
Reconfiguration Dynamique de Coterie Structurée en Arbre

Ivan Frain

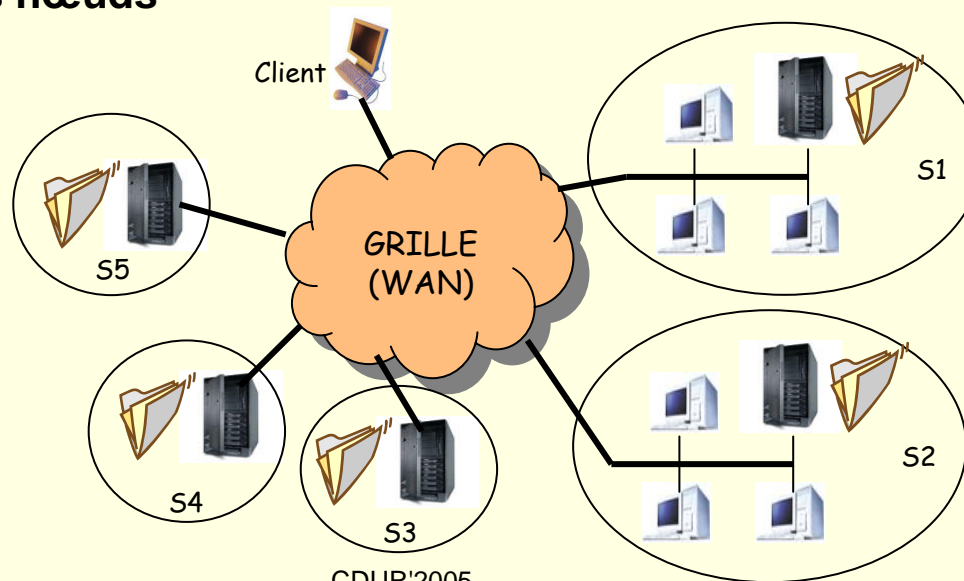
Jean-Paul Bahsoun et Abdelaziz M'zoughi
Institut de Recherche en Informatique de Toulouse (IRIT)

Sommaire

- Contexte de l'étude
- Protocoles à quorum et coterie en arbre
- Prise en compte de la charge des nœuds et problème
- Principe de permutation élémentaire
- Extension d'un algorithme de reconfiguration dynamique de coterie
- Évaluation et résultats
- Conclusion et Perspectives

Contexte : Grilles Informatiques

- Grilles de serveurs (en opposition avec grilles de type P2P)
 - une dizaine de sites distants
- Réplication des données entre sites
- Pas de maîtrise de la charge d'un nœud
 - La charge d'un serveur n'est pas uniquement issu des travaux grilles
 - La charge d'un serveur n'est pas uniquement issu de l'accès aux données
- **Maintien de la cohérence des répliques en tenant compte de l'évolution de la charge des nœuds**



Protocoles à Quorum : Définition

- **Quorum** q - Ensemble minimum de copies impliquées dans une opération de lecture ou d'écriture afin que l'opération réussisse.
- **Coterie** C - Ensemble des quorums possibles pour un groupe de copies et un protocole donné
 - Intersection : $\forall q_1, q_2 \in C : q_1 \cap q_2 \neq \emptyset$
 - Minimalité : $\forall q_1, q_2 \in C : q_1 \not\subseteq q_2$
- Cohérence forte entre répliques (réplication pessimiste)
- Différents types de protocoles à quorum: majorité, structurés en arbre, en grille, ...

Protocoles à Quorum : Exemple

■ Protocole à majorité

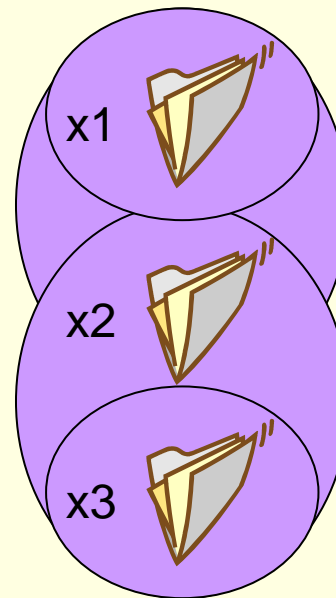
- Un quorum est composé d'une majorité de copies
- Exemple: 3 copies d'une donnée x : x1, x2 et x3

Scénario:

→ Écrire (x,'a')

Lire (x)

Écrire (x,'b')



version

valeur

~~3~~ ~~3~~ 2

~~a~~ b

~~3~~ 1

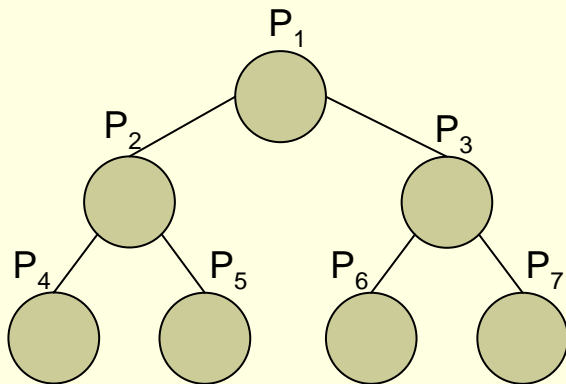
a

~~3~~ 2

b

Coterie Structurée en Arbre

- Protocole de Agrawal et Abbadì [1]

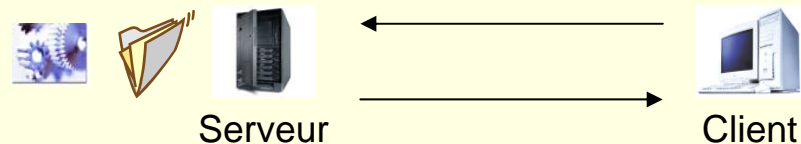


- Nœuds organisés logiquement en arbre
- Quorum : Un chemin de la racine à une feuille
- Coterie : $\{\{P_1, P_2, P_4\}, \{P_1, P_2, P_5\}, \{P_1, P_3, P_6\}, \{P_1, P_3, P_7\}\}$
- Les propriétés d'intersection et de minimalité sont vérifiées par le protocole
- $O(\log_2(n))$ messages

- Sur quel critère construire l'arbre ?

Prise en compte de la charge

- Temps d'accès à une réplique



$$T = tt_{client/serveur} + ts_{serveur} + tt_{serveur/client}$$

- Temps de service est fonction de la charge du serveur
 - Quels sont les critères de charge à retenir?
- Temps de service du serveur non négligeable

Charge d'une Coterie : Définition

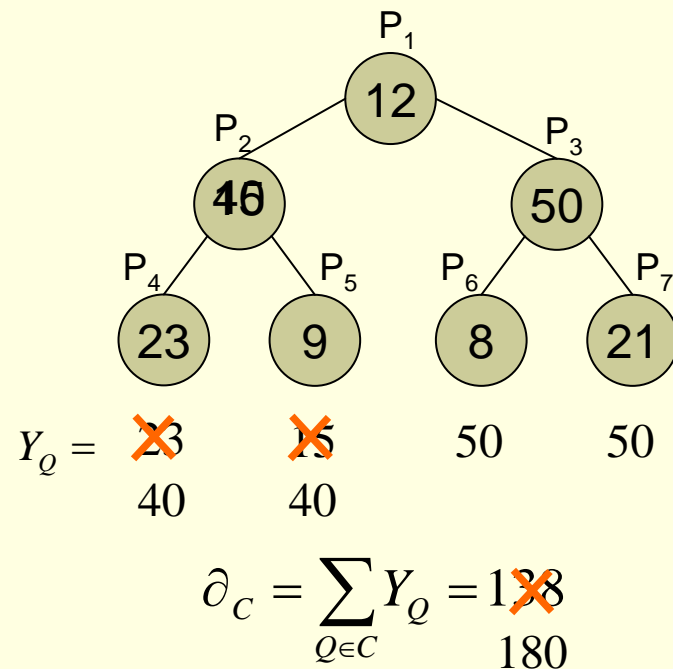
- Un nœud possédant une réplique : P
- Charge d'un Nœud : x_p
- Charge d'un Quorum

$$Y_Q = \text{Max}(x_p : P \in Q)$$

- Charge d'une Coterie

$$\partial_C = \sum_{Q \in C} Y_Q$$

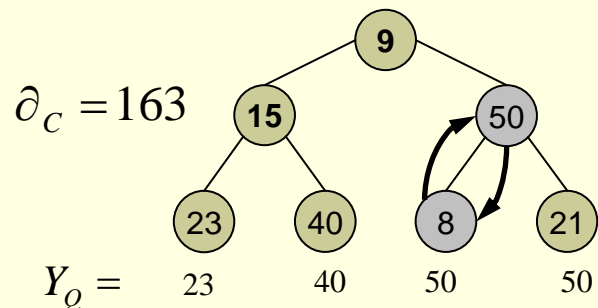
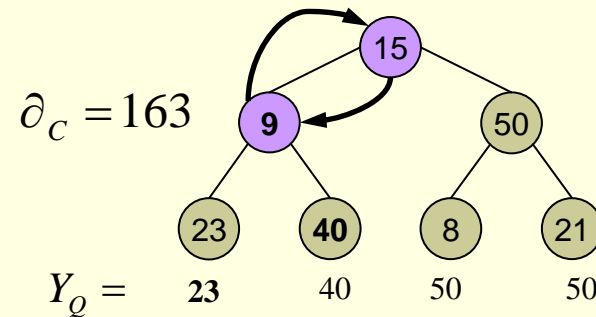
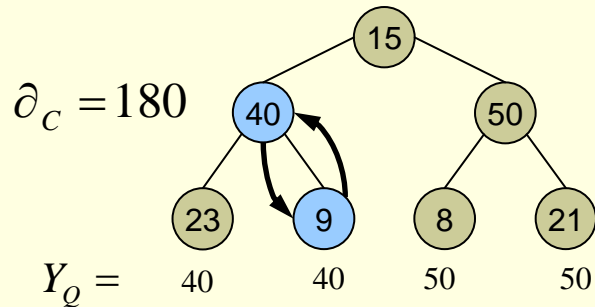
Problème Lors de l'Évolution de la Charge



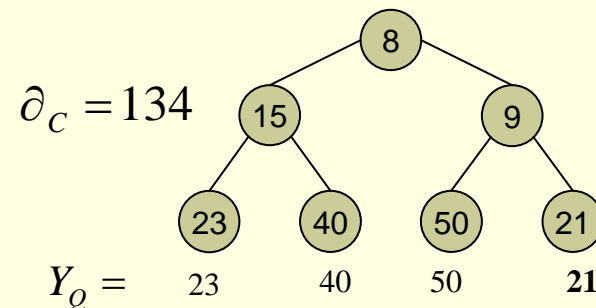
- La charge de P_2 augmente jusqu'à 40
- La charge des deux quorums augmente en conséquence
- Idem pour la charge de la coterie
- **L'augmentation de la charge d'un seul nœud peut pénaliser l'ensemble de la Coterie**
- **Comment prendre cela en compte lors de la construction de l'arbre ?**

Principe de Permutation Élémentaire

- Permuter deux nœuds parents si le père est plus chargé que le fils



...

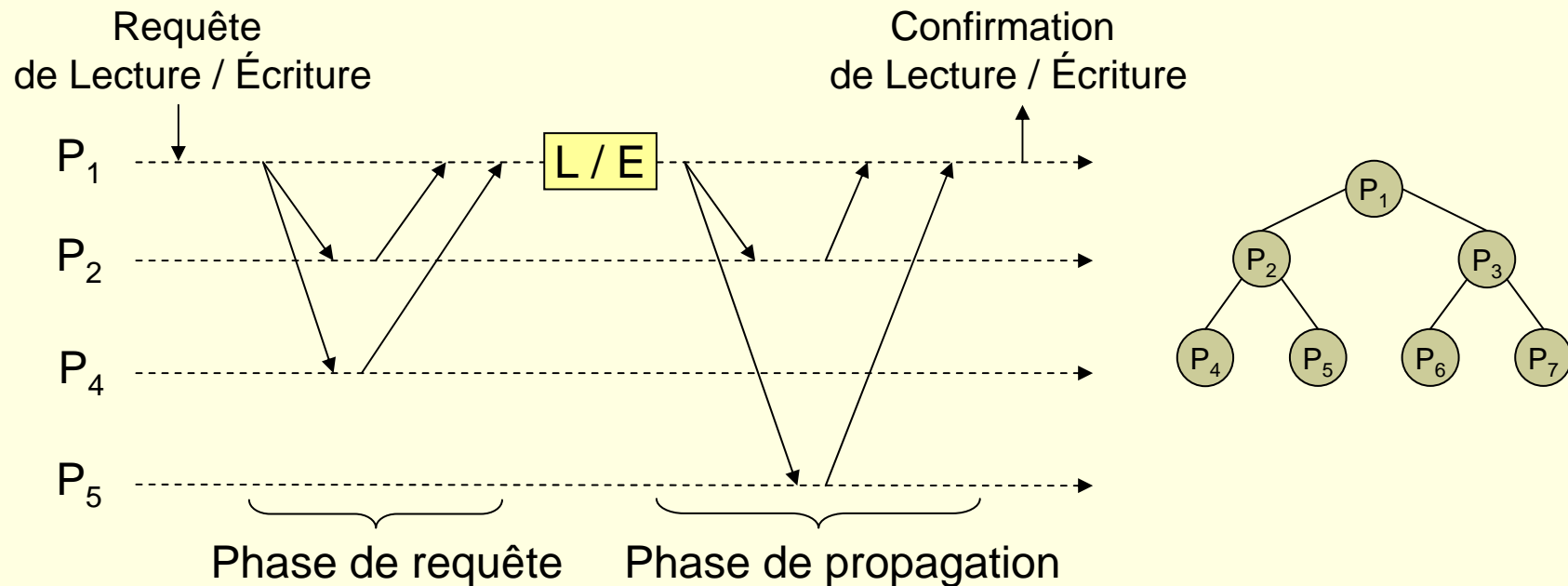


Utilisation des permutations élémentaires

- Avoir un système de lecture / écriture fonctionnel:
 - Un protocole de lecture / écriture à base de coterie
 - Un protocole supportant la reconfiguration de coterie
- Algorithme de Shvartsman et Lynch [2]
 - Multi lecteur, multi rédacteur
 - Reconfiguration dynamique de coterie
 - Interface utilisateur (lecture / écriture)
 - Interface de gestion (reconfiguration dynamique)

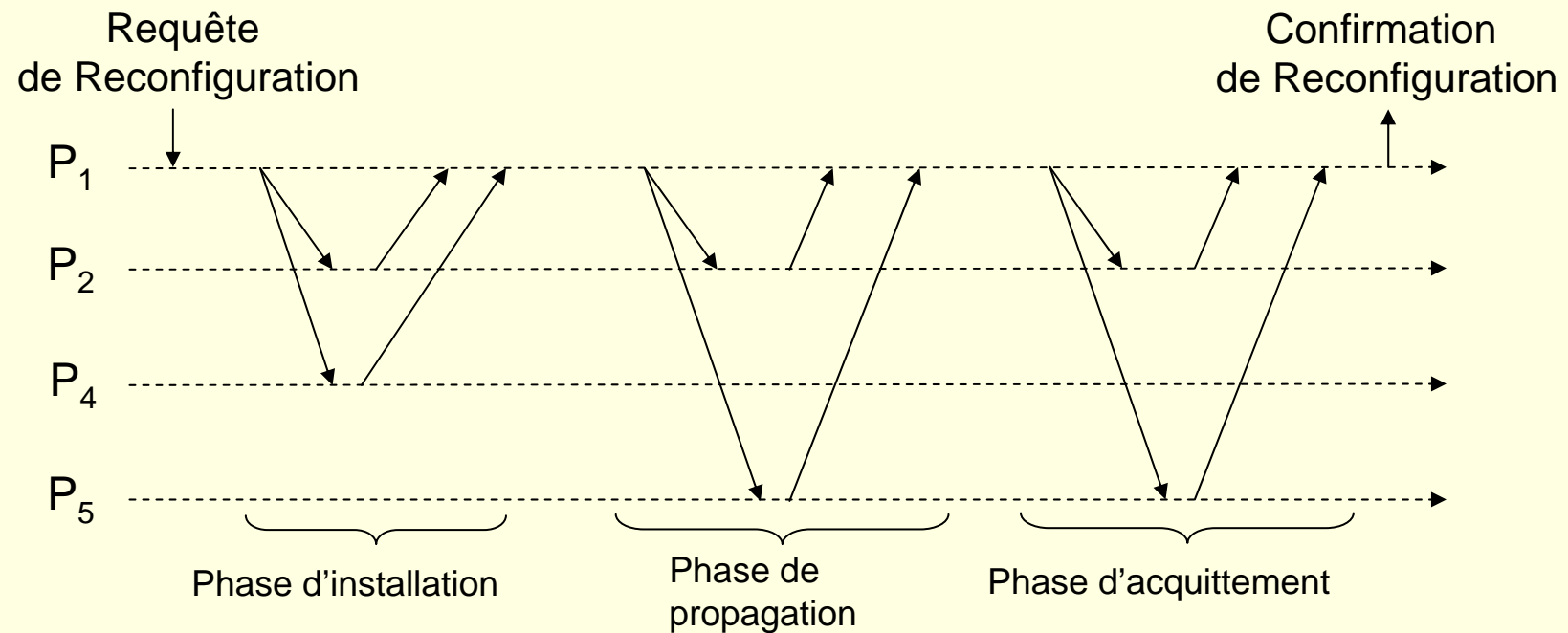
Algorithme de Reconfiguration Dynamique de Coterie (1/2)

■ Interface applicative : lecture / écriture



Algorithme de Reconfiguration Dynamique de Coterie (2/2)

■ Interface de gestion : reconfiguration

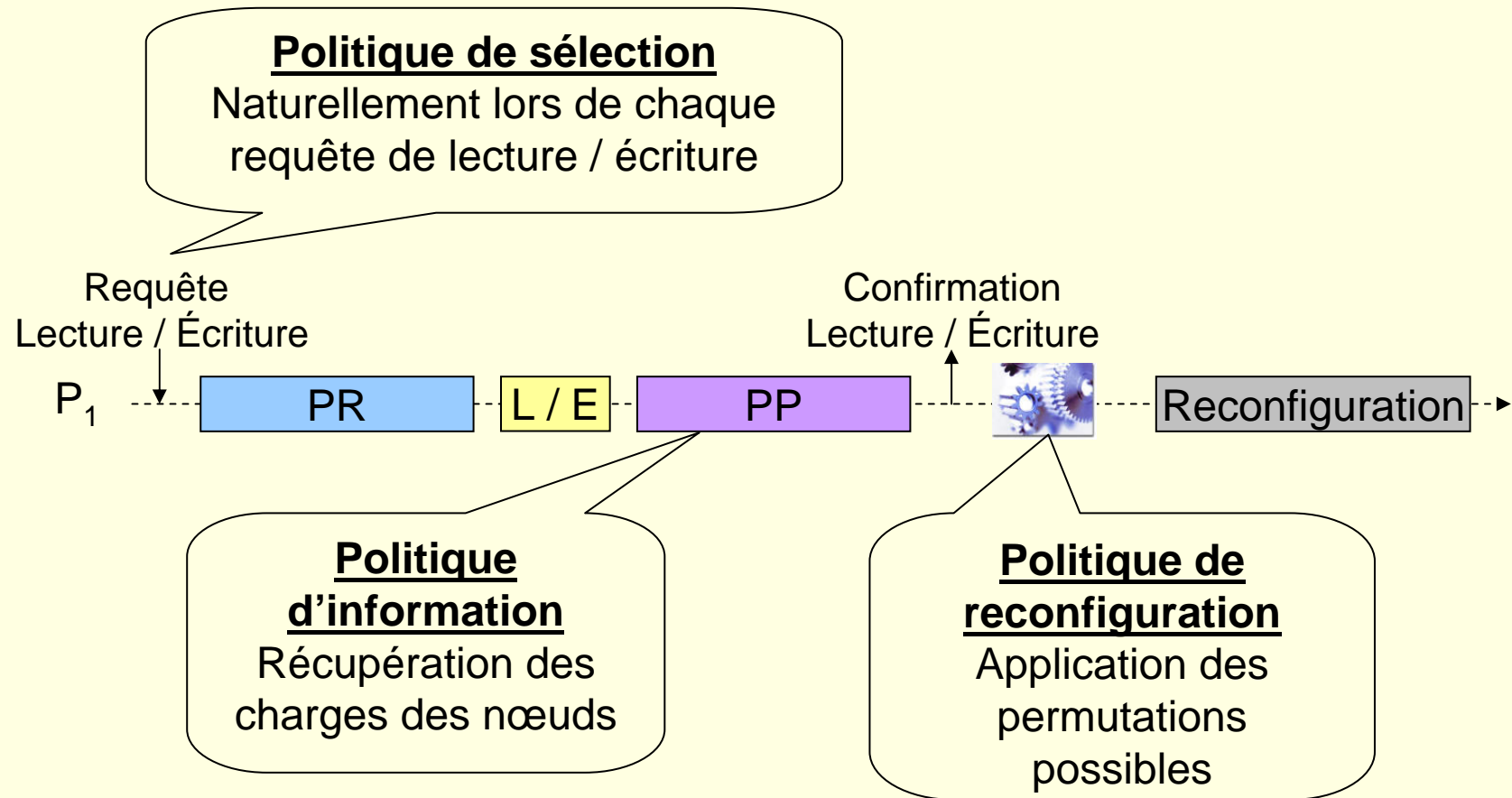


■ Prise en compte des permutations élémentaires dans ce protocole?

Extension de l'Algorithme pour les Permutations Élémentaires

- Une politique d'information
 - Récupérer de façon efficace les informations de charge des noeuds
- Une politique de sélection
 - Définir au mieux le moment de reconfiguration
- Une politique de reconfiguration
 - Choisir le nombre et l'ordre des permutations élémentaires à effectuer

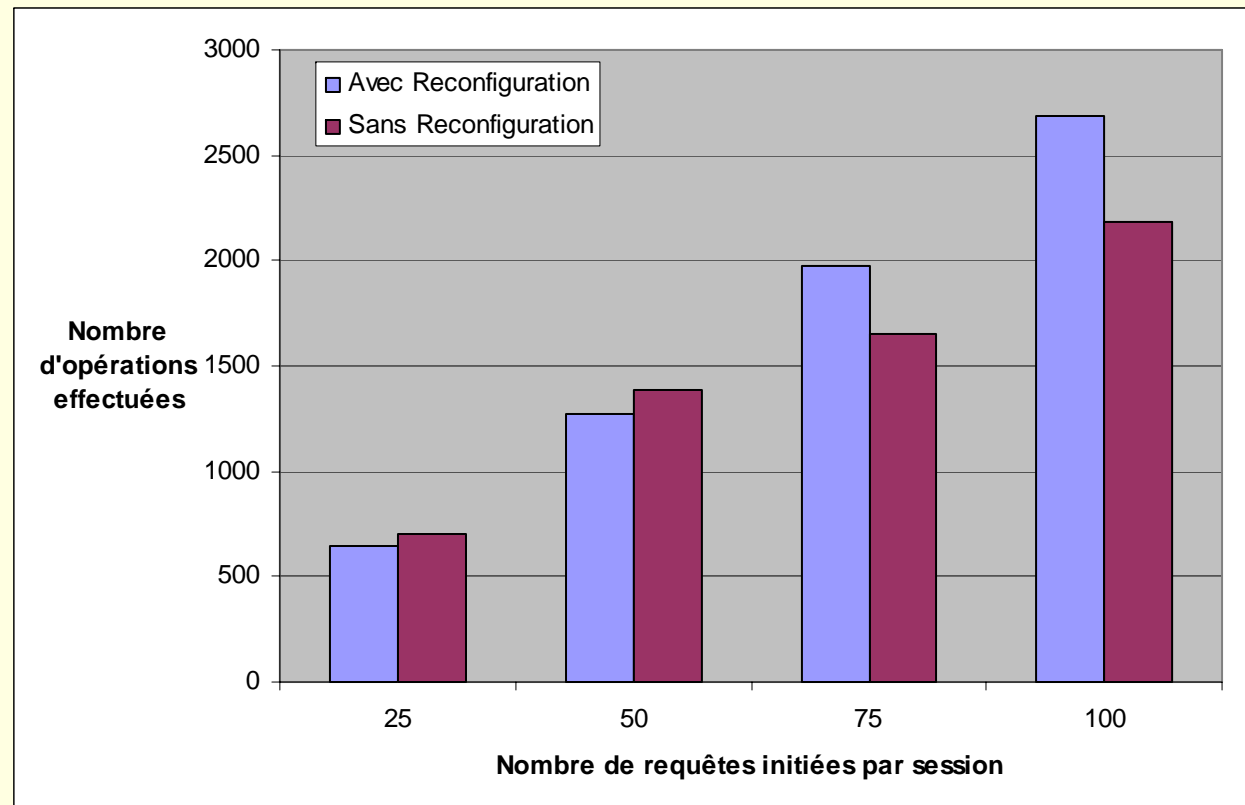
Extension de l'Algorithme pour les Permutations Élémentaires



Évaluation par Simulation

- Implantation de l'algorithme étendu dans le simulateur Neko de Urban, Défago et Schiper [3]
- Tests sur 7 nœuds (une réplique par site)
- Temps de session d'un nœud (durée pendant laquelle un nœud a une charge constante)
- Temps de simulation fixe et comptabilisation du nombre de requêtes effectuées

Résultats de Simulation



Conclusion

- Prise en compte des charges des nœuds dans les protocoles à quorum
- Principe de permutation élémentaire d'une coterie structurée en arbre
- Extension d'un algorithme de lecture / écriture atomique pour la prise en compte des permutations élémentaires
- Amélioration de près de 25% du débit d'opérations effectuées dans certains cas

Perspectives

- Évaluation en environnement réel grâce au projet ViSaGe
 - Quels sont les critères de charge pertinents?
- Comparer à d'autres protocoles à quorum (Grille, Hiérarchique...)
- Utilisation de quorum couplé avec des protocoles de réplication optimiste
- Intégration des temps de transfert pour la reconfiguration

Bibliographie

- [1] **Divyakant Agrawal and Amr El Abbadi.** *An efficient and fault-tolerant solution for distributed mutual exclusion.* ACM Transactions on Computer Systems, 9 (1) : 1-20, February 1991
- [2] **A. Shvartsman and N. Lynch.** *Robust Emulation of shared memory using dynamic quorum-acknowledged broadcasts.* In proceedings of the 27th International Symposium on Fault-Tolerant Computing. FTCS '97
- [3] **Peter Urban, Xavier Défago and André Schiper.** *Neko: A single environment to simulate and prototype distributed algorithms.* In 15th Int'l Conference on Information Networking. ICOIN-15 2001