessité de refaire un certain nombre d’opérations lorsque le résultat obtenu est erroné, ce qui allonge le temps de calcul.

Questions de réflexion

Comme c’est le répartiteur qui conserve les données de calcul et de répartition, le système est sensible aux pannes qui pourraient survenir sur le serveur sur lequel est exécuté le répartiteur. Afin de pallier en partie à cette lacune, une solution serait d’avoir de la redondance en incorporant un second serveur de répartition qui serait en attente de défaillance du premier et qui recevrait les données nécessaires à la répartition des opérations (mode esclave). Il pourrait envoyer des requêtes au premier répartiteur selon un certain intervalle afin de s’assurer qu’il est toujours connecté. Dans le cas contraire, il prend la relève avec les données de répartition les plus récentes en mémoire. Lorsque le premier répartiteur redémarre, il vérifie si un autre répartiteur est en mode maître et, dans ce cas, s’exécute en mode esclave, sinon, en mode maître.

Ce système a comme inconvénient d’augmenter significativement la quantité de message à envoyer sur le réseau, de même que d’augmenter la responsabilité du répartiteur et sa complexité. Si le second serveur plante alors que le premier n’a pas redémarrer, la totalité des données de l’opération en cours pourrait être perdue.