Laboratoire ID

Système de régulation et d'ordonnancement de requêtes d'E/S au sein des architectures parallèles

Thanh-Trung VAN
(M2R « Systèmes et Logiciels »)

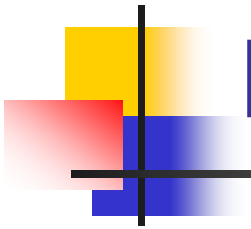
sous la direction de
Adrien LEBRE, Yves DENNEULIN
(Thanh-Trung.Van, Adrien.Lebre, Yves.Denneulin)@imag.fr

INRIA Rhône-Alpes

CNRS Centre National de la Recherche Scientifique

INP Grenoble

Université Joseph Fourier Sciences Technologie Médecine

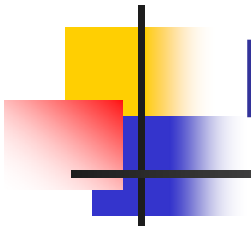


Plan

- Contexte
- aIOLi : librairie d'E/S parallèles
- aIOLi : au niveau grappe
- Résultats
- Conclusion et perspectives

Plan

- Contexte
  - Environnement
  - Notions élémentaires
  - E/S parallèles
- aIOLi : librairie d'E/S parallèles
- aIOLi : au niveau grappe
- Résultats
- Conclusion et perspectives

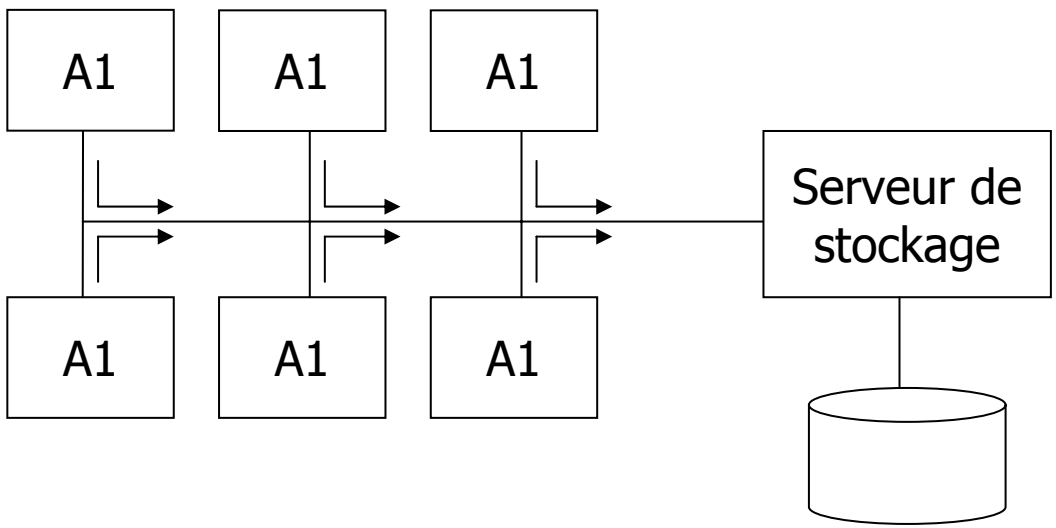


Contexte
Environnement

- Machines parallèles
  - SMP, Grappes, Grilles
- Applications scientifiques *HPC*
  - + puissance de calcul
  - + quantité de données
  - Systèmes de gestion de données spécifiques
  - Accès parallèles

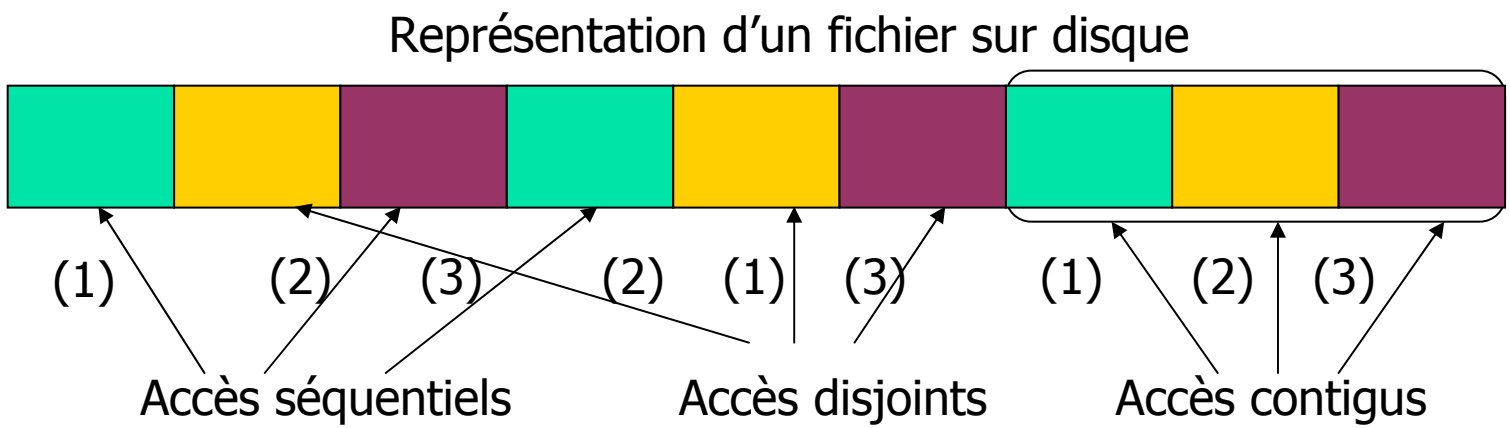
Contexte
Notions élémentaires

- Application parallèle
- Entrées/Sorties parallèles



Contexte
Notions élémentaires

- Types d'accès
  - Accès séquentiels (performance +)
  - Accès contigus (performance +++)
  - Accès disjoints (performance ---)



Contexte
E/S parallèles - Exemple

- Décomposition d'un fichier (3 processus)

Ordre de stockage dans le fichier (par ligne)

P(1,1)	P(2,1)	P(3,1)
P(1,2)	P(2,2)	P(3,2)
P(1,3)	P(2,3)	P(3,3)

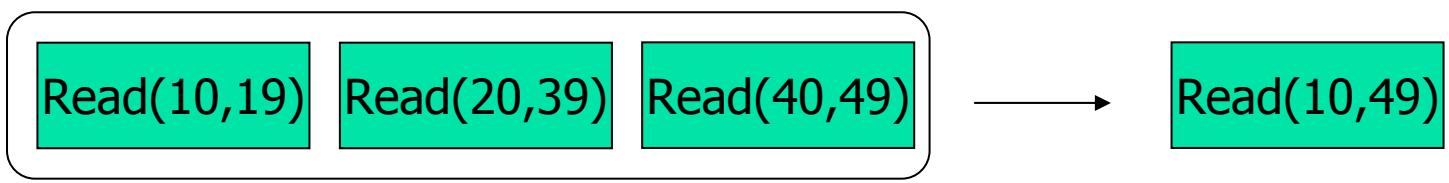
Matrice 3x3

P(1,i)	Données requises par P1
P(2,i)	Données requises par P2
P(3,i)	Données requises par P3

9 accès séquentiels/contigus/disjoints ?? $\Rightarrow$ Inefficace

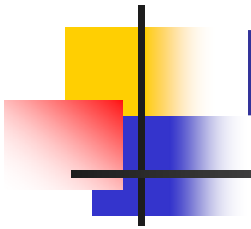
Contexte
E/S parallèles - Gestion des accès

- Ordonnancement de requêtes d'E/S
  - Réordonner dans le but d'optimiser un critère (équité entre les applications, débit de disque …)
- Méthodes d'agrégation
  - Agréger les requêtes pour effectuer des accès plus conséquents



Contexte
Systèmes existants

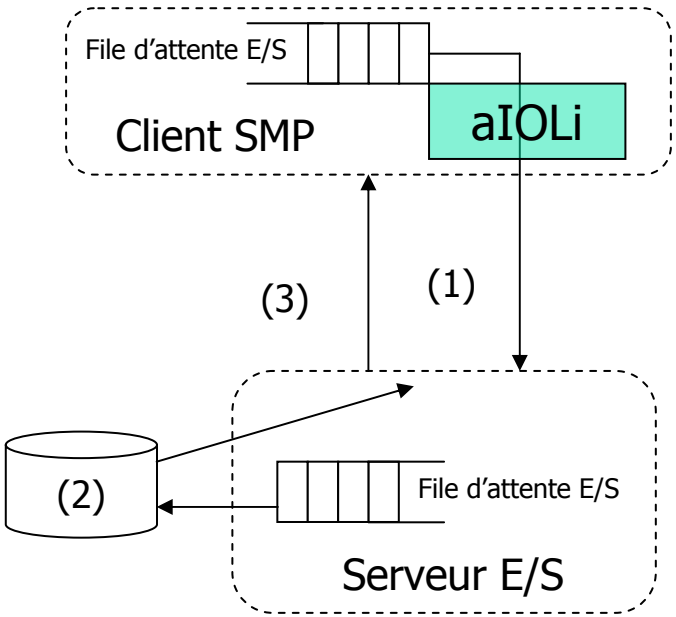
- Systèmes de fichiers parallèles
  - Performants mais +/- complexes, +/- spécifiques (dépendant de l'architecture matérielle), +/- chers
- Librairies E/S spécialisées
  - MPI I/O le standard
  - ROMIO la plus déployé
- APIs sophistiqués (+/- lourde)
  $\Rightarrow$ Besoin de solution simple



Plan

- Contexte
- aIOLi : librairie d'E/S parallèles
  - Principe
  - Evaluation
- aIOLi : au niveau grappe
- Résultats
- Conclusion et perspectives

aIOLi : version existante
Librairie d'E/S parallèles

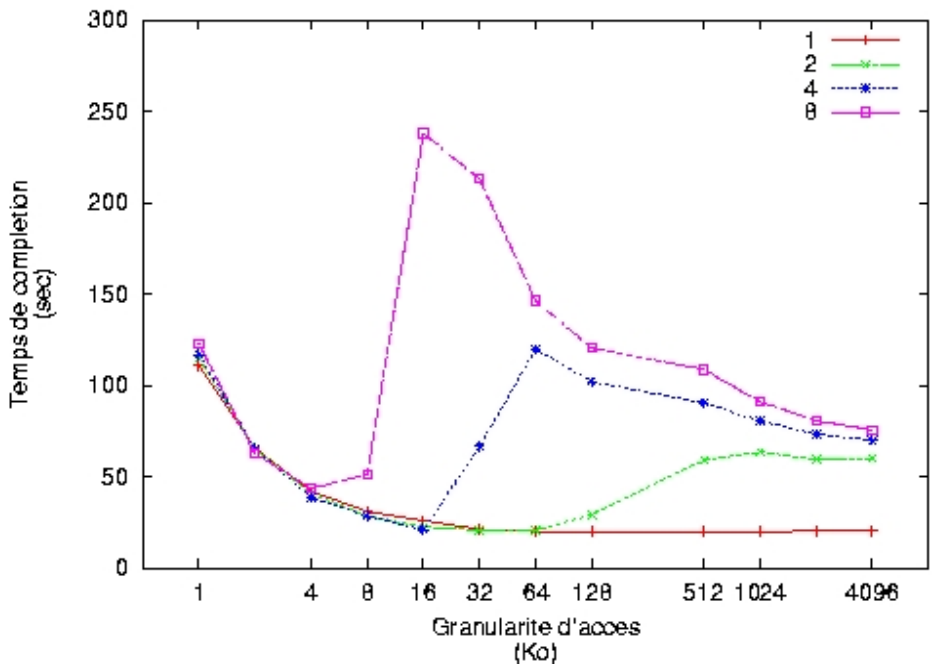


- Principes:
  - Réguler les accès (au sein d'un noeud)
  - Réordonner et agréger (si possible)
- Points forts:
  - Facile à utiliser : surchage des appels POSIX (open/read/write/lseek/close)
  - Portable (sur toutes les architectures POSIX)
  - Efficace

(1) Une requête est transmise au système de stockage
(2) Elle est exécutée au périphérique de stockage rattaché
(3) La réponse est renvoyée au client

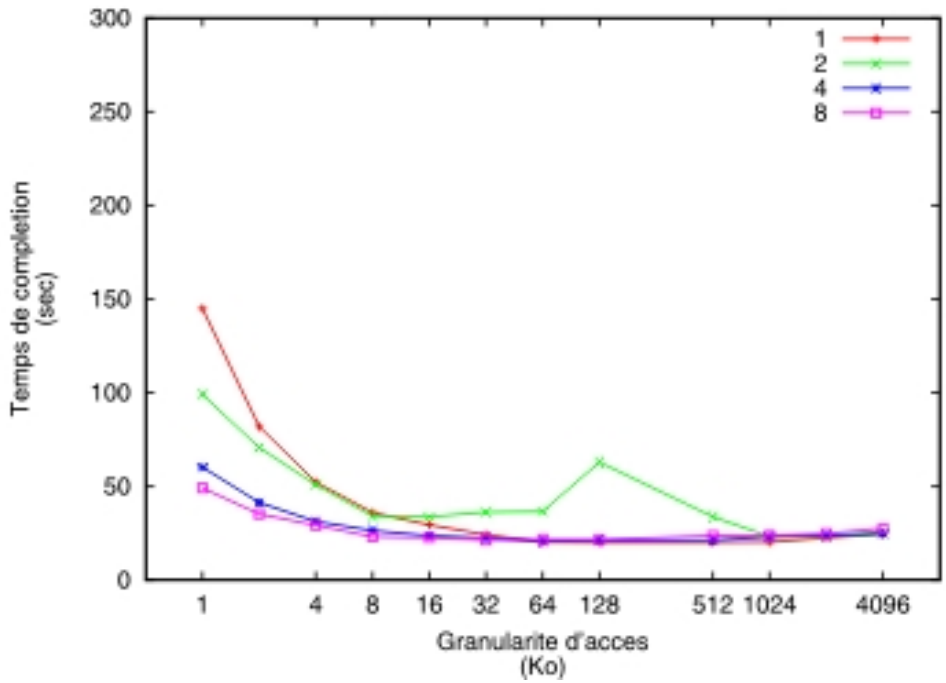
aIOLi : version existante
Evaluation

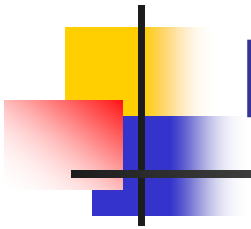
Décomposition d'un fichier de 1Go par 1, 2, 4, 8 processus
Sans aIOLi :+ de processus – de performance



aIOLi : version existante
Evaluation

Décomposition d'un fichier de 1Go par 1, 2, 4, 8 processus
Recompilé avec aIOLi :+ de processus + de performance





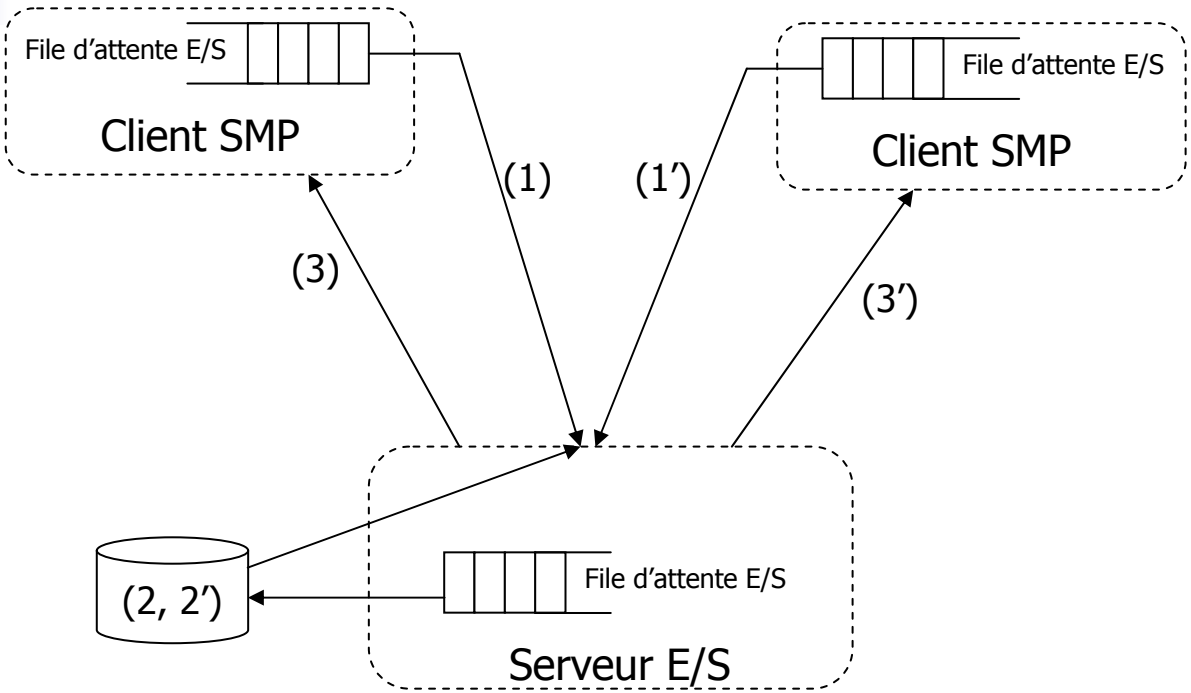
Plan

- Contexte
- aIOLi : librairie d'E/S parallèles
- aIOLi : au niveau grappe
  - Problématique
  - modèle
- Résultats
- Conclusion et perspectives

aIOLi : au niveau grappe Problématique

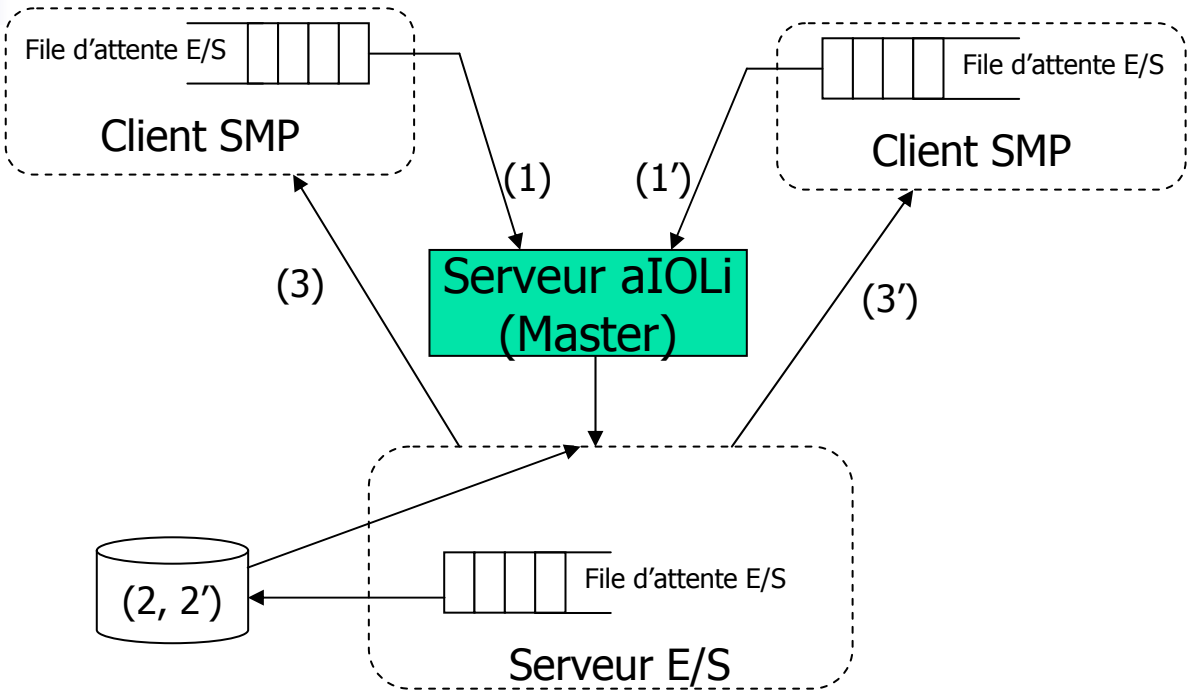
- But : Intervenir à 3 niveaux
  - Coordination intra-nœud (aIOLi à mon arrivée !)
  - Coordination inter-nœud mono-applicative
  - Coordination inter-nœud multi-applicative
- Principe:
  - Synchronisation des E/S provenant de plusieurs nœuds
  - Agrégation des requêtes
  - Ordonnancement des requêtes (mono et multi applications)

aIOLi : au niveau grappe
Synchronisation des accès



(1)(1') Une requête est transmise au système de stockage
(2)(2') Elle est exécutée au périphérique de stockage rattaché
(3)(3') La réponse est renvoyée au client

aIOLi : au niveau grappe
Synchronisation des accès

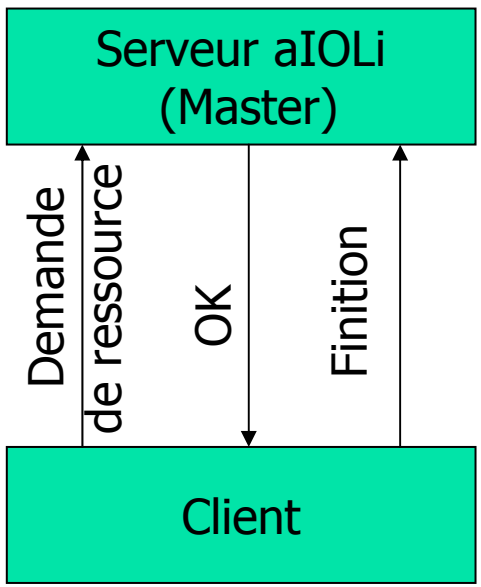


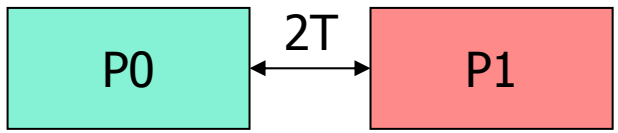
(1)(1') Une requête est transmise au système de stockage
(2)(2') Elle est exécutée au périphérique de stockage rattaché
(3)(3')La réponse est renvoyée au client

Réguler l'arrivée de requêtes ⇒ problème d'exclusion mutuelle distribuée

aIOLi : au niveau grappe
Synchronisation des requêtes

- Approche «simple»



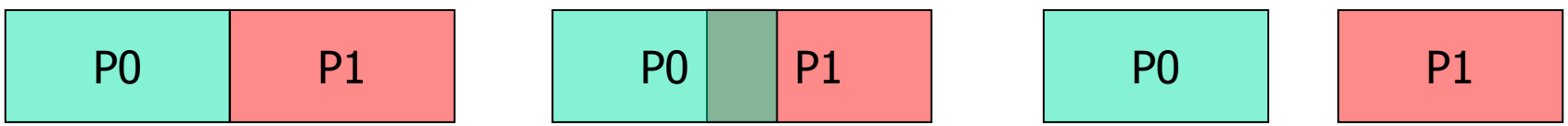


Problème: délai de synchronisation = 2T
T: temps d'envoi un message

aIOLi : au niveau grappe
Synchronisation des requêtes

- Approche améliorée:
  utilisation d'une «Prédiction de temps de transfert»:
  - Calculer le temps d'exécution d'une requête:

    T = taille_requête/débit_disque
  - Problème: précision de la prédiction



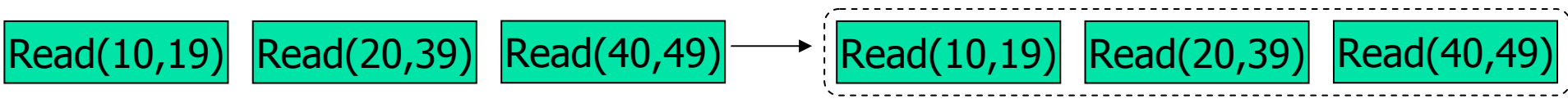
Tréel = Tprévu
a) Cas optimum

Tréel > Tprévu
b) Accès conflits

Tréel < Tprévu
c) Utilisation inefficace du disque

aIOLi : au niveau grappe Agrégation

- Agrégation physique (version précedente d'aIOLi)
  - Requiert des mécanismes de caches distribués (gestion de la cohérence, invalidation des caches, …)
    - Volontairement mis de côté – nécéssite une étude plus approfondie
- Concept d'agrégation virtuelle :
  - L'ordre ne peut être « cassé »
    - Bénéficier des caches clients et serveurs (read ahead)
    - Optimiser le temps d'accès



aIOLi : au niveau grappe
Ordonnancement de requêtes

- 2 algorithmes proposés:
  - Algorithme Shortest Job First (SJF):
    Minimiser le temps d'attente moyen
  - Algorithme Multilevel Feedback (MLF):
    Distribuer équitablement la ressource

    (utilisé pour l'ordonnancement des processus au sein des systèmes Unix)

aIOLi : au niveau grappe Ordonnancement : SJF

- Shortest Job First (SJF):
  Sélectionner la requête la plus petite
  $\Rightarrow$ problème de la famine

- Weighted Shortest Job First (WSJF)
  - Tvirtuel = $\sum Tréel_{(i)} * (M-E)/M$
  - M: Constant; E: Temps d'attente.

aIOLi : au niveau grappe Ordonnancement : MLF

- Variante de Multilevel Feedback :

  Au moment de sélection, chaque requête se voit proposer un quantum de temps q
  - Condition de sélection: temps d'exécution <= q
  - Si son temps d'exécution > q $\Rightarrow$ la prochaine fois le quantum proposé = q*2
  - L'ordre FIFO appliqué si plusieurs requêtes satisfont la condition de sélection

aIOLi : au niveau grappe
Correction des algorithmes

- Problème d'ordonnancement intra-fichier:
  - Les stratégies ne permettent pas de favoriser les agrégations au sein d'un même fichier.

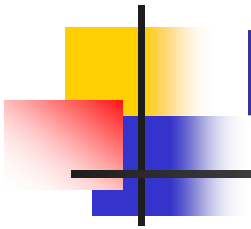
- Entre les requêtes d'un même fichier qui satisfont le critère de sélection

  $\Rightarrow$ mettre en prioritaire la requête ayant le plus petit offset.

  - WSJF: utilisation d'un coefficient de jonction.
  - MLF: remplacer le critère FIFO par le critère d'offset.

aIOLi : au niveau grappe
Gestion des accès

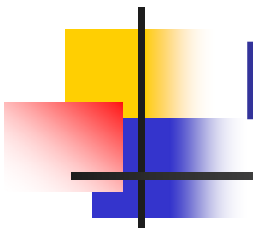
- 1. Détection des agrégations virtuelles (dépendance par offset)
- 2. Application de la stratégie d'ordonnancement
- 3. Emission d'un message de synchronisation vers le client dont la requête a été selectionée



Plan

- Contexte
- aIOLi : librairie d'E/S parallèles
- aIOLi : au niveau grappe
- Résultats
- Conclusion et perspectives

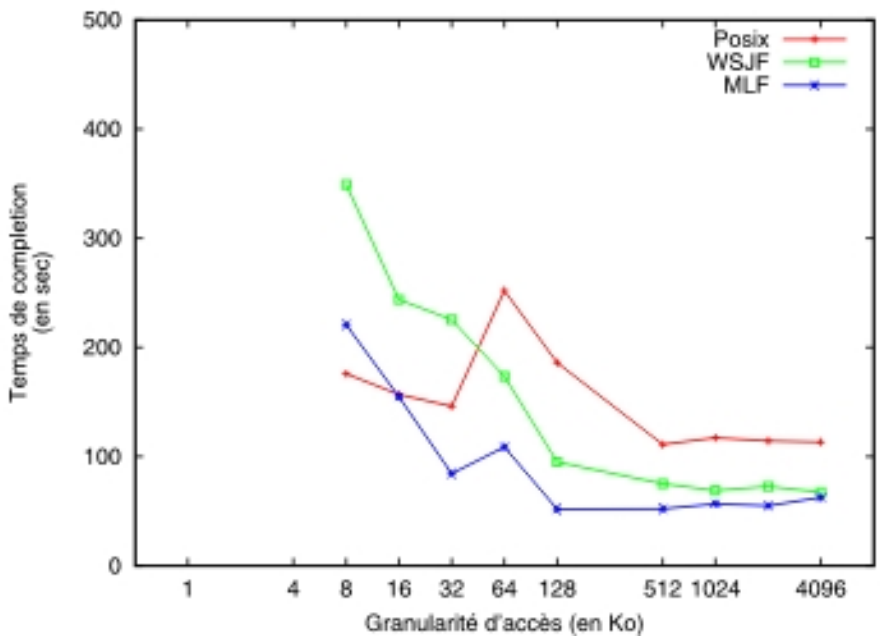
Résultats
Plateforme d'experimentation



- Système de test : grappe IDPOT (laboratoire ID-IMAG)
- Configuration : bi-processeurs (IA32), 1,5 Go RAM
- Application de test : Décomposition de fichiers sur un serveur NFS. (application MPI – MPICH)

Résultats
Mono-application (Sans prédiction)

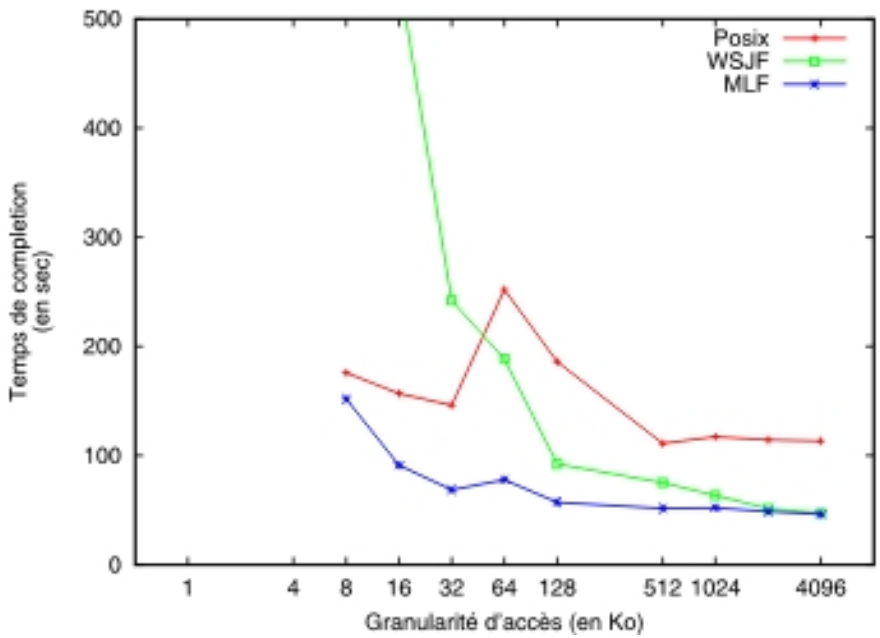
Décomposition d'un fichier de 2Go par une application MPI
(8 instances deployées sur 2 noeuds)



- 3 cas: POSIX, aIOLi avec WSJF et MLF
- Taille < 32 ko :
  pas de performance
  (délai de synchronisation)
- MLF : + efficace que WSJF

Résultats
Mono-application (Avec prédiction)

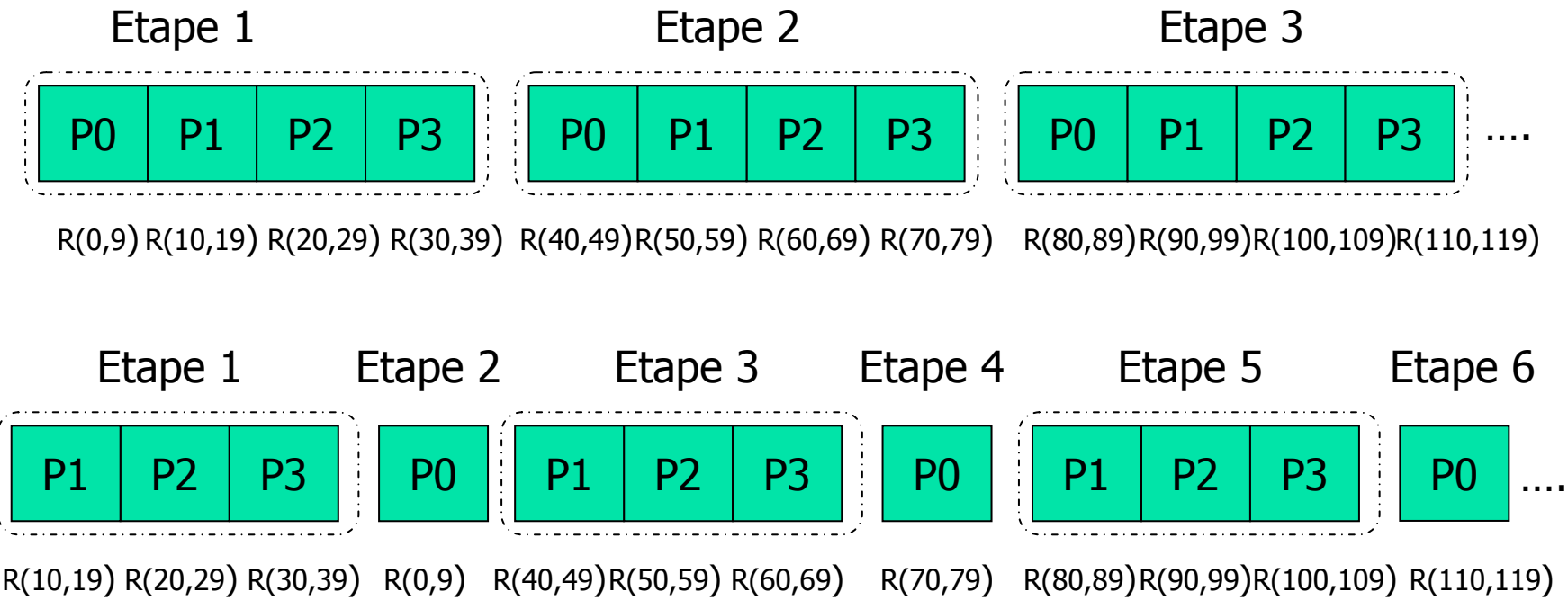
Décomposition d'un fichier de 2Go par une application MPI
(8 instances deployées sur 2 noeuds)



- WSJF: Taille < 64 ko: pas de performance
- MLF: + efficace que WSJF et POSIX
  (toutes granularités)

Résultats
Les problèmes observés

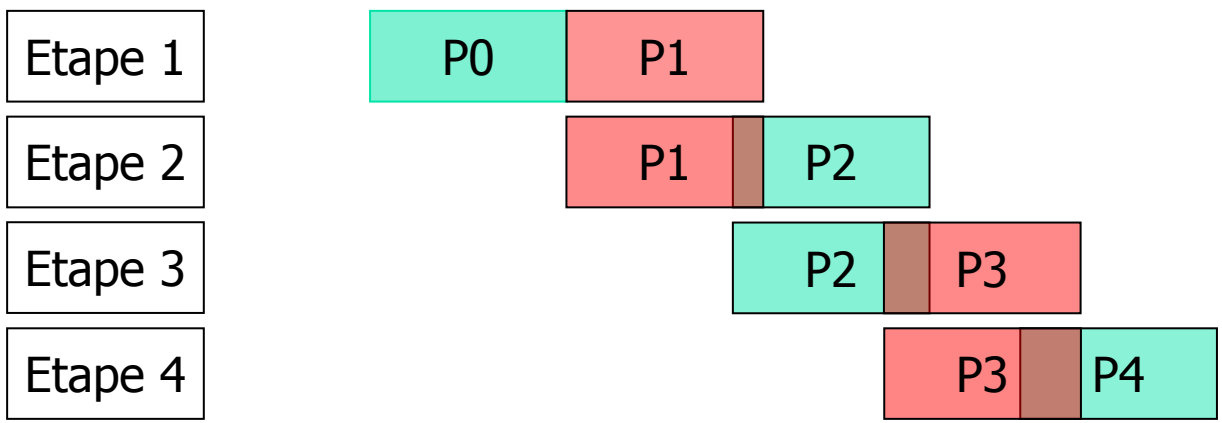
- Phénomène de décalage:



➔ peut diminuer 20% performance

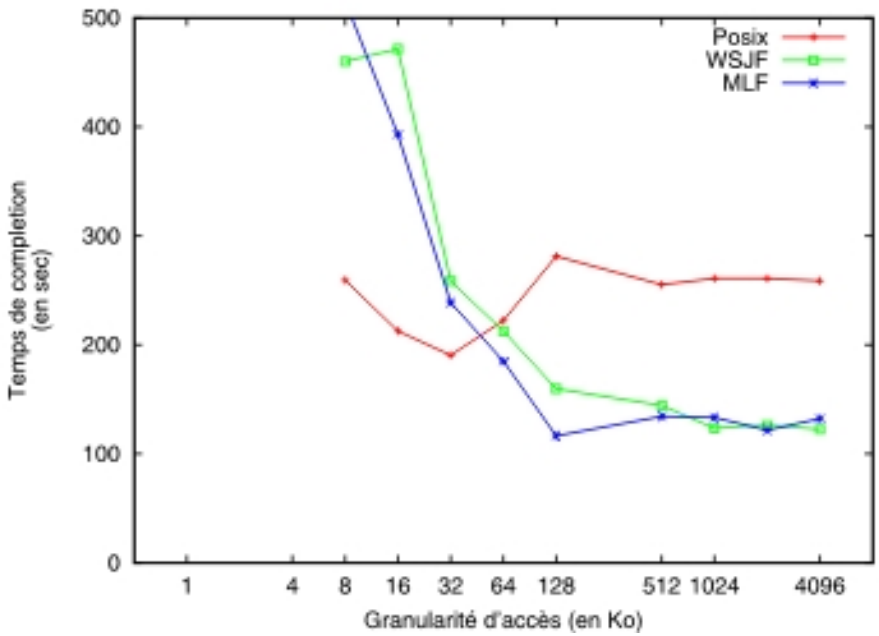
Résultats
Les problèmes observés

- Problème de prédiction: Une prédiction incorrecte peut influencer toutes les requêtes suivantes («*domino effect* »)

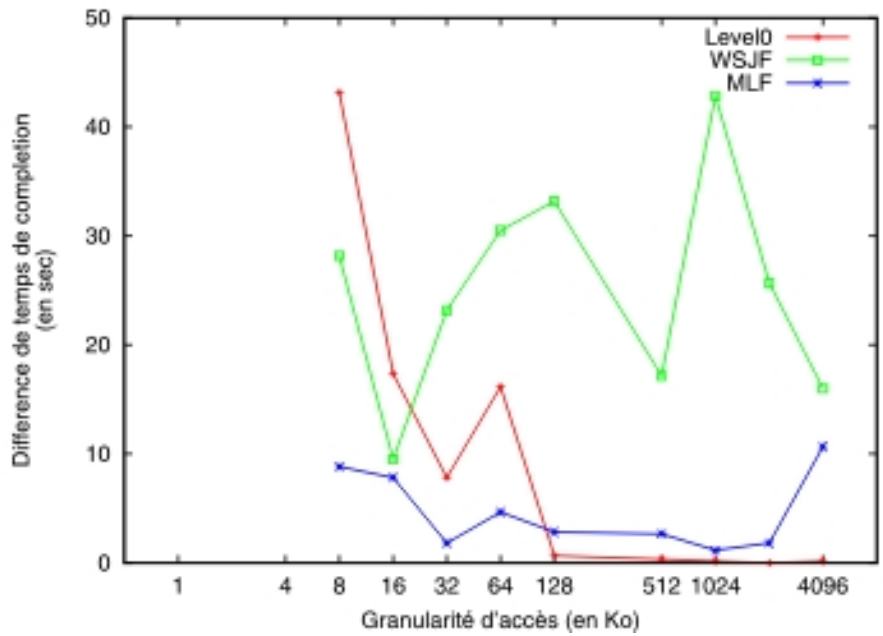


Résultats
Multi-applications (Sans prédiction)

Deux applications (4x2 processus) de décomposition parallèle de 2 fichiers sur le serveur NFS



Temps de complétion

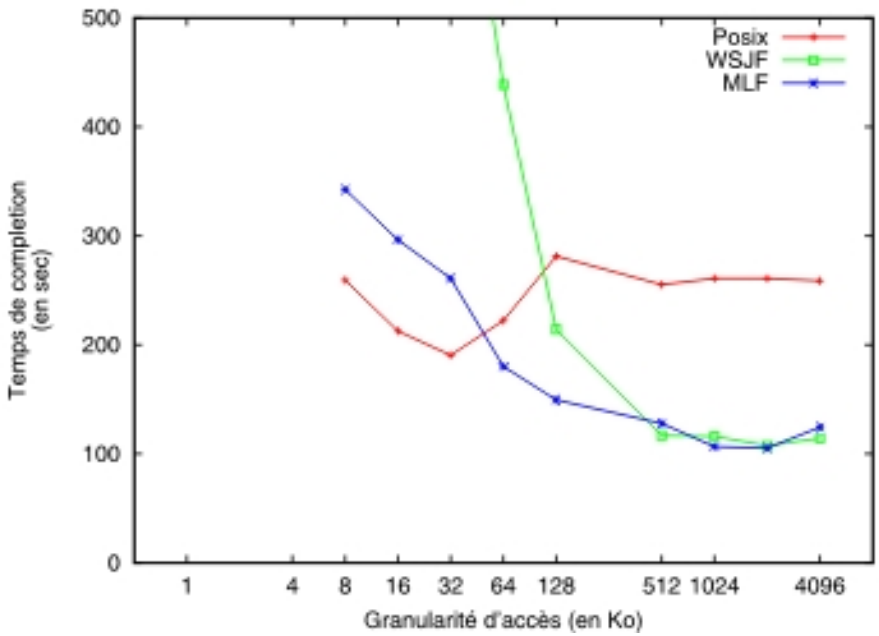


Equité

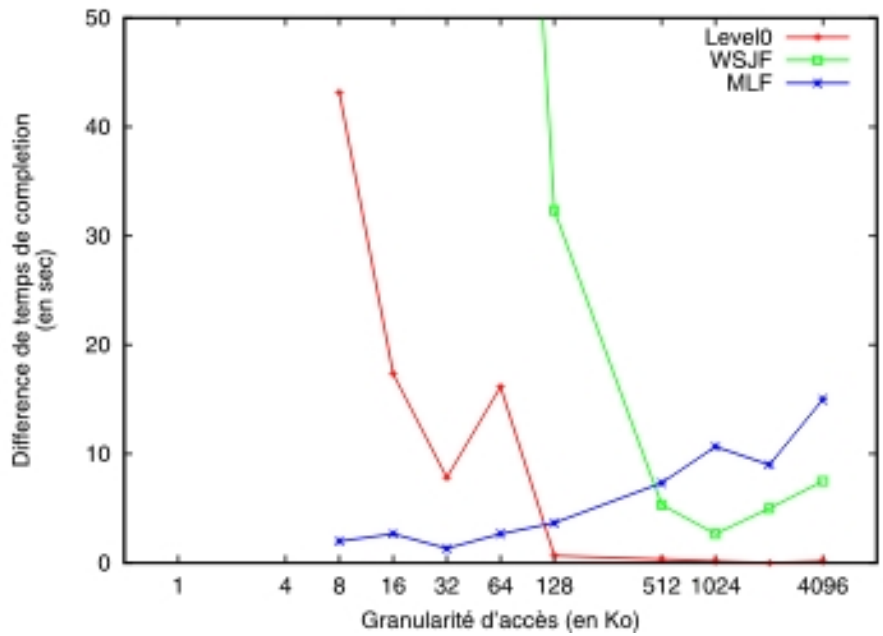
MLF: bon ratio équité/performance

Résultats
Multi-applications (Avec prédiction)

Deux applications (4x2 processus) de décomposition parallèle de 2 fichiers sur le serveur NFS

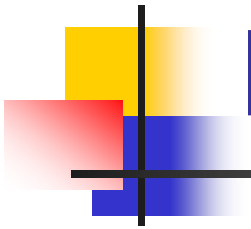


Temps de complétion



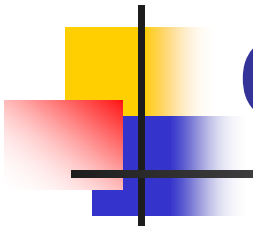
Equité

MLF: Toujours bon ratio équité/performance



Plan

- Contexte
- aIOLi : librairie d'E/S parallèles
- aIOLi : au niveau grappe
- Résultats
- Conclusion et perspectives

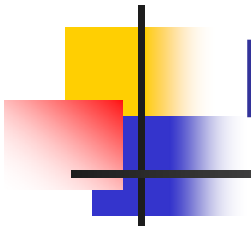


Conclusion

- Prosition prometteuse d'un service de régulation et d'ordonnancement multi-applicatifs de requêtes d'E/S

- WSJF: il y a des choses à faire (expérimentations en cours)
- MLF est prometteur

- Problème avec les petites requêtes pour les deux algorithmes
- Difficulté d'obtenir des prédictions fiables



Perspectives

- Détection des problèmes de décalage.
  - Mise en place d'une fenêtre de reflexion (délai d'attente)
- Analyse et mise en place d'un modèle de prédiction plus fin.
- Etude des contraintes et des coûts de mise en œuvre d'une agrégation physique.
- Mise en en place au niveau des grilles !
  - Etude d'une topologie hiérarchique

Merci de votre attention

Questions ??

http://aioli.imag.fr

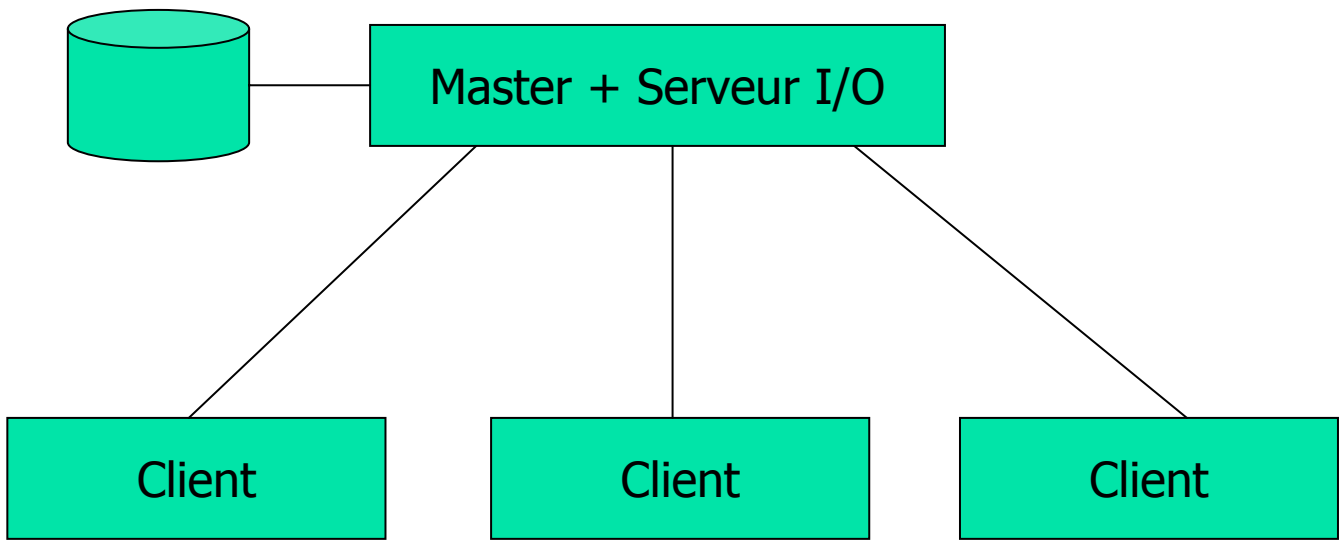
Projet LIPS

BULL- INRIA – Laboratoire ID-IMAG

aIOLi au niveau grappe
Transparents BONUS !

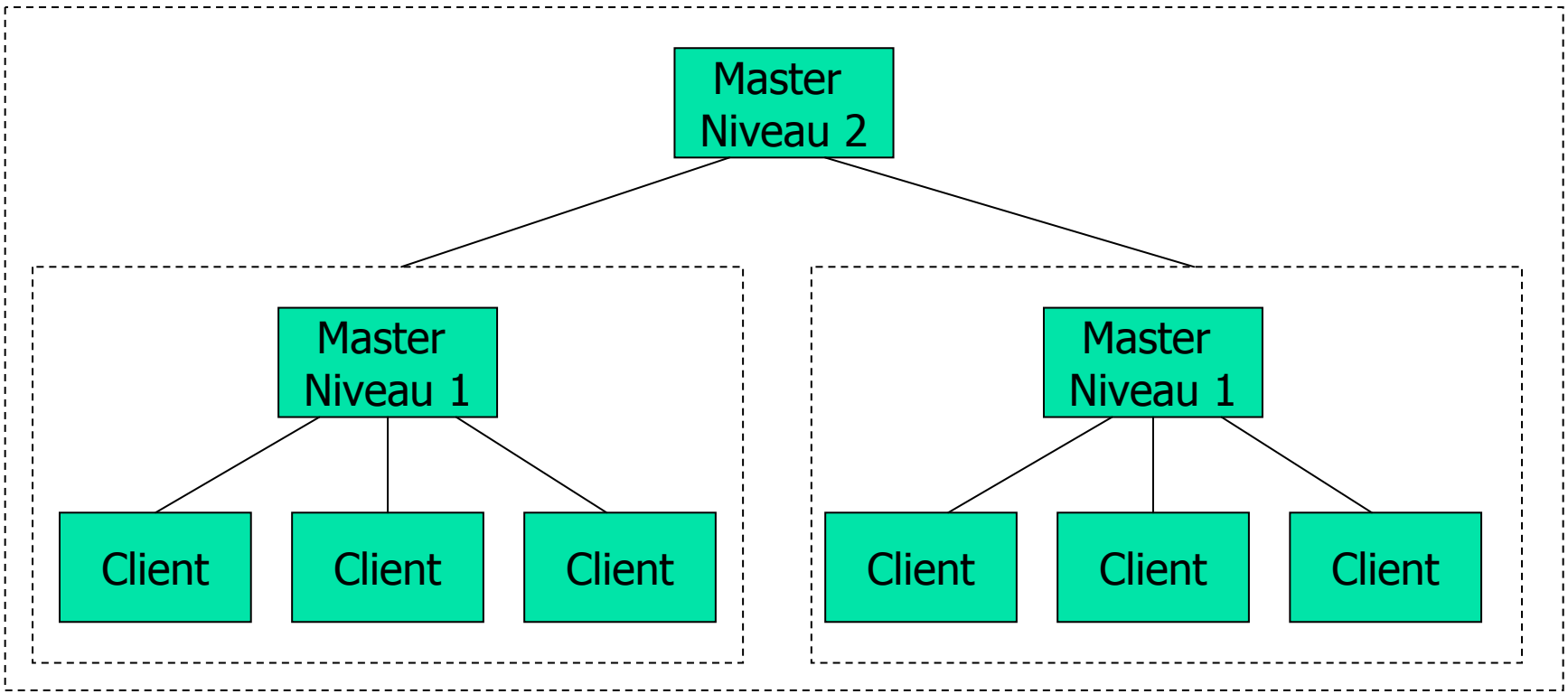
aIOLi : au niveau grappe
Différentes topologies

- Serveur « master »: centraliser et ordonnancer les accès



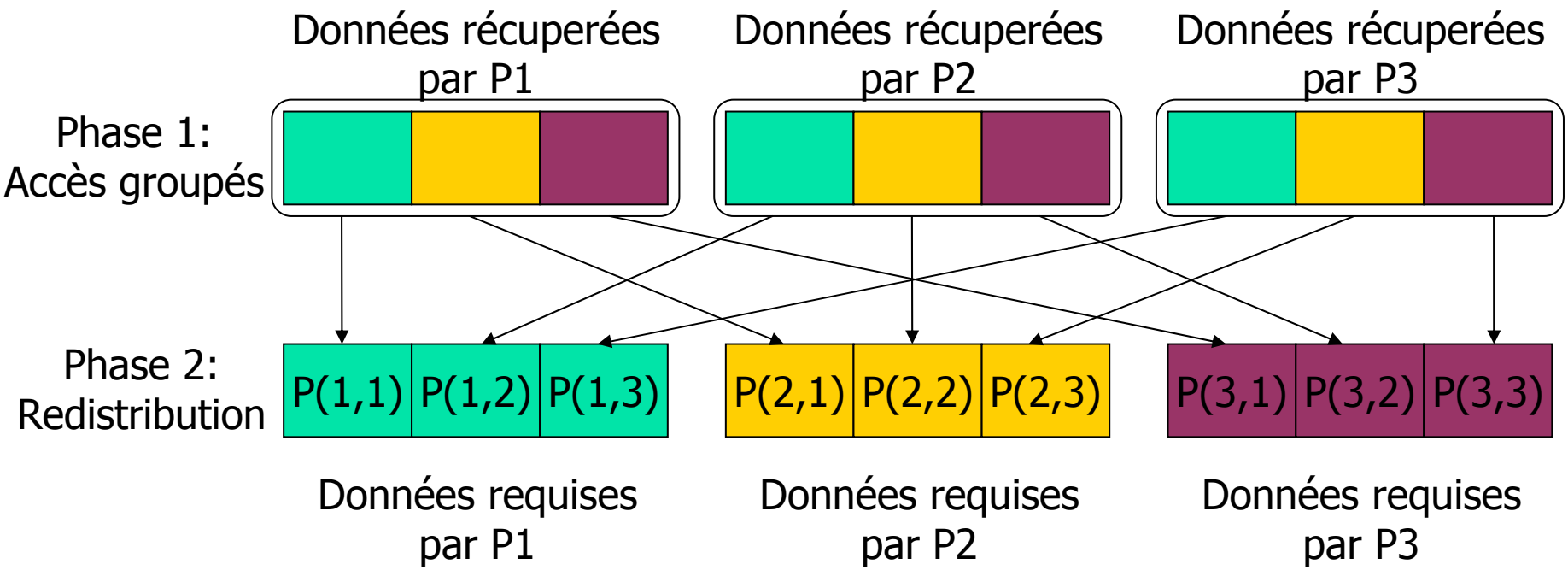
aIOLi : au niveau grappe
Différentes topologies

- Modèle hiérarchique



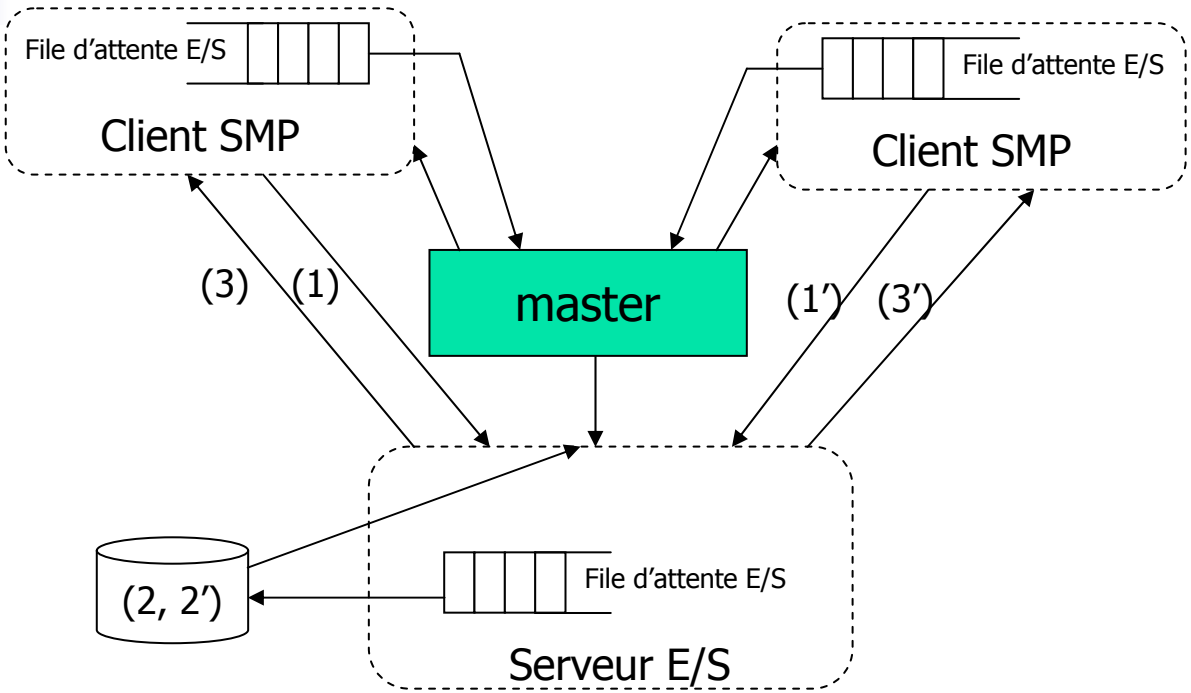
E/S parallèle
Agrégation – méthode collective

- Méthode d'agrégation : « Two-phase I/O »



3 accès contigus ⇒ plus efficace

aIOLi : au niveau grappe
Synchronisation – modèle implanté



(1)(1') Une requête est transmise au système de stockage
(2)(2') Elle est exécutée au périphérique de stockage rattaché
(3)(3')La réponse est renvoyée au client

Réguler l'arrivée de requêtes ⇒ problème d'exclusion mutuelle distribuée

FIN