Gestion dynamique des tâches dans les grappes

une approche à base de machines virtuelles

Fabien Hermenier

Équipe ASCOLA, École des Mines de Nantes

26 novembre 2009

Plan

Plan

Grappe de serveurs

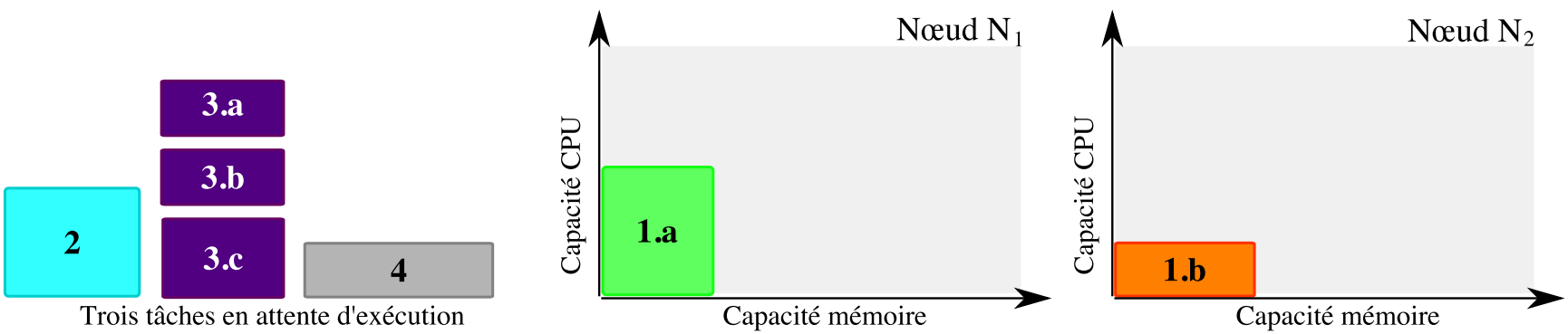
- des machines (nœuds) interconnectées mettent à disposition leurs ressources (matérielles, logicielles)
- des utilisateurs soumettent en continu des tâches dont l'exécution requiert une grande quantité de ressources (calcul scientifique)

Tâches

- des applications souvent distribuées dont les composants peuvent s'exécuter sur des nœuds différents
- les quantités de ressources (matérielles, logicielles) nécessaires à leur exécution sont décrites par l'utilisateur

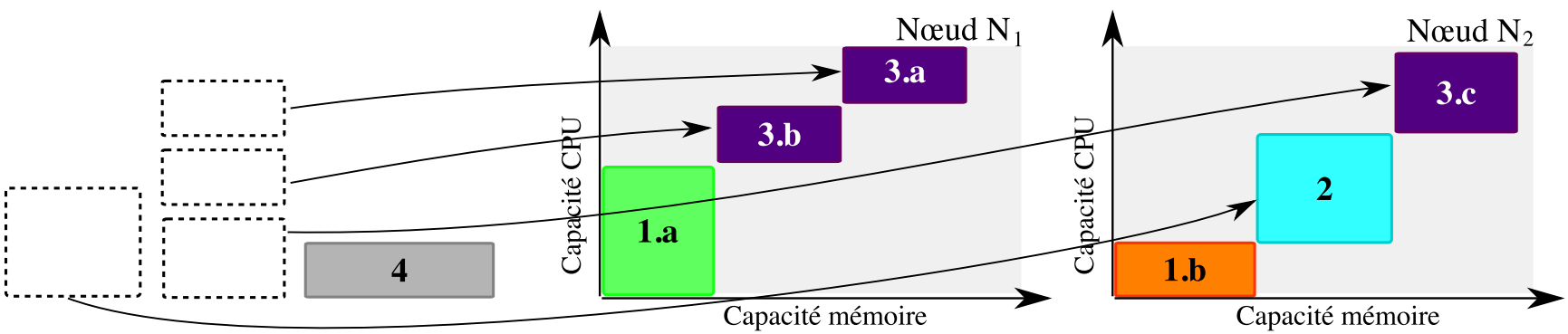
Le gestionnaire de ressources

- service responsable de l'exécution des tâches sur la grappe
- consulte les descriptions des tâches et l'état de la grappe



Le gestionnaire de ressources

- service responsable de l'exécution des tâches sur la grappe
- consulte les descriptions des tâches et l'état de la grappe
- sélectionne avec une stratégie d'ordonnancement
  - les tâches à exécuter
  - les nœuds qui hébergeront les composants des tâches



Des besoins en ordonnancement variés

Des objectifs différents

- différentes stratégies d'ordonnancement
  - réduire le temps d'attente avant l'exécution des tâches
  - réduire le nombre de nœuds nécessaire à l'exécution des tâches
  - *etc.*
- des contraintes de placement des composants spécifiques
  - placement relativement à des nœuds ou d'autres composants
  - partitionnement de la grappe en fonction des utilisateurs
  - *etc.*

Des approches actuelles peu flexibles

- différentes heuristiques *ad-hoc*
- des stratégies d'ordonnancement au champ d'application limité

Différentes approches pour l'ordonnancement

Gestion statique des tâches

(ex : EDF *[Liu et al., JACM'73]*, EASY *[Lifka, IPPS'95]*)

- allocation statique des ressources aux composants
- exécution sans préemption de chaque tâche
- avantages
  - des opérations techniquement simples
  - des ordonnancements reproductibles
- inconvénients
  - réservation fixe de ressources pour une utilisation potentiellement variable
  - tendance à la sur-réservation

Différentes approches pour l'ordonnancement

Gestion dynamique des tâches

(ex : gang-scheduling, co-scheduling [Feitelson et al., 95+])

- allocation dynamique des ressources aux composants des tâches
- actions de manipulation à la volée du placement des composants et de l'état des tâches : *migration*, *préemption*
- avantages
  - allocation des ressources en fonction des besoins
  - optimisation en continu de l'ordonnancement
- inconvénients
  - des ordonnancements non reproductibles
  - une mise en œuvre difficile

Gestion dynamique des tâches

Des opérations techniquement complexes

- allocation dynamique des ressources
  - l'application doit s'adapter à une disponibilité variable des ressources
- migration, préemption
  - l'intégrité de l'application doit être assurée

Des solutions limitées

- les approches *ad-hoc* nécessitent l'adaptation des applications
- les approches génériques supportent un type d'application restreint (ex : BLCR*[Hargrove et al., SCIDAC'06]*)
- des actions coûteuses en temps et en ressources

Problématique de la thèse

Les approches dynamiques : théoriquement plus efficaces, négligées en pratique

- des algorithmes d'ordonnancement peu adaptables
- un support pour l'exécution contraignant
- une manipulation complexe des tâches

Contributions

- une approche flexible pour l'implémentation d'ordonnanceurs
  - utilisation de la programmation par contraintes
- un support adapté à la gestion dynamique des tâches
  - utilisation des machines virtuelles
- une manipulation efficace des tâches
  - la reconfiguration dynamique

Plan

Ordonnancement flexible avec des contraintes

La programmation par contraintes

- une approche pour la modélisation et la résolution de problèmes combinatoire
- l'utilisateur modélise son problème, le solveur le résoud
- des contraintes standard, réutilisables et composables comme briques élémentaires des modèles
- une méthode de résolution exacte, générique, mais qui peut prendre du temps pour des problèmes complexes

Ordonnancement flexible avec des contraintes

Notre approche

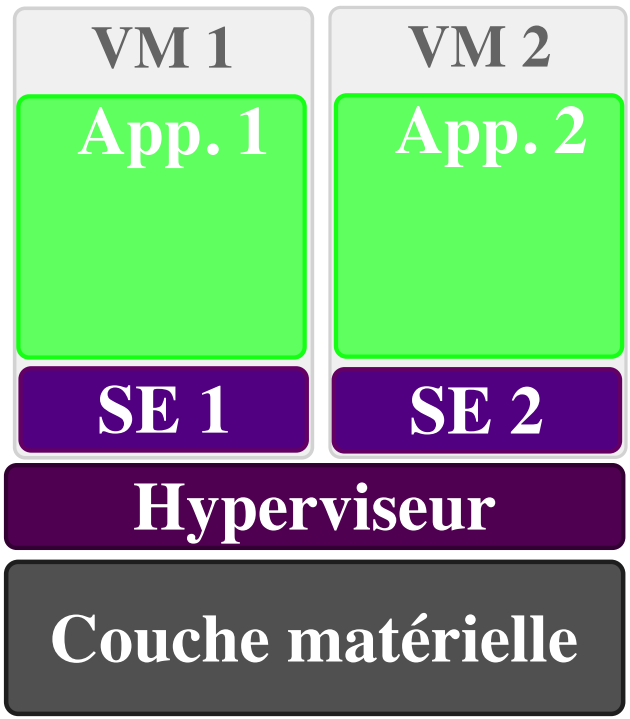
- une modélisation
  - de la grappe et de ses ressources
  - des tâches (états, besoins en ressources)
  - de l'affectation des composants des tâches aux noeuds
- des contraintes additionnelles pour définir des stratégies d'ordonnancement spécifiques
  - forcer l'état des tâches
  - forcer/interdire le placement de composants sur des nœuds
  - corréler les placements de groupes de composants

Plan

Une architecture reposant sur les machines virtuelles

La virtualisation *[Popek et Golberg, 1974]*

- un nouveau niveau d'abstraction virtualise la couche matérielle
- les applications et l'environnement d'exécution sont isolés dans des machines virtuelles (VMs)
- l'hyperviseur contrôle l'accès aux ressources des VMs



Une architecture reposant sur les machines virtuelles

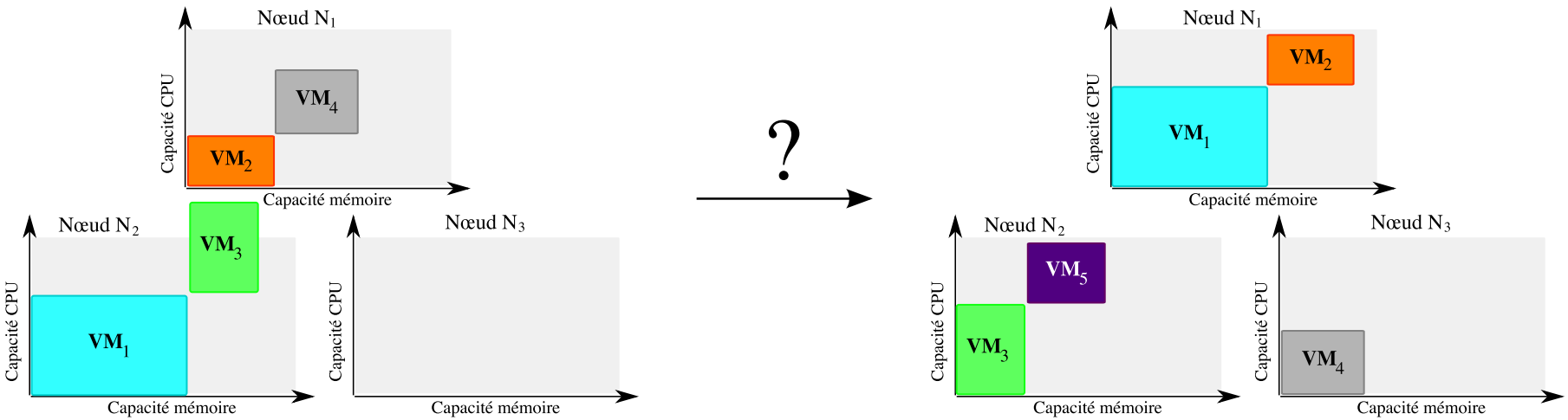
Chaque composant d'une tâche est exécuté dans une machine virtuelle

- l'effort d'adaptation des applications à la grappe est réduit
- le contrôle des tâches est transparent et non-invasif
- les primitives pour une gestion dynamique des tâches existent au niveau de l'hyperviseur
  - ré-agencement : migration à chaud *[clark et al., NSDI'05]*
  - préemption : suspension sur disque, reprise

Plan

Reconfiguration dynamique

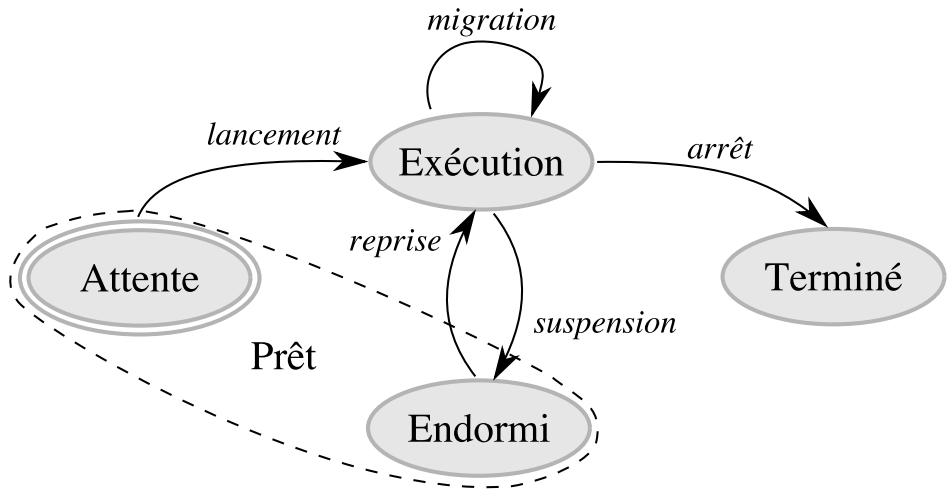
- ré-agencer les tâches
  - les besoins en ressources des composants ne sont plus satisfaits
  - un meilleur ordonnancement est possible
- problématiques
  - planifier les actions manipulant les composants (les VMs)
  - exécuter la reconfiguration le plus rapidement possible



Cycle de vie des machines virtuelles

Des actions

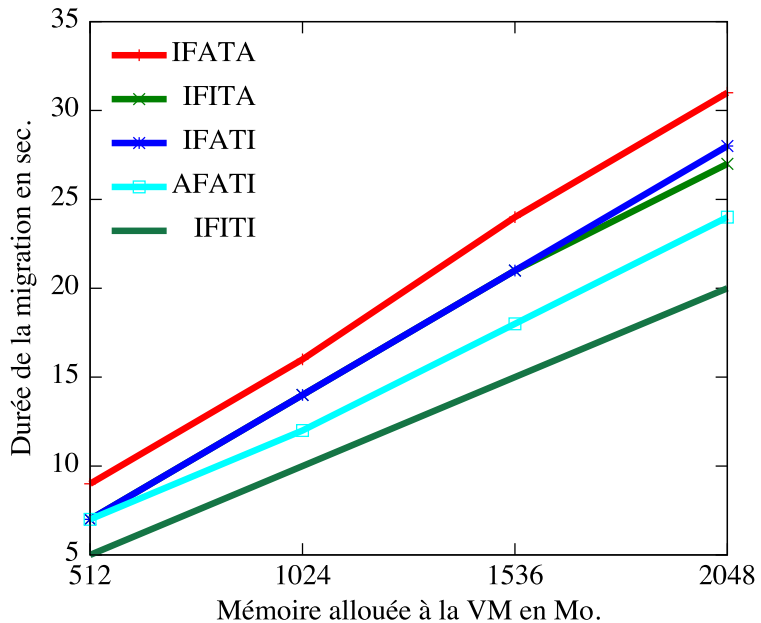
- pour changer l'état d'une VM
- pour re-positionner une VM



La reconfiguration dynamique

Exécuter une action

- à un coût
  - temps d'exécution

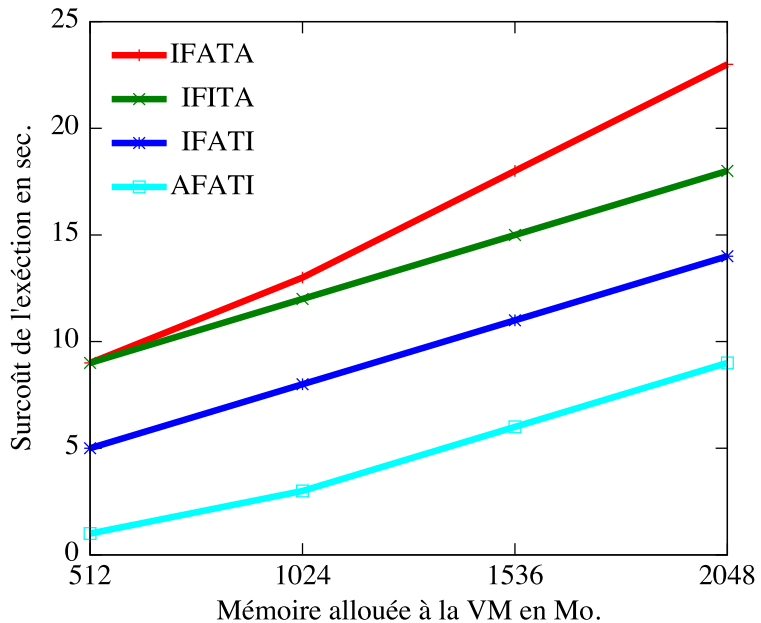


(a) Temps d'exécution d'une migration selon différents contextes

La reconfiguration dynamique

Exécuter une action

- à un coût
  - temps d'exécution
  - impact sur les performances des applications
- une fonction évalue le coût (temporelle) de chaque action



(b) Surcoût lié à une migration selon différents contextes

La reconfiguration dynamique

Exécuter une action

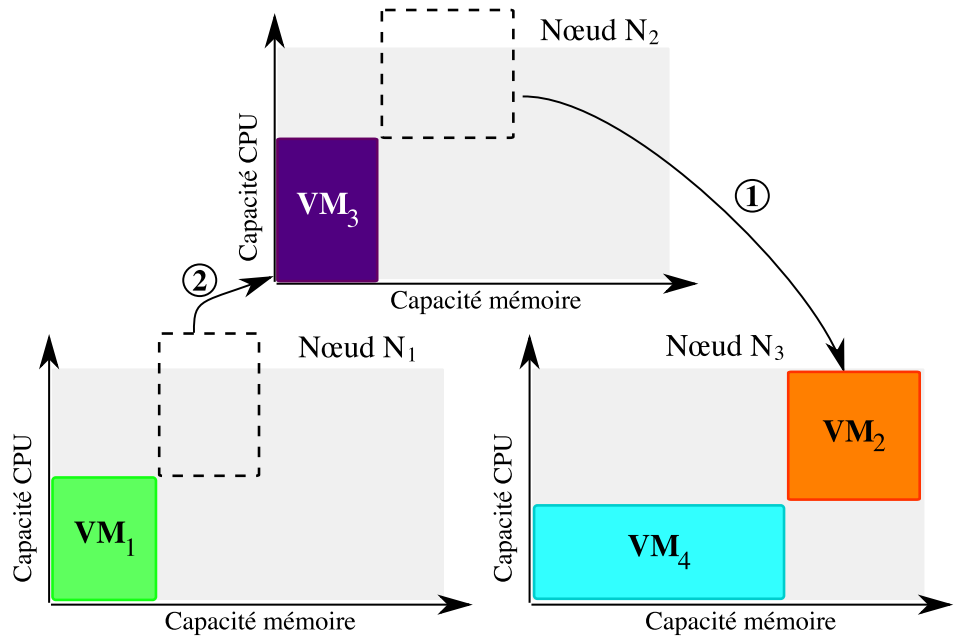
- à un coût
  - temps d'exécution
  - impact sur les performances des applications
- une fonction évalue le coût (temporelle) de chaque action
- impacte sur la disponibilité des ressources des nœuds

Action	Impact sur les ressources	
	Consommateur	Libérateur
Lancement	✓	
Arrêt		✓
Suspension		✓
Reprise	✓	
Migration	✓	✓

Planifier la reconfiguration

Assurer la faisabilité des actions

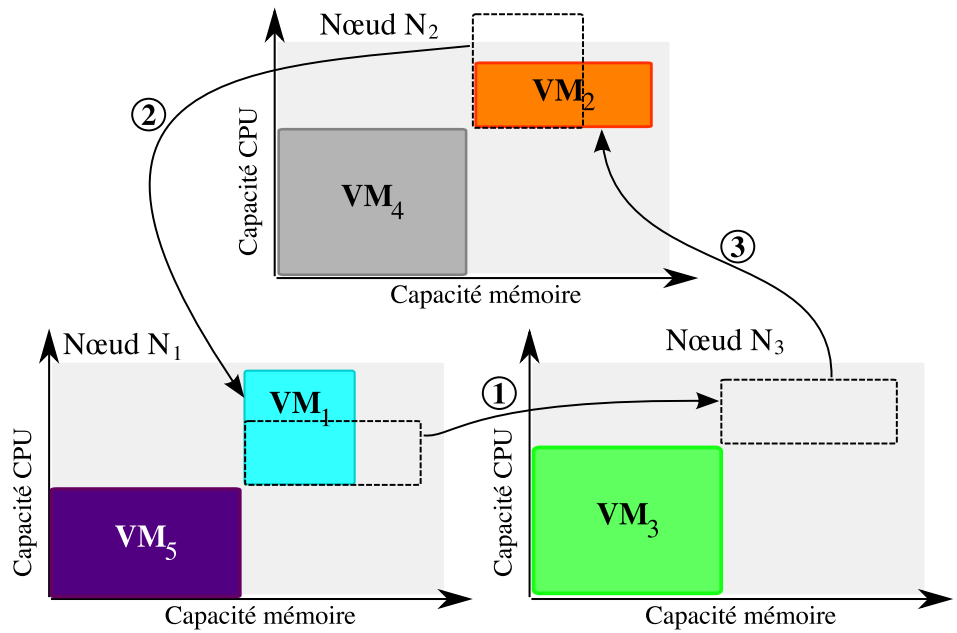
- des actions doivent être séquencées



Planifier la reconfiguration

Assurer la faisabilité des actions

- des actions doivent être séquencées
- des migrations supplémentaires sur un nœud pivot cassent des inter-dépendances



Optimiser la reconfiguration

Un besoin de réactivité

- les besoins en ressources des tâches varient dans le temps
- une reconfiguration doit être rapide pour rester cohérente
  - la somme des coûts des actions doit être la plus faible possible

Des solutions pour réduire le temps de reconfiguration

- exécuter le moins d'actions possibles
- exécuter les actions les moins coûteuses
- paralléliser les actions

Optimiser la reconfiguration

Notre approche

- des contraintes fixent l'état des VMs de la nouvelle configuration en fonction des états souhaités des tâches
- des contraintes assurent la disponibilité des ressources pour les VMs
- une heuristique calcule un plan réalisable de parallélisme maximum associé à une configuration
- une fonction objectif sélectionne la configuration de plan le moins coûteux

Plan

Entropy

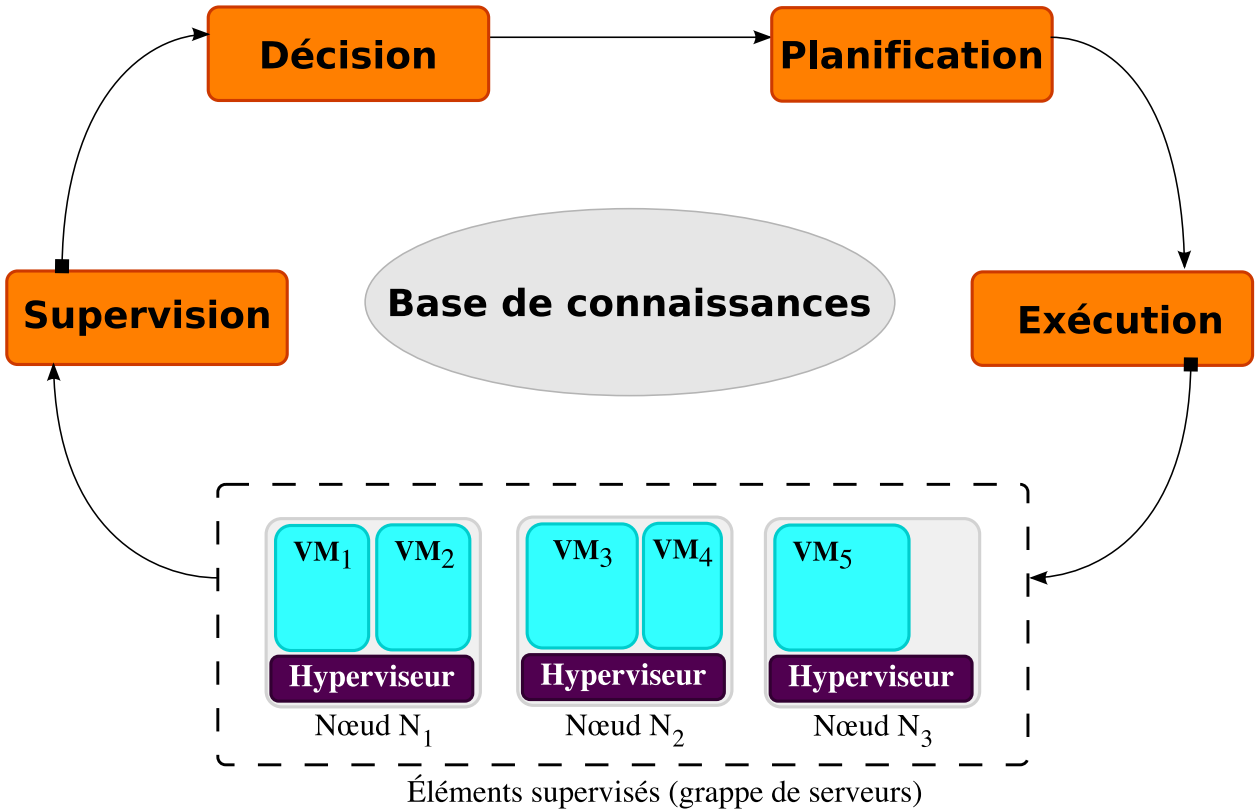
Objectifs

- un prototype pour la validation de nos travaux
- une auto-optimisation en continu de l'état et de la position de machines virtuelles

Deux scénarios développés durant cette thèse

- la consolidation dynamique
- l'ordonnancement flexible de tâches

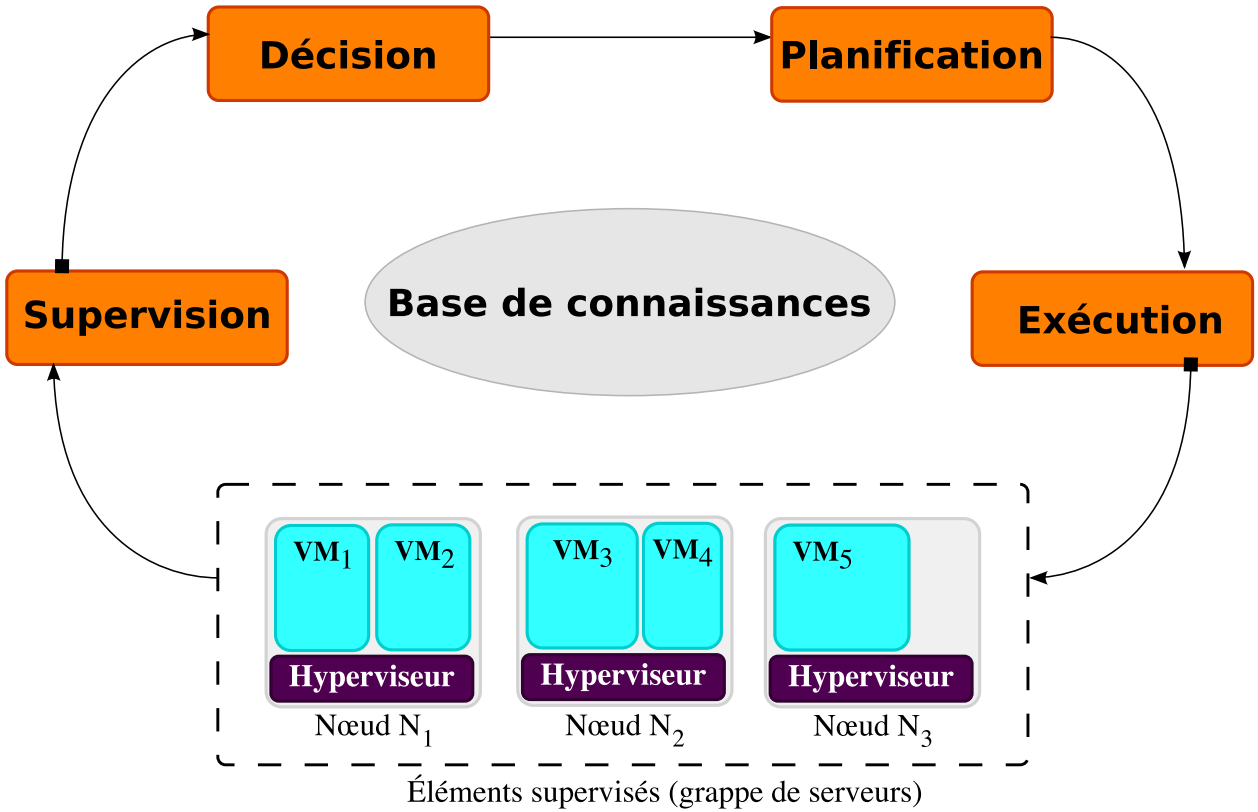
Boucle d'auto-adaptation



Base de connaissances

- une modélisation de l'ordonnancement de tâches sur une grappe avec le solveur de contraintes Choco
- une API pour définir des contraintes sur le modèle

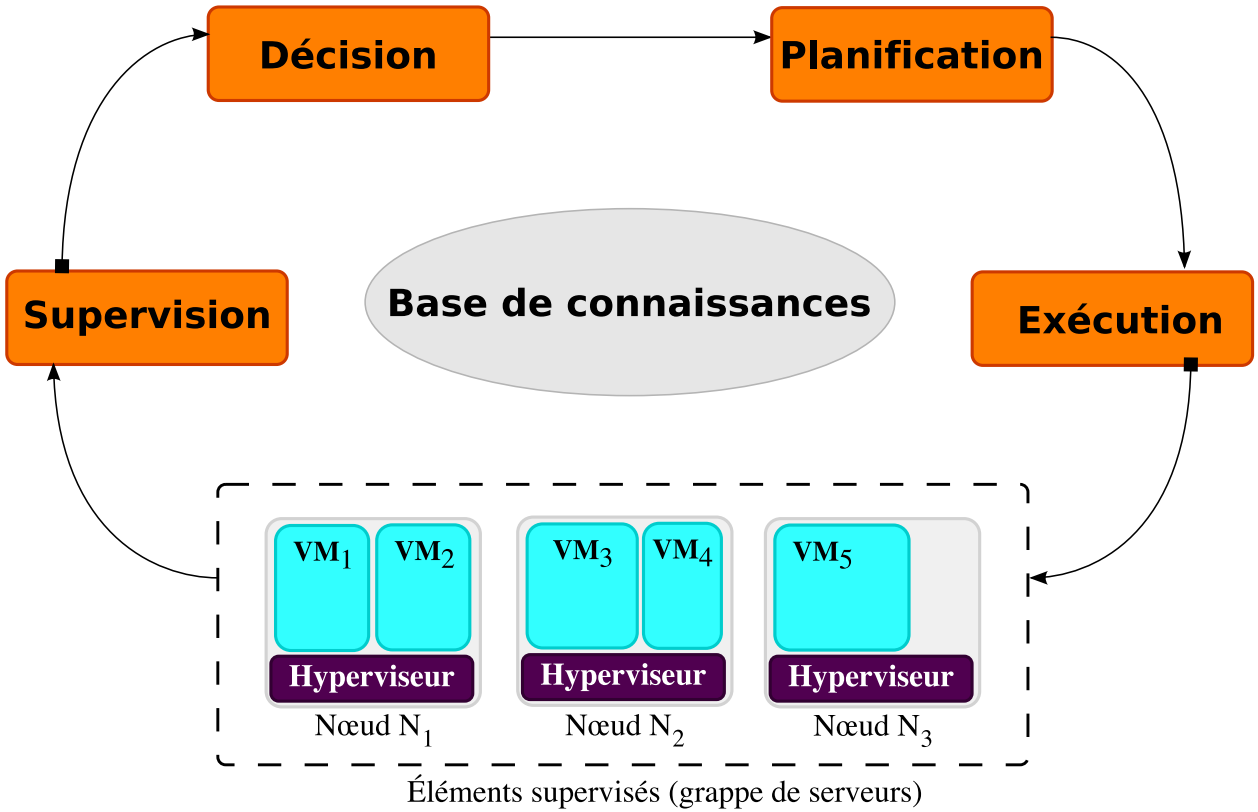
Boucle d'auto-adaptation



Supervision

- interrogation d'un système de supervision pré-installé (actuellement Ganglia)
- capture l'état courant de la grappe

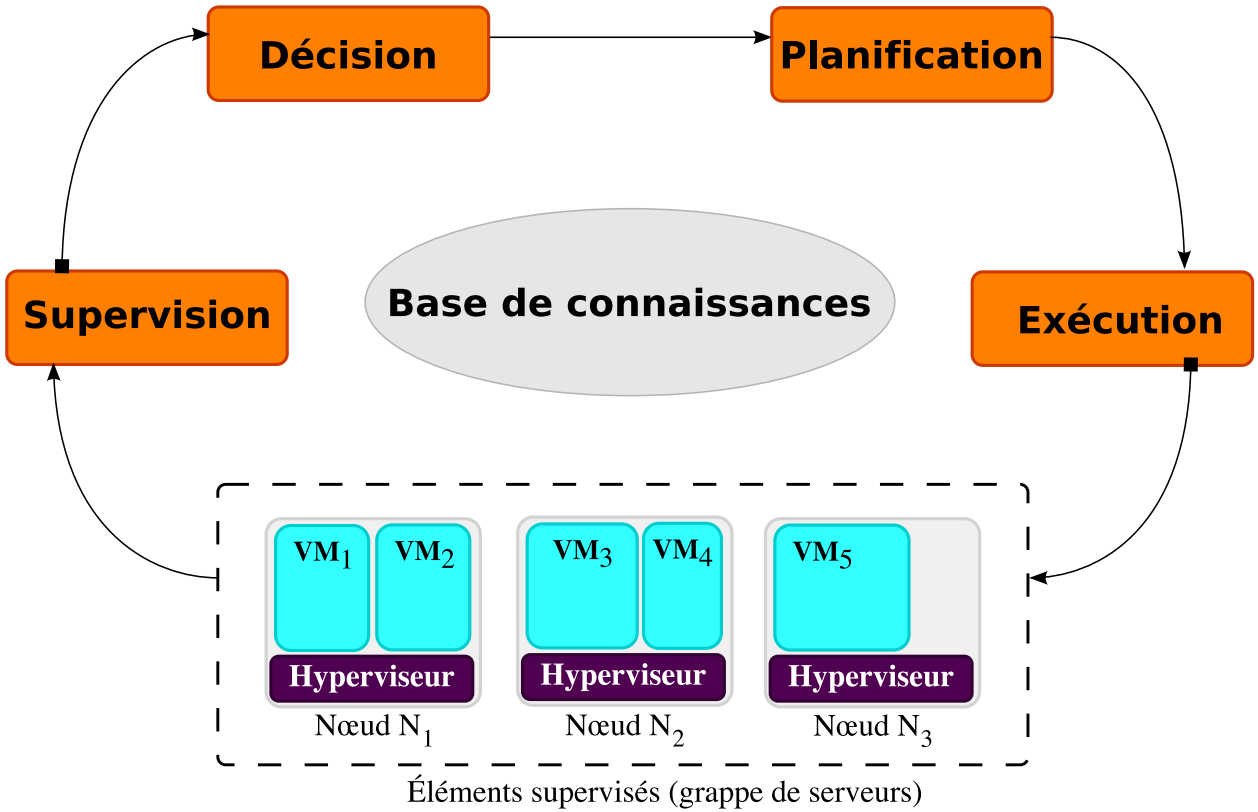
Boucle d'auto-adaptation



Décision

- analyse l'état courant de la grappe
- déduit un nouvel état pour les tâches satisfaisant des contraintes
- un module personnalisable

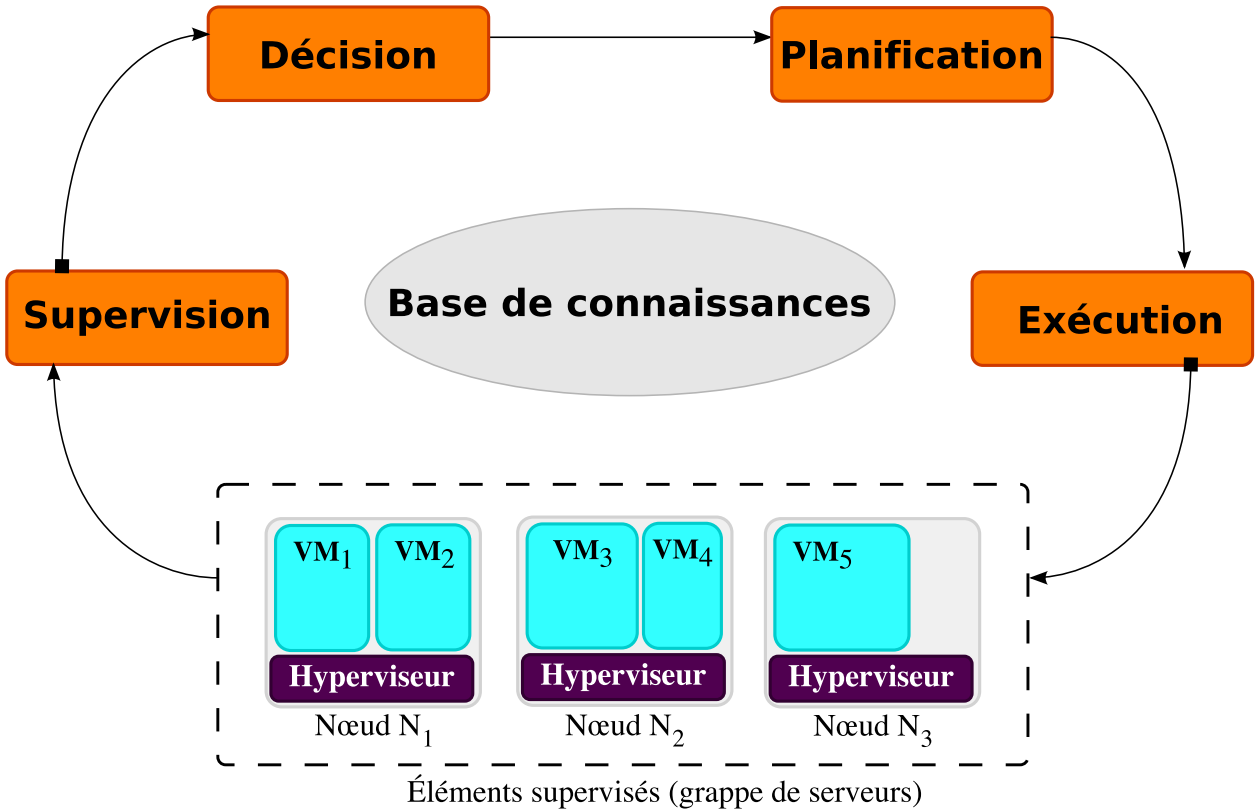
Boucle d'auto-adaptation



Planification

- reconfiguration dynamique : calcule le plan le moins coûteux pour réaliser la transition entre l'état courant et le nouvel état

Boucle d'auto-adaptation



Exécution

- adapte les actions aux spécificités de la grappe (actuellement des commandes SSH ou pour l'hyperviseur Xen)
- exécute le plan de reconfiguration

Consolidation dynamique

Disponibilité et occupation des ressources

- taux d'occupation de 53% en moyenne
  [Parallel Workload Archive]
- pour des grappes disponibles à 100%

La consolidation

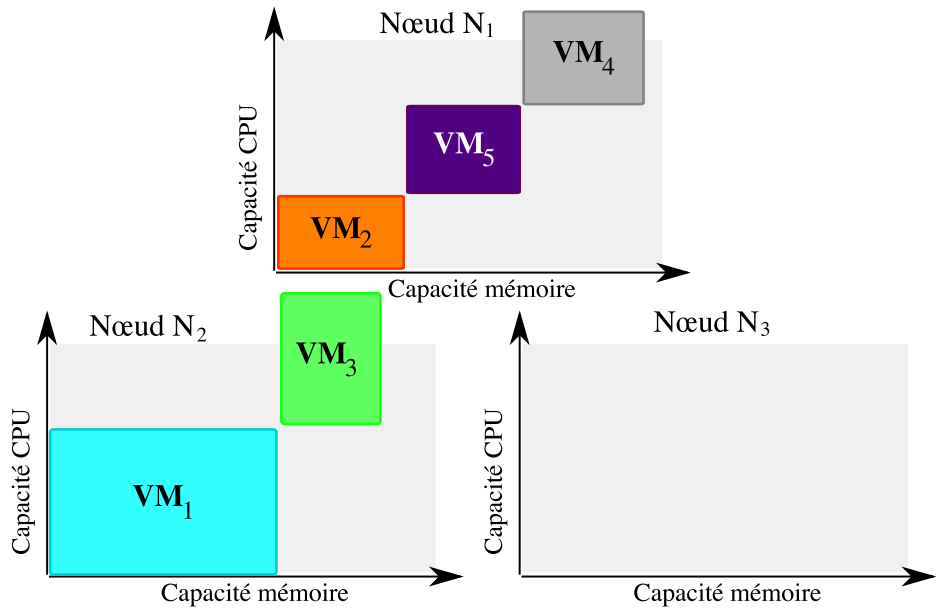
héberger plusieurs machines virtuelles sur un même nœud

- réduit la consommation énergétique en cas de concentration
- augmente la capacité d'accueil de la grappe

Consolidation dynamique

La consolidation dynamique

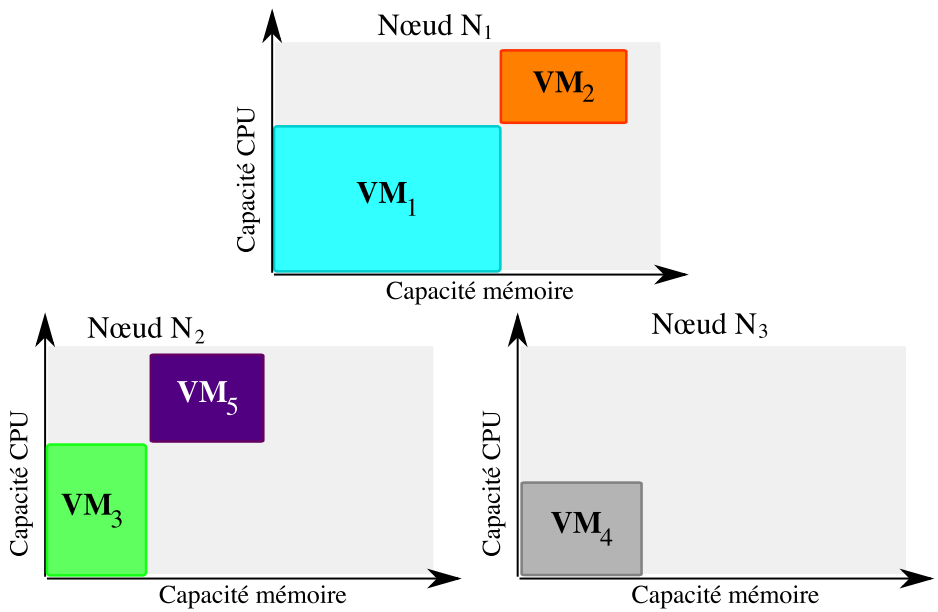
- les besoins en ressources des VMs varient et créent des agencements non-viables



Consolidation dynamique

La consolidation dynamique

- les besoins en ressources des VMs varient et créent des agencements non-viables
- des migrations à chaud ré-agencent des VMs si besoin



Consolidation dynamique

Implémentation dans Entropy

- observation des besoins courants en ressources CPU
- calcul du nombre de nœuds minimum pour héberger toutes les VMs
- la reconfiguration dynamique
  - calcule une nouvelle configuration
  - assure que les VMs seront hébergées sur le nombre minimum de nœuds

Une approche avec des contraintes

comparée à une approche heuristique standard

- ✓ plus flexible
- ✓ un résultat théoriquement meilleur
- × un temps de calcul plus long

Ordonnancement flexible de tâches

Simplifier le développement d'ordonnanceurs

- développer des ordonnanceurs pour grappe est complexe
- le développeur doit à la fois considérer
  - le problème lié à la sélection des tâches
  - les problèmes liés à l'application de l'ordonnancement choisi

Ordonnancement flexible de tâches

Implémentation dans Entropy : le changement de contexte dans les grappes

- le développeur de l'algorithme
  - se focalise sur la sélection des tâches à exécuter
  - prouve par un exemple basique qu'un agencement viable existe
- le changement de contexte se charge de la mise en place de la solution en un minimum de temps
  - suspend, reprend, lance et arrête des tâches
  - migre si nécessaire des composants

Ordonnancement flexible de tâches

Exemple d'algorithme

- sélectionne les tâches à exécuter dès qu'il existe suffisamment de ressources libres sur la grappe
- le changement de contexte assure
  - la préemption des tâches en cas de surcharge
  - une exécution au plus tôt
  - une adaptation à des besoins en ressources variables

Plan

Évaluation

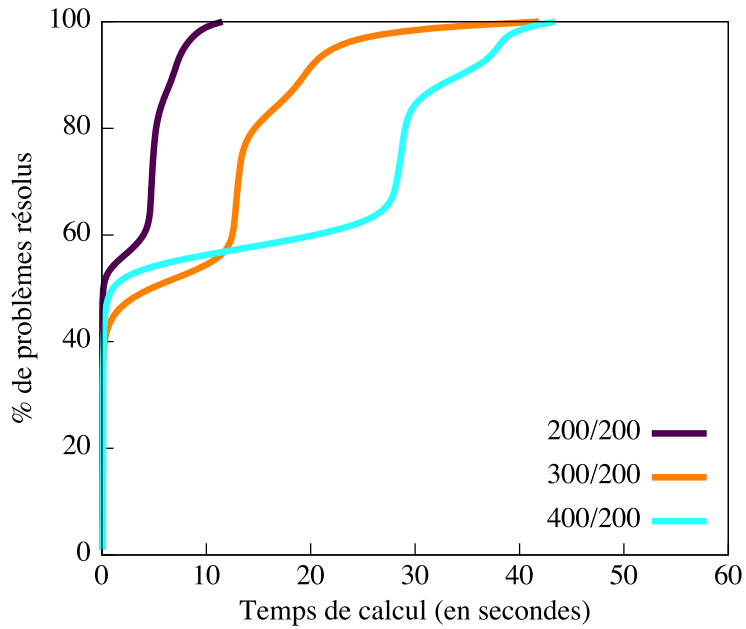
Objectifs

- valider notre approche pour la réduction du temps de reconfiguration
- estimer les temps de résolution de nos différents modules
- comparer la qualité des solutions rapportée à des approches heuristiques
- estimer l'impact sur les performances

Micro-évaluations

Comparé à l'heuristique commune *First Fit Decrease*

- un temps de résolution qui peut être important

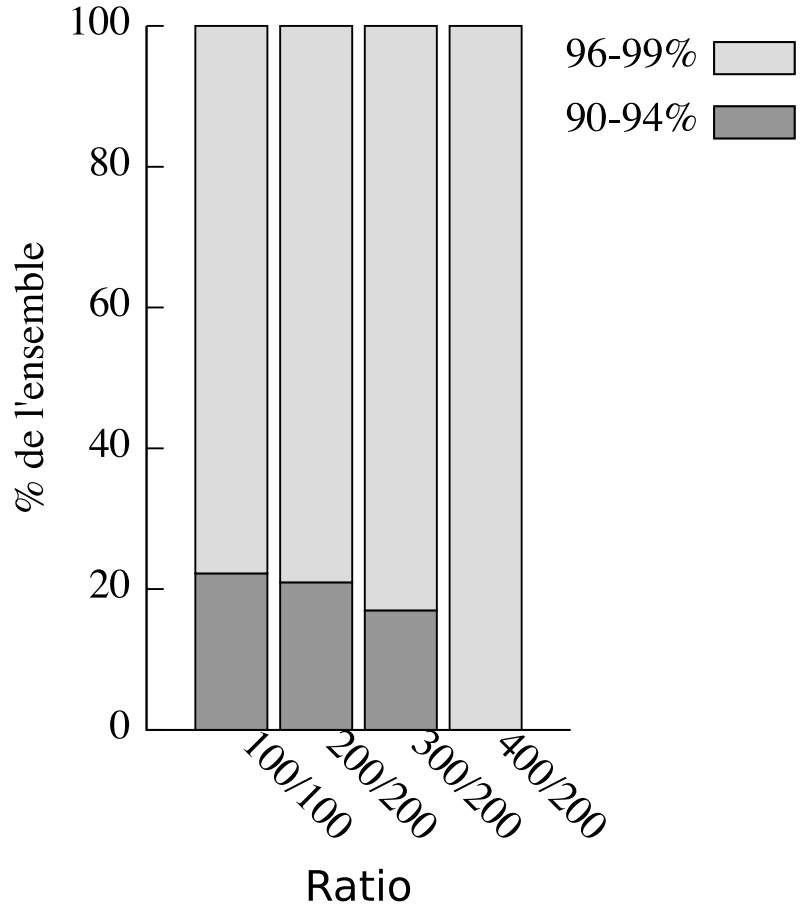


(c) Évolution de la résolution du problème de consolidation dynamique

Micro-évaluations

Comparé à l'heuristique commune *First Fit Decrease*

- un temps de résolution qui peut être important
- contrebalancé par la qualité des résultats

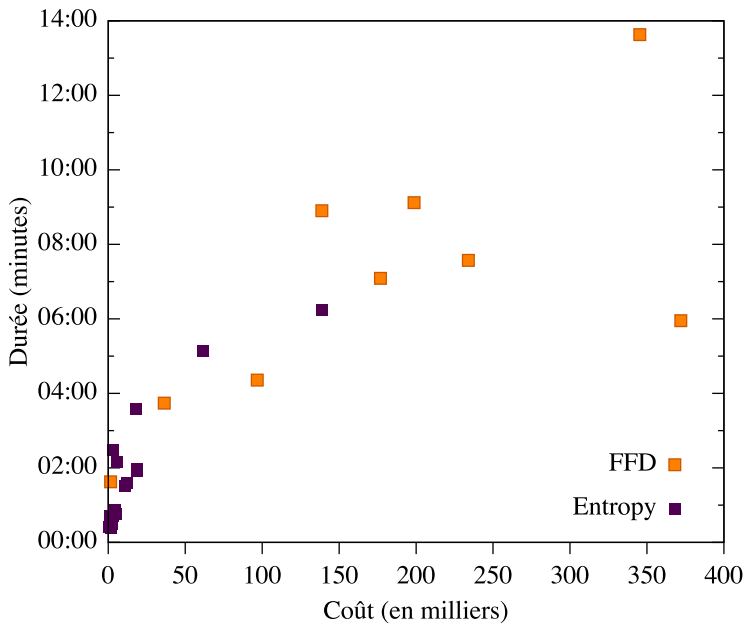


(d) Réduction du coût de la reconfiguration comparée à FFD pour différentes classes de configuration

Évaluations sur grappes

Comparé à FFD

- réduire le coût d'une reconfiguration réduit son temps d'exécution
- une plus grande réactivité
  - des reconfigurations plus fréquentes

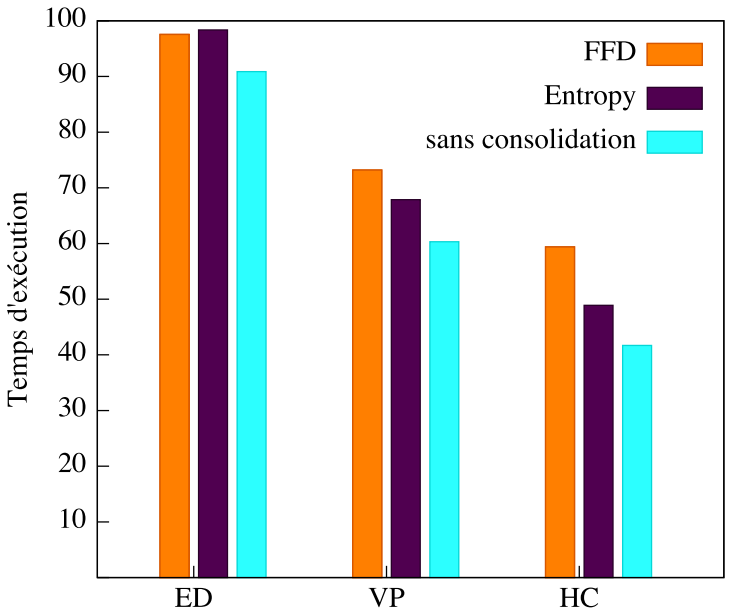


(e) Coût et durée des reconfigurations réalisées lors de l'exécution d'applications de la suite NASGrid

Évaluations sur grappes

Comparé à FFD

- réduire le coût d'une reconfiguration réduit son temps d'exécution
- une plus grande réactivité
  - des reconfigurations plus fréquentes
  - un impact réduit sur les performances (+11%)

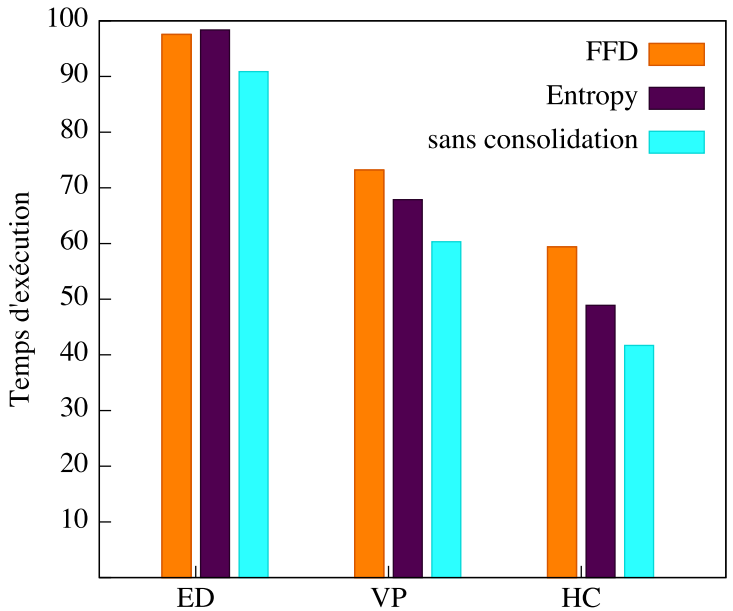


(f) Temps d'exécution d'applications de la suite NASGrid selon l'approche pour la reconfiguration

Évaluations sur grappes

Comparé à FFD

- réduire le coût d'une reconfiguration réduit son temps d'exécution
- une plus grande réactivité
  - des reconfigurations plus fréquentes
  - un impact réduit sur les performances (+11%)
- une meilleure utilisation des ressources (-50%)



(g) Temps d'exécution d'applications de la suite NASGrid selon l'approche pour la reconfiguration

Plan

Conclusion

- la virtualisation facilite l'exécution de stratégies de gestion dynamique des tâches
- la programmation par contraintes
  - facilite le développement de stratégies spécifiques
  - propose des solutions de qualité qui justifient un temps de résolution long
- le prototype Entropy valide empiriquement l'approche

Perspectives de cette thèse

Ordonnancement

- améliorer l'utilisabilité
  - une approche déclarative par un langage dédié
- augmenter la flexibilité du processus de reconfiguration
  - une approche contraintes pure

Entropy

- considérer d'autres types de ressources
  - bande passante et latence réseau
- intégrer Entropy dans des gestionnaires de grappes à base de VMs existants
  - une meilleure visibilité
  - identifier à de nouveaux besoins, cas d'utilisation, et valider la granularité de notre modèle

Bilan

- la virtualisation
  - une solution adoptée dans les centres d'hébergement
  - un candidat sérieux pour faciliter l'ordonnancement
  - des actions manipulant les machines virtuelles encore trop basiques
- l'informatique en nuage, un domaine émergent
  - l'hébergement n'est plus à la charge de l'utilisateur
  - les systèmes devront assurer un placement satisfaisant des besoins concrets
- ordonnancement reposant sur des contraintes
  - une approche adaptée à la taille de ces architectures immenses ?

Diffusion

- publications
  - conférence internationale : VEE'09
  - conférences nationale : CFSE'06, CFSE'07
  - atelier international : XHPC'06
  - poster : OSDI'08
- valorisation
  - Entropy est disponible sous licence LGPL
  - tests en cours à la Direction Générale des FInances Publiques
- un projet ANR reprenant nos travaux : Self-XL