Table des matières

Liste des figures

Liste des tableaux

Introduction générale

Contexte

Le Cloud Computing est un paradigme qui a attiré beaucoup d'attention ces deux dernières décennies grâce à son élasticité, son niveau élevé d'automatisation, son provisionnement en libre-service à la demande, et à la facturation à l'usage. Il s'appuie sur les dernières découvertes et réalisations de divers domaines de recherches, tels que le Grid Computing, l'informatique orientée service, la gestion des processus métiers (BPM : "Business Process Management"), et la virtualisation, pour fournir des services informatiques à la demande à ses utilisateurs et répondre à leurs besoins Beloglazov et al. (2012) Barham et al. (2003). Le Cloud Computing offre des infrastructures, des plateformes, et des logiciels (applications) comme des services qui sont mis à la disposition des utilisateurs sous un modèle "pay-as-you-go" ou "facturation à l'usage". Ces services sont désignés, respectivement, comme Infrastructure as a Service (IaaS), Platform as a Service (PaaS), et Software as a Service (SaaS). Ces services Buyya et al. (2009) ne sont pas seulement utilisés mais aussi installés, déployés ou répliqués à l'aide de la virtualisation qui permet de faire fonctionner sur une seule machine physique (PM) plusieurs systèmes d'exploitation (ou machines virtuelles VM), isolés les uns des autres Barham et al. (2003). Afin de fournir tous ces services et satisfaire la clientèle d'un tel système, des datacenters virtualisées sont déployés à large échelle. Ces datacenters utilisent, habituellement, la technologie de virtualisation à des fins d'isolation et de consolidation. Ces datacenters, fonctionnant sous le modèle de Cloud Computing, Hébergent une variété d'applications allant de celles qui fonctionnent pendant quelques secondes (par exemple le traitement des requêtes d'applications web telles que l'e-commerce et les réseaux sociaux) à celles qui fonctionnent pour des périodes plus longues de temps (par exemple, des simulations ou le traitement de gros volumes de données).

Avec la généralisation des services Cloud et la rapidité de son adoption par les plus grands fournisseurs des services informatiques dans le mondes, la croissance rapide du nombre des utilisateurs des services Cloud et leurs exigences vis-à-vis la qualité de ces service. Les fournisseurs ont été amenés à crée des nouveaux datacenters et/ou à densifier la puissance de ceux qui sont déjà opérationnels pour satisfaire leurs utilisateurs. L'accroissement du nombre des datacenters et la densification des ressources s'est traduite par une augmentation très importante de la consommation électrique. Selon un rapport du NRDC (*Natural Ressource Defense Council*) publié en Août 2014 DELFORGE (2014), les datacenters américains ont consommé, approximativement, 91 milliards kilowatts-heure, en 2013, l'équivalent d'une production annuelle de 34 grandes centrales électriques alimentées au charbon. Ce rapport prévoit une augmentation de la consommation annuelle d'énergie à environ 140 milliards kilowatts-heure en 2020, l'équivalent de la production annuelle de 50 centrales électriques, ce qui coûte aux entreprises américaines 13 milliards de dollars par an en factures d'électricité et émettent près de 100 millions tonnes métrique de CO_2 par an. Dans une autre étude, Gary Cook, un analyste informatique principale chez Greenpeace International, a publié dans un rapport intitulé "How Clean is Your

Cloud ?" COOK (2012), que certains datacenters consomment autant d'énergie que 180 000 foyers européens réunis. Il dit que si le Cloud était un pays, il serait classé (en 2012) 5^{eme} au monde en termes de demandes d'électricité, et ses besoins devraient être triplé d'ici 2020.Cette augmentation de l'électricité consommée par les datacenters et le sous-dimensionnement des systèmes électriques et de refroidissement se traduit par des problèmes techniques en tout genre : surchauffe et problèmes de dissipation de la chaleur, tension sur l'approvisionnement en électricité, etc.

Problématique

Jusqu'à récemment, les performances des systèmes et la qualité de services (QoS) fournis aux utilisateurs ont été les seules préoccupations lors des déploiements des datacenters sans payer suffisamment d'attention à la consommation d'énergie. Cependant, L'émergence des contraintes techniques, économiques et environnementales liées à la consommation d'énergie des datacenters pousse aujourd'hui les entreprises à améliorer l'efficacité énergétique de leurs installations existantes et à concevoir de nouveaux datacenters plus "*green*". Donc, il est nécessaire de réorienter l'attention de l'optimisation de la gestion des ressources physiques des datacenters pour l'amélioration des performances et du qualité de service vers l'efficacité énergétique, tout en maintenant une bonne qualité de service.

Les problèmes énergétiques ont été traité, au début, en utilisant : *i)* Les nouveautés des technologies matérielles tels que les processeurs multi-core, les processeurs à faible puissance, les disques SSD ...etc. *ii)* La consolidation agressive des ressources, grâce à la virtualisation, pour assurer une pleine utilisation des serveurs et réduire le nombre de serveurs actifs. Effectivement, ces solutions ont diminué la consommation d'énergie, mais en contre partie, elles ont provoqué des dégradations de la qualité de service à cause de leurs comportements (réduire les serveurs actifs et/ou la puissance de calcul). Comme il est convenu, la qualité de service et la caractéristique clé du Cloud Computing, et tous les fournisseurs des services Cloud visent à fournir une QoS fiable à leurs utilisateurs. Donc comment réduire la consommation d'énergie sans affecter la qualité de service ?, est la question qui sera traîtée dans la présente thèse.

Objectif

Cette thèse présente une stratégie de migration de machines virtuelles basée une superposition de deux théorèmes économique. La stratégie proposée vise à réduire la consommation d'énergie sans affecter la qualité de services : *1)* une instance de l'un des types du problème de correspondance stable qu'est le problème des admissions au collège Roth (1984) Gale et Shapley (1962) et qui permet de trouver un compromis entre les contraintes des prestataires de services (réduction de consommation d'énergie des PMs) et les exigences des utilisateurs (amélioration de la qualité de services fournie par les VMs) sans favoriser un côté sur l'autre à l'aide d'une procédure d'acceptation différée. Cependant, les choix affectés, par cette procédure, aux deux côtés (PMs et VMs) peuvent, parfois, être différents et même contradictoires. Ce problème est connu comme le problème de polarisation des correspondances stables Roth (1984) Martínez et al. (2004). Plusieurs travaux ont été proposé Gusfield et Irving (1989) Halldòrsson et al. (2003) Iwama et al. (2007) dans ce contexte mais ils favorisent, généralement, un côté au

détriment de l'autre. Dans ce travail, nous utilisons un algorithme proposé dans Martínez et al. (2004) pour traiter le problème de polarisation des correspondances stables dans l'instance du problème de correspondance stable connue sous le nom du "*problème des firmes-consultants*" qui permet, en cas de polarisation, de trouver un autre compromis entre les PMs et VMs. *2)* Le théorème de Coase Coase (1960) nous permet de trouver le nombre optimal de machines virtuelles à faire migrer pour sélectionner la solution entre autres possible (en fonction du nombre de machines virtuelles migrées) qui améliore le mieux la consommation d'énergie et la qualité de service.

Organisation de la thèse

A fin d'aborder tous ces aspects, nous organisons la thèse, en plus d'une introduction et une conclusion, en cinq chapitres :

Dans le premier chapitre, nous présenterons en détail les aspects énergétiques dans le cloud computing en montrant la différence entre la consommation de puissance électrique et la consommation de l'énergie ainsi que les modèles utilisés pour les quantifier. Nous montrons dans ce chapitre, aussi, que le CPU est l'élément le plus énergivore parmi les composants d'un serveur. Les contrat SLA seront décrites et leurs composants seront détaillés. Ce chapitre donne un aperçu sur la qualité des services dans le Cloud computing, et sur la difficulté de trouver un compromis entre la réduction de consommation d'energie et l'amélioration de la qualité de services.

Dans le deuxième chapitre, des généralités sur la technologie de virtualisation, les hyperviseurs et les machines virtuelles seront présentées. Une bonne partie de ce chapitre sera consacré aux techniques migrations dans le cloud computing.

Un état de l'art sur les techniques de migration des machines virtuelles dans les computing fera l'objet du troisième chapitre.

Le quatrième chapitre sera réservé à l'introduction des théorèmes économiques utilisés, à l'explication de l'utilisation de ces théorèmes dans la migration des données dans le Cloud computing, et à la description détaillée de la conception de la stratégie proposée ainsi que les services que nous avons conçus. Cette conception se fera à l'aide de formules, d'algorithmes et de diagrammes UML.

Le dernier chapitre présentera les étapes de l'implémentation de l'approche proposée. Nous y détaillerons la réalisation de certaines fonctionnalités ainsi que l'étude d'évaluation de cette stratégie. Les résultats d'expérimentation seront interprétés.

1 Consommation d'énergie et qualité de service dans le Cloud Computing

Sommaire

1.1 Introduction

Le développement des systèmes informatiques a été axé, durant les dernières décennies, sur l'amélioration des performances due à la demande des applications gourmandes de calcul dans les domaines publique, scientifique, et commerciale. Bien que le rapport performance/coût de matériels a augmenté de façon spectaculaire, la consommation d'énergie dans de tels systèmes a également connu une considérable augmentation selon la loi de Moore Li (2012). Pour obtenir des performances plus élevé par processeur afin de répondre aux besoins de leurs clients, les fabricants de microprocesseur ont poussé la densité de puissance à des vitesses exponentielles, qui va bientôt atteindre celle d'un réacteur nucléaire Venkatachalam et Franz (2005) Li (2012).

De nos jours, le Cloud Computing est le système informatique le plus adopté par les plus grands fournisseurs de services informatiques dans le monde tel que Amazon, Google, Salesforce, Microsoft, etc. Le Cloud Computing utilise la virtualisation (*voir chapitre 2*) qu'est une technologie permettant de consolider les charges sur un matériel physiquement réduit et donc de réduire fortement la consommation d'énergie. Cependant, La montée en puissance des matériels et la complexité accrue des services informatiques ont entraîné une hausse de la demande en énergie. Les coûts liés aux infrastructures et à l'énergie dans les datacenters sont ainsi devenus un facteur essentiel de la gestion de ce type de système. Une étude Koomey (2007) Kessaci et al. (2011) montre qu'en 2005, 0.6% de la consommation totale d'électricité des Etats Unis revient à la puissance utilisée par les serveurs. Cette proportion passe à 1.2% lorsque les infrastructures de refroidissement et les appareils auxiliaires sont inclus. Selon les statistiques de cette étude, la consommation totale d'électricité a doublé au cours de la période 2000 à 2005 dans le monde entier.
D'autre part, les émissions de gaz à effet de serre sont en train d'atteindre un seuil critique

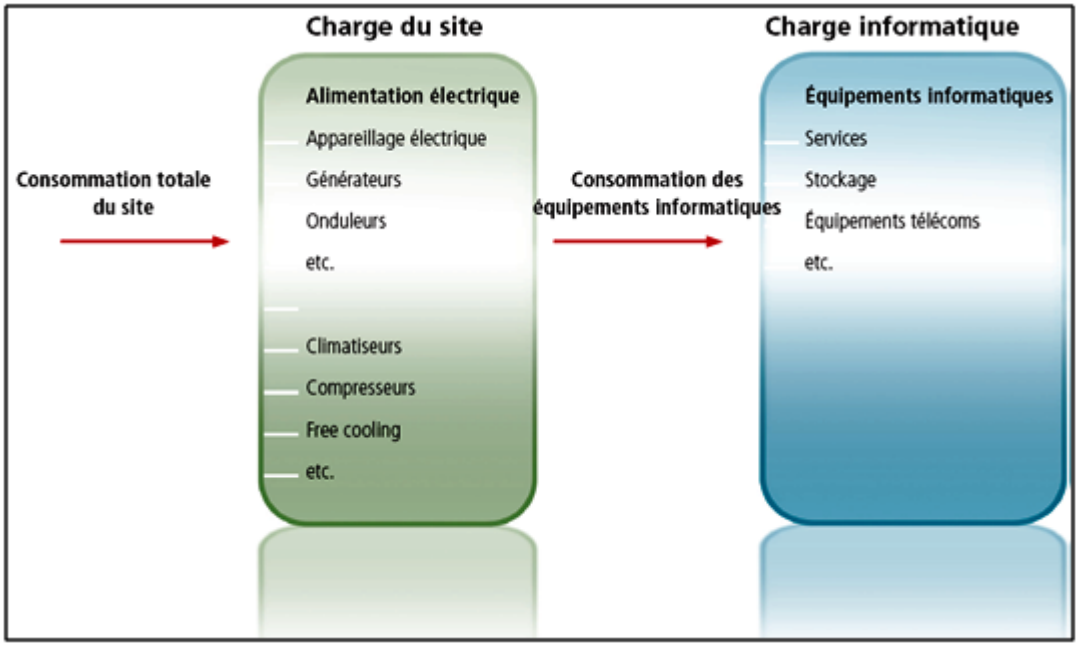


Figure 1.1 – *Schéma simplifié des principaux soussystèmes d'un datacenter ASHRAE (2010)*

Gartner (2007) Kessaci et al. (2011) et représente approximativement 2% de la quantité totale des émissions de CO_2 à travers le monde. L'accroissement de la consommation

d'énergie a un effet néfaste sur les bénéfices des fournisseurs du Cloud. Le groupe Amazon Hamiltoni (2009) Kessaci et al. (2011) a estimé le montant des coûts liés à l'énergie à 42% du budget total d'un datacenter. Ainsi, un ensemble de technologies ont donc été développées pour améliorer l'efficacité énergétique et de nouvelles options de matériels et de gestion des systèmes d'information soutiennent les stratégies d'économies d'énergie. Les mesures matériels ont surtout porté sur les solutions d'alimentation et de refroidissement.

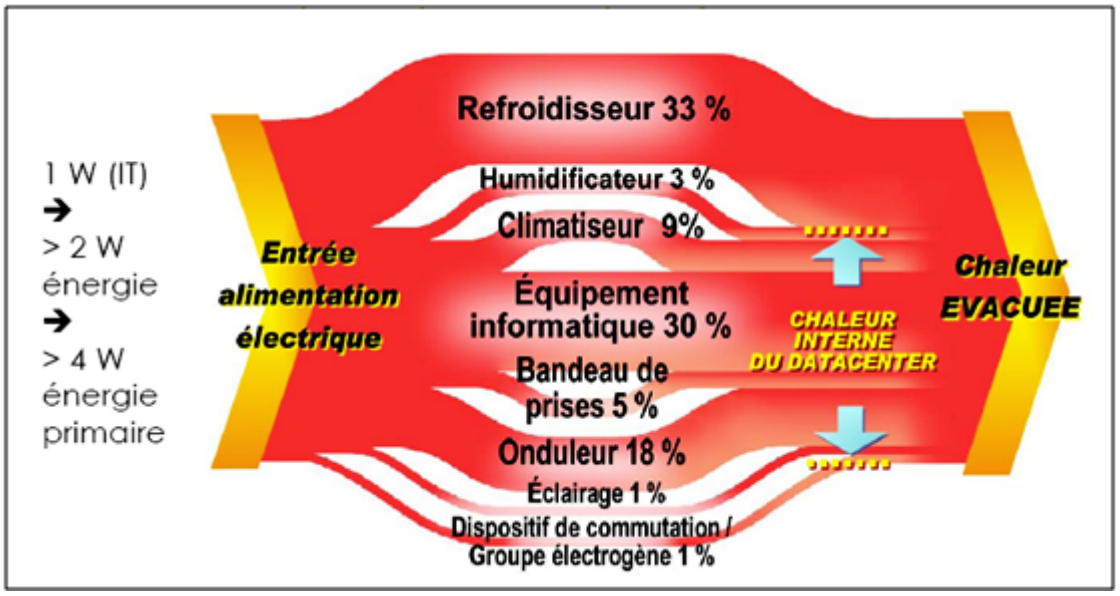


FIGURE 1.2 – *Aspects de la consommation électrique Berthoud (2011)*

Certains travaux montrent que ces mesures d'économie d'énergie entraînent d'ores et déjà une réduction significative de la demande en énergie par rapport à un scénario de statu quo Koomey (2011). D'un autre côté, ces solutions matérielles ont des répercussions sur les performances des datacenters et par conséquent sur la qualité des services (QoS) fournis aux utilisateurs du Cloud qui est gérées par un contrat SLA entre l'utilisateur final et le prestataire de services. En cas de violation d'une clause d'un contrat SLA, des pénalités sont appliquées. La pénalité non seulement pénalise le prestataire de services en réduisant leur profit, mais affecte aussi le rendement des tâches exécutées par l'utilisateur. De ce fait, trouver des solutions pour réduire la consommation d'énergie sans affecter la qualité de services est un des problèmes les plus étudiés ces dernières années Koomey (2011).
Pour identifier les défis de ce domaine et faciliter la poursuite des progrès pour satisfaire le prestataire de services ainsi que ces client, il est essentiel de synthétiser et de classer les différents aspects de la consommation d'énergie et des contrats SLA. Ce chapitre aborde les causes et les problèmes de forte consommation de puissance/énergie, et les principaux termes et qualité de services des contrats SLA. Ce chapitre se termine par une discussion du paradoxe "réduction de la consommation d'énergie/amélioration de la qualité de service" en montrant la divergence entre les deux objectifs et la contradiction des solutions proposées pour les traiter séparément.

1.2 Consommation d'énergie dans le Cloud Computing

1.2.1 Énergie et puissance électrique

Pour comprendre les mécanismes d'alimentation et de gestion de l'énergie, il est essentiel de clarifier la terminologie. Le courant électrique est le flux de charge électrique mesuré en ampères (*Ampère A* définie la quantité de charge électrique transférée par un circuit par seconde) bien que la puissance et l'énergie peuvent être définies en termes de travail qu'un système effectue. La puissance électrique est le débit auquel le système effectue une tâche, tandis que l'énergie représente la quantité totale des travaux effectués pendant une période de temps. Puissance et énergie sont mesurées, respectivement, en watts (W) et watt-heure (Wh). Une tâche est effectuée à un débit de $1W$ lorsque $1A$ est transféré à travers une différence de potentiel électrique de $1V$. Un kilowatt-heure (kWh) est la quantité d'énergie équivalente à une puissance de 1 kW (1 000 W) appliquée pendant une heure. Formellement, la puissance et l'énergie peuvent être définis comme indiqué dans ci-dessous :

$$P = \frac{W}{T} \tag{1.1}$$

$$E = PT \tag{1.2}$$

Où P est la puissance électrique, T est une période temps, E est la quantité totale de travaux effectués durant cette période de temps. La différence entre la puissance et l'énergie électrique est très importante car une réduction de la consommation de la puissance électrique ne réduit toujours pas l'énergie consommée. Par exemple, la consommation de puissance peut être réduite en abaissant les performances du CPU, et dans ce cas, un programme ou une tâche peut prendre plus de temps pour terminer son exécution en consommant la même quantité d'énergie. D'un côté, une réduction d'un pic de consommation de puissance entraîne une diminution des coûts d'approvisionnement de l'infrastructure, comme les coûts liés aux capacités des onduleurs (UPS), les unités de distribution d'alimentation (PDU), des générateurs électriques, les systèmes de refroidissement, et des équipements de distribution d'énergie. D'un autre côté, une baisse de la consommation d'énergie réduit les factures d'électricité.

La consommation d'énergie peut être réduite temporairement via les techniques de gestion dynamique de puissance électrique (Dynamic Power Management (*DPM*)) ou de façon permanente via une gestion statique de puissance électrique (*SPM*). *DPM* utilise la connaissance, en temps réel, de l'utilisation des ressources et des charges de travail pour optimiser la consommation d'énergie. Toutefois, il ne diminue pas nécessairement les pics de la consommation de puissance électrique. En revanche, *SPM* impose l'utilisation des composants matériels les plus efficaces, tels que les processeurs, les disques de stockage, les périphériques réseau, UPS, et les alimentations électriques. Ces changements structurels réduisent généralement l'énergie consommée et les pics de la consommation de puissance Beloglazov (2013).

1.2.2 Consommation statique et dynamique de la puissance électrique

La majeure partie de la consommation de puissance électrique dans les circuits CMOS (*Complementary Metal-Oxyde-Semiconductor*) comprend une puissance électrique statique et dynamique. La consommation de puissance statique est causée par des courants de fuite qui sont présents dans n'importe quel circuit actif indépendamment des scénarios

d'usage et de la fréquence de l'horloge. Cette consommation de puissance statique est, principalement, déterminée par le type de transistors et de la technologie des procédés. La réduction de la consommation de puissance statique nécessite des améliorations dans la conception de bas niveau du système.
La consommation de puissance électrique dynamique est créée par l'activité des circuits (les commutateurs des transistors, les changements des valeurs dans les registres, etc.) et dépend, principalement, d'un scénario d'usage, de la fréquence de l'horloge et des opérations d'E/S. La source de la consommation de puissance électrique dynamique est le courant des court-circuit et les capacitances commutées Beloglazov (2013). Le Courant de court-circuit génère seulement 10 à 15% de la consommation électrique totale et jusqu'à présent, aucun moyen n'a été trouvé pour réduire cette valeur sans compromettre les performances. La capacitance commutée est la principale source de consommation dynamique de puissance électrique ; par conséquent, la consommation dynamique de puissance peut être définie comme suit :

$$P_d = aCV^2f \tag{1.3}$$

Où a est l'activité de commutation, C est la capacitance physique, V est la tension d'alimentation, et f est la fréquence d'horloge. Les valeurs d'activité de commutation et la capacitance sont déterminées par la conception de bas niveau du système. La réduction combinée de la tension d'alimentation et la fréquence d'horloge réside dans les racines d'une technique DPM largement utilisée appelée l'adaptation dynamique de la tension et de la fréquence "*Dynamic Voltage and Frequency Scaling (DVFS)*". L'idée principale de cette technique est de réduire, intentionnellement, les performances du processeur, lorsqu'il n'est pas totalement utilisé, en diminuant la tension et la fréquence du CPU. Dans le cas idéal, il devrait en résulter une réduction cubique de la consommation électrique dynamique. *DVFS* est soutenu par la plupart des processeurs modernes, y compris mobile, ordinateurs de bureau, et les serveurs Pouwelse et al. (2001).

1.2.3 Dépendance "utilisation CPU/consommation de puissance"

Le processeur (CPU) est l'élément le plus énergivore du serveur. Un modèle de CPU efficace avec une gestion d'énergie efficace peut donc jouer un rôle important dans l'efficacité globale. La consommation du processeur dépend de la tension et de la fréquence de l'horloge. La gestion d'énergie au niveau du CPU ou du cœur repose donc sur l'adaptation dynamique de la tension et de la fréquence (DVFS) ou le fait d'éteindre des cœurs Beloglazov (2013).
Les données fournies par Intel Labs Minas et Ellison (2009) Beloglazov (2013) confirment que le CPU est l'élément le plus gourmand en puissance électrique suivi par la mémoire et les pertes dues à l'inefficacité d'alimentation (voir Figure1.3). Les données montrent, aussi, que le CPU ne domine plus la consommation électrique dans un serveur. Ceci résulte de l'amélioration continue de l'efficacité de puissance électrique du CPU associée à des techniques d'économie de puissance électrique (comme le *DVFS*) qui permettent au CPU de fonctionner dans des modes à faible consommation. Dans ces modes, un CPU consomme une fraction de la puissance totale tout en préservant la possibilité d'exécuter des programmes. Par conséquent, les CPUs des ordinateurs de bureau et des serveurs actuels peuvent consommer moins de 30% de leurs puissances totales dans des états de faible activité, ce qui conduit à des intervalles dynamiques de plus de 70% de leurs puissances électriques totales (forte activité) Venkatachalam et Franz (2005) Beloglazov (2013). En revanche, les intervalles de puissances électrique des autres composants sont beaucoup

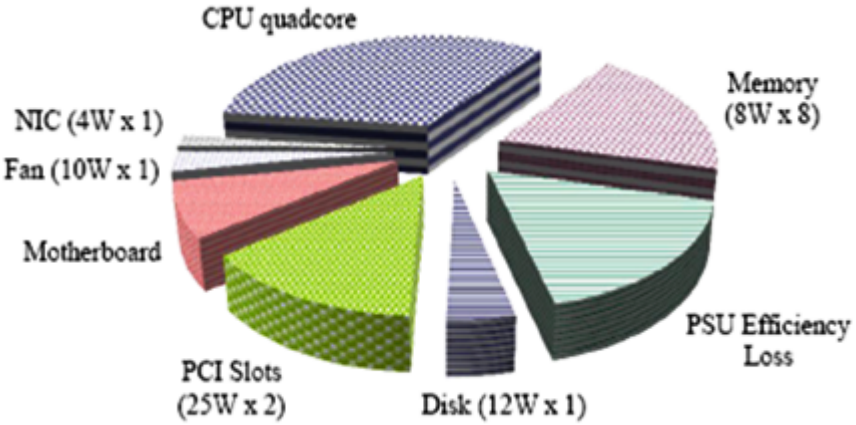


FIGURE 1.3 – *La consommation de puissance électrique par les composants d'un serveur Beloglazov (2013)*

plus étroits : moins de 50% pour les mémoires vice dynamiques (*DRAM*), 25% pour les lecteurs de disques, 15% pour les commutateurs de réseau, et négligeable pour le autres composants Fan et al. (2007a) Beloglazov (2013). La raison en est que seule le CPU prend en charge des modes actifs de faible consommation, alors que les autres composants ne peuvent être que complètement ou partiellement mis hors tension. Toutefois, la dégradation des performances d'une transition entre les modes actifs et inactifs est importante. Par exemple, un lecteur de disque en mode sommeil profond consomme presque pas de puissance électrique, mais une transition vers le mode actif encourt une latence mille fois plus élevée que la latence d'accès régulier. En résumé, le CPU se distingue comme étant le composant le plus influent sur la consommation de puissance électrique. Par conséquent, tous les techniques qui cherchent à améliorer la consommation électrique, se basent principalement sur la réduction du taux d'utilisation des CPUs. Pour élaborer de nouvelles

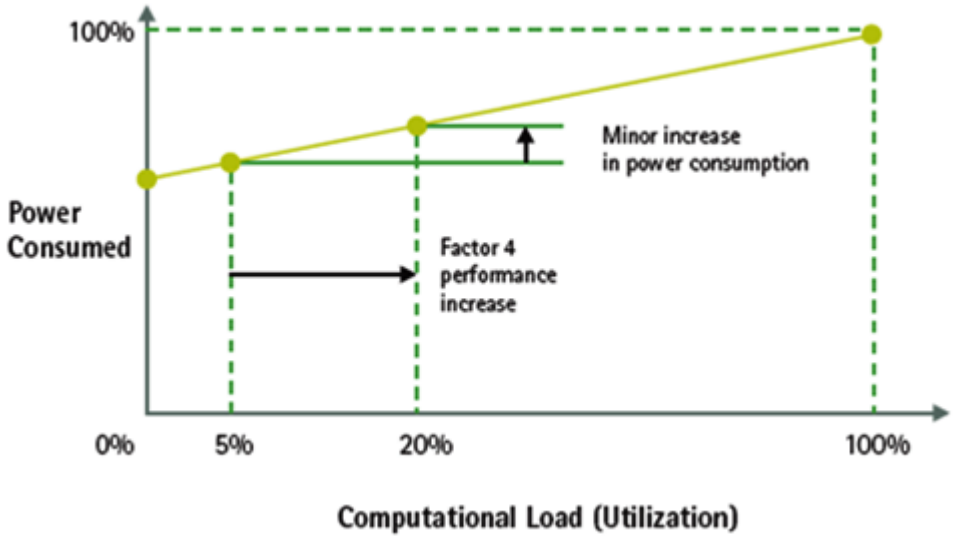


FIGURE 1.4 – *La relation entre la taux d'utilisation du CPU et la consommation de puissance électrique Accenture et WSP (2010)*

politiques pour DPM et comprendre leurs impacts, il est nécessaire de créer un modèle de consommation dynamique du puissance électrique. Un tel modèle devrait être en mesure de prédire la valeur effective de la consommation électrique par un système basé sur

quelques caractéristiques d'un système en cours d'exécution. Une des façons d'y parvenir est d'utiliser les fonctions de surveillance de puissance électrique qui sont intégrées dans les serveurs informatiques modernes. Ces fonctionnalités permettent le suivi de la consommation de puissance par un serveur en temps réel et la collecte de statistiques précises de la consommation électrique. D'après les données, il est possible d'obtenir un modèle de consommation d'énergie pour un système particulier. Cependant, cette approche nécessite la collecte de données pour chaque système ciblé. Fan et al. (2007a) ont trouvé une forte corrélation entre l'utilisation du processeur et de la consommation totale de puissance électrique par un serveur. L'idée derrière le modèle proposé est que la consommation électrique par un serveur croît linéairement avec la croissance du taux d'utilisation du CPU de la valeur de la consommation d'énergie en mode veille jusqu'à la puissance consommée lorsque le serveur est pleinement utilisée. Cette relation peut être exprimée comme suit :

$$P(u) = P_{idle} + (P_{busy} - P_{idle})u \tag{1.4}$$

Tel que P est la consommation de puissance électrique estimée, P_{idle} est la consommation électrique d'un serveur inactif et P_{busy} d'un serveur pleinement utilisé, et u est le taux d'utilisation actuel du CPU. Les auteurs ont également proposé un modèle non linéaire empirique :

$$P(u) = P_{idle} + (P_{busy} - P_{idle})(2u - u^r) \tag{1.5}$$

Où r est paramètre de calibrage qui minimise l'erreur quadratique et doit être obtenue expérimentalement : pour chaque catégorie de machines, un ensemble d'expériences de calibrages doit être effectué pour affiner ce modèle. Des expériences approfondies sur plusieurs milliers de nœuds avec différents types de charges de travail (voir Figure 1.5) ont montré que les modèles dérivés peuvent prédire avec exactitude la consommation électrique par des systèmes de serveurs avec un taux d'erreur inférieure à 5% pour le modèle linéaire et 1% pour le modèle empirique.

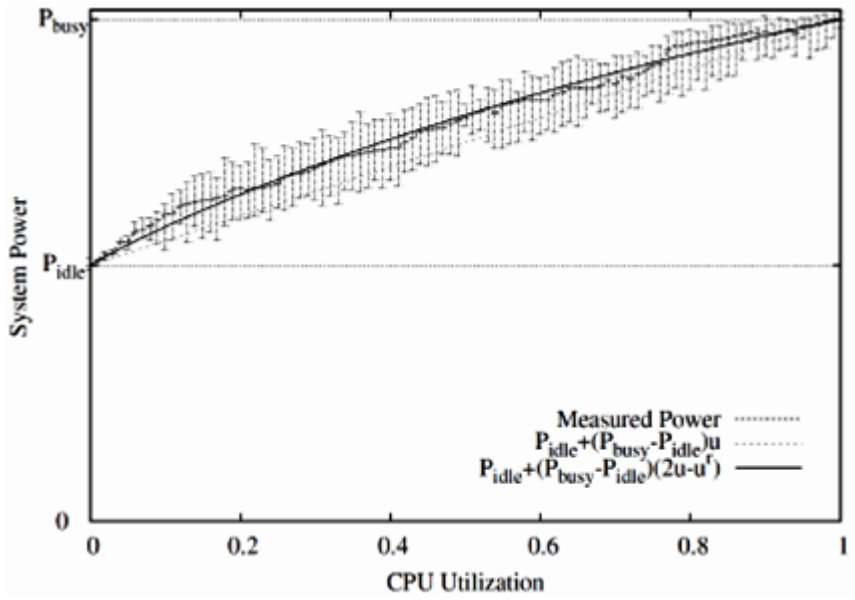


FIGURE 1.5 – *La relation entre la taux d'utilisation du CPU et la consommation de puissance électrique Fan et al. (2007a)*

Ces résultats précis, qui ont été obtenue en fixant le paramètre de calibrage r à 1.4, peuvent être expliqués par le fait que le processeur est le consommateur principal de la

puissance électrique dans les serveurs.

1.2.4 Dépendance "consommation de puissance/consommation d'énergie"

La relation entre l'énergie et la puissance est un peu comme la relation entre la distance et la vitesse :

- L'énergie est comme la distance. La quantité d'énergie utilisée sur une période de temps spécifique est comme la distance parcourue sur une période de temps spécifique.
- La puissance est comme la vitesse. Une puissance instantanée est comme une vitesse à un instant précis dans le temps (par exemple en ce moment). Une puissance moyenne sur une période de temps spécifique est comme une vitesse moyenne sur une période de temps spécifique.

La distance et la vitesse sont des mesures utiles. Et les deux sont étroitement liés (comme indiquée dans la section 1.2.1). Parfois, il est logique de parler en termes de distance, et parfois il est logique de parler en termes de vitesse. C'est la même chose pour l'énergie et la puissance, nous avons besoin des deux mais parfois il est plus logique d'utiliser l'un au lieu de l'autre.
Traditionnellement, les techniques de gestion efficace des ressources pour la réduction de la consommation d'énergie et de puissance électrique ont été appliquées aux appareils mobiles. Ça a été dicté par le fait que de tels dispositifs sont généralement alimentés par des batteries et il est essentiel d'appliquer une gestion efficace de puissance et d'énergie pour améliorer la durée de vie de leurs batterie. Néanmoins, en raison de la croissance continue de la consommation de puissance et d'énergie par les serveurs et centres de données, l'utilisation de ces techniques de gestion d'énergie et de puissance électrique ont été détourné vers de tels systèmes. Même si les problèmes causés par la consommation élevée de puissance et d'énergie sont interconnectés, ils ont leurs spécificités et doivent être considérés séparément. La différence est que la consommation de puissance électrique détermine le coût de l'infrastructure nécessaire pour maintenir le fonctionnement du système, tandis que La consommation d'énergie représente les coûts sur les factures d'électricité. Ces deux questions sont discutées en détail dans les deux prochaines sections.

Consommation de puissance électrique

La raison principale de la gestion inefficace de la puissance électrique dans les datacenters est la faible moyenne d'utilisation des ressources. Pour le montrer, les statistiques des charges de travail fournies dans le cadre du projet *CoMon* PLANETLAB (2015a), par une infrastructure de surveillance pour PlanetLab PLANETLAB (2015b), ont été analysées. Des données sur les taux d'utilisation des CPU par plus d'un millier de serveurs situés à plus de 500 endroits à travers le monde ont été utilisés dans l'analyse. Les données ont été recueillies à des intervalles de 5 minutes pendant la période du 10 au 19 mai 2010. La répartition des données des taux d'utilisation des CPU, pendant 10 jours, avec les caractéristiques de la répartition sont présentés dans la Figure 1.6.
Les résultats confirment l'observation faite par Barroso et Holzle Barroso et Hölzle (2007) : le taux d'utilisation moyen des CPU est inférieure à 50%. La valeur moyenne de l'utilisation du processeur est 36,44% avec un intervalle de confiance de 95% ce qui implique que la valeur moyenne de l'utilisation du processeur est entre 36,40% et 36,47%.

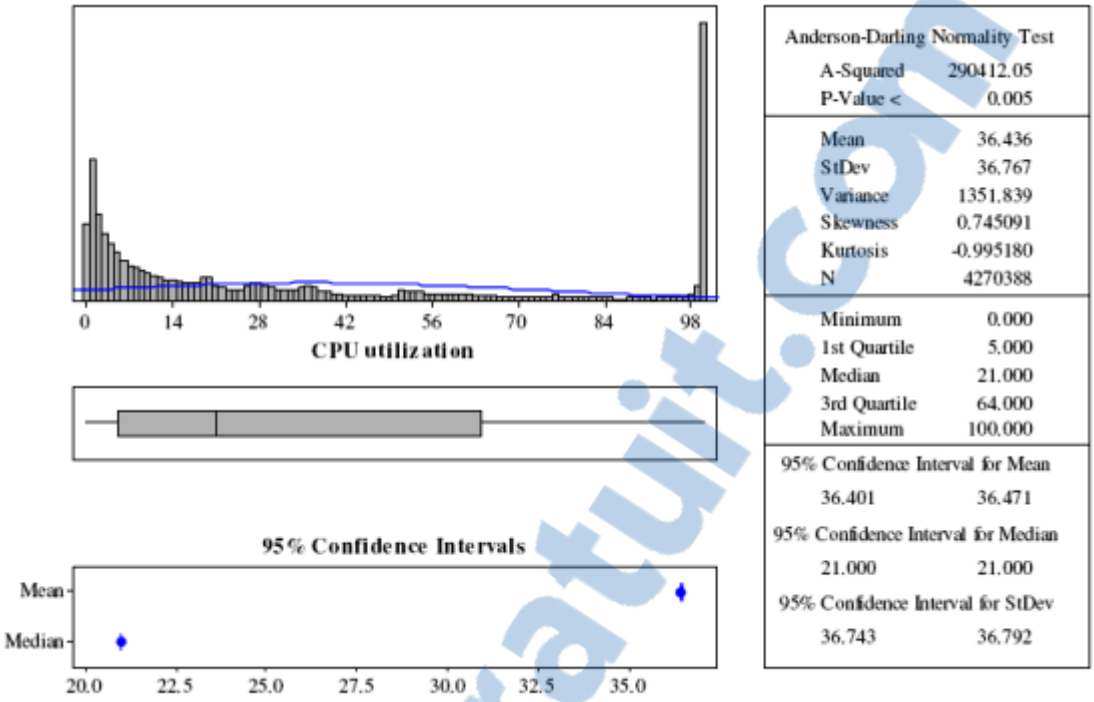


FIGURE 1.6 – *Les taux d'utilisation des CPU de serveurs de PlanetLab pendant une période de 10 jour Beloglazov (2013)*

Les principaux raisons de cette sous utilisation dans les datacenters sont la variabilité de la charge de travail et des effets statistiques. Les applications de services modernes ne peuvent pas être gardées dans des serveurs pleinement utilisés parce qu'une fluctuation, même non significative, de la charge de travail conduirait à une dégradation des performance du système et par conséquent à une incapacité à offrir la qualité de service (QoS) attendue. En outre, malgré le fait que les ressources doivent être provisionnées pour gérer les pics théoriques de charges de travail, il est très peu probable que tous les serveurs d'un des centres de données à grande échelle seront pleinement utilisées simultanément.
Les systèmes où l'utilisation moyenne des ressources est inférieur à 50% sont considérés comme inefficace en gestion de puissance électrique, puisque la plupart du temps seule la moitié des ressources sont en fait en cours d'utilisation. Bien que les ressources sont, en moyenne, utilisées par moins de 50%, l'infrastructure doit être construite pour gérer les pics de charge de travail, ce qui se produit rarement dans la pratique. Dans ces systèmes, le coût de sur-provisionnement est très important et comprend les frais de la capacité supplémentaire du systèmes de refroidissement, PDU, des générateurs, des installations de distribution de puissance électrique, onduleurs, et ainsi de suite.
Un autre problème de la forte consommation de la puissance électrique et l'augmentation de la densité des composants du serveur (ex : Blade servers d'IBM) est la dissipation de chaleur Beloglazov (2013). Une grande partie de la puissance électrique consommée par les ressources informatiques se voit transformée en chaleur qui doit être dissipée pour maintenir ces ressources dans leurs états thermiques sûres. La quantité de chaleur produite par un circuit intégré dépend de l'efficacité de la conception des composants et de la tension et la fréquence à laquelle le composant fonctionne ; en outre, le surchauffe des composants peut conduire à une diminution de leurs durées de vie.

Consommation d'énergie

Pour traiter le problème de la consommation élevée de puissance électrique, il faut minimiser les pics de charges de travail et optimiser la conception des composant d'une infrastructure et leurs utilisations. En revanche, la consommation d'énergie est définie par la consommation moyenne de puissance sur une période de temps. Par conséquent, la consommation d'énergie réelle par un centre de données ne détermine pas directement le coût de l'infrastructure. Cependant, elle se reflète dans le coût de l'électricité consommée par le système, qui est le principal composant des coûts d'exploitation des centres de données. En outre, dans la plupart des centres de données, 50% de l'énergie consommée n'atteint jamais les ressources informatiques : elle est consommée par les installations de refroidissement ou dissipée dans les conversions dans les unités de distribution d'alimentation et les onduleurs. Avec la croissance continue de la consommation d'énergie et les coûts qui y sont associés, le moment où les coûts d'exploitation dépassent le coût des ressources informatiques peut être atteint très prochainement. Par conséquent, il est crucial de développer et appliquer des stratégies de gestion efficace d'énergie des ressources dans les datacenters.
En plus des coûts d'exploitation élevés, un autre problème causé par la consommation croissante d'énergie est les fortes émissions de CO_2, qui contribuent au réchauffement global. Selon une étude menée par Gartner Gartner (2007) en 2007, La technologie de l'information et de la télécommunication était responsable de 2% d'émission mondiale de CO_2 ce qui équivaut aux émissions de l'industrie de l'aviation. L'intense couverture médiatique a permis de sensibiliser les gens à propos du changement climatique et l'effet de serre ce qui a amené de plus en plus des clients à considérer l'aspect "*Green*" dans le choix des produits et services Beloglazov (2013). Outre le souci de l'environnement, les entreprises ont commencé à faire face aux risques commerciaux causés par le fait de ne pas être convivial envers l'environnement. Par conséquent, la réduction des émissions de CO_2 devient un problème important qui doit être abordé dans le but de faciliter la poursuite des progrès et la prolifération des systèmes informatiques.

1.3 Qualité de service dans le Cloud Computing

Le Cloud Computing fournit des ressources informatiques, logicielles ou matérielles, accessible à distance, en tant que service. Avec la propagation rapide de ce modèle et la similarité en termes de services fournis par de multiples acteurs tel que Google et Amazon, en l'occurrence les SaaS du Cloud (*Softaware as a Service*) proposant des services de bureautique, ou les IaaS du Cloud(*Infrastructure as a Service*) proposant des services de calcul ou de stockage, plusieurs défis ont fait surface, notamment, la qualité de service (QoS) des services fournis aux utilisateurs finaux et les méthodes que ces utilisateurs peuvent utiliser pour comparer les services existants entre eux avant d'opter pour le plus approprié. De ce fait, l'intégration de la QoS dans un contrat SLA (*service Level Agreement*) a été proposé et adopté par la communauté Cloud, A l'instar du domaine des télécommunications qui a introduit dès les années 80 le concept de SLA, afin de supporter l'instabilité de la QoS dans un environnement hautement dynamique comme le Cloud en gérant finement la violation des termes de ce contrat Kouki et al. (2013).

1.3.1 Définition du SLA (Service Level Agreement)

Littéralement, le Service Level Agreement, que l'on traduit en français par accord de niveau de service ou contrat de niveau de service, fournit un cadre à l'intérieur duquel le vendeur et l'acheteur d'un service peuvent poursuivre une relation rentable et mutuellement bénéfique BOSE et al. (2011).
Dinesh Dinesh (2004) définit le SLA comme suit : "Une déclaration explicite des attentes et des obligations qui existent dans une relation d'affaires entre les deux organisations : le fournisseur de services et le client". Depuis les années 80's, le SLA a été utilisé dans différents domaines et la plupart des définitions disponibles sont contextuelles et varient d'un domaine à un autre. Les principales définitions du SLA dans des domaines relatives au Cloud Computing sont résumés dans le tableau ci-dessus :

Domaines	**Définitions**	**Sources**
Services Web	"*SLA est un contrat utilisé pour garantir la prestation de services web. Il définit la compréhension et les attentes de fournisseur de services et du consommateur*"	*HP Labs* Frolund et Koistinen (1998)
Réseaux	"*Un SLA est un contrat entre un prestataire service de réseau et un client qui spécifie, généralement en termes mesurables, les services que le fournisseur de services de réseau fournira et quelles sont les sanctions évaluées si le fournisseur de services ne peut pas atteindre les objectifs établis*"	*Research Project* Linlin et Buyya (2010)
Internet	"*SLA construit le fondement juridique de la prestation de services. Toutes les parties concernées sont les utilisateurs du SLA. Le consommateur de service utilise le contrat comme une description juridiquement contraignante de ce que fournisseur a promis de fournir. Le fournisseur de services l'utilise pour avoir un enregistrement précis et définitif de ce que doit être livré*"	*Internet NG* Ron et Aliko (2001)
Gestion des datacenters	"*SLA est un accord formel qui permet promettre ce qu'il est possible de fournir et fournir ce qui est promis*"	*Sun Microsystems* Linlin et Buyya (2010)

TABLE 1.1 – *Résumé des définitions du SLA classées par domaine Linlin et Buyya (2010)*

Dans le Cloud Computing, nous pouvons définir les contrats SLA comme étant une composante importante et essentiel de la relation contractuelle entre un fournisseur de services Cloud et ses clients qui englobe toutes les définitions précédentes. Selon Newsroom (2014) un contrat SLA doit contenir :

- La liste des services que le fournisseur va fournir et une définition complète de chaque service.
- Des métriques pour chaque service pour déterminer si le fournisseur de Cloud garantie la prestation de service comme convenu et un mécanisme d'audit pour surveiller les performances de chaque service.

- Des responsabilités du fournisseur de services et du client et les voies de recours disponibles pour les deux parties si les termes du contrat SLA ne sont pas remplies.
- Une description de la façon dont la SLA va changer au fil du temps.

1.4 Le paradoxe "Amélioration de la qualité de service/Réduction de la consommation d'énergie"

L'efficacité énergétique des systèmes à grande échelle, tel le Cloud computing, est devenu l'une des préoccupations primordiales pour tous les acteurs du Cloud. Par conséquent, les communautés scientifiques et professionnelles ont dirigé plusieurs recherches pour régler ce problème. De nombreux travaux ont été proposés pour diminuer la consommation d'énergie des datacenters. Ces travaux se sont concentrés sur la consolidation des ressources pour assurer une pleine utilisation des machines physiques (PMs), et en éteignant les PMs inactives (qui n'hébergent aucune machine virtuelle). Les résultats ont été très prometteurs en affichant des gains considérables en consommation d'énergie, mais ces travaux, ont provoqué des dégradations de la qualité des service à cause de leurs comportements. Comme il est connu, l'une des tâches principales d'un environnement Cloud est de fournir une QoS fiable aux utilisateurs. Les conditions de fiabilité peuvent être introduites dans des contrats SLA en déterminant par exemple le débit minimum requis, le temps de réponse désirés, ou la latence maximale délivrées par le système déployé, et même si les technologies de virtualisation modernes peuvent assurer une isolation des performances entre les machines virtuelles qui partagent le même nœud physique, due à la consolidation agressive de machines virtuelles et à la variabilité de la charge de travail, certaines VMs ne pourront pas allouer la quantité de ressources nécessaires à la demande puisque les machines physiques inactives sont déjà éteintes. Cela peut conduire à des pertes en termes de la qualité des services et des violations d'un ou plusieurs des termes du contrat SLA, ou des échecs d'exécution dans les pires des cas. Par conséquent, il est impératif que les fournisseurs de Cloud computing trouvent des compromis acceptables entre la réduction de la consommation d'énergie et l'amélioration de la qualité de service. Ainsi, comment économiser l'énergie d'un datacenter sans affecter la qualité des services qu'il fournisse ? est la question qui sera traitée dans notre travail de thèse.

1.5 Conclusion

Au cours des dernières années, l'efficacité énergétique est devenue l'une des exigences de conception les plus importantes pour les systèmes informatiques modernes, allant de simples serveurs à de centres de données dans le Cloud Computing, alors qu'ils continuent de consommer d'énormes quantités de puissance électrique. En plus des coûts d'exploitation élevés encourus par les ressources informatiques, cela conduit à des importantes émissions de CO_2 dans l'environnement.

SLA est un contrat qui relie deux ou plusieurs parties incluant des termes à respecter et des pénalités ou des sanctions qui sont appliquées en cas de violation de l'un de ces termes.

Dans ce chapitre, nous avons défini la puissance électrique et l'énergie et nous avons montré la différence entre les deux. Les principes de consommation électrique statique et dynamique ont été présentés. Nous avons montré aussi que le CPU est le composant

qui a le plus d'impact sur la consommation de puissance électrique et comment des recherches ont été capable de proposer des formules pour quantifier la puissance électrique et l'énergie. En plus ce chapitre s'est concentré aussi sur les contrat SLA et ses définitions. Á la fin de ce chapitre, nous avons souligné le paradoxe "Amélioration de la qualité de service/Réduction de la consommation d'énergie" où nous avons expliqué la difficulté de satisfaire les deux contraintes

Le chapitre suivant sera consacré à la description de la virtualisation et la migration des données dans le Cloud Computing, qui sont deux techniques de base utilisées pour la réduction de la consommation d'énergie et/ou l'amélioration de la qualité de service.

Virtualisation et migration des entités dans le Cloud Computing

2

Sommaire

2.1 Introduction

La virtualisation a été la première pierre vers l'ère du Cloud Computing. En effet, cette notion permet une gestion optimisée des ressources matérielles dans le but de pouvoir y exécuter plusieurs systèmes "virtuels" sur une seule ressource physique et fournir une couche supplémentaire d'abstraction du matériel. Les premiers travaux peuvent être attribués à IBM, qui dans les années 60, travaillait déjà sur les mécanismes de virtualisation en développant dans les centres de recherche de Cambridge et de Grenoble le tout premier hyperviseur DuCharme (2001).

La virtualisation est une technique qui permet à plusieurs systèmes d'exploitation de fonctionner simultanément sur une seule machine physique. Elle est devenue un aspect essentiel dans les serveurs modernes et des centres de données grâce à plusieurs avantages tels que le partage souple et efficace des ressources, la tolérance aux pannes, la portabilité et l'efficacité des coûts Barham et al. (2003). Dans un environnement virtualisé, les machines virtuelles (VM) agissant comme des machines physiques réelles et peuvent fonctionner en parallèle et indépendamment les unes des autres et pourtant partager les mêmes ressources physiques. Cette capacité à démarrer une image d'une VM sur n'importe quelle machine physique (PM) disponible dans un même centre de données, ou même à travers des centres de données, est un facteur clé pour des nombreux avantages promis par le Cloud Computing tels que la consolidation des ressources, la mise à l'échelle, l'élasticité, et la migration de calcul et de machines virtuelles.

La migration est une des capacités importantes de la virtualisation des systèmes, qui permet aux entités d'être migrées, de manière transparente, ainsi que leurs environnements d'exécution à travers des PMs, des serveurs ou des datacenters. Elle est un moyen important pour la gestion des applications et des ressources dans un large système virtualisé. Elle permet l'équilibrage dynamique de l'utilisation de ressources dans l'ensemble du système virtualisé à travers les hôtes physiques, et elle permet également aux applications à être déplacés de façon dynamique pour améliorer les performances et la fiabilité Wu et Zhao (2011).

Ce chapitre vise à introduire les termes "virtualisation" et "migration" dans le Cloud Computing, à présenter les principes des ces deux techniques et à montrer leurs avantages et leurs apports à la réduction de la consommation d'énergie et l'amélioration de la qualité de services.

2.2 Virtualisation dans le Cloud Computing

La virtualisation est une technologie qui sépare les fonctions de calcul et les implémentations logicielles du matériel physique. Elle offre un isolement entre le matériel et le logiciel, entre les utilisateurs et entre les processus et les ressources. Les problèmes d'isolement ne sont pas bien résolus par les systèmes d'exploitation traditionnels. Avec la virtualisation, un logiciel qui s'exécute sur le matériel peut être exécuté dans une machine virtuelle. Les services des systèmes Cloud peuvent être encapsulés dans les machines virtuelles Kecskemeti et al. (2011), et déployés par l'instanciation de ces machines Kecskemeti et al. (2010).
En général, nous pouvons dire que la virtualisation est un moyen permettant la création des objets dites virtuels. Elle peut être effectuée à différents niveaux, à partir de l'ordi-

nateur entier à la création de composants individuels. Par exemple, nous pouvons avoir un disque virtuel, une mémoire virtuelle, ou un processeur virtuel. En plus, à l'aide d'un outil de virtualisation, un système d'exploitation différent peut être hébergé et démarré dans un autre système d'exploitation. La virtualisation change presque tous les aspects de l'administration du système d'exploitation et du stockage. En utilisant la technologie de virtualisation, nous pouvons créer des ressources virtuelles à la demande donnant, ainsi plusieurs possibilités pour économiser des moyens physiques et des dépenses supplémentaires. Et nous ne devons pas oublier deux autres avantages importants de la technologie de virtualisation qui sont la tolérance aux pannes et la haute disponibilité. Spécifiquement, elle permet à plusieurs système d'exploitations de s'exécuter dans une seule machine physique (par exemple : Windows, Linux, etc.) et le nombre de machines virtuelles qui peuvent être exécutées simultanément dépend des spécifications matériels de la machine physique.

La virtualisation de systèmes d'exploitation a plusieurs intérêts :

- Utiliser un autre système d'exploitation sans redémarrer son ordinateur, afin d'utiliser des programmes ne fonctionnant pas nativement dans l'OS de la machine physique ;
- Exploiter des périphériques ne fonctionnant pas dans l'OS de la machine physique mais fonctionnant dans d'autres systèmes d'exploitation ;
- Tester des systèmes d'exploitation en cours de développement sans compromettre un environnement quotidien stable ;
- Tester des logiciels dans des environnements contrôlés, isolés et sécurisés ;
- Transporter ses systèmes d'exploitation d'un ordinateur à l'autre, une machine virtuelle fonctionnant sur n'importe quel ordinateur, disposant d'un hyperviseur compatible.

Plusieurs types de virtualisation existent, mais tous fonctionnent selon un même principe :

- Un système d'exploitation principal (appelé *système d'exploitation hôte*) est installé dans l'ordinateur et sert de système d'accueil à d'autres systèmes d'exploitation ;
- Dans le système d'exploitation hôte, un logiciel de virtualisation (appelé *hyperviseur*) est installé. Celui-ci crée des environnements clos, isolés, avec des ressources bien précises : ces environnements clos sont appelées *des machines virtuelles* ;
- D'autres systèmes d'exploitation (appelés *systèmes d'exploitation invités*) peuvent alors être installés dans des machines virtuelles. Leur instance est totalement isolée du système hôte et des autres systèmes invités.

À l'heure actuelle, les machines virtuelles sont déployées dans la plupart des centres de données tant que serveurs normaux, mais avec beaucoup moins d'entretien et coûts d'administration Marshall (2011).

2.2.1 Types de virtualisation

Il existe actuellement différents types de virtualisation et chaque type répond à un besoin, une utilisation particulière et dispose de ses propres contraintes et avantages. Pour aborder la suite, il est important de comprendre le vocabulaire suivant :

- **Système hôte** : Machine physique (et son système) qui virtualise le système invité.
- **Hyperviseur** : Logiciel permettant de créer et gérer des machines virtuelles. Il est installé sur la machine hôte.
- **Système invité** : Machine virtuelle s'exécutant sur un hôte.

Virtualisation complète

La virtualisation complète LEFEVRE (2014) consiste à présenter au système d'exploitation invité un matériel virtuel complet s'exécutant de la même manière qu'un système physique. Cela permet à un système d'exploitation invité de s'exécuter sans modifications préalables et sans avoir conscience d'être une machine virtuelle. Les système hôtes et invités doivent supporter la même architecture, par exemple un système invité x64 sur un système hôte x64. Cette solution apporte une certaine isolation entre les machines virtuelles et ces machines s'exécutent en parallèle sur un ou plusieurs serveurs. Cela permet de centraliser les coûts et d'exploiter au maximum le matériel. C'est la technologie la plus utilisée dans les environnements complexes puisqu'elle permet de virtualiser différents types de systèmes d'exploitation (Windows, Linux, etc.) sur une même plate-forme et sans modifications (voir Figure 2.1).
Parmi les hyperviseurs qui utilisent ce type de virtualisation, nous pouvons citer : **KVM**, **VirtualBox**, **VMWare Worksation**.

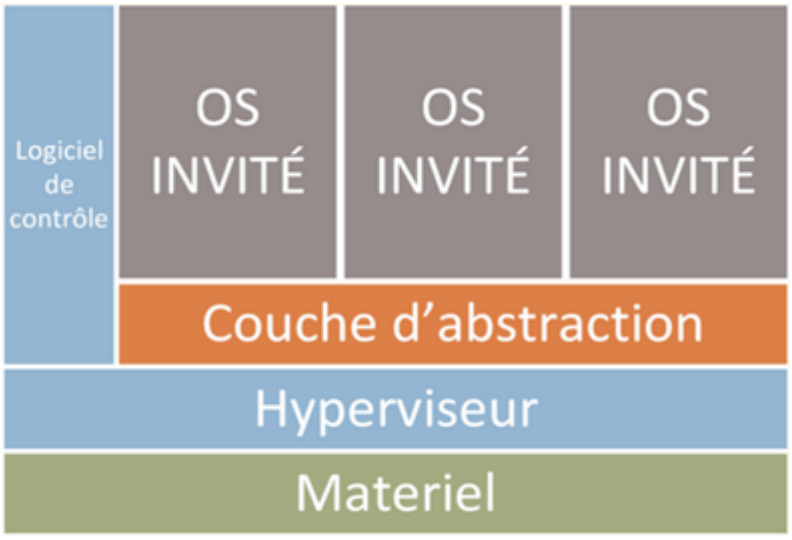


FIGURE 2.1 – *La virtualisation complète LEFEVRE (2014)*

Emulation

C'est la forme la de virtualisation la plus basique. L'hyperviseur présente un matériel virtuel complet au système invité qui peut être d'une architecture CPU complètement différente du systèmes hôte. L'hyperviseur convertit ensuite les instructions CPU interceptées de la machine virtuelle en instructions compréhensibles par son CPU. C'est la technique utilisée par les émulateurs de consoles de salons qui disposaient souvent d'une architecture non standardisée (voir Figure 2.2). Cette technique offre des performances relativement médiocres dues à la différence d'architecture CPU LEFEVRE (2014).
Certains hyperviseurs adoptent ce type de virtualisation, à savoir : **QEMU**, **Microsoft VirtualPC**, **MacOnLinux**.

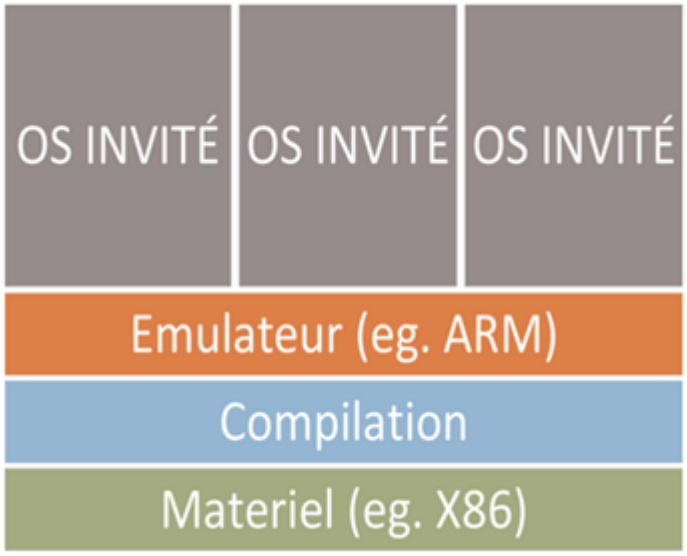


FIGURE 2.2 – *L'émulation LEFEVRE (2014)*

Para-virtualisation

La para-virtualisation LEFEVRE (2014) se situe entre l'exécution sur machine physique et la virtualisation complète. Pour améliorer les performances, l'hyperviseur permet à la machine virtuelle de communiquer avec lui (voir Figure 2.3). Il fournit une interfaces de communication à la machine virtuelle ce qui permet d'exécuter des opérations plus rapidement via l'hyperviseur plutôt que de les exécuter en environnement virtuel.
Nous pouvons citer quelques hyperviseurs utilisant ce type de virtualisation, comme : **Xen**, **VM/CMS d'IBM**, **ESX de VMWare**.

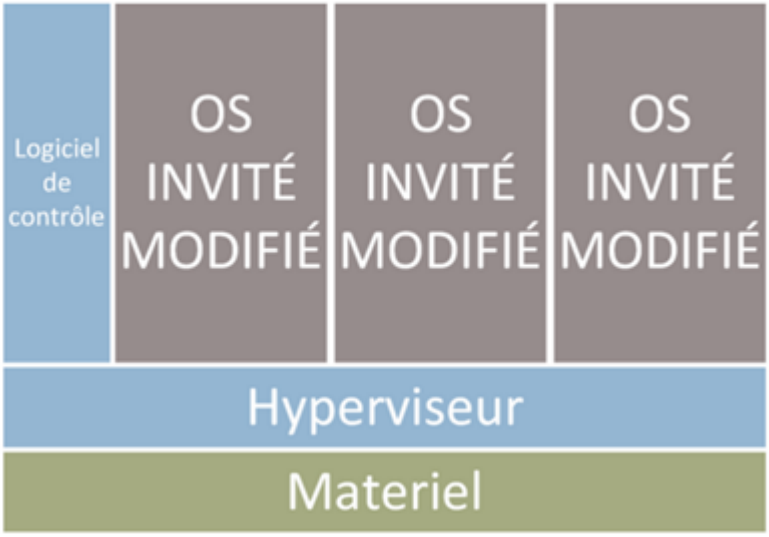


FIGURE 2.3 – *La para-virtualisation LEFEVRE (2014)*

Environnement virtualisé (Operating system-level virtualization) ou Conteneurs et Isolateurs

Chaque environnement exécuté possède son propre espace mémoire, afin d'être isolé des autres environnements, les autres ressources systèmes sont partagées (pilotes, noyau,

etc.). Les instructions ne sont pas interceptées, et l'environnement exécuté doit impérativement être compatible avec celui du système hôte. (Nous ne pouvons pas, par exemple, exécuter Microsoft® Windows® dans un environnement virtuel au-dessus d'un hôte de type Linux). Cette solution est performante, du fait du peu de surcharge système (overhead), mais les environnements virtualisés ne sont pas complètement isolés (voir Figure 2.4). En effet si un des systèmes isolés est défaillant et met en péril le noyau, le système hôte s'écroule et entraîne dans sa chute l'ensemble des systèmes virtuels. les conteneurs ou les isolateurs améliore sensiblement la sécurisation du système dans la mesure où en cas de corruption ou d'attaque d'un des serveurs virtuels, l'attaquant n'aura accès qu'à l'environnement isolé Colin et Desbureaux (2009).
Nous pouvons citer comme exemples des environements virtualisés : **Linux Vserver**, **Chroot**, **BSD Jail**, **OpenVZ**

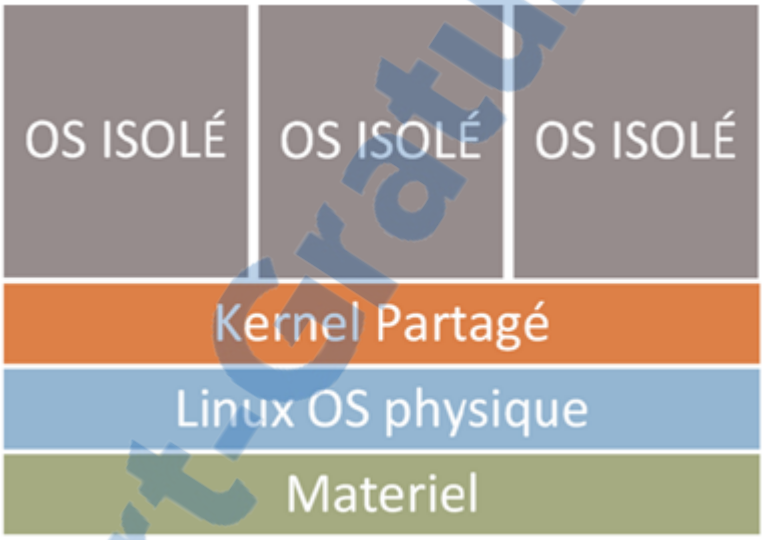


FIGURE 2.4 – *Les conteneurs ou les isolateurs Colin et Desbureaux (2009)*

2.2.2 Hyperviseurs

Les acteurs de marché de solutions de virtualisation ont attribué plusieurs définitions aux hyperviseurs Desai et al. (2013) ExpertGlossary (2015) :

- L'hyperviseur est la couche logicielle qui fait abstraction du matériel à partir du système d'exploitation afin de permettre à plusieurs systèmes d'exploitation de fonctionner sur le même matériel. L'hyperviseur s'exécute sur le système hôte permettant ainsi d'exécuter des machines virtuelles sur le même système hôte ***-Red Hat-***.
- Une fine couche de logiciel, qui fournit généralement des capacités de partitionnement virtuel qui s'exécute directement sur le matériel, mais sous les services de virtualisation de niveau supérieur ***-VMWare-***.
- C'est un terme alternatif pour le gestionnaire de machine virtuelle, utilisé car il signifie "au-delà de superviseur", puisqu'il est responsable de la gestion des multiples noyaux "superviseur" ***- xen.org-***.

Nous pouvons citer plusieurs hyperviseurs existant sur le marché, à savoir : **XEN**, **KVW**, **ESX**, **VMWare Workstation**, **QEMU**, **etc**.

2.2.3 Rôle de la virtualisation dans le Cloud Computing

Contrairement au modèle SaaS (*Software as a Service*), les modèles PaaS (*Platform as a Service*) et IaaS (*Infrastructure as a Service*) permettent aux utilisateurs d'installer leurs propres applications sur des espaces de stockages fournis. Dans le modèle IaaS, les utilisateurs peuvent même avoir leurs propres systèmes d'exploitation, ce qui rends la technologie de virtualisation indispensable et considérée comme une technologie clé pour fournir ces services.
Fournir tout un système matériel réel pour chaque utilisateur est difficile car il nécessite un espace physique dans les racks des serveurs pour le placer, une alimentation suffisante pour le faire fonctionner, un système de refroidissement et des coûts d'installation et de déploiement. Ainsi, dans le cloud computing, l'utilisation de la virtualisation a retenu l'attention ; au lieu d'installer une machine physique pour chaque client, il suffit de créer une machine virtuelle et de la lui attribuer de manière rapide et transparente, ce qui permet d'éviter les exigences décrites précédemment.

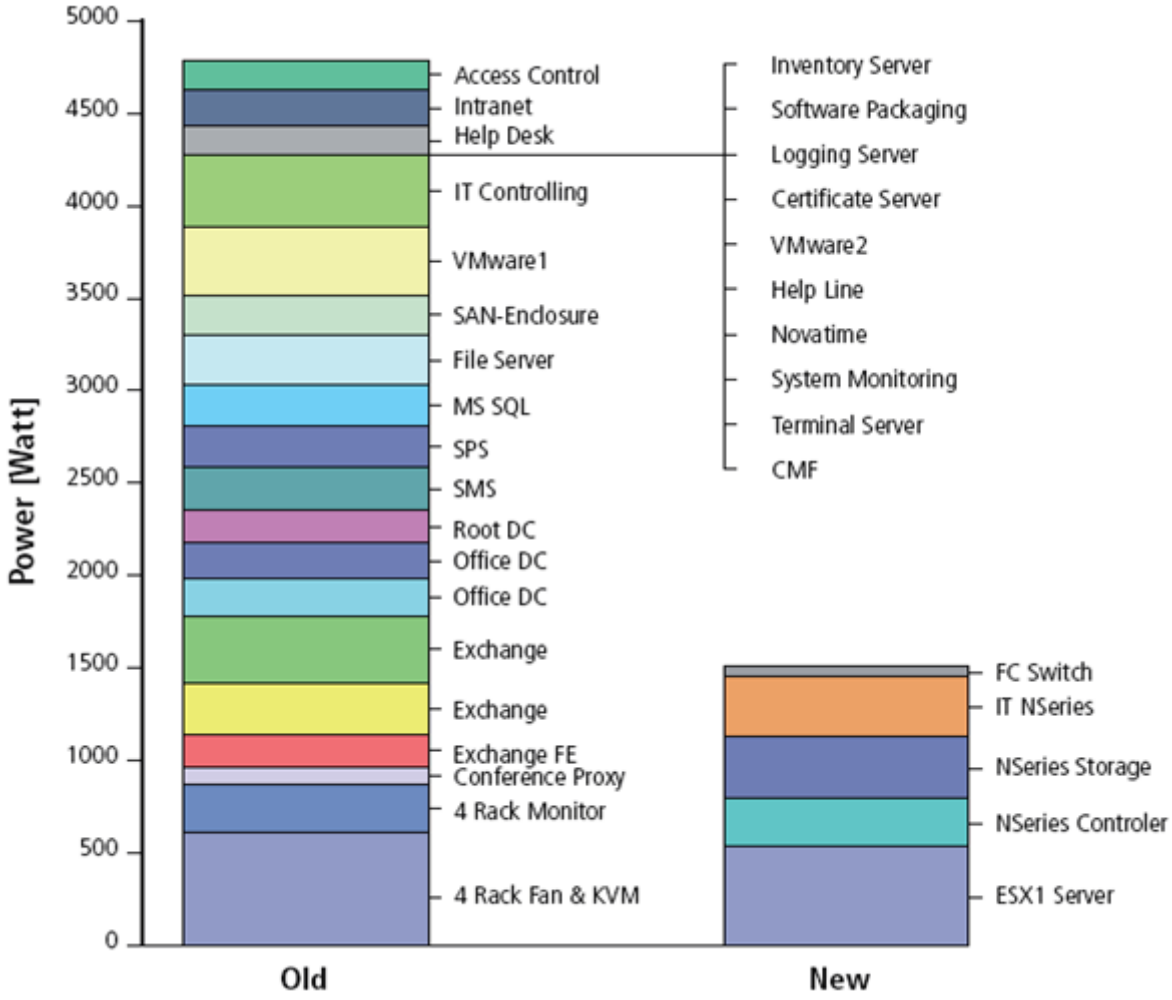


FIGURE 2.5 – *Réduction de la consommation de puissance électrique par la virtualisation Shappl et Durandy (2011)*

Potentiel d'économies d'énergie et de puissance électrique de la virtualisation

La virtualisation est l'une des technologies les plus puissantes pour réduire la demande en énergie dans les datacenters et les serveurs. La consolidation des matériels, en concentrant la charge de travail sur un nombre réduit de serveurs physiques, permet souvent d'économiser 40 à 80% d'énergie, voire plus. La technologie actuelle offre la possibilité de mettre en œuvre une virtualisation avec un facteur de consolidation d'au moins 10-20%, en fonction des systèmes et des exigences spécifiques. La Figure 2.5 montre un exemple de consolidation de serveurs par virtualisation au ministère Allemand de l'environnement. Ces mesures spécifiques ont permis d'économiser environ 68% d'énergie, avec une réduction des matériels à deux pour les serveurs physiques sous VMware ESX Galvin (2009).

Un autre exemple de virtualisation par IBM Xen (2009) avec les serveurs lames (*Blade servers*) montre que les économies d'énergie pourraient dépasser 90% si toutes les mesures appropriées étaient envisagées au niveau du matériel et de l'infrastructure. Ces exemples montrent bien que la consolidation par virtualisation est l'une des principales options pour améliorer sensiblement l'efficacité énergétique des datacenters. Toutefois, comme dans les autres approches purement informatiques, le potentiel d'économie ne peut être pleinement atteint que s'il s'accompagne de mesures parallèles sur l'infrastructure, notamment l'alimentation et le refroidissement.

2.2.4 Avantages de la virtualisation

Depuis de nombreuses années, les performances des équipements informatiques n'ont cessées d'évoluer pour atteindre aujourd'hui une puissance extraordinaire. Les applications proposées de nos jours ont besoin de beaucoup de ressources mais paradoxalement n'utilisent qu'une fraction du potentiel de certains serveurs. Selon Microsoft, il est souvent possible de regrouper jusqu'à 5 serveurs sur une seule machine sans perte de performances. La virtualisation apporte donc de nombreux avantages Muda (2010) HORALEK et al. (2011) :

- Elle permet de diminuer le nombre de machines physiques, ce qui entraine un retour sur investissement intéressant (les investissements pour le déploiement ou le renouvellement des serveurs sont donc nettement moindres en termes d'achat de matériel).
- Elle fournit une allocation dynamique de la puissance de calcul en fonction des besoins de chaque application et de chaque utilisateur.
- En termes d'espace nécessaire, un serveur capable de faire fonctionner différents systèmes d'exploitation sur une seule machine réduit en moyenne de moitié l'espace réservé aux serveurs dans une entreprise.
- En plus de cette réduction de place, la diminution du nombre de machines physiques entraine une réduction de la consommation énergétique. Nous sous-estimons souvent le coût d'alimentation en électricité des serveurs et de la climatisation nécessaire. Ces coûts, cumulés tout au long de la vie de vos serveurs, sont loin d'être négligeables rapportés au coût d'acquisition d'un serveur. Un seul serveur physique coûtera nettement moins cher en climatisation et électricité sur une période de 4 ans que 3 ou 4 serveurs.

- En outre, moins de machines veut dire moins de contrats de supports matériels (souvent très cher sur les serveurs, où la maintenance doit intervenir rapidement en cas de panne matérielle).
- En plus de ces avantages, la virtualisation permet une gestion simplifiée du parc serveurs et d'utiliser les ressources de ce parc de manière optimale (répartition des VMs sur les machines physiques en fonction des charges respectives).
- Les applications étaient autrefois étroitement liées aux serveurs sur lesquels elles s'exécutaient. La technologie de virtualisation créant une couche d'abstraction entre le matériel physique et les logiciels, elle permet l'exécution et la cohabitation de plusieurs serveurs bien distincts sur une même machine. Ainsi, des applications métiers développées en interne ne s'exécutant que sur un ancien OS (comme NT4 par exemple) peuvent être conservées sans garder les contraintes liées à l'ancien serveur physique.
- La virtualisation permet de réduire le temps et le coût souvent élevés consacrés à l'administration des serveurs. La gestion du parc machine est plus facile, ce qui allège la charge des administrateurs.
- Elle simplifie la mise en place de plateformes de tests ou de production en réduisant le temps de mise à disposition d'un serveur (la possibilité d'installer, tester, déployer et migrer facilement des machines virtuelles d'une machine physique à une autre.
- Elle augmente la disponibilité des serveurs avec une reprise d'activité plus rapide que pour une machine physique puisque grâce à la portabilité des machines virtuelles évoquée ci-dessus, il est très simple et rapide de reprendre l'activité suite à un désastre. Il suffit de sauvegarder les machines virtuelles et de les mettre sur une autre machine physique supportant la virtualisation (temps d'installation de l'ordre d'une journée) pour pouvoir relancer votre activité.
- Enfin, elle permet d'améliorer la sécurité et l'isolation d'un environnement (l'attaquant peut casser des systèmes d'exploitation virtuels, mais ne pourra pas casser les systèmes d'exploitation hôtes qui lui sont invisibles)

2.3 Migration des entités dans le Cloud Computing

La migration est le processus de déplacer une entité d'un serveur hôte ou d'un emplacement de stockage vers un autre. Cette entité peut être un disque de stockage virtualisé, une machine virtuelle, des services ou des processus, ou des objets de base de données. La migration représente un outil important pour la gestion d'applications et des ressources dans un système à grande échelle tel que le Cloud Computing. Elle fournit un équilibrage de charge, une tolérance aux pannes, et une bonne gestion de la consommation d'énergie.

2.3.1 Types de migration

Nous distinguons trois catégories de migration selon le processus, et trois types selon la nature de l'entité à faire migrer :

Processus de migration

1. **Migration à chaud (Live migration) :** La migration à chaud permet de déplacer une machine virtuelle d'un nœud physique à un autre sans interruption de service, ce qui rend le processus totalement transparent pour l'utilisateur. Ce processus se compose de six étapes principales (voir Figure 2.6) Clark et al. (2005b) Clark et al. (2005a) :

   - **Étape 0 : Prémigration.** Une machine virtuelle active existe sur la machine physique A.
   - **Étape 1 : Réservation.** L'hôte source envoie la requête de migration à l'hôte de destination et attend sa confirmation de disponibilité des ressources. L'hôte de destination réserve par la suite la mémoire nécessaire à la machine virtuelle.
   - **Étape 2 : Copie itérative de pages mémoire.** Pendant la première itération toutes les pages mémoire de la machine virtuelle source sont copiées sur l'hôte de destination dans la mémoire qui lui est allouée. Dans les itérations ultérieures, seulement les pages mémoire modifiées (*dirty pages*) durant la phase de transfert précédente sont copiées.
   - **Étape 3 : Stop et copie.** La machine virtuelle est mise en pause afin de permettre une ultime copie des pages mémoire modifiées. L'hôte transfère également les registres processeur ainsi que l'état des périphériques à la machine de destination. À la fin du transfert, la VM cible devient une copie parfaite de la VM source.
   - **Étape 4 : Redirection.** L'hôte de destination indique à l'hôte source qu'il a correctement reçu l'image de la VM.
   - **Étape 5 : Activation de la machine virtuelle.** La machine virtuelle migrée est activée et peut continuer son exécution.

   En fin de processus, un message de mise à jour est envoyé au switch physique afin d'adapter sa table de routage. De ce fait, les paquets à destination de la machine virtuelle ne passent plus par le même port physique.

2. **Migration à froid (Cold migration) :** La migration à froid est la migration d'une machine virtuelle hors tension. Avec la migration à froid, vous avez la possibilité de déplacer les disques associés d'un centre de données à un autre. Les machines virtuelles ne sont pas tenues d'être sur un support de stockage partagé. Le processus de migration à froid est simple à mettre en œuvre (comme c'est le cas pour le produit VMware), et il peut se résumer comme suit Gurobi (2011) BOSE et al. (2011) :

   - Les fichiers de configuration, y compris le fichier NVRAM (paramètres du BIOS), les fichiers journaux, ainsi que les disques de la machine virtuelle, sont déplacés de l'hôte source vers l'hôte de destination
   - La machine virtuelle est enregistrée avec le nouvel hôte de destination.
   - Une fois la migration terminée, l'ancienne version de la machine virtuelle est supprimée de l'hôte source.

   Remarque
   Il est important de souligner que les deux principales différences entre la migration à chaud et la migration à froid sont que la migration à chaud a besoin d'un stockage partagé pour les machines virtuelles dans l'ensemble du serveur, en plus dans la

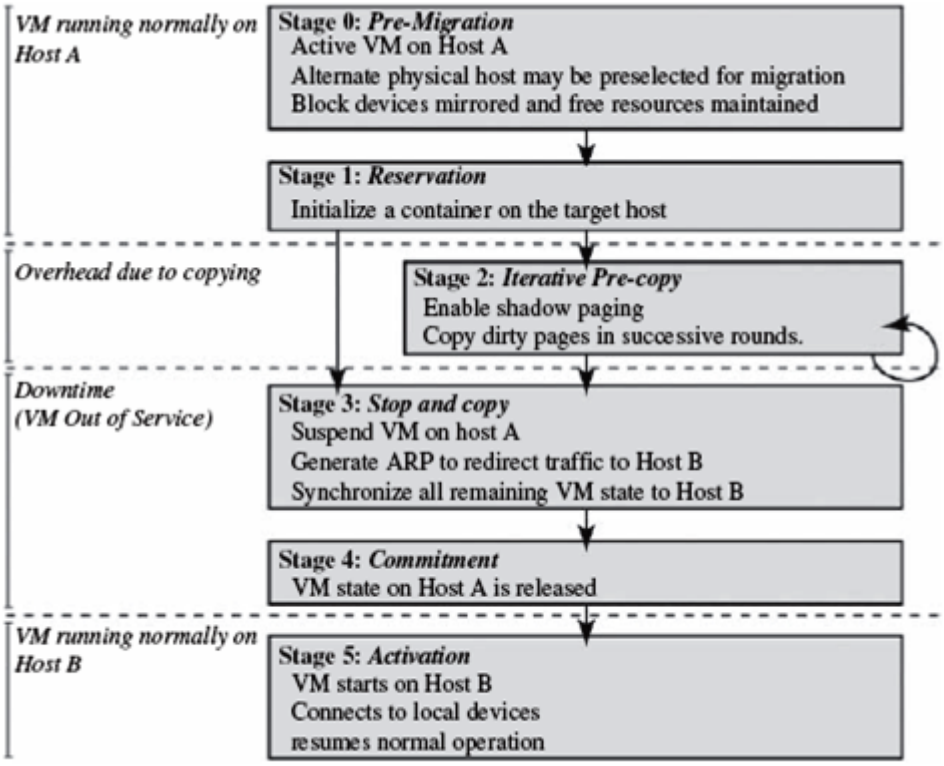


FIGURE 2.6 – *Le processus de migration à chaud Clark et al. (2005a)*

migration à chaud d'une machine virtuelle entre deux hôtes, il y aurait certaines vérifications de compatibilité CPU, contrairement à la migration à froid où ces contrôles ne s'appliquent pas.

3. **Migration des disques virtualisés (Live storage migration) :** Ce type de migration représente le déplacement des disques virtuels ou d'un fichier de configuration d'une machine virtuelle en cours d'exécution à un nouveau centre de données sans l'interruption de la disponibilité du service de la machine virtuelle Buyya et al. (2011).

Nature de l'entité à faire migrer

1. **Migration des services ou des processus :** Elle est définie par le transfert de service et la possibilité de déplacer l'exécution d'un service ou un processus d'une machine réelle à une autre sans interruption. La migration de processus a été prévue pour améliorer les performances du serveur lorsque ce processus nécessite plus de ressources dont le serveur ne dispose pas. Ainsi, le processus sera déplacé vers un autre serveur qui dispose de ces ressources.
   Un processus est une instance d'un programme, qui pourrait être une application d'un utilisateur ou un programme lié au système, fonctionnant sur un système d'exploitation. Ces processus partagent les ressources matérielles qui sont gérées par le systèmes d'exploitation.
   Quand un processus est exécuté sur un système d'exploitation, un espace de mémoire pour stocker l'image de ce processus est alloué. Ensuite, le processus s'exécute sur l'ordinateur, stockant des données nécessaires sur les segments de mémoire

pour traiter les instructions nécessaires au CPU. Beaucoup de processus partagent des ressources autres que l'espace mémoire tel que les descripteurs de fichiers pour la lecture et l'écriture, les sockets pour la communication dans un réseau IP, etc. l'exécution des programmes dépendent de plusieurs paramètres, à savoir les bibliothèques partageables, les caractéristiques de l'OS et les espaces de noms, y compris les systèmes de fichiers et les IDs de processus. Certains processus stockent un grand bloc de données qui peuvent consommer un nombre non négligeable de pages mémoire Muda (2010).
De nos jours, les tailles des images de processus sont devenues plus larges, et les relations entre multiples processus qu'ils soient ou non dans le même dispositifs physiques sont de plus en plus courantes. Ainsi, on peut dire que la gestion des ressources consommées par un processus est devenu très complexe dans les systèmes d'exploitation modernes.
En plus des ressources visibles, les processus consomment du temps CPU, qui est l'un des principales ressources sur un système informatique. Même si plusieurs processus semblent être en cours d'exécution sur un seul ordinateur, chaque cœur du CPU peut être utilisé par, seulement, un seul processus à la fois. L'exécution en parallèle de nombreux processus est devenue possibles avec le système de partage de temps ("*Time Sharing System*") et les processus qui ne sont actuellement pas exécutés sur le CPU sont mis dans un état "sommeil" en attendant leurs tours respectifs.

2. **Migration dans les bases de données :** Les nouvelles applications sur Internet connaissant un grand succès et des défis comme :

   - La croissance exponentielle de volume de données qu'ils doivent gérer.
   - La gestion des accès multiples à ces données (Architecture « multitenant Maenhaut (2014) » est une expression anglaise qui signifie « multilocataire »).

   Ces applications nécessitent de créer de nouveaux systèmes de gestion de bases de données (SGBDs) qui sont adaptables au Cloud. C'est ce qui a donné naissance aux SGBD NoSQL (*Not only SQL*) Bruchez (2015). La migration des bases de données consiste à exporter le schéma de la base de données puis transférer les données vers un autre un nœud pour garantir une bonne qualité de services (Service Level Agreement) pour les utilisateurs, et un bon équilibrage de charge. Bien que la plupart des SGBDs NoSQL supporte la migration des données, ils utilisent des techniques de migration lourdes comme la migration à froid (Cold migration or stop and migrate) qui consistent à mettre en attente (downtime) toutes les transactions pendant la période de la migration engendrant une dégradation des performances du service.

   Un système de gestion de base de données relationnels (SGBDR) permet de réaliser des transactions atomiques, cohérentes, isolées, et durables (ACID) (Atomicity, Consistency, Isolation, Durability). Les capacités ACID garantissent que si plusieurs utilisateurs font de manière simultanée des modifications des données, toutes les modifications vont être prises en compte, dans un ordre précis et maîtrisé de manière à avoir un résultat cohérent (intégrité des données) avec l'historique des modifications faites par chacun. Mais la mise en œuvre stricte des capacités ACID entraine forcément un coût et un niveau de performance faible dans un environnement distribué où le nombre de requêtes est très grand et l'accès concurrent est fréquent Bruchez (2015).

NoSQL (Not Only SQL) désigne une catégorie de SGBD non-relationnels proposant des alternatives au langage SQL et au modèle relationnel utilisés dans les bases de données traditionnelles. Ces nouveaux SGBD sont apparues il y a quelques années pour répondre aux besoins des sites à fort trafic comme Google, Amazon ou Facebook. C'est ce qui leurs a permet de disposer de bases de données plus adaptées au stockage d'un volume massif de données. Ils offrent de meilleures performances en lecture/écriture que les bases de données relationnelles et possèdent une grande évolutivité (en terme de scalabilité horizontale). En contrepartie, ils ne possèdent pas toutes les propriétés ACID qui garantissent la fiabilité d'une base de données. NoSQL ne remplace donc pas le modèle relationnel mais offre des solutions intéressantes dans certaines situations où les performances priment sur la cohérence des données Bruchez (2015).

3. **Migration des machines virtuelles :** C'est une action permettant de déplacer une machine virtuelle d'une machine physique vers une autre. Une machine virtuelle fonctionnant sur un premier hôte physique peut être migrée vers un autre hôte physique. La migration elle-même englobe le transfert de l'état persistant de la machine virtuelle, le transfert de l'état instable de la machine virtuelle (par exemple contenu de la RAM et de l'état du processeur et la redirection du trafic réseau). Une fois que le transfert d'état est terminé, la machine virtuelle continue à fonctionner dans la nouvelle machine physique. La migration des VMs peut être effectuée soit par une migration à froid, par comme une migration à chaud.

Comparaison entre la migration des processus et la migration des machines virtuelles

Même si la migration des processus et la migration des machines virtuelles sont appelées "Migration", il existe de nombreuses différences entre ces deux styles de migration comme montré dans le Tableau 2.1 :

Entité	***Processus***	***Machine virtuelle***
Synchronisation du contenu de mémoires	*Réduite*	*Grande*
Gestion du fichier descripteur	*Compliquée*	*Plus facile*
Continuité des connexions au réseau	*Plus difficile*	*Plus facile*
Gestion des thread	*Difficile*	*Facile*
Vérification de l'environnement utilisateur	*obligatoire*	*Inutile*

TABLE 2.1 – *Différences entre la migration des processus et la migration des machines virtuelles*

L'une des différences majeures entre les deux styles est l'entité qui est transférée durant la procédure de migration. la migration d'un processus consiste à transférer, uniquement, l'image mémoire de l'application en cours d'exécution, tandis que la migration d'une machine virtuelle transfère l'image mémoire de l'entière VM. Sur le site Mosix, les auteurs soutiennent que la migration de machine virtuelle est coûteuse en temps et en mémoire Kecskemeti et al. (2011) Muda (2010) et il est vrai que la migration d'une machine virtuelle dont un large espace mémoire est alloué prend beaucoup plus de temps. En termes de synchronisation de mémoire, la migration de processus peut sembler meilleur que la migration des VMs, en fonction de la charge de travail qui provoque la migration. Cependant, sur les machines virtuelles, les application peuvent être exécutées sans modification tant qu'elles sont pris en charges par le système d'exploitation de l'hôte de destination. Les bibliothèques partagées et d'autres fichiers qui sont nécessaires à l'exécution des processus seront également migrées avec la machine virtuelle. Ainsi , il est inutile de s'inquiéter

de savoir si oui ou non ces bibliothèques ou fichiers sont disponibles sur l'hôte de destination.
En revanche, plusieurs limitations sont attribuées à la migration des processus tel que le traitement de la mémoire partagée Marshall (2011) Muda (2010) et les applications multithread LEFEVRE (2014) Muda (2010). L'utilisation de la mémoire partagée avec la migration des processus est un obstacle car il devient difficile de changer le contenu de la mémoire sur un nœud pendant l'exécution du processus sur l'autre nœud ; le même espace mémoire utilisé sur l'hôte source, peut être ou ne pas être disponible sur l'hête de destination. Lorsque les applications deviennent compliquées en matière de gestion des ressources, la migration des processus peut ne pas être aussi efficace qu'elle devrait être.
Les limitations de la migration des processus peuvent être atténuées en choisissant de migrer la machine virtuelle entièrement plutôt que le processus seul. Toutefois, il est nécessaire de mentionner que la tailles des données transférées durant la procédure de migration est trop importante par rapport à la migration des processus. Cependant, les réseaux ont évolué et la bande passante est devenu beaucoup plus large, donc la question du transfert d'une grande taille des données n'est plus d'actualité. Ainsi, même avec cet inconvénient de taille de données transférées, nous pouvons dire que la migration des machines virtuelles est plus efficace et facile à gérer que le migration des processus.

2.3.2 Avantages et inconvénients de la migration dans le Cloud Computing

Plusieurs avantages peuvent être mise en valeur par l'utilisateur de la migration, néanmoins nous pouvons citer quelques inconvénients Clark et al. (2005b) Clark et al. (2005a) Wubin (2012) :

Avantages de la migration dans le Cloud Computing

- L'adaptation automatique à l'évolution de la charges de travail (en hausse, en baisse).
- La tolérance aux pannes.
- Garantir l'élasticité et la mise à l'échelle.
- L'équilibrage de charge.
- La possibilité de faire des tâches de maintenances et d'entretiens sans interruption de services.
- La possibilité d'utiliser la migration dans le cas d'un Disaster Recovery : cela implique la mise en place d'un système similaire dans un site secondaire et la synchronisation, en temps réel ou en différé, entre les machines virtuelles.
- La capacité de faire des mises à jour et des mises à niveau de façon transparente aux clients.

Inconvénients de la migration dans le Cloud Computing

- Problème de réseaux lors de migration.
- Les conflits d'adressage.
- Problèmes de communication entre les VMs après la migration.
- Problème de routage.

2.4 Conclusion

La virtualisation est une technologie clé dans le cloud computing qui a contribué, de près, à son émergence. L'une des composantes principales de cette technologie est les machines virtuelles parce qu'elles contiennent tous les systèmes d'exploitation et les applications nécessaires au déploiement des services pour les clients.

Ce chapitre a présenté les concepts de base qui ont été le soubassement de l'émergence du cloud computing, à savoir la virtualisation et la migration. les principes des ces technologies et techniques ont été décrites et leurs contributions à la réduction de la consommation énergétiques et à l'amélioration de la qualité de services à été montré.

Nous présenterons, dans le chapitre suivant, quelques travaux existants dans littérature qui utilisent la virtualisation et la migration des machines virtuelles comme des techniques de bases dans des strétgies proposées afin d'améliorer la qualité de service et/ou réduire la consommation d'énergie, ainsi que pour réaliser d'autres objectives.

3 Etat de l'art sur les approches de migration des machines virtuelles dans le Cloud Computing

Sommaire

3.1 Introduction

Le Cloud Computing a émergé comme une technologie de pointe pour fournir des services informatiques sur la base du paiement à l'utilisation "pay per usage". La demande croissante du cloud computing a abouti à la création de grands centres de données à large échelle. Pour gérer les requêtes continues des utilisateurs du Cloud et pour fournir une bonne qualité de service, les centres de données sont souvent suréquipés, ce qui mène à une faible utilisation des ressources et à un gaspillage d'une quantité importante d'énergie puisque ces centres de données nécessitent une énorme quantité de puissance électrique pour leurs fonctionnements. L'énergie utilisée dans les opérations de ces centres de données engendre des coûts opérationnels élevés pour les fournisseurs des services Cloud et endommage l'environnement à cause des émissions CO_2 provenant de ces datacenters. Jusqu'à très récemment, les recherches sur les centres de données ont été axés sur l'amélioration des performances des systèmes pour répondre aux attentes des utilisateurs sans prêter attention à la quantité d'énergie consommée et du gaz CO_2 émise par ces infrastructures comme montré dans la Figure 3.1 : De très nombreuses recherches

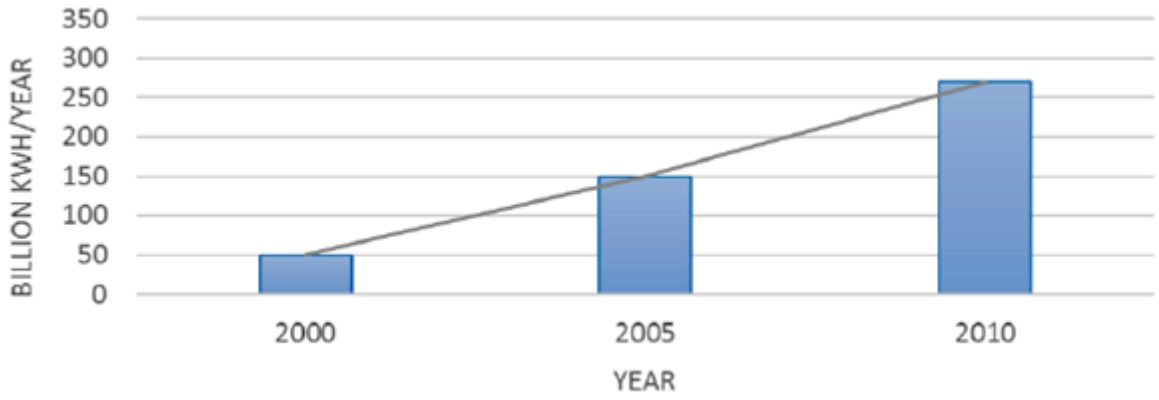


Figure 3.1 – *La consommation d'énergie des centres de données du monde entre 2000 et 2010 Singh et al. (2014)*

ont été menées dans le domaine de la gestion d'énergie et l'efficacité énergétique des ressources dans les systèmes informatiques. Comme les techniques de gestion de puissance électrique et de gestion de l'énergie sont étroitement liées, dans ce chapitre, elles sont appelées les techniques de gestion d'énergie.

Comme montré dans la Figure3.2, les techniques de gestion d'énergie peuvent être divisées en deux catégories : techniques statiques (*SPM*) et techniques dynamiques (*DPM*). Du point de vue matériel, SPM contient toutes les méthodes d'optimisations qui sont appliquées au moment de la conception sur les niveaux des circuits, logiques et architecturals du système Devadas et Malik (1995) Venkatachalam et Franz (2005) Beloglazov (2013). L'optimisation au niveau du circuit se concentre sur la réduction de la puissance de l'activité de commutation de portes logiques individuelles et des circuits combinatoires du transistor par l'application de conception complexe des grilles et le redimensionnement des transistors. L'optimisation au niveau logique est destinés à la puissance de l'activité de commutation de ce niveau et des circuits séquentiels. Les méthodes de niveau architectural comprennent l'analyse de la conception du système et l'intégration subséquente des techniques d'optimisation d'énergie.

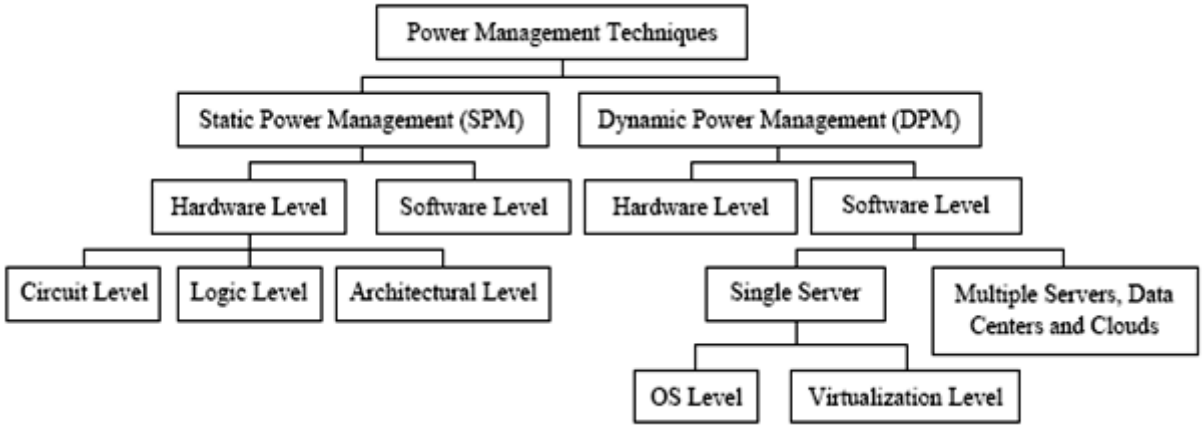


FIGURE 3.2 – *Taxonomie des techniques de gestion de puissance électrique et de l'énergie Beloglazov (2013)*

En plus de l'optimisation de la conception du système au niveau matériel, il est extrêmement important d'examiner attentivement la mise en œuvre des programmes qui doivent être exécutés sur le système car même avec du matériel parfaitement conçu, une conception logiciel de mauvaise qualité peut conduire à des pertes dramatiques des performances et d'énergie. Puisque il est extrêmement difficile de modifier la conception du matériels dans un système qui est déjà opérationnel, il semble judicieux de s'orienter vers la conception logicielle et l'amélioration des techniques dynamique de gestion d'énergie.

Ce chapitre se concentre sur les techniques de DPM qui incluent des méthodes et des stratégies d'adaptation de l'exécution du comportement du système en fonction des besoins actuels en ressources ou toute autre caractéristique dynamique de l'état du système. Une hypothèse importante permettant l'application des techniques DPM est que les systèmes subissent des charges de travail variables pendant leur fonctionnement permettant ainsi le réglage dynamique des états d'énergie selon les exigences de rendement actuelles. Une autre hypothèse souvent faite est que la charge de travail peut être prédite à un certain degré, ce qui permet la déduction des futurs états du système et prendre les mesures appropriées.
La Figure 3.2 montre aussi que les techniques DPM peuvent être distinguées par le niveau auquel elles sont appliquées : matériel ou logiciel. DPM matériel varie selon les différents composants matériels, mais peuvent généralement être classées en des techniques de la mise à l'échelle dynamique des performance "*Dynamic Performance Scaling (DPS)*" comme DVFS (*Dynamic Voltage and Frequency Scaling*) Lee et al. (2012) Pouwelse et al. (2001) et des techniques de désactivation (partielles ou complète) dynamique des composants "*Dynamic Component Deactivation (DCD)*" pendant les périodes d'inactivité. En revanche, les techniques DPM de niveau logiciel utilisent une interface aux capacités du système, en l'occurrence le Cloud computing, telle que la virtualisation, les machines virtuelles et la migration, permettant de proposer des politiques de gestion de ressources en fonction des statistiques récupérées auparavant. Ces politiques visent à satisfaire des exigences des fournisseurs de services Cloud ou/et de leurs utilisateurs.
Nous nous concentrons de manière particulière sur les techniques DPM de niveau logiciel qui cherchent à réduire la consommation d'énergie tout en améliorant la qualité de service.

3.2 Classification des approches de migration des VMs dans le Cloud Computing

Les approches et les stratégies de migration des machines virtuelles peuvent être divisées, principalement, en deux catégories : la catégorie qui cherche à améliorer le processus standard de migration, et l'autre utilise le processus tel qu'il est pour améliorer des performances définies au préalable par les fournisseurs de service et/ou leurs utilisateurs (Dans notre cas, la consommation d'énergie et la qualité de service (QoS)).

3.2.1 Première catégorie : Optimisation du processus de migration

Dans cette catégorie, les stratégies proposées visent à améliorer le processus de migration en apportant des modifications sur l'une ou plusieurs de ces étapes. Parmi ces stratégie, nous pouvons citer :

Selon les auteurs Liu et al. (2009), le processus standard de migration (ou comme ils le désigne "*Memory to Memory*") est très lourd et ne peut être utilisé que dans un réseau LAN à cause de la taille des pages mémoires envoyées durant ce processus. Liu et al. ont montré dans leur étude que l'envoi de ces pages mémoires dans un réseau WAN peut engendrer un temps considérable d'interruption de service. De ce fait, ils proposent une approche de migration à chaud de machines virtuelles basée sur les systèmes "Checkpointing/recovery" and "Trace/Replay", qui sont utilisés habituellement pour la détection des intrusions et des pannes, pour minimiser le temps d'interruption de service et rendre le processus de migration encore plus transparent. L'approche proposés dans ce travail Liu et al. (2009) consiste à enregistrer toutes les informations et les changements d'état d'une VM dans un fichier log. L'enregistrement commence à partir d'un checkpoint qui désigne le moment à partir duquel l'exécution est devenu indéterministe. Après l'envoi de la machine virtuelle, l'hôte source envoie le fichier log, au lieu des pages mémoires, à l'hôte de destination pour mettre à jour l'état de la VM. La Figure 3.3 montre l'ensemble du processus de migration d'une machine virtuelle en cours d'exécution de l'hôte A vers l'hôte B. Le processus de migration est considéré comme une interaction transactionnelle entre les deux hôtes impliquant les phases suivantes :

- **Initialisation :** choisir l'hôte qui a les ressources nécessaires pour héberger la VM à faire migrer.
- **Réservation :** l'hôte source envoie une requête de migration à l'hôte de destination, choisie lors de l'initialisation, pour que ce dernier réserve les ressources nécessaires pour la VM en question.
- **Checkpointing :** l'état du système (mémoire virtuelle, base de registre, etc.), à cet instant, est sauvegardé dans un fichier image.
- **Le transfert itératif des fichiers logs :** après le Checkpointing, l'exécution des événements devient indéterministe, donc l'hôte source envoie périodiquement ses fichiers logs à l'hôte de destination.
- **L'attente et la chasse :** cette étape consiste à suspendre le transfert itératif et à créer un fichier log spécifique appelé V_{thd} contenant le dernier état de la VM de l'hôte source.
- **Stop et copier :** la VM de l'hôte source est suspendue et le fichier log V_{thd} est envoyé à l'hôte de destination.

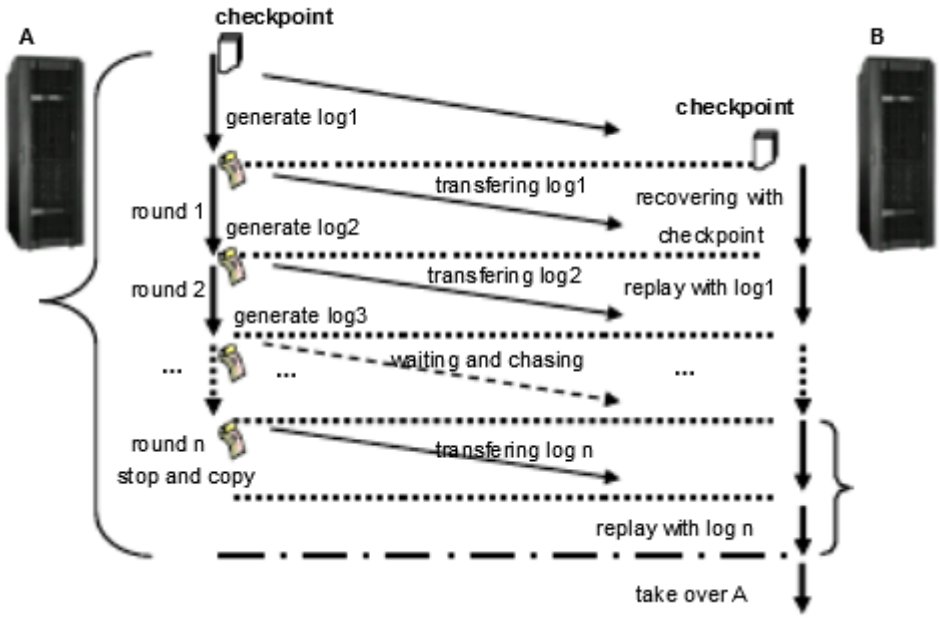


FIGURE 3.3 – *Processus de migration de VMs basé un système "Trace/Replay" Liu et al. (2009)*

- **Activation de la machine virtuelle :** la machine virtuelle migrée est activée et peut continuer son exécution.
- **Mise à jour des informations réseau :** un message de mise à jour est envoyé au switch physique afin d'adapter sa table de routage. De ce fait, les paquets à destination de la machine virtuelle ne passent plus par le même port physique.

Dans Michael et al. (2009), les auteurs considèrent le processus standard de migration comme une approche "précopie" parce que l'état du VM et les pages mémoires modifiées ou *dirty page* sont envoyées après la copie de la VM dans l'hôte de destination (voir Figure 3.4). Ils proposent une approche "postcopie" qui consiste à envoyer, premièrement, les états d'exécution de la VM tels que les états du CPU et par ses pages mémoires. Le principale avantage de cette approche est que les pages mémoires sont transférées, au plus, une fois et ainsi la surcharge de transmission des *dirty page* est évitée. Les auteurs du travail Michael et al. (2009) ont présenté la conception, l'implémentation, et l'évaluation de la migration "postcopie" à chaud des machines virtuelles dans un réseau LAN. Ils ont découpé le processus de migration en trois phase :

- **Le temps de préparation :** c'est le temps entre l'initiation de l'opération de migration et le transfert de l'état du processeur de la machine virtuelle vers l'hôte de destination, durant lequel la VM continue son exécution. Pour l'approche "Pré-copie", ce temps comprend toute la phase de copie itérative, même s'il est négligeable dans cet approche.
- **Le temps d'interruption de service :** c'est le temps durant lequel l'exécution de la VM est interrompue. Au minimum, il inclus le temps de transfert de l'état du processeur et les pages mémoires. Dans l'approche "Pré-copie", ce temps n'inclu que le temps de transfert des pages mémoires modifiées (dirty pages).
- **La reprise :** c'est le temps entre la reprise d'exécution de la VM sur l'hôte de destination et la fin du processus de migration. Durant cette phase toutes les dépendances

sur l'hôte source sont supprimées. Pour l'approche "Pré-copie", cette phase consiste seulement à reprogrammer la VM sur l'hôte de destination et de supprimer celle sur l'hôte source.

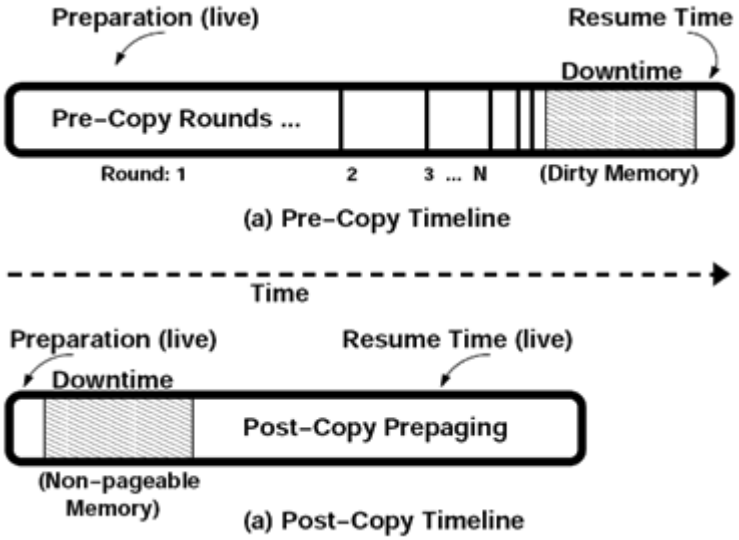


FIGURE 3.4 – *différence entre le temps de migration de la "Pré-copie" approche et la "Post-copie" approche Michael et al. (2009)*

Les auteurs ont montré que leur approche réduit considérablement le temps totale de migration tout en conservant l'état active de la VM pendant la migration. Cependant, cette approche augmente, même un peu, le temps d'interruption de service.

L'approche proposée par Akiyama et al. (2012) est un mécanisme de réutilisation de mémoire qui vise à réduire la quantité de données transférées dans un processus de migration à chaud. Dans le cas d'une consolidation dynamique des VMs, une VM peut être migrée vers un hôte où elle a été déjà hébergée auparavant ; l'image de mémoire d'une VM est sauvegardée dans l'hôte même après qu'elle soit migrée pour que cette image soit réutilisée au cas où cette VM revient à cet hôte. Les faibles quantités de données transférées contribuent à un réduction de temps de migration et une optimisation des algorithmes de placement puisque le nombre potentiel d'hôtes pour une VM est restreint. Les évaluations ont montré que cette approche réduit la quantité de données transférées lors d'une migration à chaud et par conséquent le temps de migration ainsi que la consommation d'énergie d'un système de consolidation dynamique des VMs (voir Figure 3.5).

3.2.2 Deuxième catégorie : Optimisation de la consommation d'énergie et la qualité de service

Les approches de cette catégorie utilisent le processus standard de migration afin de répondre aux besoins des fournisseurs des services Cloud et de leurs utilisateurs. Pour rester dans notre contexte, nous présentons dans ce qui suit les approches et les stratégies qui ont été proposées pour réduire la consommation d'énergie et/ou améliorer la qualité de service (QoS) et répondre au termes d'un contrat SLA :
Li et al. (2009) proposent "*EnaCloud*" qui supporte la planification des applications et la migration à chaud pour minimiser le nombre de machines physiques actifs afin d'économiser l'énergie. Cette approche vise également à réduire le nombre de migrations des ma-

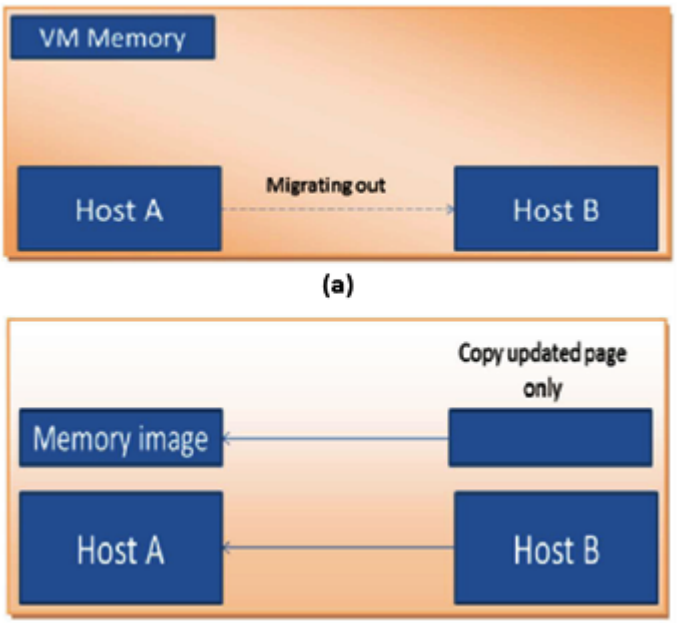


FIGURE 3.5 – *Un cas de consolidation de VM dans l'approche MiyakoDori Akiyama et al. (2012)*

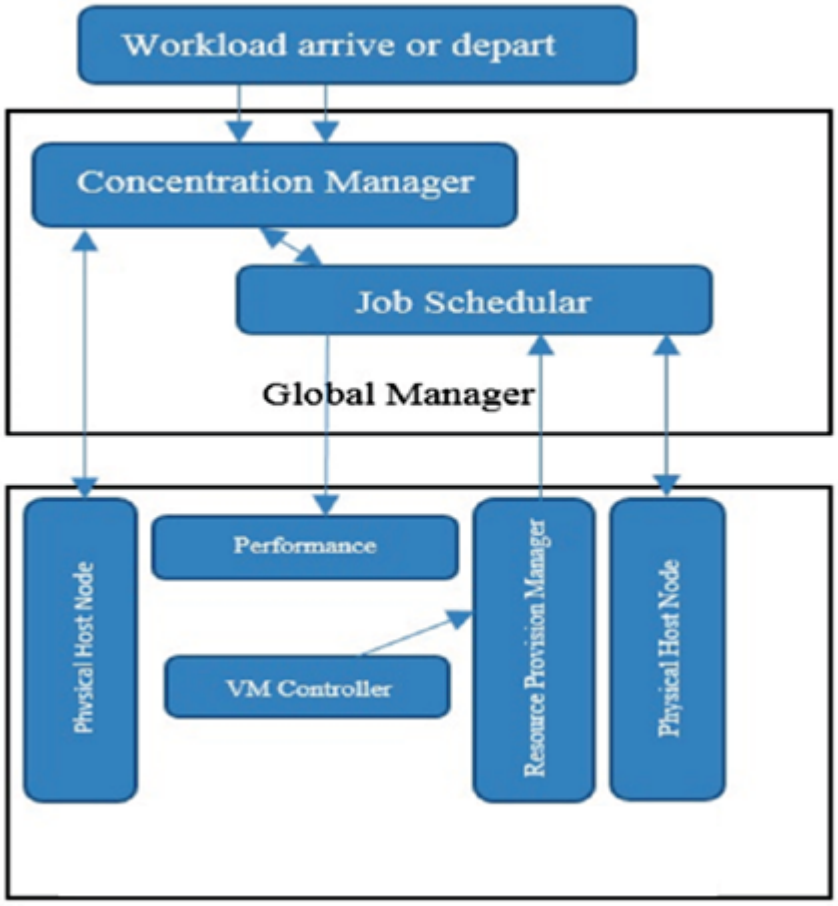


FIGURE 3.6 – *L'architecture d'EnaCloud Li et al. (2009)*

chines virtuelles. Dans EnaCloud, il y a un contrôleur central (*broker*) qui lance l'exécution d'un gestionnaire de charge et d'un planificateur de jobs (voir Figure 3.6). Le planificateur de jobs reçoit la charge de travail, redimensionne les événements et les livre à un gestionnaire, appelé "*Concentration Manager*" qui génère alors, en fonction de ce qu'il a reçu, une série d'insertions et de migrations pour un schéma de placement d'applications qui sont transmises au planificateur de jobs. Le planificateur les envoie, ensuite, au contrôleur de machine virtuelle selon les schémas de placement.
Chaque nœud se compose de contrôleur de machine virtuelle, un gestionnaire d'allocation de ressources et un analyseur des performances :

- Le contrôleur de VM invoque l'hyperviseur pour exécuter des commandes telles que le démarrage, l'arrêt, ou la migration de la VM.
- L'analyseur de performance collecte les données lors du fonctionnement de la VM pour évaluer sa performance.
- Le gestionnaire d'allocation de ressources redimensionne les machines virtuelles sur la base des statistiques de performance collectées par l'analyseur de performance.

Ici, deux types de nœuds sont pris en considération : des nœuds de calcul et des nœuds de stockage. les nœuds de stockages stockent les données et les fichiers alors que les nœuds de calcul sont considérés comme homogènes et hébergent une ou plusieurs machines virtuelles. Le programme s'exécutant dans la VM et le système d'exploitation sous-jacent sont désignés comme la charge de travail. un nœud actif est appelé "*open-box*" et un nœud inactif est appelé "*close-box*". Dans EnaCloud, les charges de travail sont redimensionnées et étroitement regroupées pour réduire le nombre des open-boxs. Ces charges de travail sont distribuées sur les nœuds grâce à la migration qui a, ici, deux principales objectives : La minimisation du nombre des open-boxs et la réduction du temps de migration.

Les auteurs Lee et al. (2012) propose une approche dynamique d'économie d'énergie. Leurs approches comportent trois étapes : la surveillance d'utilisation des CPUs, l'algorithme DVFS Lee et al. (2012) Pouwelse et al. (2001), et la migration à chaud. L'utilisation de processeur de chaque serveur est surveillée dans ce système. Selon la valeur mesurée, un processus approprié sera déterminé pour réduire la consommation d'énergie. Les charges des serveurs sur-utilisés peuvent être migrée vers des serveurs moins chargées tout en éteignant les serveurs inactifs. Les résultats obtenus ont été prometteurs en terme de consommation d'énergie Lee et al. (2012). Cependant, la fréquence de migration et le placement des VMs basé sur les charges de travail causent une dégradation des performances du système. Pour y remédier, les auteurs proposent trois altératives :

- L'optimisation basée sur plusieurs ressources du système comme les CPU, RAM, bande passante.
- L'optimisation du réseau : prendre en considération la communication entre les VMs et leurs sémantiques avant les décisions de replacement pour diminuer le trafic sur le réseau et le transfert des données.
- L'optimisation thermique : la température des nœuds doit être prise en compte avant la réallocation des VMs. Cette initiative vise à éviter les "*hot spots*" en réduisant la charge sur les serveurs sur-utilisés et de diminuer la charge sur les systèmes de refroidissement.

Beloglazov et al. présentent dans Beloglazov et Buyya (2010) un système de gestion des ressources dans une architecture décentralisée des datacenters virtualisés qui réduit

la consommation d'énergie et fournit la qualité de service requise. La stratégie proposée vise à diminuer la consommation d'énergie en consolidant les machines virtuelles selon les topologies de réseau virtuel (*VM network*), le taux actuel d'utilisation des ressources, et l'état thermiques des serveurs. Les auteurs ont présenté trois étapes d'optimisation de placement des VMs :

- La première étape consiste à contrôler l'utilisation des ressources et à la réallouer des VMs pour minimiser le nombre de machines physiques utilisées et ainsi minimiser la consommation d'énergie du système. Toutefois, la consolidation agressive de machines virtuelles peut conduire à une violation des exigences de performance, donc les auteurs ont proposé plusieurs heuristiques pour la sélection des machines virtuelles à faire migrer et ont étudié le compromis entre les performances et l'économie d'énergie. L'idée principale de la politique est de fixer deux seuils d'utilisation du CPU : un seuil supérieur et un autre inférieur, et de garder l'utilisation totale du CPU créée par les machines virtuelles partageant la même machine physique entre ces deux seuils. Si l'utilisation dépasse le seuil supérieur, certaines machines virtuelles doivent être migrées pour réduire les risques d'une violation SLA. Si le seuil inférieur est violé, toutes les VMs présentes sur la machine physique dont le seuil a été violé doivent être migrer ailleurs et cette machine est arrêtée pour économiser de l'énergie.

- Due à la réallocation continue, quelques VMs peuvent être mal placées menant à une charge excessive sur le réseau à cause de la quantité d'informations nécessaires pour les migrations. Par conséquent, il est crucial de considérer le comportement et les communications entre les VMs dans les décisions de réallocation. Le but de cette étape est de placer les VMs d'une manière à minimiser la quantité des données échangées via le réseau.

- Un système de refroidissement d'un centre de données consomme une quantité importante d'énergie, par conséquent, la troisième phase d'optimisation proposée vise à optimiser le fonctionnement du système de refroidissement. Due à la consolidation, des nœuds de calcul subissent des charges élevées conduisant à une surchauffe et donc nécessitent un refroidissement important. Cette étape consiste à surveiller l'état thermique des machines physiques (nœud de calcul) à l'aide des capteurs et de gérer le système de refroidissement en fonction des données récoltées.

Des heuristique ont été proposées pour résoudre le problème de réallocation des VMs. Ces heuristiques ont été évaluées par simulation sur CloudSim. Les expérimentations montrent, qu'avec des seuils adéquats, la stratégie donnent de bons résultats.

Dans Beloglazov et Buyya (2012), les auteurs ont amélioré l'algorithme proposé dans Beloglazov et Buyya (2010) tout en maintenant le principe des deux seuils : le seuil de sous-utilisation et celui de sur-utilisation. À cette fin, les auteurs proposent :

- Un algorithme optimal et déterministe en ligne pour la migration des VMs et les problèmes de consolidations dynamiques.

- Une nouvelle heuristique adaptative pour les problèmes d'efficacités énergétiques de performances.

- Deux métriques pour mesurer la violation du SLA ont été utilisées :

  - **SLATAH ("*SLA violation Time per Active Host*") :** le pourcentage du temps pendant lequel un hôte actif était en sur-utilisation de 100% (utilisation de son CPU était de 100%).
  - **PDM ("*Performance Degradradation due to Migration*") :** la dégradation de la performance globale par VMs à cause des migrations.

- Trois métriques pour la détection des hôtes sur-utilisés :
  - **MAD :** le médian de déviation absolu Huber et al. (1981).
  - **IQR :** l'intervalle interquartile Huber et al. (1981).
  - **LR :** la régression local Cleveland (1979).

Ces travaux ont été évaluées en comparant trois politiques de sélection des VMs : *(1)* la politique du temps minimum de migration (*The Minimum Migration Time (MMT)*), *(2)* la politique du choix aléatoire (*The Random Choice*), et *(3)* la politique de corrélation maximale (*The Maximum Correlation*). les résultats ont montré que l'algorithme proposé combiné avec la politique *MMT* de sélection de VMs surpasse de loin les algorithmes de consolidation dynamique de VMs par rapport aux métriques de violation du SLA et d'énergie en raison de au niveau réduit des violations du SLA et de nombre de migrations des VMs.

Xu et Li (2011) proposent un algorithme de correspondance stable appelé "Egalitarian" pour la migration des machines virtuelles dans le Cloud Computing. Cette approche cherche, aussi, à trouver un compromis entre les VMs et les serveurs en se basant sur la charge de migration des serveurs et le temps de transfert des VMs entre ces serveurs. En cas de polarisation, la correspondance stable "Egalitarian" calcule la somme des rangs des VMs dans les listes de préférences des serveurs pour déterminer les serveurs susceptibles d'héberger ces VMs. Par contre, les auteurs n'ont pas pris en considération le coût de migration ou la consommation d'énergie. Dans notre travail, nous utilisons la correspondance stable pour déterminer le schéma de migration et nous intégrons le théorème de Coase pour réduire le coût de migration, et aussi une extension de correspondance stable Martínez et al. (2004) pour déterminer le meilleur compromis possible entre VMs et PMs sans favoriser un côté au détriment de l'autre.

3.3 Aperçu de l'approche proposée dans cette thèse

Nous proposons une stratégie de migration de machines virtuelles basée sur une superposition de deux théorèmes économique qui vise à réduire la consommation d'énergie sans affecter la qualité de services :

1. une extension du problème du correspondance stable qu'est le problème des admissions au collège Roth (1984) Gale et Shapley (1962) qui permet de trouver un compromis entre les contraintes de prestataires de services (consommation d'énergie des PMs) et les exigences des utilisateurs (qualité de services fournie par les VMs) sans favoriser un côté sur l'autre à l'aide d'une procédure d'acceptation différée. Cependant, les choix affectés, par cette procédure, aux deux côtés (PMs et VMs) peuvent, parfois, être différents et même contradictoires. Ce problème est connu comme le problème de polarisation des correspondances stables Roth (1984) Martínez et al. (2004). Plusieurs travaux ont été proposés Xu et Li (2011) Gusfield et Irving (1989) dans ce contexte mais ils favorisent un côté au détriment de l'autre. Dans ce travail, nous utilisons une autre extension du problème de mariage stable

proposé dans Martínez et al. (2004) qui permet, en cas de polarisation, de trouver un autre compromis entre les *PMs* et *VMs*.

2. le théorème de Coase Coase (1960) nous permet de trouver le nombre optimal de machines virtuelles à faire migrer pour sélectionner la solution entre autres possible (en fonction du nombre de machines virtuelles migrées) qui améliore le mieux la consommation d'énergie et la qualité de service (QoS) et qui réduit, en conséquence, la charge de migration, c'est-à-dire, le nombre de machines virtuelles migrées via le réseau.

3.4 Etudes comparatives

Nous avons essayé de faire des comparaisons entre les approches listées ci-dessus et notre approche. Ces comparaisons sont montrées dans le tableau 3.1.

3.5 Conclusion

Maximiser l'efficacité énergétique sans affecter les performances des systèmes est l'une des tâches les plus difficiles auxquels sont confrontés les fournisseurs de Cloud. Dans ce chapitre, nous avons présenté quelques travaux pertinents menés par la communauté scientifique. Ces travaux cherchent soit à améliorer le processus standard de migration, soit à optimiser la consommation d'énergie et/ou la qualité de service. Nous avons, aussi, abordé les bases de notre approche. Le chapitre suivant sera consacré à la description de l'approche proposée dans ce travail de thèse.

Approche	**Avantages**	**Paramètres et ressources considérées**	**Répercussions**
Liu et al. (2009)	Réduire le temps d'interruption de service, minimiser la quantité d'informations échangées dans le réseau	Les états d'exécution des *VMs*, des fichiers log	Lenteur au lancement de la *VM* sur l'hôte de destination à cause de la réexécution des instructions contenues dans le fichiers log
Michael et al. (2009)	Réduire le temps total de migration, minimiser la quantité d'informations échangées dans le réseau	Les états d'exécution des *VMs*, les états des CPU et les pages mémoires des *VMs*	Augmenter le temps d'interruption de service
Akiyama et al. (2012)	Réduire la quantité des données transférées via le réseau et le temps total de migration	Les états d'exécution des *VMs*, les pages mémoires des *VMs* et les espaces de stockages sur les *PMs*	Consommation de trop d'espaces de stockage à cause de la sauvegarde des images mémoires des *VMs*
Li et al. (2009)	Economiser l'énergie et diminuer le nombre de migrations	Mémoire, stockage et charges de travail	Dégradation de la QoS à cause de la mise hors tension continue des *PMs* (*consolidation agressive*)
Lee et al. (2012)	Réduire la consommation d'énergie en minimisant le nombre de *PMs* actives	CPU, mémoire et charges de travail	Dégradation de la QoS à cause de la haute fréquence de migrations (*consolidation agressive*) et du placement des *VMs* basé sur la charge de travail
Beloglazov et Buyya (2010)	Réduction de la consommation d'énergie, amélioration de la QoS et équilibrage de charge	CPU, réseau et état thermique des *PMs*	Quantité importante de données transférées à cause du nombre élevé de migrations et avec des seuils non-adéquats, le système subit une dégradation considérable de la QoS et de la consommation d'énergie
Beloglazov et Buyya (2012)	Réduction de la consommation d'énergie, amélioration de la QoS, équilibrage de charge et diminution de la charge de migration	CPU, réseau, état thermique des *PMs* et le systèmes de refroidissement	Avec des seuils non-adéquats, le système subit une dégradation considérable de la QoS et de la consommation d'énergie
Xu et Li (2011)	Réduire le temps de transfert des *VMs* et le nombre de sauts entre serveurs effectués par une *VMs* lors d'une migration	réseau, taille des *VMs* et mémoire	Pas de prise en considération de la consommation d'énergie et de la QoS
Kella et Belalem (2014)	Réduction de la consommation d'énergie, amélioration de la QoS, équilibrage de charge et diminution du nombre de *VMs* migrées	CPU, aailles des *VMs*, mémoire et les charges de travail	Avec des nombres de *VMs* et de *PMs* non-adéquats, le système subit une dégradation considérable de la QoS et de la consommation d'énergie

TABLE 3.1 – *Comparaison entre les approches listées ci-dessus et notre approche*

Description et modélisation de la stratégie proposée de migration

4

Sommaire

4.1 Introduction

Nous avons vu dans le chapitre précédent quelques travaux de la littérature qui traitent le problème de la consommation élevée d'énergie dans les datacenters et son impact sur l'amélioration de la qualité de service. Ces travaux traitent la migration des machines virtuelles comme des problèmes d'affectation ou d'allocation, et ils se sont basés sur des heuristiques et des méta-heuristiques. Par contre, dans d'autres travaux, les auteurs ont préféré établir des modèles de coût basés sur des paramètres statistiques en récoltant des données et en observant le comportement des serveurs et des machines virtuelles vis-à-vis des charges de travail.

Le problème de migration de machines virtuelles est traité comme un problème d'affectation où il est question d'attribuer des machines virtuelles à des machines physiques. Pour traiter ce problème, nous avons proposé une stratégie de migration basée sur une superposition de deux célèbres théorème dans l'économie : *le problème de correspondance stable* et *le théorème de Coase* afin de réduire la consommation d'énergie et améliorer la qualité de service dans le Cloud Computing. Donc il est primordial de décrire ces théorèmes avant de présenter notre approche. Par conséquent, nous commençons ce chapitre par une description détaillée du problème de correspondance stable et ses types ainsi que le théorème de Coase. Nous verrons dans cette partie aussi, la manière dont ces deux théorèmes sont projetés sur une stratégie de migration de machines virtuelles dans le Cloud Computing pour alléger les inquiétudes des fournisseurs (consommation d'énergie) et répondre aux attentes des utilisateurs (QoS).

La deuxième partie du chapitre est consacrée à l'explication de notre approche, en décrivant ses différentes phases et comment nous avons intégré la procédure d'acceptation différée du problème de correspondance stable et appliqué le théorème Coase pour une répartition optimale des *VMs* qui garantit le meilleur des compromis entre la réduction de la consommation d'énergie et l'amélioration de la qualité de service.

4.2 Préliminaires et vue d'ensemble

Nous présenterons dans cette section les deux théorèmes sur lesquels se basent notre travail, à savoir "*le problème de correspondance stable*" et "*le théorème de Coase*" :

4.2.1 Problème de correspondance/allocation stable

Historique

L'économie est généralement étroitement liée à l'argent, mais la vie économique s'étend au-delà de ce qui peut être ou est monétisé, comme le montrent les gagnants du prix Nobel de l'économie en 2012, "*Lloyd Shapley*", Professeur à l'Université de Californie à Los Angeles et "*Alvin Roth*", Professeur à l'Université de Stanford, pour la théorie des allocations stables et sa pratique dans la conception des marchés Committee (2012). Cette théorie se distingue du fait qu'elle décrit, principalement le marché sans argent.
Les économistes étudient *comment les sociétés allouent des ressources ?* Certains problèmes d'allocation sont résolus par le système de prix : des salaires élevés attirent les travailleurs dans une profession en particulier, et les prix élevés de l'énergie incite les consommateurs à économiser l'énergie. Cependant, l'utilisation du système de prix se heurte, dans de

nombreux cas, à des obstacles juridiques et éthiques. Considérons par exemple, l'attribution des places dans les écoles publiques pour les enfants, ou l'attribution des organes humains à des patients qui ont besoin de greffes. En outre il y a beaucoup de marchés où le système des prix fonctionne, mais l'hypothèse traditionnelle de la concurrence parfaite n'est même pas approximativement satisfaite. En particulier, de nombreux biens sont indivisibles et hétérogènes, de sorte que le marché pour chaque type de biens devient très mince. Comment ces marchés étroits allouent-ils des ressources alors qu'ils dépendent des instituions qui régissent les transactions ?

L'analyse des mécanismes d'allocation repose sur une idée plutôt abstraite : Si des individus rationnels, qui connaissent leurs intérêts et qui se comportent en conséquence, s'engagent simplement dans des échanges mutuels sans restriction, alors le résultat devrait être efficace. Si ce n'est pas le cas, certains personnes élaboraient de nouveaux échanges qui leurs seraient plus favorables. Une allocation où les individus concernés perçoivent aucun intérêt ou gain à faire d'avantage échanges est appelé "*stable*" Roth (1984) Gale et Shapley (1962). La notion de stabilité est un concept central dans la théorie de jeux coopératifs qui est considérée comme une zone abstraite de l'économie mathématique qui vise à déterminer comment un groupe d'individus rationnels peut choisir une allocation tout en coopérant entre eux. Lloyd Shapley est considéré comme le principal architecte de cette branche de la théorie de jeux en développant ses principaux concepts dans les années 1950 et 1960 Committee (2012).

Les fondations pour le cadre théorique ont été établies en 1962, lorsque David Gale et Shapley Lloyd ont publié un court article sur une classe des problèmes d'allocation Gale et Shapley (1962). Ils ont considéré un modèle de deux ensembles d'agents : des travailleurs et des firmes, qui doivent être associés les un aux autres. Si un travailleur est embauché par l'employeur A, mais ce travailleur aurait préféré l'employeur B, qui aurait également aimé embaucher ce travailleur (mais ne l'as pas fait), donc il y a des gains inexploités de cet échange. Si l'employeur B avait engagé ce travailleur, les deux auraient eu un meilleur arrangement. Gale et Shapley ont proposé une procédure d'acceptation différée qui conduit toujours à une correspondance stable. La procédure montrent comment les agents sur un côté du marché (par exemple employeurs) font des offres à ceux de l'autre côté, qui acceptent ou rejettent ces offres selon certains règles.

La pertinence empirique de cette théorie a été reconnu plus tard par Alvin Roth dans une étude publié en 1984 Roth (1984) Roth et Vate (1990) Roth et Peranson (1999). Roth a constaté que le marché américains des nouveaux médecins résidents a toujours souffert d'une série de défaillance à cause des mauvaises attributions des résidents aux hôpitaux, et qu'une chambre de compensation centralisée ("*centralized clearinghouse*")avait amélioré la situation en mettant en œuvre une procédure essentiellement équivalente à la procédure d'acceptation différée de Gale et Shapley. L'étude de 1984 de Roth Roth (1984) a montré comment le concept de stabilité prévoit un principe organisateur et aide les économistes à comprendre pourquoi les marchés fonctionnent parfois correctement et parfois non.
Par la suite, Roth et ses collègues ont utilisé cette théorie, en combinaison avec des études empiriques, des expériences contrôlées en Laboratoire et des simulations informatiques, pour examiner le fonctionnement des autres marchés Roth et Sotomayor (1989). Leur recherche a non seulement éclairé le fonctionnement de ces marchés, mais est également révélée utile dans la conception des institutions qui aident les marchés à bien fonctionner, souvent en implémentant une version ou une extension de la procédure de Gale and Sha-

pley. Cela a conduit à l'émergence d'une nouvelle et vigoureuse branche de l'économie appelée "*La conception des marchés*".

Le problème de correspondance stable et ses généralisations ont été largement étudiés en optimisation combinatoire et en théorie des jeux Roth (1984). La raison est que ces modèles sont utiles pour décrire des situations économiques et sociales. En outre, comme des application réelles, les programmes d'allocation ont été établis dans plusieurs domaines, nous pouvons citer : CARMS (Canadian Resident matching Service) au Canada carms (2015) Iwama et Miyazaki (2008), JRMP (Japan Residency Matching Program) au Japon jrmp (2015) Iwama et Miyazaki (2008), SPO (Scottish PRHO Allocations) au Scotland SPO (2015) Iwama et Miyazaki (2008). En 2003, Roth a aidé la ville de New York à refaire de son système d'affectation des écoles supérieures et il a réussi à réduire de 90% le nombre d'étudiants qui finissent habituellement affectés à des écoles qui n'étaient même pas sur leurs listes de préférences et selon un article publié dans BBC magazine "*Al Roth : An economist who saves lives*" en 2012 Knight (2012), le système d'échange de Roth basé sur une extension de la procédure d'acceptation différée maximise le nombre de greffes de rein à partir d'un pool donné de paires patient-donneur.

Principe

Considérons le problème de correspondance stable (*SMP : Stable Matching Problem*) comme un graphe G, où les agents sont représentés par des sommets. Deux agents sont reliés par une arête si chacun de ces agents accepte l'autre comme étant son partenaire. Pour chaque sommet s, soit $<_s$ un ordre linéaire des arêtes incidentes à s, et chaque agent à des préférences strictes de ses possibles partenaires. Les préférences strictes des partenaires sont des préférences où il n y a aucune relation entre ces partenaires.

On dit que l'agent s préfère f à e (en d'autres mots f domine e) si $e <_s f$ est vrai. Une correspondance μ est un ensemble d'arêtes et de paires distinctes de sommets. Si une arête $e = \{r, s\}$ appartient à μ alors r et s sont reliés dans μ, et r et s sont des partenaires dans le marché. Un agent est seul, si son sommet est découvert dans μ, c'est à dire qu'il n'y a aucune correspondante arête incidente à ce sommet.

Une correspondance μ est stable si toute arête non appariée, e appartient à μ, est dominée par une arête appariée, f appartient à μ. Alternativement, une correspondance stable peut être définie comme une correspondance sans une bloquante arête. Une arête $e = \{r, s\}$ est bloquante pour une correspondance μ si r est soit non relié ou préfère e à l'arête qui le relie à son partenaire actuel dans μ, et dans le même temps, s est soit non relié ou préfère e à l'arête qui le relie à son partenaire actuel dans μ. La stabilité dans un marché signifie qu'aucune paire d'agents ne peut pas bénéficier d'une rupture avec leurs partenaires actuels et d'un nouveau partenariat mutuel avec d'autres partenaires.

Exemple Committee (2012)
Supposons deux côtés du marché : des étudiant en médecine et des départements médicaux. Chaque département a besoin d'un stagiaire et chaque étudiant en médecine veut un stage. Une correspondance est, donc, une attribution de stages aux candidats. Naturellement les étudiants ont des préférences de départements auxquels ils veulent être affectés. Une correspondance est dite inacceptable pour un agent (département ou étudiant) s'il existe des partenaires encore seuls ou libres qu'il préfère à son partenaire

actuel (département ou étudiant).

En général, une correspondance est stable si aucune coalition ou partenariat ne peut l'améliorer. Dans l'exemple précédent, un matching stable doit satisfaire deux conditions :

1. Aucun agent (étudiant ou département) ne trouve la correspondance inacceptable.
2. Aucune paire département-étudiant ne préfère être dissoute pour que ses gens puissent trouver d'autres partenaires, plutôt que de rester avec leurs partenaires actuels.

Selon Gale et Shapley (1962), la condition *(1.)* est une *rationalité individuel* qui assure que chaque agent soit relié au partenaire qui lui est le plus profitable selon la situation du marché, et la condition *(2.)* est appelée *la stabilité des paires* qui résulte en partie de la première condition et qui signifie que la correspondance actuelle garantit la stabilité du marché. Les deux conditions impliquent qu'aucun nouveau partenariat, ni n'importe quel autre partenaire d'une paire département-étudiant, ne peut améliorer la correspondance actuelle.

4.2.2 Types du problème de correspondance stable

Nous distinguons trois types du problème de correspondance stable : (*i*) le "*one-to-one problème*", (*ii*) le "*many-to-one problème*", et (*iii*) le "*many-to-many problème*". Chacun de ses types a plusieurs instances selon son domaine d'application. Dans notre présent travail, nous projetons une instance du deuxième type, connu sous le nom du "Problèmes des admissions au collège Gale et Shapley (1962)" sur le domaine de migration des machines virtuelles dans les Cloud Computing. Mais pour plus de simplicité et de clarté, nous commençons par présenter la plus basique des instances du problème de correspondance stable qu'est "Le problème de mariage stable Gusfield et Irving (1989)".
Nous présenterons dans ce qui suit chaque type à travers l'une de ses instances :

One-To-One problème (le problème du mariage stable)

Le problème de mariage stable Gusfield et Irving (1989) est un problème connu qui consiste à apparier des hommes et des femmes pour atteindre un certain type de stabilité. Chaque personne fournit un ordre de préférences strict des membres du sexe opposé. L'objectif est de former des couples (homme-femme) afin qu'il n'y a pas deux personnes de sexe opposé qui auraient voulu être mis ensemble plutôt qu'avec leurs partenaires actuels.
Une instance de taille n du problème de mariage stable implique deux ensembles disjoints de taille n, des hommes et des femmes. Chaque personne détient une listes contenant tous les membres de sexe opposé dans un ordre de préférence stricte. Une personne p préfère q à r, où q et r sont du sexe opposé à p, si et seulement si q précède r dans la liste de préférence de p.

Définition 1 (Mariage) : une correspondance μ est une correspondance one-to-one entre des hommes et des femmes. Si un homme h et une femme f sont appariés dans μ, alors h et f sont appelés partenaires dans μ, et on écrit $h = \mu(f)$ et $f = \mu(h)$; où $\mu(f)$ est μ-partenaire de f et $\mu(h)$ est μ-partenaire de h.
Définition 2 (Paire bloquante) : Soit un mariage μ. Une paire (h, f), où h est un homme et f est une femme, est une paire bloquante si h et f ne sont pas des partenaires dans μ, mais f préfère h à son $\mu(f)$ ($h >_f \mu(f)$) et h préfère f à son $\mu(h)$ ($f >_h \mu(h)$).

Définition 3 (Mariage stable) : Un mariage μ est stable s'il ne contient aucune paire bloquante.

Exemple
Considérons un problème de mariage stable de taille 4, où les hommes et les femmes sont étiquetés arbitrairement et séparément 1,...,n, et les listes de préférences des hommes et des femmes sont présenté dans la Figure 4.1 tel que h-PreList(i) est la liste de préférence de l'homme i, et f-PreList(j) est la liste de préférence de la femme j. La correspondance

h-PreList(*1*): {f2, f4, f1, f3}	f-PreList(*1*): {h2, h1, h4, h3}
h-PreList(*2*): {f3, f1, f4, f2}	f-PreList(*2*): {h4, h3, h1, h2}
h-PreList(*3*): {f2, f3, f1, f4}	f-PreList(*3*): {h1, h4, h3, h2}
h-PreList(*4*): {f4, f1, f3, f2}	f-PreList(*4*): {h2, h1, h4, h3}
Les listes de préférences des hommes	*Les listes de préférences des femmes*

FIGURE 4.1 – *Une instance de taille 4 d'un problème de mariage stable*

$(h1, f4), (h2, f3), (h3, f2), (h4, f1)$ est stable. Ce matching est défini comme un ensemble de paires (Homme, Femme) ordonnées et sa stabilité peut être vérifiée en considérant chaque homme à son tour comme étant un membre d'une paire bloquante. L'homme $h1$ peut former une paire bloquante avec la femme $f2$ qu'il préfère à sa partenaire $f4$, mais $f2$ préfère son actuel partenaire $h3$ à $h1$. $h2$ et $h3$ sont appariés à leurs femmes favorites, donc aucun d'eux ne peut être dans une une paire bloquante. Finalement, $h4$ peut former une paire bloquante avec $f4$, mais elle préfère rester avec son partenaire actuel $h1$.
Une autre possible correspondance stable est $(h1, f4), (h2, f1), (h3, f2), (h4, f3)$. La stabilité de ce matching peut être vérifiée de la même façon. D'autre part, la correspondance $(h1, f1), (h2, f2), (h3, f3), (h4, f4)$, par exemple, est instable à cause de la paire bloquante $(h1, f4)$, $h1$ préfère $f4$ à sa partenaire actuelle et réciproquement, $f4$ préfère $h1$ à son partenaire actuel. D'autres correspondances instables peuvent avoir plus d'une paire bloquante ; par exemple, le matching $(h1, f1), (h2, f2), (h3, f4), (h4, f3)$ a six paires bloquantes : $(h1, f2)$, $(h1, f4)$, $(h2, f1)$, $(h2, f4)$, $(h3, f2)$, $(h4, f4)$.

Many-To-One problème (le problème des admissions au collège) Gale et Shapley (1962)

1. **Présentation du probléme**
   Un collège (ou une université) examine les candidatures de n candidats dont il ne peut admettre qu'un quota de q candidats. Après avoir évaluer leurs qualifications, le bureau des admission doit décider quels sont ceux à admettre. La procédure qui consiste à offrir des postes à seulement les q meilleurs candidats n'est pas toujours satisfaisante, car il n'est pas sûre que tous ceux qui ont été admis accepteront réellement d'intégrer ce collège. Par conséquent, pour qu'un collège puisse recevoir q acceptations, il sera obliger à offrir plus de q postes aux demandeurs. Le problème de déterminer qui et combien d'étudiants à admettre peut nécessiter certaine conjecture parce qu'il n'est peut être pas connu si un candidat a postulé dans d'autres collèges ; si cela est connu, il n'est peut être pas connu comment l'étudiant classe les collèges dans lesquels il a postulé ; et même si cela est aussi connu, il ne sera pas connu lequel des autres collèges lui offrira un poste. Le résultat de toute cette incertitude est que les collèges peuvent seulement s'attendre à ce que la nouvelle classe admise sera raisonnablement proche du nombre et de la qualité des étudiants recherchés.

La procédure des admissions habituelle présente des problèmes pour les candidats ainsi que pour les collèges. Un candidat qui est obligé à soumettre une liste des collèges où il a postulé selon ses préférences, peut sentir de dire que ce collège est troisième dans sa liste de préférence ruinerait ses chances d'être admis dans celui-ci. La liste d'attente est l'une des solution qui ont été appliquées, selon laquelle un candidat peut être informé qu'il n'est pas admis, mais peut être admis si un poste se libère. Cependant, cela introduit de nouveaux problèmes. Supposons, par exemple, qu'un candidat est accepté par un collège et placé dans la liste d'attente d'un autre qu'il préfère. Doit-il jouer la garantie en intégrant le premier collège ou prendre le risque d'attendre une réponse favorable du deuxième ? Est-il éthique d'intégrer le premier collège sans informer le deuxième puis se rétracter et choisir le deuxième collège si celui-ci l'accepte plus tard ?
Gale et Shapley Gale et Shapley (1962) ont proposé une procédure (*la procédure d'acceptation différée*) pour éviter toutes ces difficultés. Cette procédure consiste à assigner à chaque collège, précisément, son quota de candidats (étudiants) d'une manière qui satisfait les deux côtés, et qui supprime toutes les incertitudes.

2. **Modélisation du problème**
Les premiers éléments du modèle sont deux ensembles finis et disjoints, $C = C_1, ..., C_n$ et $E = E_1, ..., E_m$, de collèges et d'étudiants, respectivement. Les règles du jeu sont que n'importe quel étudiant et collège peuvent se mettre d'accord que l'étudiant intégrera le collège, un collège peut choisir de conserver l'un de ses postes vacant, et tout étudiant peut rester libre (non-relié à un collège) s'il le souhaite. Chaque collège c a un entier positif q_c appelé son quota qui est le nombre de postes qu'il a à offrir, et une liste de préférence d'ordre stricte $P(c)$ sur l'ensemble $E \cup \{nr\}$. Chaque étudiant e a aussi sa liste de préférence d'ordre stricte $P(e)$ sur l'ensemble $C \cup \{nr\}$ où nr représente des éléments qui resteront non reliés ou qui n'ont pas trouver de partenaires. Le résultat est défini par une correspondance $\mu : C \cup E \rightarrow C \cup E \cup \{nr\}$ tel que $|\mu(e)| = 1$, $\forall e \in E, |\mu(c_i)| = q_i$, $\forall c_i \in C$, et, pour chaque $c \in C$et $e \in E$, $\mu(e) = c$ si et seulement si $e \in \mu(c)$. C'est un résultat qui assigne un sous-ensemble d'étudiants à un sous-ensemble de postes vacants et laisse le reste des étudiants et des postes vacants non-reliés, par exemple, si un collège avec un quota q est attribué à un certain nombre $k < q$ d'étudiants dans une correspondance μ, alors $q - k$ éléments de $\mu(c)$ est égale à nr. Le résultat est, donc, une attribution des étudiants aux collèges, de sorte que chaque étudiant est attribué à, au plus, un collège, et chaque collège est relié à, tout au plus, son quota d'étudiants.

Une correspondance μ est *individuellement rationnelle* si pour chaque étudient e, $\mu(e) >_e nr$, et si pour chaque c et $\alpha \in \mu(c), \alpha >_c nr$. μ est instable si elle n'est pas individuellement rationnelle ou s'il existe un collège c_i et un étudiant e_j tel que : $c_i >_{e_j} \mu(e_j)$ et $e_j >_{c_i} \alpha$ où $\alpha \in \mu(c_i)$. Une correspondance qui n'est pas instable est appelée "*une correspondance stable*", et l'ensemble des correspondances stables générées en fonction des listes de préférences d'ordre stricte $P(c)$ et $P(e)$ sera désigné μP. Un étudiant e et un collège c seront dits compatible l'un à l'autre s'il existe une correspondance $\mu \in \mu P$ tel que $e \in \mu(c)$. Nous remarquons, facilement, que le problème de mariage stable est un cas spécial du problème des admissions au collège qui survient quand chaque collège c_i dans C a un quota q_i=1.

3. **Projection du problème à la migration des machines virtuelles dans les infrastructures Cloud**
Supposons que : (i) la migration des machines virtuelles (VMs) est déclenché par

une dégradation des performances du système ou par un pic de consommation d'énergie, ce qu'est le cas dans notre travail, (*ii*) chaque VM dispose d'un ensemble uniforme de ressources, qu'il peut éventuellement utiliser, en termes de CPU, mémoire et espace disque, (*iii*) nous disposons des techniques de monitoring de la charge du trafic sur les machines physiques (PMs), qui sont généralement fournis par les fournisseurs de services Clark et al. (2005a) Clark et al. (2005b).
Nous étudions la migration des VMs comme un problème des admissions au collège. Le marché est alors composé de deux ensemble disjoints : un ensemble de VMs "VS" de cardinalité "V", et un ensemble de machines physiques PMs "PS" de cardinalité "P". Chaque VM (étudiant) cherche à être migré vers une PM (collège). tandis que chaque PM peut héberger plusieurs VMs. La capacité d'une PM est le nombre maximal de VMs q_{pm} qu'elle peut contenir jusqu'à la limite de provisionnement des ressources par VM. Nous présumons que :

$$\sum_{i=1}^{P} q_{pm} \geq V \tag{4.1}$$

pour que chaque VM peut avoir, au moins, une correspondance possible.
Pour modéliser les intérêts communs et contradictoires des individus d'un marché, le concepts de préférences est utilisé dans la littérature de correspondance stable. Dans le cas des VMs, les préférences peuvent être basées sur un large éventail de considérations pratiques telles que : le nombre de sauts dans le réseau, la taille de l'image de VM stockée, ou la charge des machines physiques. Nous verrons dans la section 4.4.2 comment les listes de préférences sont générées dans le cas étudié.
Soit $PmPre = (PmPre_1, ..., PmPre_P)$ et $VmPre = (VmPre_1, ..., VmPre_V)$, les vecteurs des listes de préférences des machines physiques et des machines virtuelles, respectivement, où les préférences des machines physiques représentent les intérêts des fournisseurs de services, à savoir la réduction de consommation d'énergie, et les préférences des machines virtuelles expriment ceux des utilisateurs des services, à savoir l'amélioration de la qualité de service. On peut définir la stabilité de correspondance de notre modèle des admissions au collège comme suit :
Une correspondance μ est stable s'il n'y a aucune incitation pour n'importe quelle paire de s'écarter de μ. Autrement dit, il n'y a aucune paire $(VM, PM) \in VSXPS$ tel que : (*i*) VM préfère PM à son partenaire dans la correspondance actuelle $\mu(VM)$; et (*ii*) PM préfère héberger VM, éventuellement, au détriment d'une autre machine virtuelle qu'il préfère moins selon $PMPre_{PM}$.
Nous étudierons dans la section 4.2.3 l'algorithme proposé par Gale et Shapley Gale et Shapley (1962) qui permet de trouver la correspondance stable au problème des admissions au collège, et nous nous sommes basés cet algorithme pour trouver la meilleurre allocation possible des VMs à travers les machines physiques.

Many-To-Many problème (le problème des firmes et consultants) Martínez et al. (2004)

Considérons un ensemble de firmes et de consultants où chaque firme souhaite embaucher un ensemble de consultants et chaque consultant souhaite travailler pour un ensemble de firmes. Chacune des firmes a des préférences sur l'ensembles des consultants, et pareillement chacun des consultants a des préférences sur l'ensemble des firmes. Ceci est un "Many-To-Many" problème car chaque élément d'un côté du marché (firmes ou consultants) veut ôtre liés à plusieurs partenaires de l'autre côté. Ainsi, une correspondance, dans ce cas, est une affectation d'un ensemble de consultants à des firmes, et d'un ensemble de firmes à des consultants de sorte qu'une firme f est affectée à un consultant

c si et seulement si c est également affecté à f.

Autrement dit, il y a deux ensembles disjoints d'agents, un ensemble de n firmes F=$f_1, ..., f_n$ et un ensemble de m consultants C=$c_1, ..., c_m$. Chaque firme f a une liste de préférence d'ordre stricte $P(f)$ sur l'ensemble 2^C. Chaque consultant c a aussi sa liste de préférence d'ordre stricte $P(c)$ sur l'ensemble 2^F où chaque élément des listes de préférence peut être un ensemble vide $\emptyset$ ou un sous ensemble des ensembles mères F où C, par exemple une firme f_1 peut préférer embaucher c_1 et c_2 plutôt qu'embaucher c_3 seul et un consultant c_3 peut préférer travailler pour f_2 et f_3 plutôt que travailler pour f_1, f_4, et f_5. Le résultat est défini par une correspondance $\mu : F \cup C \rightarrow 2^F \cup 2^C \cup \{nr\}$ tel que pour chaque $f \in F$ et $c \in C$. $\mu(c) \in c$ si et seulement si $c \in \mu(f)$. Etant donné les relations de préférences $P(f)$ et $P(c)$, si un ensemble de partenaires appartenant à une liste de préférence et est privilégié à l'ensemble vide ($P(f_i) >_{f_i} \emptyset$ ou $P(c_j) >_{c_j} \emptyset$) alors cet ensemble est appelé acceptable. Par conséquent, une firmes a la possibilité de ne pas embaucher des consultants du tout plutôt que d'embaucher un ensemble inacceptable de consultants, et de même, un consultant peut préférer rester au chômage plutôt que travailler pour un ensemble inacceptable de firmes.
Exemple

$$P(f_i) = \{c_1c_4, c_2, \emptyset, c_3\} \tag{4.2}$$

$$P(c_j) = \{f_3f_1, f_2, \emptyset, f_4f_5\} \tag{4.3}$$

Ceci indique que la firme f_i préfère c_1 et c_4 ensemble à c_2 seul, et préfère n'embaucher personne plutôt que d'embaucher c_3. Le consultant c_j préfère être embauché par f_3 et f_1 plutôt que d'être embauché par f_2 et préfère rester au chômage plutôt que travailler pour f_4 et f_5.
Dans cet exemple, les ensembles acceptables pour f_i et c_j sont respectivement $\{c_1, c_4\}$, $\{c_2\}$, et $\{f_3, f_1\}$, $\{f_2\}$.

Nous constatons que contrairement au "Many-To-One" problème, les agents du "Many-To-Many" problème n'ont pas de quota qui détermine, au préalable, le nombre de partenaires qu'ils préfère avoir. Le nombre et la qualité de partenaires désirée sont tous les deux intégrés aux listes des préférences des agents.

4.2.3 Procédure d'acceptation différée

En 1962, David Gale et Lloyd Shapley, deux mathématiciens économistes, ont conçu un élégant algorithme pour résoudre le problème du mariage stable et l'ont appelé "*la procédure d'acceptation différée*" ou "*l'algorithme Gale-Shapley*" Gusfield et Irving (1989) Gale et Shapley (1962) :

Algorithme 1 L'algorithme Gale-Shapley

Initialisation : tous $h \in H$ et tous $f \in F$ sont célibataires
Tant que ¬Un homme est célibataire et n'a pas demandé toutes les femmes **Faire**
  Choisir un tel homme h
  $f \leftarrow$ est la 1^{re} femme sur la liste de préférence de h que h ne lui a pas encore fait sa demande
  Si $\neg f$ est célibataire **Alors**
    Apparier h et f
  Sinon Si $\neg f$ préfère h à son partenaire actuel h' **Alors**
    Apparier h et f
    Rejeter h'
  Sinon
    Rejeter h
  Fin Si
Fin Tant que
Résultat : un ensemble de paires (*homme* , $femme$) stables

Un aspect crucial de cet algorithme est que les propositions ne sont pas immédiatement acceptées, mais tenues à l'acceptation en différée. Tout homme dont la proposition a été rejetée peut faire une proposition à une autre femme. La procédure continue jusqu'à ce que aucun homme ne souhaite faire d'avantage de propositions, et c'est à ce moment là que toutes les femmes acceptent les propositions qu'elles détiennent déjà.

Analyse de l'algorithme

Une analyse détaillée de l'algorithme *Gale-Shapley* a été fournie dans Northeastern University (2015) qui démontre que l'algorithme se termine rapidement et trouve toujours une correspondance stable :

Théorème 1. *L'algorithme se termine après, au plus, n^2 itérations.*

Preuve. *Logiquement, l'algorithme fait des progrès continuellement. Dans chaque itération, un homme fais sa proposition à la prochaine femme dans sa liste de préférence et à qui il n'a jamais fais de proposition. Et puisque, il y a n hommes, chaque liste préférence et de taille n, donc automatiquement il n'y aura pas plus de n^2 propositions. Ainsi le nombre max d'itérations possible est : n^2.*

Lemme 1. *Chaque femme f restera engagé dès qu'elle reçoit la première proposition. À partir de ce point, le partenaire à qui elle va être engagé, ne pourra être que meilleur que qui le précède.*

Preuve. *Une fois qu'une femme f reçoit une proposition, elle est prise, et elle va continuer à l'être. Si l'algorithme annule l'engagement actuel de f, c'est sûrement pour l'apparier avec un autre homme. En plus de ça, si la paire (h, f) est remplacée par $(\acute{h}, f)$, c'est seulement si f préfère $\acute{h}$ à h, donc son nouveau choix est toujours meilleur que celui du précédent.*

Lemme 2. *Si un homme h est célibataire à un moment donné de l'exécution de l'algorithme, c'est sûrement qu'il y a des femmes à qui il n'a pas encore fait sa proposition.*

Preuve. *La preuve est par contradiction. Supposons qu'à un moment donné, il y a un homme h célibataire mais a fait sa proposition à toutes les femmes dans sa liste de préférence. Cela signifie qu'à ce moment là, chaque femme a reçu au moins une demande de la part d'un homme, et d'après*

le lemme 1, nous obtenons que chaque femme est engagée. Ainsi, nous avons n femmes engagées et donc n hommes engagés, ce qui implique que h est également engagé ce, qui contredit l'hypothèse de départ disant que h est célibataire.

Théorème 2. *À la fin de l'exécution, l'algorithme Gale-Shapley renvoie toujours une correspondance stable.*

Preuve. *Premièrement, une preuve que l'algorithme renvoie une correspondance parfaite. La preuve est par contradiction. Supposons que l'algorithme retourne un matching imparfait, ce qui implique qu'il y aurait un homme h célibataire à la fin de l'exécution de l'algorithme. Selon le lemme 2, ça signifie que h n'a pas encore envoyé de demande à certaines femmes. Mais dans ce cas, l'algorithme ne sortira pas de la boucle "**tant que**" et l'algorithme ne finira jamais, donnant la contradiction souhaitée.*
Deuxièmement, une preuve que la correspondance générée est stable. Encore une fois, la preuve est par contradiction. Supposons qu'il y a deux hommes h et $\acute{h}$ et deux femmes f et $\acute{f}$ tel que $(h, f) \in \mu$ et $(\acute{h}, \acute{f}) \in \mu$, mais h préfère $\acute{f}$ à f et $\acute{h}$ préfère f à $\acute{f}$. Selon l'algorithme Gale-Shapley, f est la dernière femme à qui h a fait sa proposition. Puisque h préfère $\acute{f}$ à f, h aurait déjà envoyé une proposition à $\acute{f}$ avant de l'envoyer à f. À ce moment, ou plus tard, $\acute{f}$ était engagée à un homme, disons $\acute{h}$ qui elle préfère plus que h. À la fin, $\acute{f}$ est engagée à $\acute{h}$. Selon le lemme 1, on trouve que $\acute{f}$ préfère $\acute{h}$ à $\acute{h}$ et préfère $\acute{h}$ à h, ce qui implique que $\acute{f}$ préfère $\acute{h}$ à h contredisant la supposition initial ($\acute{f}$ préfère h à $\acute{h}$).

Gale et Shapley (1962) ont observé le comportement du l'algorithme proposé pour le problème du mariage stable et l'ont modifié, d'une manière à ce que les agents d'un côté du marché peuvent accepter plusieurs partenaires pour l'appliquer au problème des admissions au collège où les collèges représentent les hommes et les étudiants désignent les femmes. Cette extension de la procédure d'acceptation différée, comme montré ci-dessous, comprend une séquences des opérations de *candidature* et de *rejet*. À chaque itération, un étudiant e_i non-admis postule dans le collège c_j, le premier dans sa liste de préférence et à qui il n'a pas encore envoyé de candidature, et il sera provisoirement assigné à ce collège (Sachant que cette assignation n'est pas définitive). Si le collège c_j a reçu, avec cette dernière candidature, sa $(q_j + 1)^{eme}$ candidature alors il sera obligé de rejeter le pire des étudiants qui lui sont assignés actuellement e_k. En plus , si c_j atteint son quota q_j alors chaque étudiant e_i moins préféré que le pire des étudiants assignés actuellement à c_j, e_k, sera supprimé de la liste de préférence de c_j et vice versa, ainsi que toutes les paires (e_i, c_j) :

Algorithme 2 L'algorithme Gale-Shapley pour le problèmes des admissions au collège

Initialisation :
$\mu \leftarrow \emptyset$;
Tous $e_i \in E$ est non-admis et tous $c_j \in C$ n'a pas encore reçu de candidature ;
Tant que $\neg\exists\ e_i$ non admis **Faire**
  $c_j \leftarrow$ le premier collège dans e-PreList(i) ;
  $\mu \leftarrow \mu \cup \{(e_i, c_j)\}$;
  Si $\neg c_j$ a reçu plus de demandes que son quota q_j **Alors**
    $e_k \leftarrow$ le pire étudiant dans $\mu(c_j)$ selon c-PreList(j) ;
    $\mu \leftarrow \mu \setminus \{(e_k, c_j)\}$;
  Fin Si
  Si $\neg c_j$ est plein **Alors**
    $e_k \leftarrow$ le pire étudiant dans $\mu(c_j)$ selon c-PreList(j) ;
    Pour ($\forall e_j \in$ c-PreList(j) et $e_k >_{c_j} e_i$) **Faire**
      Supprimer e_i de c-PreList(j) ;
      Supprimer c_j de e-PreList(i) ;
    Fin Pour
  Fin Si
Fin Tant que

Nous utiliserons cet algorithme comme base pour établir notre procédure d'acceptation différée afin de déterminer, en cas de besoin, vers quelles machines physiques déplacer les machines virtuelles concernées par une migration.

4.2.4 Polarisation des correspondances stables

L'algorithme Gale-Shapley peut être exécuté de deux manières différentes : soit c'est les hommes qui font des propositions aux femmes, soit le contraire, c'est les femmes qui font des propositions aux hommes. Dans le deuxième cas, le processus commence avec chaque femme proposant à son meilleur choix parmi les hommes d'être son partenaire. Chaque homme se penche ensuite sur les différentes propositions qu'il a reçu (le cas échéant), et conserve ce qu'il considère comme la proposition la plus intéressante (mais il ne l'accepte pas définitivement) et rejette toutes les autres. Les femmes, qui ont été rejetées dans le premier tour, proposent à leurs deuxièmes meilleurs choix, tandis que les hommes gardent leurs nouvelles meilleurs offres et rejettent les autres, et cela continue jusqu'à ce que les femmes ne veulent plus faire de propositions en obtenant leurs meilleurs partenaires possibles.

La manière d'exécution de l'algorithme s'est avérée avoir d'importantes conséquences sur la répartition des partenaires ; il est très important de savoir le côté du marché qui va initier l'algorithme (les hommes ou les femmes). Si les femmes font les propositions, le résultat, que nous appellons "*f-optimal*", sera mieux pour elles que si les hommes font les propositions, puisque certaines femmes se retrouvent avec des hommes qu'elles préfèrent à leurs partenaires dans le cas où c'est les hommes qui font des propositions. En effet, la correspondance résultante est la meilleure correspondance possible pour les femmes, et inversement, l'algorithme inverse -où les hommes proposent- conduira au pire résultat, que nous appellons "*h-optimal*", du point de vue de certaines femmes. Cette contradiction des correspondance stables est appelée "*la polarisation des correspondance stable*".

Nous retrouvons la polarisation de correspondance stable, aussi, dans le problème

des admissions au collège et le problème des firmes et consultants. Dans le problème des admissions au collège, il est connu que c'est les étudiants qui postule pour des postes au sein des collèges, donc c'est eux qui font des propositions. Ainsi, le résultats obtenus, appelé "*e-optimal*", sera mieux pour eux que si c'est les collèges qui font des offres aux étudiants (dont le résultat est "*c-optimal*").

Selon McVitie et Wilson (1971) Iwama et Miyazaki (2008), *h-optimal* est souvent la pire correspondance pour les femmes et vice versa, *f-optimal* est souvent la pire correspondance pour les hommes. Par conséquent, il est naturel d'essayer de chercher une correspondance qui non seulement stable mais aussi "*bonne*" pour les deux côtés.
Plusieurs propositions ont été faite pour traiter le problème de polarisation des correspondances stables Martínez et al. (2004) Gusfield et Irving (1989) Halldòrsson et al. (2003) Iwama et al. (2007). Nous ne présenterons ici que la solution proposée dans Martínez et al. (2004) dont nous avons utilisé le principe pour traiter le problème de polarisation des correspondances stables dans le cas de la migration des machines virtuelles dans le Cloud Computing.

Martínez et al. (2004) ont présenté un algorithme qui permet traiter la polarisation des correspondances stables dans le cas du problème firmes-consultants. Cet algorithme consiste à trouver toutes les correspondances stables possibles en éliminant, successivement, des sous ensembles non-désirés des listes de préférences et l'algorithme s'arrête lorsque toutes les firmes engagent les consultants qui veulent travailler pour elles, autrement dit, *firme-optimal = consultant-optimal*.
En gros, l'algorithme proposé dans Martínez et al. (2004) fonctionne en appliquant successivement la procédure suivante : premièrement, étant donné en entrée un profil original des préférences substituables, il calcule par l'algorithme d'acceptation différée les deux correspondances stable *firme-optimal* et *consultant-optimal*. Deuxièmement, il identifie toutes les paires firme-consultant (f, c) où la firme f embauche le consultant c dans $firme - optimal$ mais pas dans $consultant - optimal$ et successivement, pour chacun de ces paires (f, c), il modifie les préférences de la firme f en éliminant tous les sous ensembles contenant le consultant c mais tous les autres sous ensembles inchangés. Ils ont appelé cette élimination des sous ensemble "*la troncature de préférence original de (f, c)*". En utilisant la procédure d'acceptation différée, il calcule, à nouveau, la correspondance stable $firme - optimal$ en utilisant les nouvelles listes de préférences tronquées. Troisièmement, bien que cette nouvelle correspondance $firme - optimal$ pourrait ne pas être stable par rapport au profil original des préférences, il est stable à condition que si le consultant c avait à choisir le meilleur sous-ensemble de l'ensemble constitué de l'union des deux correspondances $firme - optimal$ (l'original et le nouveau), il choisirait le nouveau. Si la nouvelle correspondance est choisie (et donc, il est stable par rapport au profil original), il garde la nouvelle $firme - optimal$ et les nouvelles préférences tronquées, et il procède à nouveau dès le début en utilisant le profil modifié comme entrée. L'algorithme s'arrête quand il n'y a aucune paire (f, c) où la firme f embauche un consultant c dans la correspondance stable $firme - optimal$ (par rapport au profil des préférences tronquées) mais pas dans $consultant - optimal$.

Dans le présent travail, nous utilisons cet algorithme sur le modèle "problème des admissions au collège" pour traiter les possibles polarisations des correspondances stables dans le choix des machines physiques susceptibles d'héberger les machines virtuelles migrées.

4.2.5 Théorème de Coase

Principe du théorème

Le théorème de Coase a évolué à partir d'un argument illustratif de Ronald Coase qu'est "*le problème du coût social*" Coase (1960) jusqu'en devenir une pièce maîtresse dans la loi et l'économie moderne. Ronald Coase a inventé un nouveau concept "*les coûts du transaction*" pour préparer le terrain pour sa vision. Ce concept lui a permet de gagner un prix Nobel de l'économie en 1991 pour sa découverte et la clarification de l'importance des coûts de transaction et des droits de propriété pour la structure institutionnelle et le fonctionnement de l'économie Nobelprize.org (2015). Ces coûts de transaction peuvent consister, de manière générale, en le temps, l'argent, et l'effort que quelqu'un perd pour réaliser ce qu'il veut Schlafly (2007) ou, d'une manière plus précise, à l'ensemble des coûts nés de la coordination des agents économiques (les deux parties d'une transaction) tels que des effets externes (ou comme les économistes les appellent "*externalités*") d'une transaction, ceux liés à l'incertitude des agents (et donc le coût de l'accès à l'information mais aussi des coûts de la transaction proprement dits, comme les coûts de négociation, de rédaction, etc.).
Ce sont cependant les externalités, les effets négatifs externes, de l'action d'un agent économique qui posent le plus de difficultés. Ces effets négatifs externes sont souvent illustrés par l'exemple d'une pollution : *je produis des marchandises en brûlant du pétrole, je dégage de la fumée qui est susceptible de nuire à mes voisins*. Tel est l'exemple type de ces effets négatifs externes. Une fois les coûts d'une transaction ont été découverts et décrits, la vision de Coase devint possible. Pour Coase, l'externalité résulte au fond d'un conflit au sujet de l'usage d'une ressource rare (il faut voir le problème d'une façon symétrique) et reflète une définition incomplète des droits de propriété. Il démontre dans son théorème, que quand il y'a des différends sur des usages concurrents des propriétés d'une ressource et que les droits de propriété sont suffisamment spécifiés, la négociation entre les parties concernées conduira à un résultat efficace indépendamment de la partie dont les droits de propriété lui sont attribués tant que les coûts des transactions liés à la négociation sont négligeables et la personne qui finira par utiliser la ressource rare sera toujours celle qui en fait l'usage le plus profitable.

Pour l'utilité du théorème de Coase du point de vue conceptuel, voici ce que Mackaay et Rousseau en disent dans leur livre Mackaay et Rousseau (2008) :"*Le théorème est un point de départ dans un sens conceptuel également. Il met en évidence l'importance de bien définir les droits pour faciliter la résolution de différends sur des usages concurrents qui ne manqueront pas de surgir à mesure que les découvertes et le mouvement des personnes modifient la rareté des choses de notre monde. L'analyse révèle l'effet incitatif de la propriété. Le théorème ouvre aussi la porte du droit de la responsabilité, en soulignant son rôle dans la recherche de la solution la moins onéreuse des utilisations concurrentes. Le droit des obligations contractuelles trouve également sa place dans l'univers coaséen, en ce qu'on lui impute une mission de réduire les coûts de transaction, de manière à faciliter l'évolution des ressources vers leurs usages les plus productifs*".

Exemple 1 : Watkins (2015)
Pour illustrer le théorème de Coase, nous allons utiliser un simple exemple. Considérant un chemin de fer, sur le quel passe des locomotives de charbon à brûlure de vapeur, qui traverse secteur cultivateur dans la saison de récolte. Les dommages de récolte de chaque passage de train sont $200. Supposez que le coût de faire passer des trains sur une ligne à côté d'un secteur cultivateur est comme suit :
Si les revenus d'un passage du train est de $350 et si aucune compensation n'est exigée

Nombre de trains par jour	Coût privé	Coût des dommages de récoltes	Coût social
1	$100	$200	$300
2	$200	$400	$600
3	$400	$600	$1000
4	$700	$800	$1500
5	$1100	$1000	$2100
6	$1600	$1200	$2800

pour les dommages de récolte (ou *externalités*), combien de passages par jour permettra de maximiser les gains ?
Cette question peut être répondue en comparant les revenus aux coûts privés et en trouvant le nombre de courses qui maximise les bénéfices :
Comme peut être vu dans le tableau ci-dessus, le bénéfice maximal est réalisé par quatre

Nombre de trains par jour	Revenu	Coût privé	Bénéfice
1	$350	$100	$250
2	$700	$200	$500
3	$1050	$400	$640
4	$1400	$700	$700
5	$1750	$1100	$650
6	$2100	$1600	$500

passages de train. Cependant, dans ce cas les coûts des dommages de récolte sont négligés. En ajoutant les coûts des dommages de récolte aux coûts privés de la société de chemin de fer, ses bénéfices seront différents :
Si on emploie les termes de Ronald Coase, on peut dire que deux trains par jour est

Nombre de trains par jour	Revenu	Coût privé + coût des dommages de récolte	Bénéfice
1	$350	$300	$50
2	$700	$600	$100
3	$1050	$1000	$50
4	$1400	$1500	-$100
5	$1750	$2100	-$350
6	$2100	$2800	-$700

le nombre socialement optimal de passages de trains, mais quatre passages semble être ce qui se produirait en l'absence d'une responsabilité légale au sujet des dommages de récolte. Mais dans ce cas, la société de chemin de fer sera la seule à endosser la responsabilité alors que la ressource rare ici (la récolte) est partagée entre deux entités (la société de chemin de fer et les fermiers). Coase a suggéré que les fermiers pourrait payer la société de chemin de fer pour réduire le nombre de passages de trains. Supposons que les fermiers payeront $1200 pour qu'aucun train ne traverse le secteur cultivateur et ils déduiront $200 de ce paiement à chaque passage de train. la rentabilité de la société serait comme suit :
La société de chemin de fer réalise un bénéfice maximum de $1300 avec deux passages

Nombre de trains par jour	Revenu	Coût privé	Paiement de fermiers	Bénéfice
1	$0	$0	$1200	$1200
1	$350	$100	$1000	$1250
2	$700	$200	$800	$1300
3	$1050	$400	$600	$1250
4	$1400	$700	-$400	$1100
5	$1750	$1100	-$200	$850
6	$2100	$1600	-$0	$500

de trains par jour, qui est le nombre *socialement optimal* de passages de trains. ça c'est l'essence du théorème de Coase : les mêmes niveaux de production (ou de bénéfices) sont réalisés si le causeur des externalités négatives est légalement responsable des coûts de ces externalités et que ceux touchés par ces externalités effectuent un paiement au causeur pour réduire la quantité des externalités négatives et vice versa, si les fermiers et la société échangent leurs positions, l'externalités négative seraient "*les revenus de la société*" et donc c'est la société de chemin de fer qui effectuerait un paiement aux fermiers pour chaque nombre de passages de trains par jour à condition que les fermiers endossent la responsabilité légales de l'externalité négative.

Théorème de Coase & l'approche proposée

Comme nous l'avons introduit dans le chapitre précédent, notre approche est une superposition de principalement deux modèles économiques : "*le problème de correspondance stable*" et "*le théorème de Coase*". Le premier a été expliqué en détails dans les sections précédentes. Ce modèle est déclenché suite à une dégradation des performances ou à un pic de consommation d'énergie et génère une correspondance stable entre les VMs et PMs sous forme de schéma d'affectation. Mais est-il approprié de migrer toutes les VMs sélectionnées ? Sera-t il profitable à la consommation d'énergie globale ou à la qualité de service de migrer toutes ces VMs même si ça semble être la meilleure solution sur le coup ? N'est-il pas exagéré de migrer toutes les VMs de certaines PMs (serveurs) ou du datacenter, alors qu'ils ne sont même pas défaillants, vers d'autres emplacements ? ça ne mènera t'il pas à une dégradation de l'état et des performances des autres PMs et des datacenters ? La quantité de données migrée n'engendrera-t il de congestion dans le réseau et par conséquent, ralentir le trafic réseau ? Prenons un exemple en se basant sur le cas de Google, qui selon certains rapports dispose de plus d'un million de serveurs (PMs) dispersés sur 36 datacenters, et donc on se retrouve avec une moyenne de 27 500 serveurs par datacenters et on assume que chaque serveur héberge actuellement 20 VMs. Donc, si l'un des datacenters subit des dégradations considérables de ses performances ou des pics de consommation d'énergie, on sera obligé de migrer toutes les VMs qui se trouve dans ce datacenter vers les autres datacenters. On aura 550 000 VMs à migrer via le réseau WAN. Par conséquent, il est préférable de réduire la charge de migration (ce qui revient le nombre VMs à migrer) afin d'alléger le processus de migration et garder un état sain du système.

Tous ces points soulevés nous ont conduit à considérer le premier schéma d'affectation ou la correspondance stable comme un résultat intermédiaire et non final, et à chercher une méthode permettant de réduire le nombre de machines virtuelles à faire migrer mais

sans affecter la consommation d'énergie et la qualité de service. Dans ce contexte, nous avons utilisé le théorème de Coase comme modèle où les *PMs* et les *VMs* sont les deux agents économiques, l'externalité négative est la charge de migration, et la propriété ou la ressource rare partagée est définie selon le paramètre déclencheur du processus de migration :

- *Une dégradation des performances (QoS) :*
  - **La propriété :** la qualité de service (QoS) dont les *VMs* ont la responsabilité légale.
  - **Le paiement :** les *PMs* favorisent la migration et l'hébergement des *VMs* qui ont les pire QoS parmi les *VMs* sélectionnées pour améliorer leurs qualités de services.
  - **Le coû privé :** une infime hausse dans la consommation d'énergie.
- *Un pic de consommation d'énergie :*
  - **La propriété :** la consommation d'énergie dont les *PMs* ont la responsabilité légale.
  - **Le paiement :** les *VMs* favorisent les *PMs* qui consomment le moins d'énergie parmi les emplacement potentiels afin d'éviter des futures pics de consommation d'énergie même si ça peut affecter la qualité de service.
  - **Le coû privé :** une possible dégradation de performance.

Dans ce travail, nous avons exploité le principe du théorème de Coase, non seulement pour déterminer le nombre de VMs à migrer mais aussi de les désigner (Voir section 4.4.4), autrement dit, nous avons réussi à déterminer explicitement les machines virtuelles à faire migrer tout en trouvant un compromis entre la consommation d'énergie et la qualité de service, et en réduisant la charge de migration.

4.3 Architecture du Cloud Computing pour la stratégie de migration proposée

Notre approche est basée sur le modèle de Cloud Infrastructure as a Service (IaaS). En effet, nous avons utilisé dans notre travail une architecture deux tiers avec d'un côté le fournisseur du Cloud distribué et de l'autre les clients ou les utilisateurs des services Cloud. Ces derniers ont accès au Cloud en demandant des ressources au fournisseur. Le service proposé par le fournisseur de Cloud dans notre approche est fourni par des machines virtuelles stockées dans les machines physiques qui constituent les datacenters du Cloud. Le but de ce travail est d'aider le fournisseur à réduire sa consommation d'énergie tout en garantissant une excellente qualité de service à ses utilisateurs. L'originalité de cette approche est de proposer un algorithme de migration des machines virtuelles basé sur une superposition de deux célèbres théorèmes économiques : (*i*) le problème de correspondance stable et (*ii*) le théorème de Coase, ceci dans le but trouver un compromis entre les contraintes énergétiques des fournisseurs de services Cloud et les exigences des utilisateurs de ces services. Les objectifs traités par notre approche sont la réduction de la consommation d'énergie tout en veillant à garantir une qualité de service (QoS) satisfaisante .

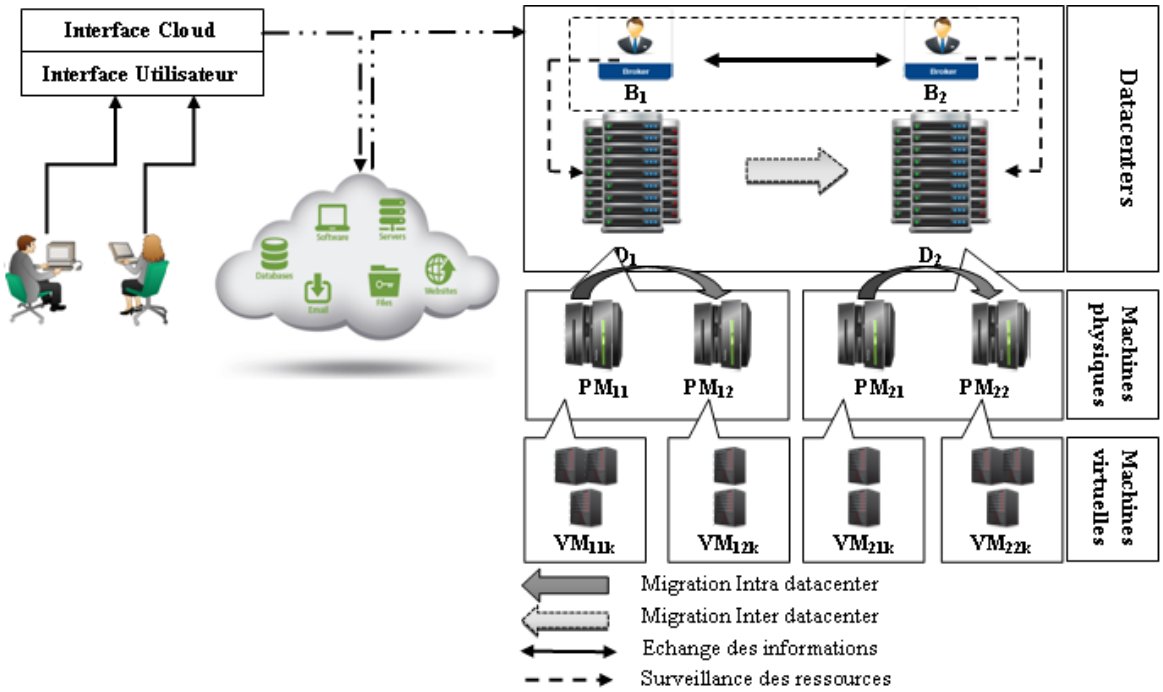


FIGURE 4.2 – *Architecture du cloud computing pour notre stratégie de migration des VMs*

Notre problème met en relation deux tierces parties. La première partie est le fournisseur de Cloud. Ce dernier possède un ensemble $D = \{D_1, D_2, ..., D_n\}$ de n datacenters géographiquement répartis sur différentes zones dans le monde. Chaque datacenter D_i contient un ensemble $PM_i = \{PM_{i1}, PM_{i2}, ..., PM_{im}\}$ de machines physiques et est géré par un Broker B_i. Chaque machine physique PM_{il} est caractérisée par la performance de son CPU "$CPU(PM_{il})$" représentée en $MIPS$ et un espace de stockage libre $CapPM_{il}$ en octet. La seconde partie est représentée par les clients avec leurs requêtes qui doivent être exécutées sur l'un des datacenters de l'ensemble D. Le Broker B_i du datacenter D_i qui reçoit une requête d'un utilisateur choisira la machine virtuelle VM_{ijk} de l'ensemble $VM_{ij} = \{VM_{ij1}, VM_{ij2}, ..., VM_{ijp}\}$ qui va exécuter cette requ^te, et qui est stockée dans la machine physique PM_{ij}. Chaque VM_{ijk} est représentée par $CPU(VM_{ijk})$, la performance de son CPU, en MIPS et sa taille Svm_{ijk} en octet.
Le Broker de chaque datacenter vérifie périodiquement l'état intérieur et extérieur du datacenter pour décider, ensuite, de la nécessité du type migration des machines virtuelles (voir Figure 4.2) :

- **Migration intra datacenter :** si l'état d'une ou plusieurs PMs du datacenter est critique.
- **Migration inter datacenter :** si l'état global du datacenter est critique. Ce type de migration est possible grâce aux informations échangées entre les Brokers.

4.4 Stratégie proposée de migration des machines virtuelles

Pour mieux disséquer la stratégie proposée, nous la présenterons comme une suite de quatre étapes, Chaque étape répond à une question pertinente :

- **Le moment de migration :** qui répond à la question "*Quand migrer ?*" pour déterminer le moment adéquat pour lancer le processus de migration.

- **Le placement des VMs :** chercher les PMs appropriées qui peuvent héberger les VMs qui vont être migrer pour répondre à la question "*Où migrer ?*".
- **Le processus de migration des VMs :** dans lequel on cherche "*Comment migrer ?*" ou "*Quelle approche utilisée pour migrer les VMs ?*".
- **La détermination des VMs à faire migrer :** qui répond à la question "*Combien et quelles VMs à faire migrer ?*" pour sélectionner les VMs à faire migrer afin d'atteindre une meilleure amélioration des performances du système.

Afin de répondre à ces questions, nous divisons la stratégie de migrations des VMs en quatre principales phase : (*i*) la phase de surveillance, (*ii*) la phase de pré-négociation ou de préparation, (*iii*) la phase de négociation, et (*iv*) la phase de sélection des machines virtuelles.

4.4.1 Phase de surveillance

Dans ce travail, nous cherchons à atténuer les préoccupations énergétiques des fournisseurs de services Cloud et à répondre aux exigences des utilisateurs. Dans ce but, nous proposons deux seuils qui vont déterminer le moment de migration : un pour la consommation d'énergie, et l'autre pour la qualité de service.

Chaque Broker vérifie périodiquement l'état de son datacenter (voir Figure 4.3), et il lancera une migration si et seulement si l'un des deux seuils (seuil de consommation d'énergie ou seuil de temps de réponse) est dépassé. La combinaison du théorème de "*correspondance stable*" et de l'utilisation de ces deux seuils permet de fournir un certain équilibre entre les contraintes des fournisseurs et les exigences des utilisateurs en assurant la satisfaction des deux parties, et dans le pire de cas, il n'y aura pas de déceptions.

Considérons un datacenter D_i qui contient m machines physiques et de n machines virtuelles, le seuil du temps de réponse ($S_{Tresp}(D_i)$) et le seuil de consommation d'énergie ($S_{Cenergy}(D_i)$) sont calculés, respectivement, comme suit :

$$S_{Tresp}(D_i) = \frac{\sum_{k=1}^{n} T_{resp_k}}{n} + [\frac{\sum_{k=1}^{n} T_{resp_k}}{n} * Const] \tag{4.4}$$

$$S_{Cenergy}(D_i) = \frac{\sum_{j=1}^{m} EnergyCons_j}{m} + [\frac{\sum_{j=1}^{m} EnergyCons_j}{m} * Const] \tag{4.5}$$

Tels que :
$Const$: est une constante ($30\% \leq Const \leq 70\%$).
T_{resp_j} : est le temps moyen requis par une VM_k pour exécuter toutes ces cloudlets. Où une cloudlet représente n'importe quelle requête ou tâche soumise au cloud.
$EnergyCons_j$: représente la consommation d'énergie de PM_j.

Les seuils $S_{Tresp}(Di)$ et $S_{Cenergy}(Di)$ sont les seuils calculés lors d'une vérification de l'état interne du datacenter, autrement dit, les valeurs de ces deux seuils vont déterminer s'il y'aura une migration intra datacenter ou non. Pour calculer les seuils globaux du Cloud et afin de déterminer l'état externe du datacenter D_i, le Broker B_i doit récolter des informations sur les moyennes des temps de réponses et les consommations d'énergie des

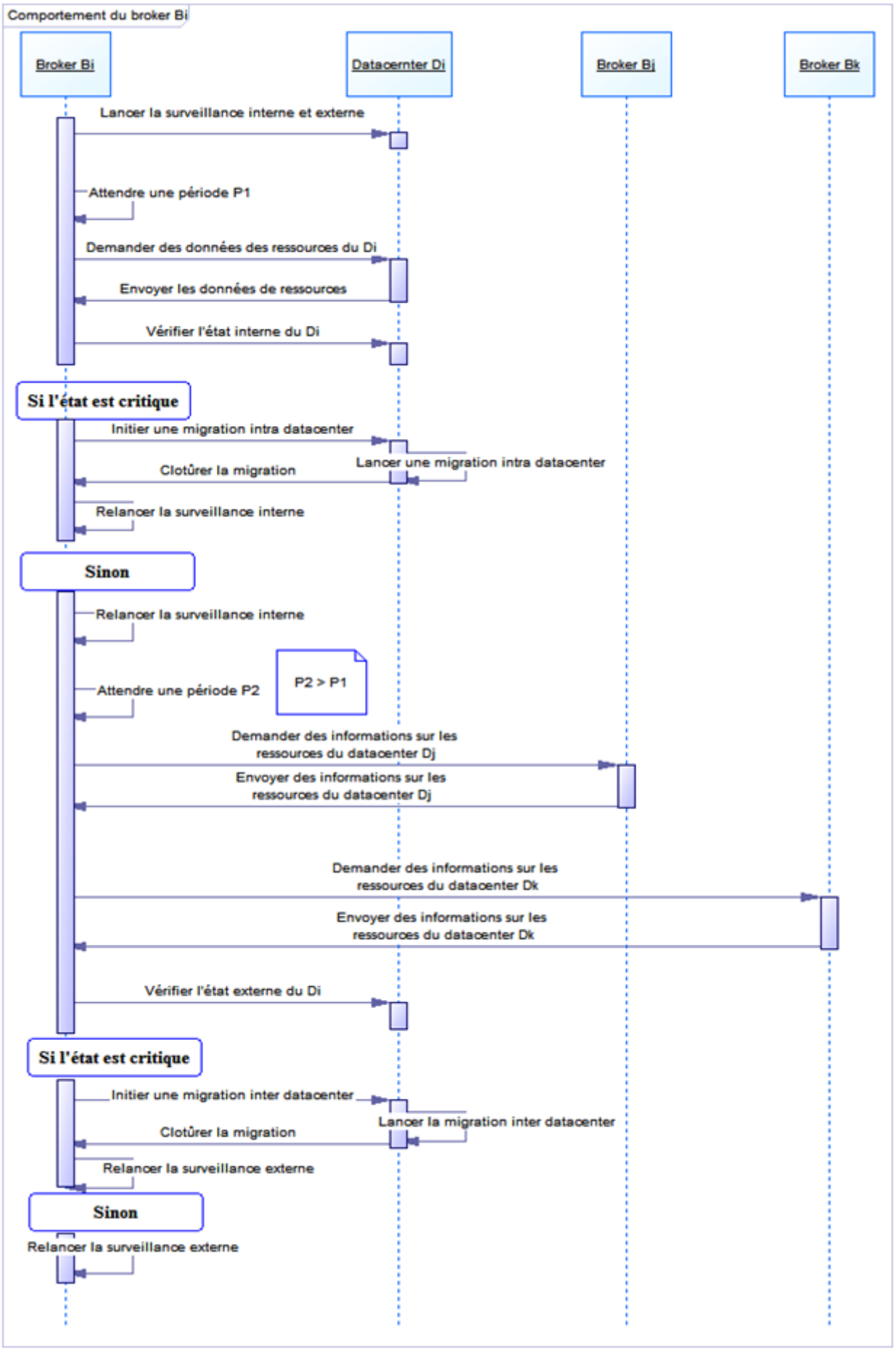


FIGURE 4.3 – *Le diagramme de séquence du comportement du broker*

autres $(w-1)$ datacenters auprès des autres brokers (comme indiqué dans la Figure 4.3). Les seuils globaux du système du temps de réponse et de la consommation d'énergie sont calculés, respectivement, comme suit :

$$GS_{Tresp} = \frac{\sum_{i=1}^{w} T_{resp_{D_i}}}{w} + [\frac{\sum_{i=1}^{w} T_{resp_{D_i}}}{w} * Const] \tag{4.6}$$

$$GS_{Cenergy} = \frac{\sum_{i=1}^{w} EnergyCons_{D_i}}{w} + [\frac{\sum_{j=1}^{m} EnergyCons_{D_i}}{w} * Const] \tag{4.7}$$

Où :
$T_{resp_{D_i}}$: est le temps moyen requis par toutes les VMs du datacenter D_i pour exécuter toutes leurs cloudlets.
$EnergyCons_{D_i}$: représente la consommation d'énergie du datacenter D_i.

Le temps de réponse

Nous considérons le temps de réponse d'une cloudlet CLD_i comme le temps nécessaire à une machine virtuelle VM_k pour recevoir cette cloudlet, l'exécuter, et renvoyer la réponse à l'utilisateur (Voir les formules 4.8, 4.9, et 4.10) :

$$T_{resp}(CLD_i) = T_{exe}(CLD_i) + T_{trn}(CLD_i) + T_{att}(CLD_i) \tag{4.8}$$

Avec :
$T_{exe}(CLD_i)$: est le temps d'exécution de la cloudlet CLDi qui est calculé par la formule 4.9.

$$T_{exe}(CLD_i) = \frac{CLD_iSize}{VM_kSize} \tag{4.9}$$

$T_{trn}(CLD_i)$:est le temps de transfert de la cloudlet CLD_i vers VM_k.

$$T_{trn}(CLD_i) = \frac{CLD_iSize}{VM_kBW} \tag{4.10}$$

Remarque : dans la formule 4.9, CLD_iSize est mesuré en MI (*Million d'Instructions*) et VM_kSize en $MIPS$ (*Million d'Instruction Par Seconde*). Par contre dans la formule 4.10, CLD_iSize est mesuré en Mb (*Mégabits*) avec VM_kBW en $Mbps$ (*Mégabits Par Seconde*).

La consommation d'énergie

Selon Richa et Purohit (2011), de nombreuses études Fan et al. (2007b), Kusic et al. (2008) montrent qu'il existe une relation linéaire entre la consommation d'énergie et l'utilisation du processeur. Ces études affirment qu'une puissance moyenne consommée par un serveur inactif représente 70% de l'énergie consommée par un serveur pleinement utilisé. Ainsi, la consommation d'énergie d'une machine physique est calculée comme indiqué dans la formule 4.11 :

$$P(u) = P_{max} * (0.7 + 0.3 * u) \tag{4.11}$$

Où :
$Pmax$: est 250 W pour les serveurs modernes. u : est le taux d'utilisation de processeur et il est calculé par la formule 4.15.

Le résultat de cette phase de surveillance est un ensemble de machines virtuelles susceptibles d'être migrées. Ces machines virtuelles appartiennent aux machines physiques dont la consommation d'énergie a dépassé le seuil de consommation d'énergie ou le temps de réponse a dépassé le seuil du temps de réponse.

4.4.2 Phase de pré-négociation ou de préparation

Après avoir déterminer les machines virtuelles qui vont être migrées, cette phase permet de sélectionner les machines physiques (PMs) qui vont jouer le rôle d'hébergeur pour ces VMs. Dans cette phase, une liste pour chaque VM est établie, contenant les PMs où elle veut être migrée par ordre stricte de préférence. De même, une liste pour chaque PM est établie, contenant les VMs qu'elle veut héberger selon un ordre stricte de préférence. Ainsi, nous pouvons diviser cette phase en deux sous phase :

1. **La détermination des emplacements potentiels**
   Le choix des emplacements potentiels dépends de deux critères :

   - **Le type de migration :**
     - 1^{er} *cas -Migration intra datacenter- :* si la migration est intra datacenter, les emplacement potentiels sont les machines physiques du datacenter en question et dont la consommation d'énergie et le temps de réponse n'ont pas dépassé les seuils calculés.
     - 2^{eme} *cas -Migration inter datacenter- :* si la migration est inter datacenter, les emplacements potentiels doivent être choisis parmi les machines physiques des datacenters dont l'état est toujours stable (non critique).
   - **Le seuil dépassé :**
     - 1^{er} *cas -Migration intra datacenter- :* Si le seuil qui a été dépassé est celui de la consommation d'énergie, nous favorisons les machines physiques qui ont déjà des VMs en exécution comme emplacement potentiel. Cette proposition a pour objectif d'éviter d'utiliser les machines physiques qui n'ont aucune ressource utilisée et par conséquence réduire la consommation d'énergie du datacenter qui héberge cette machine.
     - 2^{eme} *cas -Migration inter datacenter- :* Si le seuil qui a été dépassé est celui du temps de réponse, toutes les machines physiques sont concernées par la migration.

   Dans les quatre cas précédent, la condition suivante doit être vérifiée :

   $$\sum_{j=1}^{m} Cap(PM_j) > VSize \tag{4.12}$$

   Tels que :
   $CapPM_j$: est l'espace libre dans une machine physique j qui a été choisie comme emplacement potentiel d'une ou plusieurs VMs. $VSize$: La somme des tailles de toutes les VMs qui vont être migrées.

2. **L'établissement des listes de préférences**
   Cette étape consiste à créer les listes de préférences des deux parties de la migration (*les deux parties marché*) :

(a) Une liste de préférence $VmPre_k$ pour chaque machine virtuelle VM_k, concernée par la migration, qui contient les machines physiques, qui ont été choisi comme emplacements potentiels, mises dans un ordre ascendant selon deux méthode :

- **Le temps de transfert (Voir formule 4.13) de VM_k vers ces machines physiques :** dans ce cas, nous avons utilisé une matrice pour représenter la bande passante entre chaque deux machines physiques de l'ensemble de toutes les machines physiques du datacenter :

$$T_{Trans}(VM_k, S(VM_k), D(VM_k) = S_{VM_k} * \sum_{c=1}^{q-1} \frac{1}{BP(c, c+1)} \tag{4.13}$$

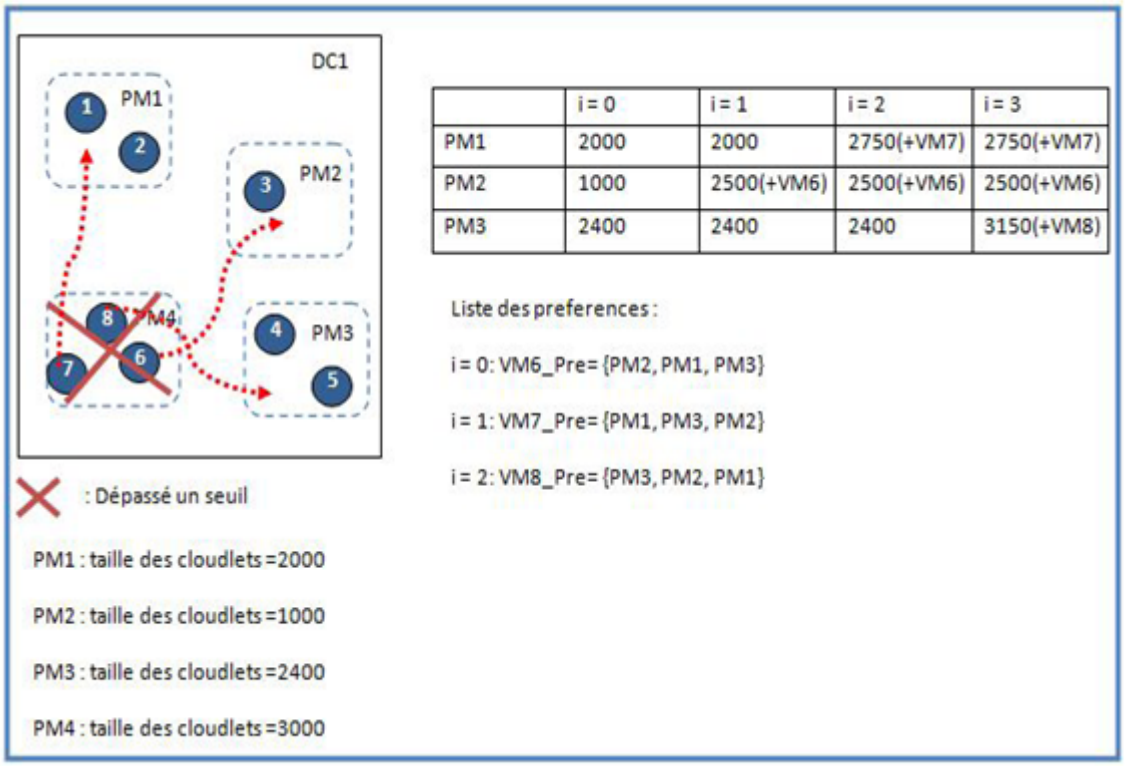
Tels que :
SVM_k : est la taille de VM_k.
$S(VM_k)$: est la machine contenant VM_k.
$D(VM_k)$: est la machine physique de destination.
$BP(c, c+1)$: représente la bande passante entre la machine entre deux machines physiques voisines PM_c et PM_{c+1}.
q : est le nombre de saut entre la PM source et la PM de destination.

On dit qu'une PM_g domine PM_h dans $VmPre_k$ ou $PM_g <_{VmPre_k} PM_h$ si et seulement si $T_{Trans}(VM_k, S(VM_k), PM_g) < T_{Trans}(VM_k, S(VM_k), PM_h)$ et donc la machine virtuelle VM_k préfère la machine physique PM_g à la machine physique PM_h.

- **La taille des cloudlets en MI (*Million d'Instructions*) :** le but de cette méthode est d'établir une liste des préférences contenant des PMs, pour chacune des VMs hébergée par la PM_i dont le seuil a été dépassé. Pour chaque VM, le Broker met toutes les PMs (à l'exception de PM_i) dans une liste dans un ordre croissant selon la somme des tailles des cloudlets qui ont été affectées à ses VMs. Le Broker ajoute par la suite, la taille des cloudlets de cette VM à la somme des tailles de la première PM de la liste de préférence de celle-ci. Cette méthode permet de :
  - Garantir un équilibrage de charge en distribuant les VMs de manière équitable selon la taille des cloudlets, sur l'ensemble des PMs.
  - Évitant de surcharger une PM par rapport à une autre, ce qui réduit la chance d'un éventuel dépassement de seuil.

  Pour illustrer ce principe, nous proposons l'exemple de la Figure 4.4 : un datacenter contient quatre (4) machines physique PM_1, PM_2, PM_3 et PM_4, où PM_4 a dépassé un seuil.
  Dans ce cas, les machines virtuelles de la PM_4 vont établir des listes des préférences contenants les autres PMs.
  Soit VM_6, VM_7, VM_8 les machines virtuelles hébergées par la PM_4 :
  - Taille des cloudlets de VM_6 = 1500 MI.
  - Taille des cloudlets de VM_7 = 750 MI.
  - Taille des cloudlets de VM_8 = 750 MI.

(b) Une liste de préférence $PmPre_j$ pour chaque machine physique PM_j concernée par la migration. Cette liste contient les VMs, qui vont être migrées, mises dans un ordre ascendant selon l'impact de ces VMs sur l'utilisation du CPU de la



	i = 0	i = 1	i = 2	i = 3
PM1	2000	2000	2750(+VM7)	2750(+VM7)
PM2	1000	2500(+VM6)	2500(+VM6)	2500(+VM6)
PM3	2400	2400	2400	3150(+VM8)

FIGURE 4.4 – *Exemple de l'établissement des listes de préférences en utilisant les tailles des cloudlets*

machines PM_j $(ImpactUtilization(VM_k, PM_j))$:

$$ImpactUtilization(PM_j, VM_k) = \frac{CPU(VM_k)}{CPU(PM_j)} \tag{4.14}$$

Tels que :
$ImpactUtilisation(VM_k, PM_j)$: est le taux d'utilisation du CPU de la machine PM_j par VM_k.
$CPU(VM_k)$: est la quantité de MIPS que requiert la machine virtuelle VM_k.
$CPU(PM_j)$: est la quantité de MIPS que la machine physique PM_j peut supporter.

En utilisant $ImpactUtilisation(VM_k, PM_j)$, nous pourrons calculer le taux d'utilisation CPU moyen (paramètre u dans la formule 4.11) de la machine physique PM_j et ainsi calculer sa consommation d'énergie en utilisant la formule 4.11. Le taux d'utilisation moyen de PM_j est calculé par la formule suivante :

$$CpuUtilization(PM_j) = \frac{\sum_{l=1}^{w} ImpactUtilization(VM_l)}{W} \tag{4.15}$$

Avec :
W : est le nombre de machines virtuelles stockées dans PM_j.

La fin de cette deuxième phase marque la fin des préparations du processus de migration. La prochaine phase portera la description de notre procédure d'acceptation différée et le traitement du cas de polarisation des correspondance stable.

4.4.3 Phase de négociation

Cette phase est le noyau de notre approche. Elle implique l'utilisation d'une procédure d'acceptation différée pour le problème de migration des machines virtuelles vers des machines physiques afin de trouver la meilleure correspondance stable entre les PMs et les VMs, en utilisant leurs listes de préférences établies au préalable.

Dans cette phase, l'un des deux parties (PM ou VM) fait des propositions à l'autre, pour s'associer avec lui, et ce dernier selon ses préférences décide s'il va accepter ou rejeter ces propositions. Les deux algorithmes présentés ci-dessus, décrivent la phase de négociation entre la PM et la VM, dont l'algorithme 3 (illustré par la Figure 4.5) représente la procédure de PM après la proposition d'une VM, et l'algorithme 3 (illustré par la Figure 4.6) représente la procédure de VM qui envoie des propositions aux PMs. À la fin de l'exécution de ces deux algorithmes, chaque VM_k et PM_j obtiendra sa liste des partenaires $VmList_k$ et $PmList_j$ respectivement.
Remarque : Les deux algorithmes s'exécutent en parallèle (Procédure PM_j et procédure VM_k).

Algorithme 3 Procédure PM_j

Tant que ¬le Broker n'a pas envoyé un message "Stop" **Faire**
  Attendre une proposition d'une machine virtuelle
  Soit VM_k la VM qui a envoyé la proposition.
  Si $\neg CapPM_j \geq S_{VM_k}$ **Alors**
    Accepter la proposition de VM_k
    Ajouter VM_k à la liste $PmList_j$ par ordre de préférence
    Si $\neg CapPM_j < S_{VM_k}$ **Alors**
      Pour Tous élément VM_g classé après VM_k dans $PmPre_j$ **Faire**
        envoyer un message de rejet à VM_g
        supprimer VM_g de la liste de préférence $PmPre_j$
      Fin Pour
    Fin Si
  Sinon
    Si $\neg VM_k$ appartient à $PmPre_j$ **Alors**
      Envoyer un rejet au dernier élément VM_b du $PmList_j$.
      Envoyé une confirmation au VM_k
      Ajouter VM_k à la liste $PmList_j$ par ordre de préférence
      Supprimer VM_b de la liste $PmList_j$
      Pour Tous élément VM_g classé après VM_k dans $PmPre_j$ **Faire**
        envoyé un message de rejet à VM_g
        supprimer VM_g de la liste de préférence $PmPre_j$
      Fin Pour
    Sinon
      envoyé un message de rejet à VM_k
    Fin Si
  Fin Si
Fin Tant que

Le résultat de ces deux algorithmes est une correspondance stable optimale pour les VMs (*VM-optimal* ou μ_{vm}). En appliquant les même algorithmes, mais cette fois-ci se sont les PMs qui font les propositions, nous obtenons une correspondance stable optimale

pour les machines physiques (*PM-optimal* ou μ_{pm}). Cependant, les deux correspondances

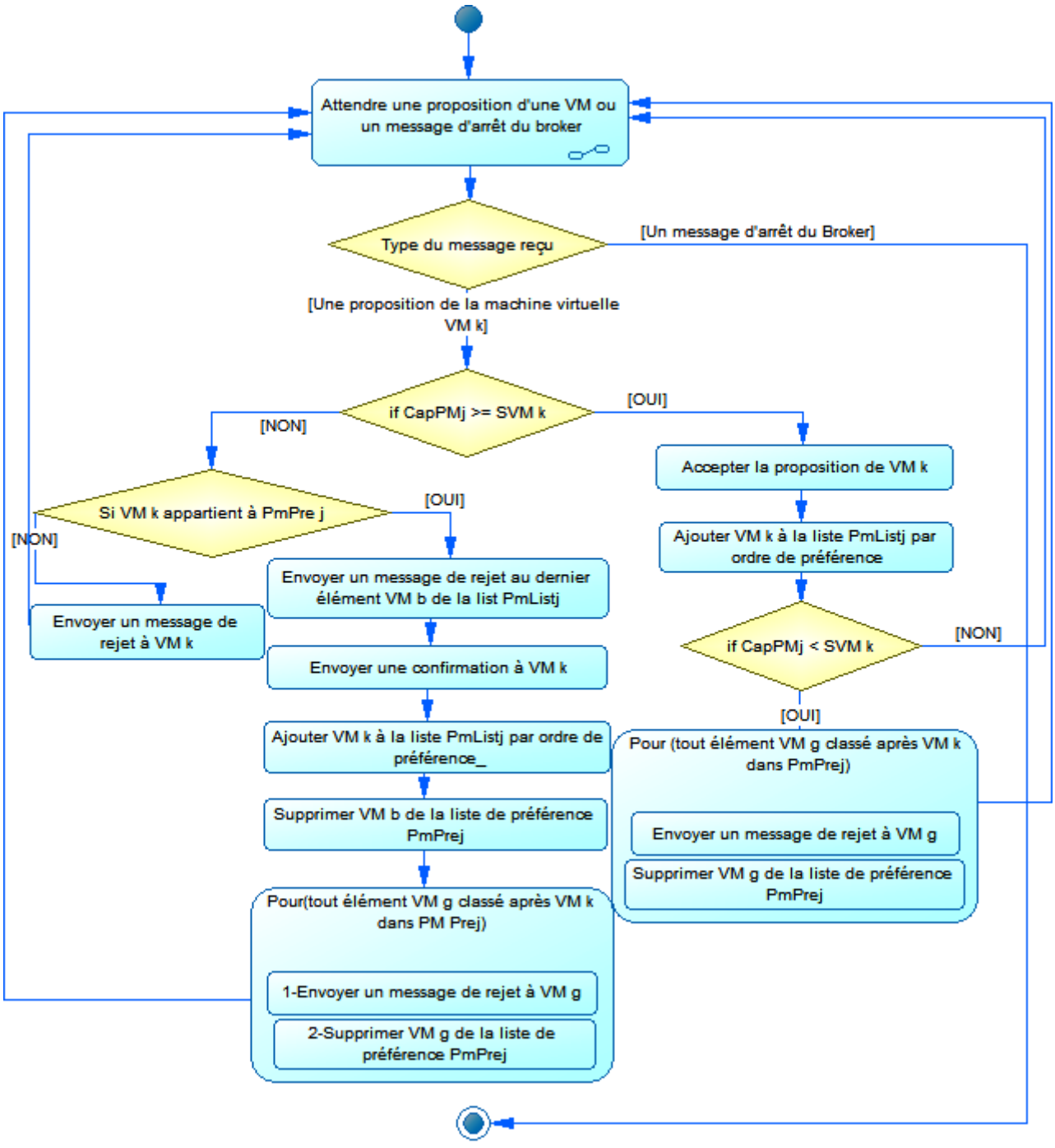


FIGURE 4.5 – *Diagramme d'activité de la procédure d'acceptation de PM_j*

stables (*VM-optimal* et *PM-optimal*) peuvent être :

- *Contradictoires :* la meilleure solution pour un côté peut être, en même temps, la pire pour l'autre. C'est à dire, chaque VM a un partenaire dans *VM-optimal* qui est différent de celui de $PM - optimal$ et chaque PM hébergent des VMs dans *PM-optimal* qui différent de celles dans *VM-optimal*, Ce problème est connu comme la polarisation des correspondances stables (voir section 4.2.4).

- *Différentes : VM-optimal* et *PM-optimal* contiennent des paires communes et d'autres contradictoires.

Algorithme 4 Procédure VM_k

```
1: Tant que ¬End = False Faire
2:   Tant que ¬VmPre_k n'est pas vide Faire
3:     Choisir PM_j le premier élément de VmPre_k
4:     Proposer à PM_j d'être un partenaire de VM_k
5:     Supprimer PM_j de VmPre_k
6:     Ajouter PM_j à la liste VmList_k.
7:     Attendre la réponse PM_j
8:     Si ¬PM_j accepte la proposition Alors
9:       Rien
10:    Sinon
11:      Supprimer PM_j de VmPre_k
12:      Si ¬PM_j est dans la liste VmList_k Alors
13:        Supprimer PM_j de VmList_k
14:      Sinon
15:        End ← True
16:      Fin Si
17:    Fin Si
18:  Fin Tant que
19: Fin Tant que
```

Pour surmonter ce problème, nous avons utilisé un algorithme qui a été proposé dans Martínez et al. (2004) pour résoudre le problème de la polarisation d'une variante plusieurs-à-plusieurs de la correspondance stable (Many-to-Many stable matching problem), connu sous le nom du "*problème du firmes-consultants*" et qui a été présenté dans la section 4.2.2, en calculant l'ensemble des plusieurs-à-plusieurs correspondances stables basé sur les changements effectuer sur les listes de préférences (Voir Section 4.2.4).

Dans cet algorithme, le côté qui fait des propositions doit faire une troncature de sa liste de préférences en enlevant son meilleur choix de cette liste et relancer la procédure d'acceptation différée pour obtenir sa correspondance stable optimale (*VM-optimal* ou *PM-optimal*). Après, l'autre côté doit suivre le même processus et faire une troncature de sa liste de préférences, puis relancer la procédure d'acceptation différée. Ces démarches seront répétées jusqu'à ce que les deux correspondances stable convergent ou dans le meilleur des cas, nous obtiendrons des résultats identiques (*VM-optimal = PM-optimal*). Si après une troncature des listes de préférences, une migration n'est plus possible :

- Soit parce que la Formule 4.12 n'est pas plus respectée, c'est à dire, les PMs restantes dans les listes des préférences des VMs ne pourront pas héberger toutes les VMs (cas de troncature des listes de préférences des VMs "$VmPre$").
- Soit parce qu'une ou plusieurs VMs ne figurent plus dans les listes de préférences des PMs (cas de troncature des listes de préférences des PMs "$PmPre$").

Alors on arrête le processus de négociation et nous choisissons l'une des deux correspondances stables actuelles, *VM-optimal* ou *PM-optimal*, selon le seuil qui a déclenché la migration. Si l'état est devenu critique à cause du seuil de temps de réponse, le schéma

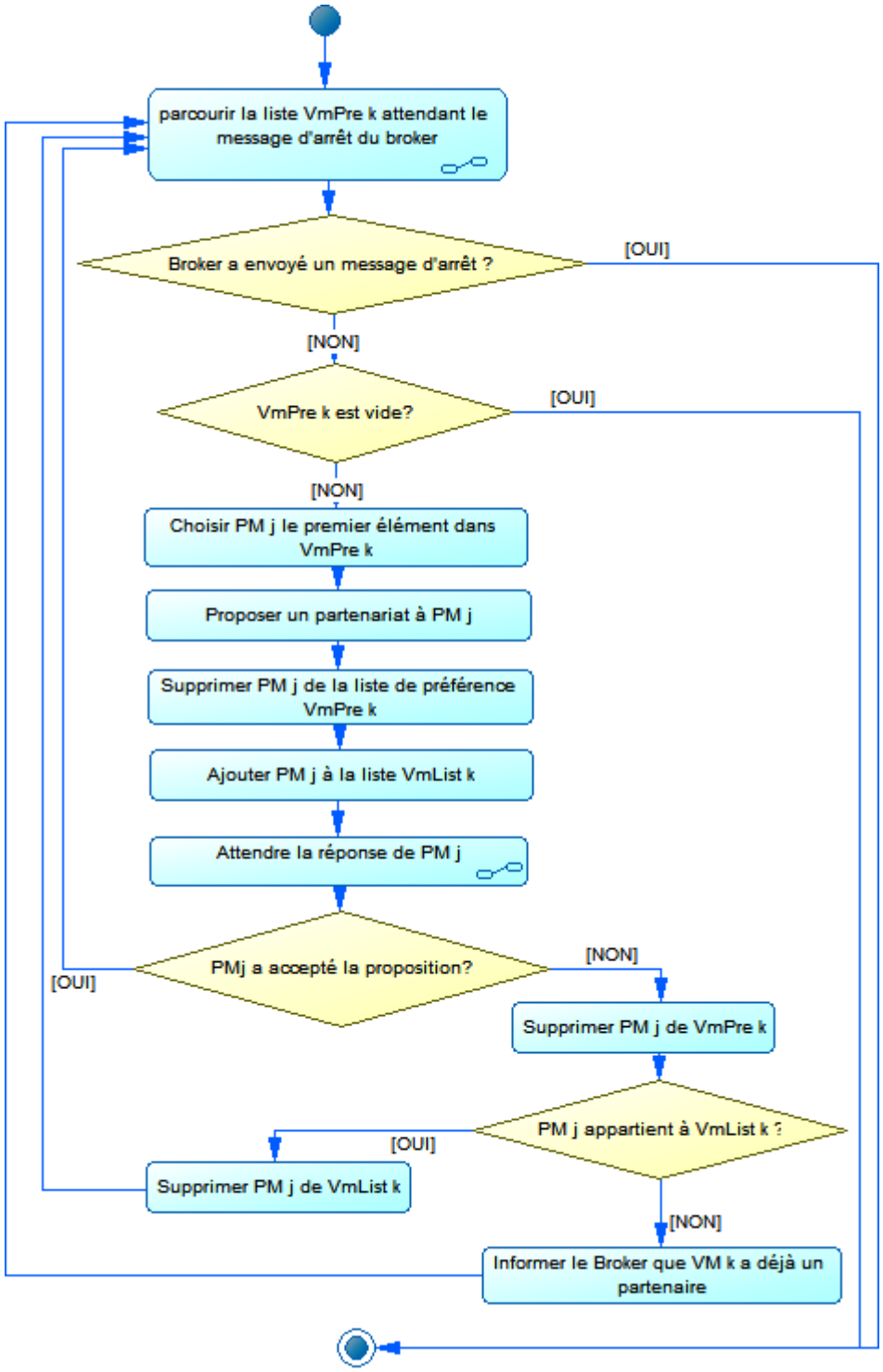


FIGURE 4.6 – *Diagramme d'activité de la procédure de proposition de* VM_k

d'affectation présenté dans la correspondance stable *VM-optimal* sera retenu, sinon si c'est à cause du seuil de consommation d'énergie alors le schéma d'affectation retenu sera celui de *PM-optimal*.

Le schéma de la Figure 4.7 représente, de manière détaillée, les principaux étapes du traitement de polarisation des correspondances stables dans le cas de migration des machines virtuelles vers des machines physiques.

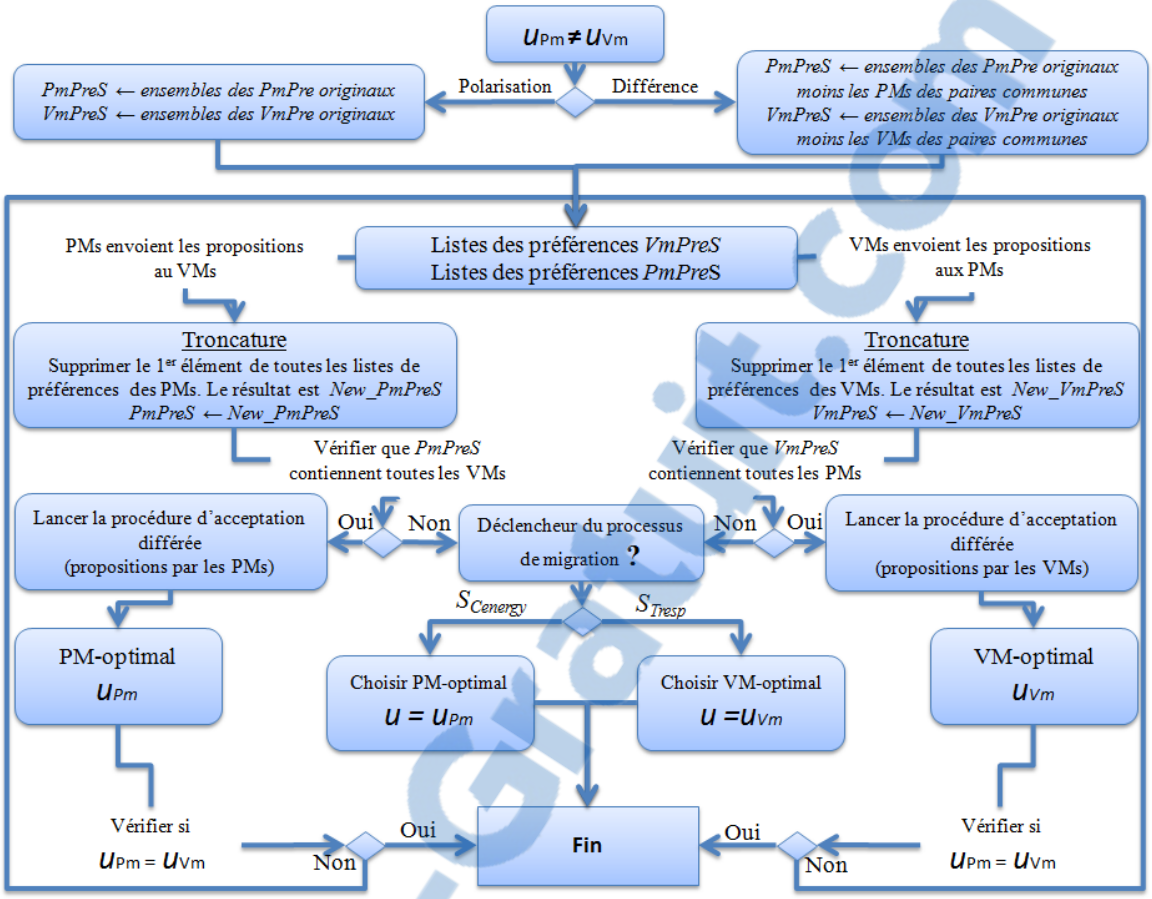


FIGURE 4.7 – *Schéma des traitements appliqués au problème de polarisation*

4.4.4 Phase de séléction des VMs

Cette phase est la dernière à être exécuté, après avoir répondu aux trois questions : Quand migrer ? Comment migrer ? Où migrer ? pour répondre à la dernière question : Combien et quelles *VMs* à faire migrer ?. La sélection des *VMs* est une phase utilisée pour déterminer les machines virtuelles, parmi celles qui ont été désignées comme des entités de migration, qui vont être migrées pour obtenir la meilleure répartition des *VMs* sur les emplacements potentiels. Il s'agit donc de trouver la répartition qui garantit le compromis le plus optimale parmi ceux possibles entre la consommation d'énergie et la qualité de service.

Nous allons utiliser le théorème de Coase pour déterminer la répartition optimale des machines virtuelles sur les datacenters pour avoir le meilleur compromis possible entre la consommation d'énergie et le temps de réponse.

L'utilisation du théorème de Coase dans cette phase comporte plusieurs avantages :

- Réduire de la charge de migration et ainsi alléger le réseau.
- Permettre un équilibrage de charge entre les *PMs* et une meilleure exploitation des ressources en évitant de déplacer toutes *VMs* hébergées par les machines physiques dont la valeur calculée (consommation d'énergie ou temps de réponse) a dépassé l'un des deux seuils.

- Augmenter l'intervalle entre deux violations des seuils et ainsi garder le systèmes dans un état sain le plus longtemps possible.

Comme nous l'avons déjà montré dans la section 4.2.5, si le déclencheur de la migration est le seuil de consommation d'énergie, la propriété sera la consommation d'énergie et le revenu sera donc le gain en consommation d'énergie, tandis que le coût privé sera une possible dégradation dans la qualité de service (Temps de réponse). Et inversement, si le déclencheur de la migration est le seuil de temps de réponse, le temps de réponse sera désigné comme propriété, et une possibilité d'augmentation de la consommation d'énergie comme coût privé.

Reprenons l'exemple de la Figure4.4, et supposons que le déclencheur du processus de migration est un pic de consommation d'énergie et que la phase de négociation a produit la correspondance stable μ suivante : μ = (PM_1, VM_7), (PM_2, VM_6), (PM_3, VM_8).
Soit C1 la consommation d'énergie d'une PM avant la migration, et C2 est sa consommation estimée après la migration. Nous calculons le gain de consommation d'énergie par la formule 4.16 :

$$Gain(PM_j) = \left(1 - \left\lfloor \frac{\sum_{i=1}^{j} C_2(PM_i)}{\sum_{i=1}^{j} C_1(PM_i)} \right\rfloor \right) * 100 \tag{4.16}$$

De l'autre côté, la perte est calculée par la formule 4.17 :

$$Loss(PM_j) = - \left(1 - \left\lfloor \frac{\sum_{i=1}^{j} C_1(PM_i)}{\sum_{i=1}^{j} C_2(PM_i)} \right\rfloor \right) * 100 \tag{4.17}$$

Puisque le déclencheur de la migration est le seuil de la consommation d'énergie, les PMs seront par ordre croissant placés selon leurs consommations d'énergie.

Table 4.1 – *Exemple d'implémentation du théorème de Coase*

	Gain ou perte de consommation d'énergie	**Gain ou perte de temps de réponse**	**Gain total**
$PM_1(VM_7)$	50%	-15%	50+(-15)=35%
$PM_2(VM_6)$	20%	20%	40%
$PM_3(VM_8)$	25%	-10%	15%

Le gain de PM_1 de la consommation d'énergie est calculé en divisant la $C2(PM_1)$ sur $C1(PM_1)$ tandis que le gain de PM_2 de la consommation d'énergie est calculé en divisant la somme de $C2(PM_1)$ et $C2(PM_2)$ sur la somme de $C1(PM_1)$ et $C1(PM_2)$ et ainsi de suite. La même méthode est utilisée pour obtenir le gain de temps de réponse. Nous remarquons dans le tableau que la ligne 2 a le meilleur gain avec 40%. Donc, la répartition finale des VMs est : (PM_1, VM_7), (PM_2, VM_6). Le VM_8 ne sera pas migrée vers PM_3.

4.5 Conclusion

Ce chapitre a présenté une description détaillée de l'approche proposée de migration des machines virtuelles dans notre architecture du Cloud. Nous avons montré la manière

dont des théorèmes issus de l'économie ont été intégrés dans l'approche proposée. Ces théorèmes ont été abordés dans la première partie de ce chapitre de façon simple et illustrés avec des exemples.

Nous avons présenté dans la deuxième partie des algorithmes, des diagrammes, et des schémas expliquant la manière qui a été utilisé pour exploiter les théorèmes abordés dans la première partie afin d'atteindre les objectifs visés par notre travail.

Le chapitre suivant sera consacré à la description de l'implémentation de la stratégie proposée dans le simulateur CloudSim Calheiros et al. (2011). Quelques expérimentations effectuées seront discutées pour évaluer notre stratégie.

Implémentation et interprétation des résultats

5

Sommaire

5.1 Introduction

Les plates-formes cibles sur lesquelles porte notre étude sont des environnements à large échelle distribués qui peuvent être situés sur de nombreux domaines administratifs. Par conséquent, il peut être relativement difficile de conduire des expériences reproductibles de longue durée dans un tel cadre. Nous avons donc choisi de faire usage d'un simulateur afin de tester notre approche. Un autre avantage de l'utilisation d'un simulateur est la possibilité de pouvoir maîtriser l'ensemble des paramètres de la plate-forme simulée, ce qui est peut être difficile voir impossible dans un environnement réel. Notre choix s'est porté sur un simulateur largement utilisé dans le contexte du Cloud Computing qu'est le CloudSim.

Dans ce chapitre nous allons présenter le simulateur CloudSim ainsi que certaines de ses principales fonctionnalités, par la suite, nous définirons les métriques utilisées pour tester l'approche proposée de migration des VMs, ensuite, nous analyserons les résultats obtenus des simulation, et enfin, nous discuterons ces résultats pour comprendre le comportement de notre approche.

5.2 Environnement de développement

5.2.1 Langage de programmation Java

Java est à la fois un langage de programmation informatique orienté objet et un environnement d'exécution portable créé par James Gosling et Patrick Naughton employés de Sun Microsystems avec le soutien de Bill Joy (co-fondateur de Sun Microsystems en 1982), présenté officiellement le 23 mai 1995 au SunWorld Iván Devosa (2010). Le langage Java a la particularité principale que les logiciels écrits avec ce dernier sont très facilement portables sur plusieurs systèmes d'exploitation tels que : Unix, Microsoft Windows, Mac OS ou Linux avec peu ou pas de modifications. C'est la plate-forme qui garantit la portabilité des applications développées en Java Iván Devosa (2010).
Les applications Java peuvent être exécutées sur tous les systèmes d'exploitation pour lesquels a été développée une plate-forme Java, dont le nom technique est JRE (*Java Runtime Environment*). Cette dernière est constituée d'une JVM (*Java Virtual Machine*), le programme qui interprète le code Java et le convertit en code natif. Mais le *JRE* est surtout constitué d'une bibliothèque standard à partir de laquelle doivent être développés tous les programmes en Java. C'est la garantie de portabilité qui a fait la réussite de Java dans les architectures Client-Serveur en facilitant la migration entre serveurs, très difficile pour les gros systèmes IPETI (2015).
Java est devenu aujourd'hui une direction incontournable dans le monde de la programmation, parmi les différentes caractéristiques qui sont attribuées à son succès Iván Devosa (2010) :

- L'indépendance de toute plate-forme : le code reste indépendant de la machine sur laquelle il s'exécute. Il est possible d'exécuter des programmes Java sur tous les environnements qui possèdent une Java Virtual Machine.
- Java est également portable, permettant à la simulation d'être distribuée facilement sans avoir à recompiler le code pour les différents systèmes.
- Le code est structuré dans plusieurs classes, dont chacune traite une partie différente de la simulation.

- Il assure la gestion de la mémoire.
- Java est multitâche : il permet l'utilisation des Threads qui sont des unités d'exécution isolées.

5.2.2 Le simulateur CloudSim

CloudSim est un simulateur qui s'appuie sur la recherche et le développement dans le domaine émergent du Cloud Computing. Nous avons opté pour ce simulateur grâce à ses fonctionnalités Calheiros et al. (2011) :

- Un support pour la modélisation et la simulation d'environnements à grande échelle du Cloud Computing, y compris les DataCenters.
- Une plate-forme autonome pour la modélisation du Cloud Computing, des DataCenters, des Brokers, de l'ordonnancement et des politiques d'allocation des ressources.
- Un support pour la simulation des connexions réseau entre les éléments du système de simulation
- La disponibilité d'un moteur de virtualisation qui facilite la création et la gestion indépendante des services ainsi que l'hébergement des services virtualisés sur un nœud d'un Datacenter.
- La flexibilité pour commuter entre l'allocation en espace partagé et en temps partagé et la prise en charge de la répartition des cœurs de traitement aux services virtualisés.
- Simulation de la définition de matériel de centre des données (DataCenter) en termes de machines physiques composées de processeurs, de dispositifs de stockage, de mémoire et de largeur de bande interne.
- Simulation des spécifications, de la création et de la destruction de machines virtuelles.

L'architecture du CloudSim

La structure logicielle de CloudSim est représentée par une architecture en couches (voir Figure 5.1). Cette architecture est composée de 4 couches :

- La couche la plus basse représente l'outil SimJava, le moteur de simulation d'événements discrets qui met en œuvre les principales fonctionnalités requises pour des structures de simulation de haut niveau comme la formation d'une file d'attente, le traitement des événements, la création de composants système (les services, les machines (Hôte), le centre de données (Datacenter), le Broker, les machines virtuelles) , la communication entre les composants et la gestion de l'horloge de simulation Calheiros et al. (2011).
- La couche suivante englobe les bibliothèques qui implémentent le Toolkit GridSim, qui est un simulateur de grilles. Ce dernier supporte :
  - Les composants logiciels de haut niveau pour modéliser les multiples infrastructures de grille, y compris les réseaux et les profils de trafic associés.

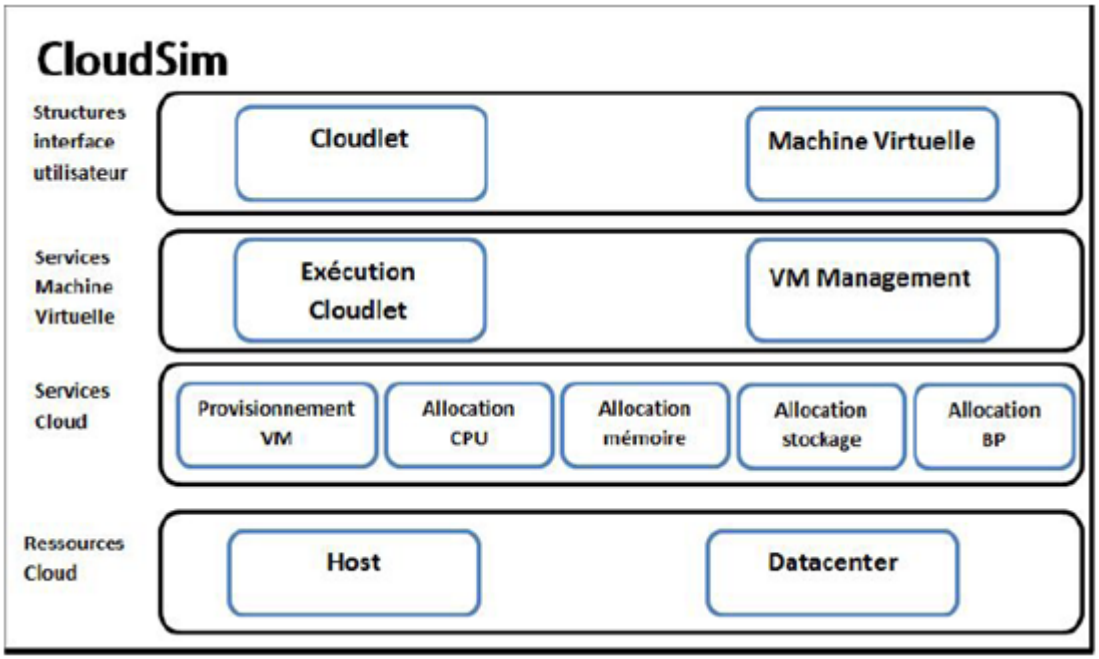


FIGURE 5.1 – *Architecture de CloudSim Calheiros et al. (2011)*

  - Les composants fondamentaux de grille comme les ressources, les ensembles de données, les traces de charge de travail et les services de l'information.

- La prochaine couche identifie l'outil CloudSim. Elle permet l'extension des fonctionnalités principales exposées par la couche GridSim. CloudSim fournit le nouveau support pour la modélisation et la simulation des environnements virtualisés des centres de données de Cloud comme des interfaces dédiées de gestion de VMs, de mémoire, de stockage et de bande passante Calheiros et al. (2011).
  La couche CloudSim gère l'instanciation et l'exécution des entités principales (VMs, machines, centres de données, application) pendant la période de la simulation. Cette couche est capable d'instancier et de gérer simultanément et de manière transparente la gestion de passage à l'échelle de l'infrastructure de Cloud consistant en milliers de composants système.
  Les fonctionnalités fondamentales comme l'allocation de machines aux VMs à base de demandes des utilisateurs, la gestion d'exécution d'application et le contrôle dynamique sont traitées par cette couche. Un fournisseur de Cloud, qui veut étudier l'efficacité des différentes politiques dans la répartition de ses hôtes, lui faudrait mettre en œuvre ses stratégies dans cette couche en étendant les fonctionnalités du programme de base d'allocation de VM. Un hôte du Cloud peut être partagé en même temps entre plusieurs machines virtuelles (VMs) Calheiros et al. (2011).

- La couche la plus haute dans la pile de simulation est l'*interface utilisateur* qui expose la configuration des fonctionnalités liées aux hôtes : le nombre de machines, leur spécification, les applications (le nombre de tâches et leurs conditions), les VMs, le nombre d'utilisateurs, les politiques d'ordonnancement du Broker.

Les classes de CloudSim

Le simulateur CloudSim est composé de plusieurs classes que nous pouvons classer en deux catégories : des classes qui modélisent les entités comme le Datacenter, le Broker,

etc. Et des classes modélisant les politiques d'allocation.
Parmi les classes fondamentales qui forment les blocs constitutifs du simulateur CloudSim comme est présenté dans la Figure 5.2, nous pouvons citer seulement celles utilisées :

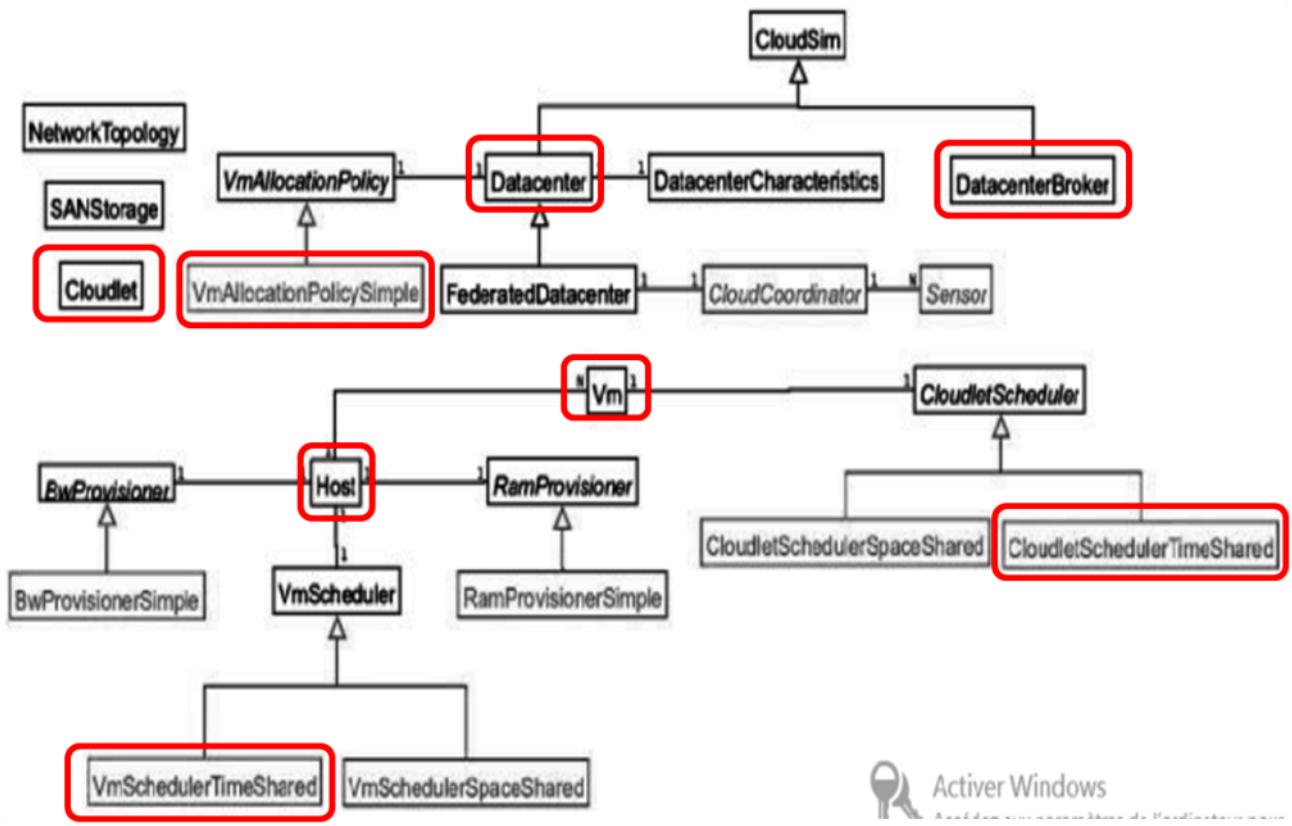


FIGURE 5.2 – *Diagramme de classe de la conception de simulateur CloudSim Calheiros et al. (2011)*

1. **Cloudlet**
   Cette classe modélise les services d'application du Cloud (comme la livraison, réseaux sociaux, et le workflow d'affaires). CloudSim représente la complexité d'une application en fonction de ses besoins informatiques. Chaque service d'application a une taille d'instruction pré-assigné et la quantité de flux de transfert de données qu'il doit entreprendre au cours de son cycle de vie. Cette classe peut également être étendue pour supporter la modélisation de la performance et d'autres paramètres de composition pour les applications telles que les transactions dans les applications orientées base de données Calheiros et al. (2011).

2. **Vm**
   Cette classe représente une instance de machine virtuelle (VM), qui est gérée et hébergée par une machine physique. Chaque composant VM a accès à un composant qui stocke les caractéristiques suivantes liées à une VM : mémoire accessible, le processeur, capacité de stockage, et les politiques de provisionnement interne de la machine virtuelle Calheiros et al. (2011).

3. **Host**
   Cette classe modélise une ressource physique comme le serveur de stockage ou de calcul. Elle encapsule des informations importantes telles que la quantité de mémoire et de stockage, le type de cœurs de traitement (pour représenter une machine multi-core), une politique d'allocation pour le partage de la puissance de traitement

entre les machines virtuelles, et les politiques d'approvisionnement de mémoire et de bande passante pour les machines virtuelles Calheiros et al. (2011).

4. **Datacenter**
   Elle encapsule un ensemble d'hôtes qui peuvent être soit homogènes ou hétérogènes par rapport à leurs configurations matérielles (mémoire, noyaux, capacité et stockage) Calheiros et al. (2011).

5. **DatacenterBroker**
   Cette classe modélise le Broker, qui est responsable de la médiation entre les utilisateurs et les prestataires de services selon les conditions de QoS des utilisateurs. Il déploie les tâches des services à travers les Clouds et utilise le service d'information du Cloud *CIS* (*Cloud Information Services*) pour une allocation des ressources qui répond aux besoins de QoS des utilisateurs. Les chercheurs et développeurs des systèmes doivent étendre cette classe pour évaluer et de tester les politiques personnalisés Calheiros et al. (2011).

6. **DatacenterCharacteristics**
   Cette classe contient des informations sur la configuration des ressources des centres de données Calheiros et al. (2011).

7. **SimEntity**
   Il s'agit d'une classe abstraite, elle représente l'entité de simulation qui est capable d'envoyer des messages à d'autres entités et de gérer les messages reçues ainsi que les événements. Toutes les entités doivent étendre cette classe et redéfinir ses trois principales méthodes : *startEntity()*, *processEvent()* et *shutdownEntity()*, qui définissent les actions pour l'initialisation de l'entité, le traitement des événements et la destruction de l'entité Calheiros et al. (2011).

8. **VMAllocationPolicy**
   C'est une classe abstraite qui modélise les politiques (d'espace partagé, de temps partagé) exigées pour allouer la puissance de traitement aux VMs. Les fonctionnalités de cette classe peuvent facilement être ignorées pour accommoder des politiques spécifiques à l'application de partage de processeur Calheiros et al. (2011).

9. **VMProvisioner**
   Cette classe abstraite représente la politique d'approvisionnement qu'un moniteur de VM utilise pour allouer les VMs aux hôtes. La *VMProvisioner* sélectionne l'hôte qui répond à la quantité de mémoire demandée, le stockage pour le déploiement de la VM Calheiros et al. (2011).

5.3 Implémentation de l'approche proposée dans CloudSim

Nous présentons dans cette partie une vue globale des démarches à effectuer pour réaliser une simulation en détaillant l'intégration de l'approche dans les étapes suivantes :

- **Initialisation du package CloudSim :** elle consiste à créer un *Cloud Information Service* (*CIS*) qui est une entité qui fournit des services d'enregistrement des ressources Cloud, d'indexation, et de découverte. La liste des hôtes devient prêtes à traiter des cloudlets une fois enregistrées auprès de cette entité. Autres entités communiquent avec le *CIS* pour le service de découverte de ressources, qui retourne une liste des identifiants des ressources enregistrées.

- **Création des datacenters et leurs machines physiques :** cette étape permet de :
  - Spécifier les caractéristiques de chaque datacenter : son architecture, les systèmes d'exploitation et les hyperviseurs utilisés, la politique d'allocation des tôches (space or time shared), ainsi que les coûts d'usage des ressources de calculs, de mémoire, de stockage , et de bande passante.
  - Créer un ensemble de machines physiques pour chaque datacenter, telle que chaque machine physique peut posséder une ou plusieurs unités de calcul (*CPU*).
- **Création des Brokers :** créer un Broker pour chaque datacenter. Chaque Broker agit comme un gestionnaire de datacenter qui a le rôle de gestionnaire des machines virtuelles, physiques, et des cloudlets. Le Broker peut créer, ou détruire des VMs, et répartir les cloudlets sur les VMs pour l'exécution.
- **Création des machines virtuelles :** déployer des machines virtuelles sur les datacenters créés.
- **Création des cloudlets :** créer un ensemble de cloudlets, où chaque cloudlet est caractérisée par : un identifiant unique, une longueur, une taille de fichier d'entrée, une taille de fichier de sortie, et les modèles d'usage du CPU, la RAM, et la bande passante.
- **Détermination de la politique d'allocation de VMs :** cette étape consiste à choisir la politique d'allocation des VMs qui sera appliquée après un intervalle fixé au préalable. C'est dans cette phase que nous implémentons notre propre politique d'allocation des VMs basée sur les quatre phases du chapitre précédent : phase de surveillance, phase de pré-négociation, phase de négociation, et phase de sélection des VMs. Ainsi, le Broker de chaque datacenter fait appel à notre politique après chaque intervalle et sauvegarde les valeurs calculées des temps de réponse et de consommations d'énergie dans un vecteur, pour l'utiliser plus tard dans l'analyse des résultats.

5.4 Expérimentations et résultats

Afin d'évaluer l'approche proposée, nous avons réalisé trois séries d'expériences sur CloudSim. En supposant que l'application est déployée dans un seul datacenter. Nous nous concentrons, dans ces expériences, sur :

- L'impact du nombre de machines physiques (serveurs), des machines virtuelles et des cloudlets sur le temps de réponse et la consommation d'énergie.
- L'impact de l'utilisation du théorème de Coase sur la charge de migration (Nombre de machines virtuelles migrées), le temps de réponse et la consommation d'énergie.

Visant ces objectifs, trois approches ont été comparées :

1. **Approche NoSMP (Pas de "Stable Matching Problem") :** c'est une approche où il n y'a pas de migrations du tout.
2. **Approche SMP (Utilisation de "Stable Matching Problem") :** l'approche a été implémentée en excluant le théorème de Coase, c'est à dire, seulement, les trois premières phases de l'approche ont été implémentées sans prendre en compte la sélection des VMs.

3. **Approche *SMP-CT* (de "Stable Matching Problem" + "Coase Theorem") :** toutes les quatre étapes de l'approche proposée ont été implémentées.

Dans ce qui suit, nous calculons les gains ou les pertes apportés par l'approche proposée *SMP-CT* en termes de temps de réponse, de consommation d'énergie, et de charge de migration, donc les deux autres approches *NoSMP* et *SMP* seront comparées à *SMP-CT*. Soit *Val(SMP-CT)* la somme des valeurs relevées d'une série d'expérimentation (expérimentation sur le temps de réponse ou la consommation d'énergie) en utilisant l'approche *SMP-CT*, $Val(SMP)$ la somme des valeurs relevées d'une série d'expérimentation en utilisant l'approche *SMP*, et $Val(NoSMP)$ la somme des valeurs relevées d'une série d'expérimentation en utilisant l'approche *NoSMP*.
Si *Val(SMP-CT)* < $Val(SMP)$, nous calculons le gain par la formule 5.1 :

$$Gain = \left(1 - \left[\frac{Val(SMP\text{-}CT)}{Val(SMP)}\right]\right) * 100 \tag{5.1}$$

Si *Val(SMP-CT)* $\geq Val(SMP)$, nous calculons la perte par la formule 5.2 :

$$Perte = -\left(1 - \left[\frac{Val(SMP)}{Val(SMP\text{-}CT)}\right]\right) * 100 \tag{5.2}$$

Le gain et la perte du *SMP-CT* par rapport à *NoSMP* sont calculés de la même façon.

5.4.1 L'impact du nombre de machines physiques et de l'utilisation du théorème de Coase

Dans ces simulations, nous avons utilisé un datacenter qui contient 200 identiques machines virtuelles stockées dans un même ensemble de machines physiques identiques. La taille de l'ensemble change à chaque simulation. Les PMs varient entre 50 et 300. Le Broker du datacenter distribue un ensemble de 2000 cloudlets dans chaque simulation (Tableau 5.1)
Le tableau TABLE 5.2 représenté par la Figure 5.3 et le tableau TABLE 5.3 représenté

Paramètres	**Valeurs**
Nombre de datacenter	1
Nombre de PMs	[50, 100, 150, 200, 250, 300]
Nombre de VMs	200
Nombre de cloudlets	2000
Nombre de CPU par PM	2
Nombre de CPU par VM	1
RAM de PM	2 Go
RAM de VM	1 Go

TABLE 5.1 – *Les paramètres de simulation de la première série d'expérimentation*

par la Figure 5.4 montrent, respectivement, la comparaison des temps de réponse et de consommation d'énergie entre la stratégie sans migration, la stratégie proposée complète, et la stratégie de migration utilisant la correspondance stable sans le théorème de Coase.
Le tableau Table 5.4 schématisé par la Figure 5.5 montre l'impact du théorème de Coase sur le nombre de machines virtuelles migrées.

Comme illustré dans le tableau 5.2 et la Figure 5.3, les deux approches *SMP-CT* et *SMP* donnent de meilleurs résultats par rapport à l'approche *NoSMP*, en terme d temps

Nombre de PMs	50	100	150	200	250	300	Gain/Perte
NoSMP	0.48	0.439	0.31	0.68	0.62	0.4	24.001%
SMP-CT	0.425	0.432	0.12	0.568	0.516	0.165	
SMP	0.425	0.4333	0.17	0.61	0.56	0.22	7.94%

TABLE 5.2 – *Les résultats des temps de réponse de la première série d'expérimentation*

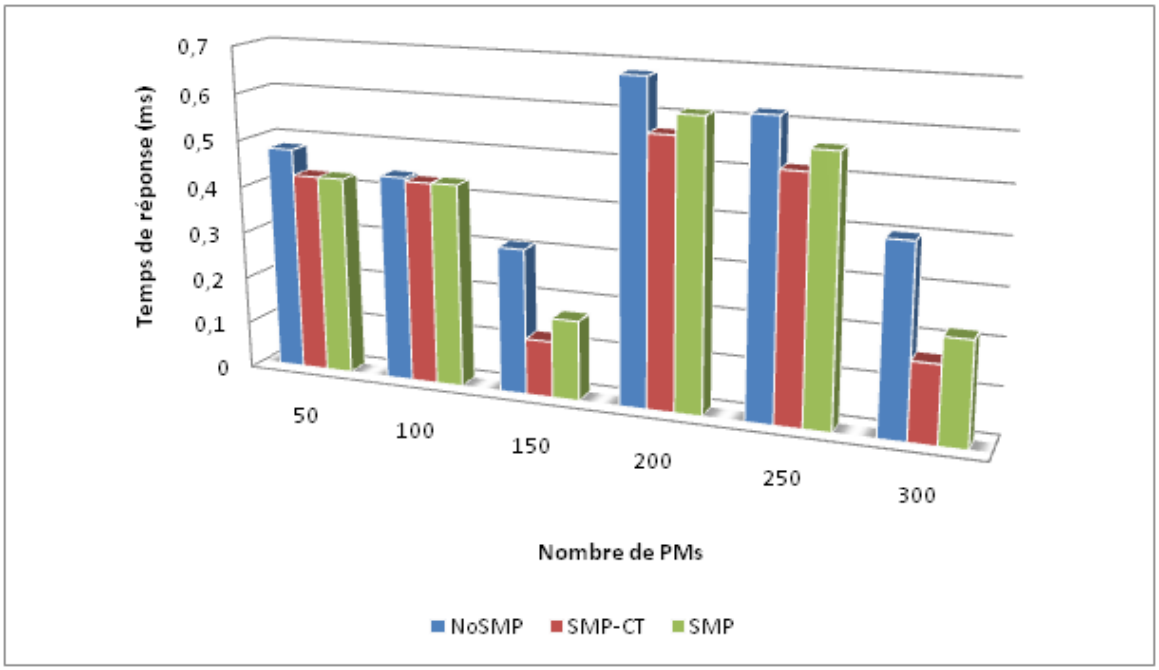


FIGURE 5.3 – *L'impact du nombre de PMs et du théorème de Coase sur le temps de réponse*

de réponse avec un gain moyen de 24%, et ceci grâce à la migration périodique des VMs vers des PMs sous-utilisées. L'utilisation du théorème de Coase permet de booster les résultats de 7.94% suite à la sélection optimal des machines virtuelles.

Nous remarquons des résultats du tableau 5.3 et de la Figure 5.4, que l'utilisation d'une stratégie de migration basée sur le problème de correspondance stable permet d'économiser plus de 14% d'énergie par rapport à une stratégie sans migration. Nous constatons, aussi, une perte de 3.30% dû à l'utilisation du théorème de Coase.

Le graphe de la Figure 5.4 peut être splitté en deux intervalles [50...150] où les approches *SMP-CT* et *SMP* donnent des excellents résultats, et]150...300] où leurs résultats commence à se dégrader jusqu'à ce que l'approche *NoSMP* soit la meilleure. cette divergence de résultats entre les deux intervalles est liés à la relation nombre de VMs, nombre de PMs et à l'effort de migration effectué, ces deux point seront discutés en détails dans

Nombre de PMs	50	100	150	200	250	300	Gain/Perte
NoSMP	9559.13	16853.34	18077.55	18274.53	23939.66	18286.17	14.39%
SMP-CT	2340.39	10001.68	9996.18	18237.26	22522	26818	
SMP	2456	9586.05	8388	17936.87	22462	26115	- 3.30%

TABLE 5.3 – *Les résultats de consommation d'énergie de la première série d'expérimentation*

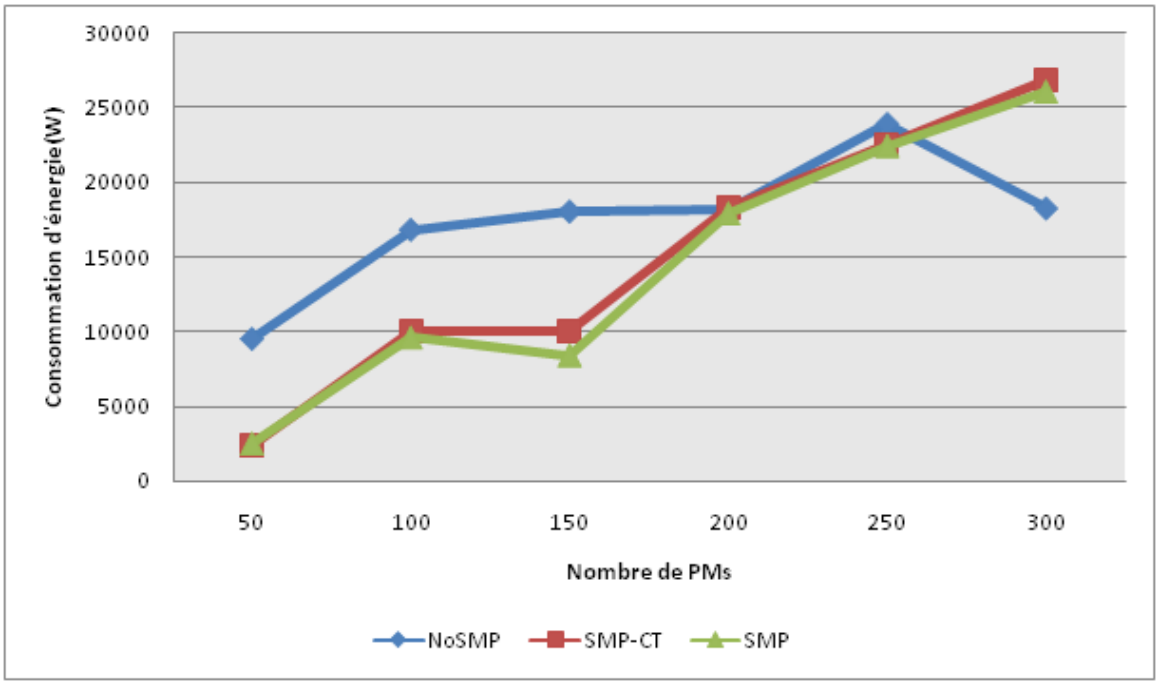


FIGURE 5.4 – *L'impact du nombre de PMs et le théorème de Coase sur la consommation d'énergie*

la section 5.4.4.

Nombre de PMs	**50**	**100**	**150**	**200**	**250**	**300**	**Gain/Perte**
SMP-CT	46	121	180	181	193	195	
SMP	50	131	186	191	193	195	3.17%

TABLE 5.4 – *Les résultats du nombre de VMs migrées de la première série d'expérimentation*

Le graphe de la Figure 5.5 montre que l'utilisation du théorème de Coase peut être très bénéfique pour le système. Puisque, en plus des améliorations apportées en termes de temps de réponses, il peut réduire l'échange d'informations dans le réseau et alléger le trafic en réduisant le nombre de machines virtuelles migrées.

5.4.2 L'impact du nombre de machines virtuelles et de l'utilisation du théorème de Coase

Ces expériences ont été réalisées avec 250 machines physiques, qui forment un seul datacenter, 2000 cloudlets, et un ensemble variable de machines virtuelles (Tableau 5.5) : Les résultats de cette deuxième série d'expérimentations du temps de réponse, de consommation d'énergie, et de charge de migration sont présentés, respectivement, dans les Figures 5.6, 5.7, 5.8.
Les résultats de cette deuxième série d'expérimentations montrent que :

- Environ 17% du temps de réponse peut être gagné en appliquant l'approche *SMP-CT* en comparaison avec l'approche *NoSMP*.
- Le théorème de Coase peut améliorer les performances du système de 25.94%.

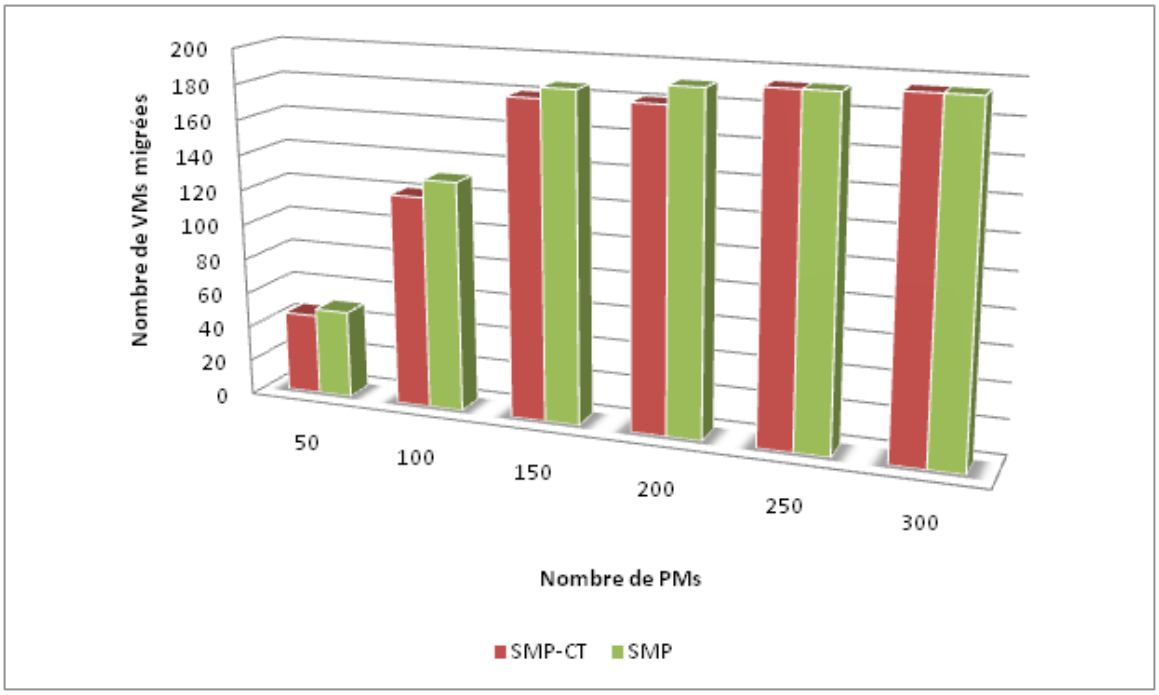


FIGURE 5.5 – *L'impact du nombre de PMs et le théorème de Coase sur le nombre de VMs migrées*

Paramètres	**Valeurs**
Nombre de datacenter	1
Nombre de PMs	250
Nombre de VMs	[100, 200, 300, 400, 500, 600]
Nombre de cloudlets	1000
Nombre de CPU par PM	2
Nombre de CPU par VM	1
RAM de PM	2 Go
RAM de VM	1 Go

TABLE 5.5 – *Les paramètres de simulation de la deuxième série d'expérimentation*

Nombre de VMs	**50**	**100**	**150**	**200**	**250**	**300**	**Gain/Perte**
NoSMP	0.1	0.2	0.29	0.62	0.47	0.46	17.29%
SMP-CT	0.23	0.51	0.21	0.18	0.28	0.36	
SMP	0.23	0.51	0.26	0.52	0.44	0.43	25.94%

TABLE 5.6 – *Les résultats des temps de réponse de la seconde série d'expérimentation*

Nombre de PMs	**100**	**200**	**300**	**400**	**500**	**600**	**Gain/Perte**
NoSMP	9073.018	18269	27995.2	35180.61	43549.91	45902.41	33.97%
SMP-CT	22036.5	22522	17407.93	22324.33	17820.04	16716.66	
SMP	22036.5	22522	17949.17	22324.33	18724.42	19656.94	3.56%

TABLE 5.7 – *Les résultats de consommation d'énergie de la seconde série d'expérimentation*

- L'approche *SMP-CT* est bien meilleure, en général, par rapport à l'approche sans migration. cet avantage est estimé à 33.97%.

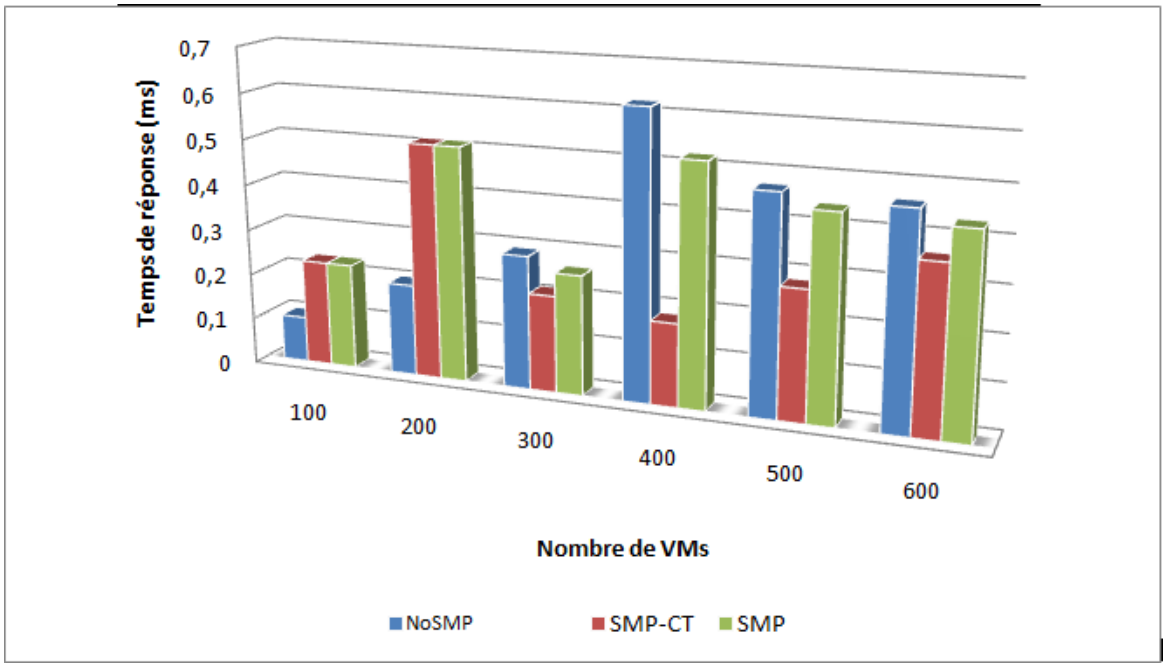


FIGURE 5.6 – *L'impact du nombre de VM et le théorème de Coase sur le temps de réponse*

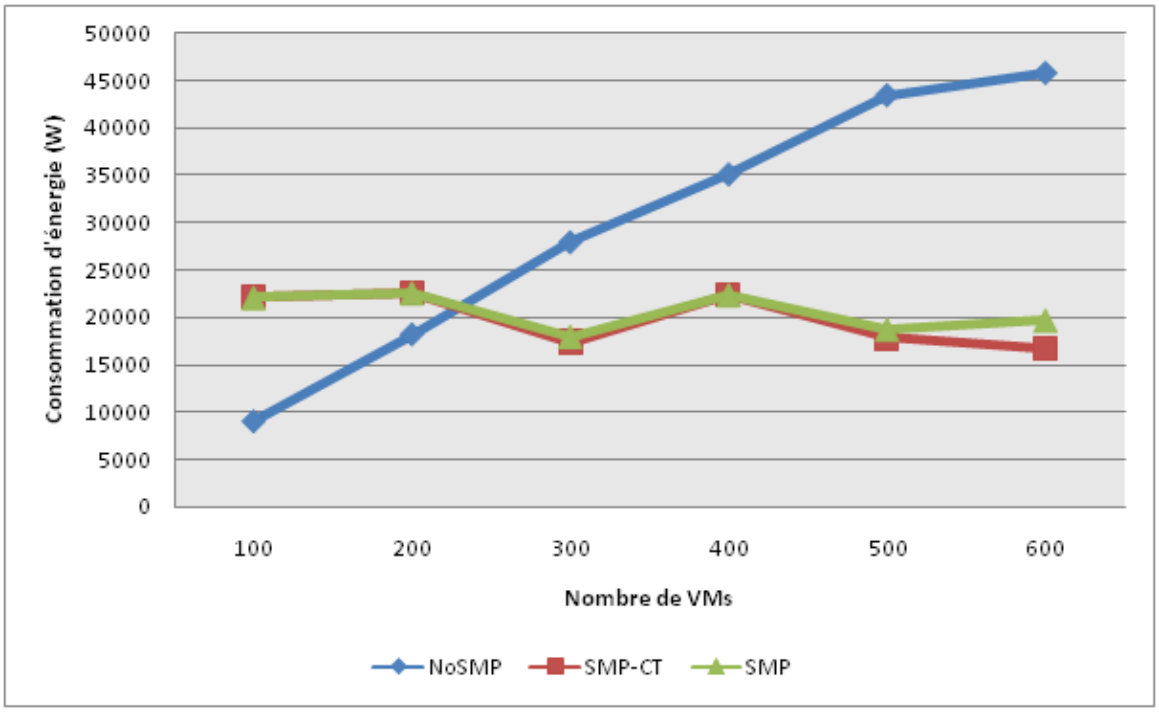


FIGURE 5.7 – *L'impact du nombre de VM et le théorème de Coase sur la consommation d'énergie*

- L'utilisation du théorème de Coase aide à économiser d'avantage d'énergie. Le tableau 5.7 représenté par la Figure 5.7 estime que *SMP-CT* est bien meilleure, point de vue énergie, que l'approche *SMP* de 3.56%.

Nous remarquons aussi que les figures 5.6 et 5.7 peuvent êre divisé en deux intervalles : [100...200] et [300...600]. Dans le premier, la migration est considérée comme un inconvénient pour le système, tandis que dans le deuxième, l'approche proposée a réalisé des gains considérables en termes de temps de réponse et de consommation d'énergie (Voir section 5.4.4).

Nombre de PMs	100	200	300	400	500	600	Gain/Perte
SMP-CT	100	193	286	360	362	287	
SMP	100	193	292	360	442	335	7.78%

TABLE 5.8 – *Les résultats du nombre de VMs migrées de la deuxième série d'expérimentation*

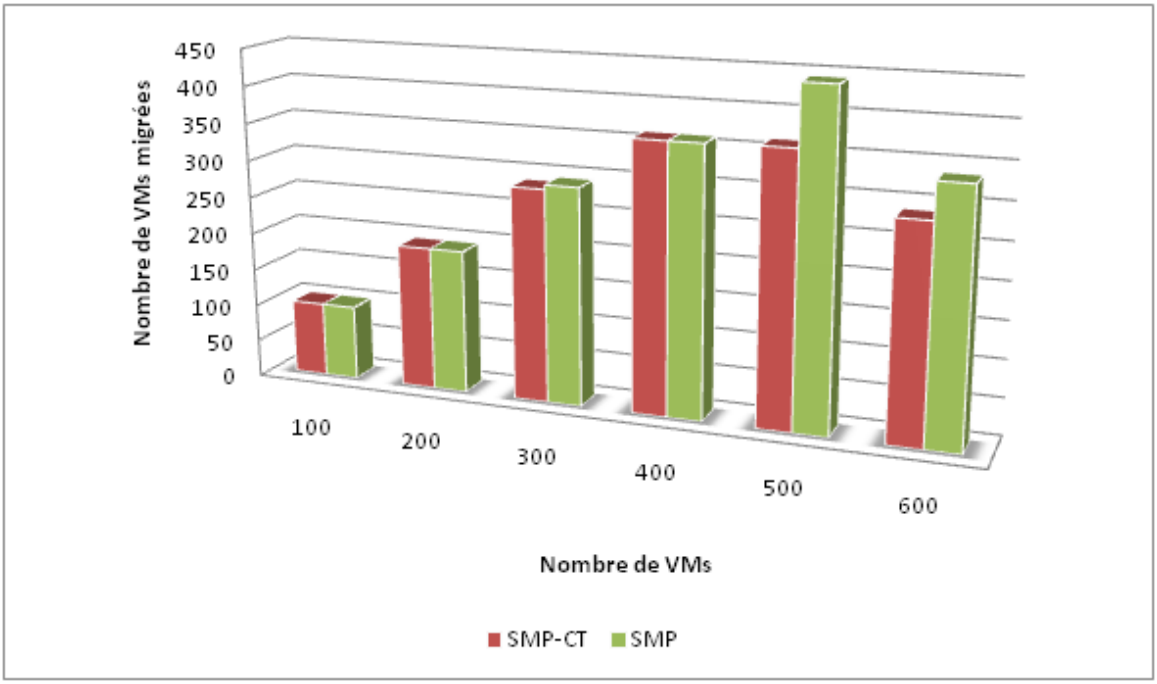


FIGURE 5.8 – *L'impact du nombre de VMs et le théorème de Coase sur le nombre de VMs migrées*

- la charge de migration peut être diminuée de 7.78% en utilisant le théorème de Coase.

5.4.3 L'impact du nombre de cloudlets

Pour cette série d'expérimentation, le datacenter se compose de 100 machines virtuelles réparties sur 60 machines physiques identiques, et un ensemble de cloudlets, comme indiqué dans le tableau Table 5.9 :

Paramètres	Valeurs
Nombre de datacenter	1
Nombre de PMs	60
Nombre de VMs	100
Nombre de cloudlets	[500, 1000, 1500, 2000, 2500, 3000]
Nombre de CPU par PM	2
Nombre de CPU par VM	1
RAM de PM	2 Go
RAM de VM	1 Go

TABLE 5.9 – *Les paramètres de simulation de la troisième série d'expérimentation*

Les résultats de simulation sont présentés dans le tableau Table 5.10, la Figure 5.9 et la Figure 5.10. Les résultats montrent que le nombre de cloudlets n'a aucun impact sur le

temps de réponse et la consommation d'énergie. Dans tous les cas, notre approche apporte les meilleurs résultats.

Nombre de cloudlets	**500**	**1000**	**1500**	**2000**	**2500**	**3000**
NoSMP	0.4	0.39	0.39	0.39	0.39	0.39
SMP	0.21	0.21	0.22	0.33	0.23	0.30

TABLE 5.10 – *Les résultats des temps de réponse de la troisième série d'expérimentation*

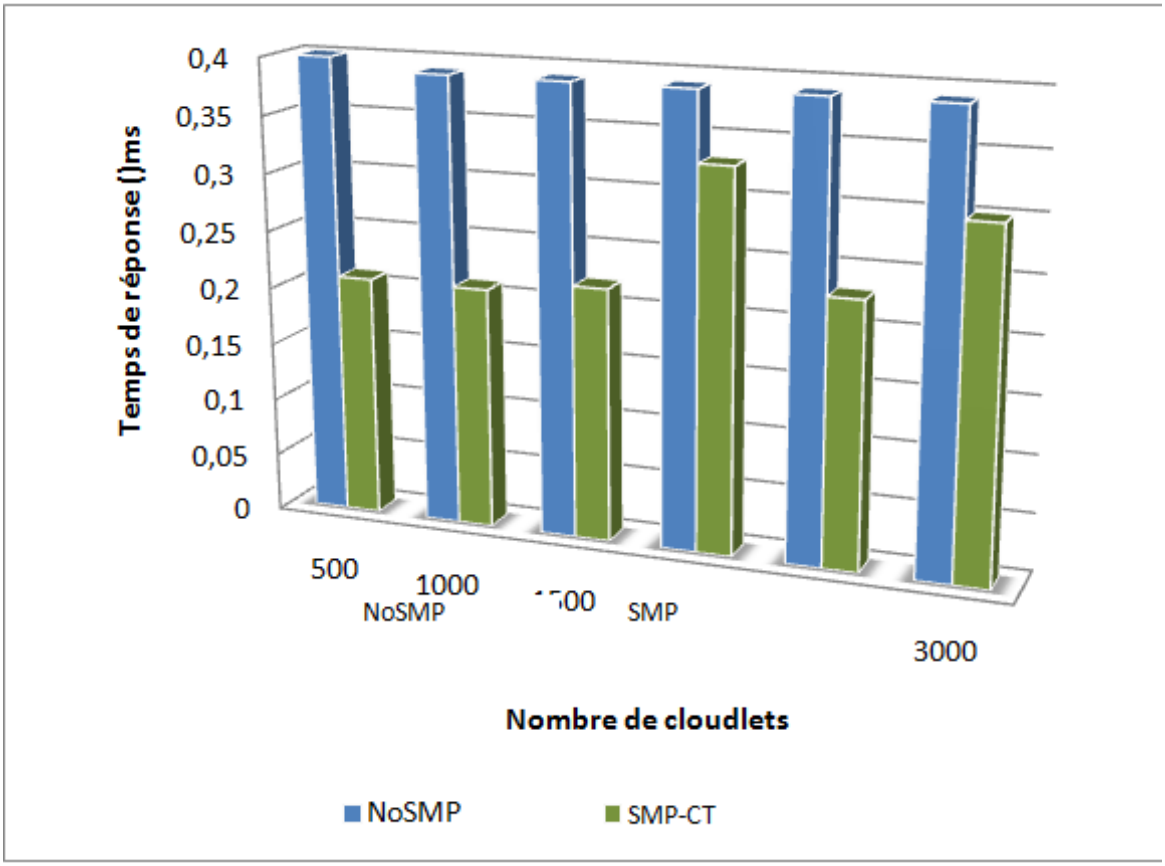


FIGURE 5.9 – *L'impact du nombre de cloudlets sur le temps de réponse*

5.4.4 Discussion

De ces trois séries de simulation, nous pouvons conclure que les seuls paramètres qui ont un impact réel sur le temps de réponse et la consommation d'énergie sont le nombre de PMs et le nombre de VM. Dans cette section, nous allons discuter les raisons de l'impact du nombre de VMs et le nombre de PMs sur l'énergie calculée et le temps de réponse.

Nous remarquons à partir de ces séries de simulation précédentes et leurs résultats, concernant la consommation d'énergie, que les meilleurs résultats sont fournis lorsque le nombre de PMs est compris entre 30% du nombre VMs < nombre de PMs < 70% du nombre VMs, pourquoi ?

C'est à cause du comportement de notre approche. Le modèle proposé doit calculer périodiquement deux seuils (de consommation d'énergie et de temps de réponse).

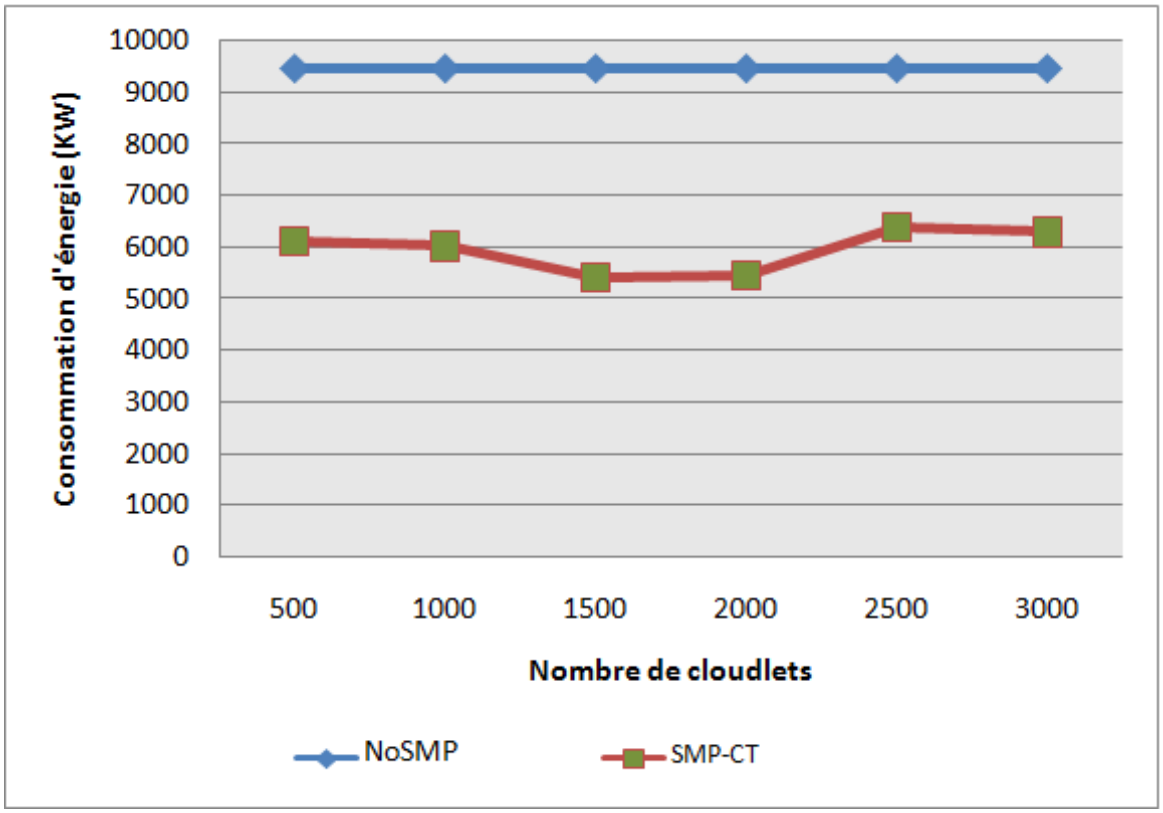


FIGURE 5.10 – *L'impact du nombre du cloudlets sur la consommation d'énergie.*

Lorsque le nombre de PMs ne respecte pas cette condition (30% du nombre de VMs < nombre du PMs < 70% de nombre de VMs) l'un des deux seuils sera inutile, comment ?

1. Si nombre de PMs < 30% de nombre VMs, les deux seuils seront dépassés constamment, donc le modèle proposé va migrer des VMs à chaque période. La migration ici va devenir gênante car les PMs seront tout le temps pleinement utilisées, ce qui entraînera à la fois une dégradation des performances constante et une hausse dans la consommation d'énergie (Figures 5.7 et 5.6 Le premier intervalle).

2. 2. Dans le second cas, où le nombre de PMs > 70% nombre de VMs. Le seuil de consommation d'énergie ne sera pratiquement jamais dépassé, et par conséquent, l'approche proposée va favoriser le cas où le seuil de temps de réponse est dépassé en migrant les VMs vers des différentes PMs afin d'améliorer le temps de réponse. Ce qui impliquera une absence de migration après un certain temps puisque les VMs sont dispatchées sur toutes les PMs et donc notre approche sera presque identique à une approche sans migration mais le nombre de machines physiques actifs est plus grand qu'une approche sans migration à cause des premières migrations (le deuxième intervalle de la Figure 5.4).

5.5 CONCLUSION

Dans ce chapitre, nous avons présenté le simulateur CloudSim utilisé pour implémenter l'approche proposée ainsi que les résultats obtenus des simulations. Aussi, nous avons réalisé plusieurs séries d'expérimentations dans le but de comparer la stratégie

proposée qui implémente les deux théorèmes économiques : le problème de correspondance stable et le théorème de Coase, avec deux autres stratégies de migration : la stratégie qui utilise seulement le problème de correspondance stable sans le théorème de Coase et une stratégie qui ne fait pas de migration, tout en variant différents paramètres comme : le nombre de machines physiques, le nombre de VMs, et le nombre de cloudlets, ainsi que le nombre de cycles. Les résultats de la comparaison ont montré que notre approche de migration permet, généralement, de réduire : la consommation d'énergie, et d'améliorer le temps de réponse dans l'environnement de Cloud Computing.

Nous avons aussi remarqué de toutes les expérimentations que l'utilisation du théorème de Coase permet d'améliorer considérablement la qualité de service (amélioration de 7.94% dans la première série d'expérimentations et 25.94% dans la deuxième), ainsi de réduire la charge de migration (un gain de 3.17% et 7.78%, dans respectivement, la première et la deuxième série d'expérimentations). Cependant, dans certains rares cas, il provoque une hausse de consommation d'énergie (Figure 5.4) même si cette hausse est généralement négligeable.

Conclusion générale

Le Cloud Computing est un nouveau paradigme qui consiste à proposer des ressources informatiques sous forme de services à la demande accessibles de n'importe où, n'importe quand et par n'importe qui. Ce paradigme s'appuie principalement sur la virtualisation qui permet de faire fonctionner sur une seule machine physique plusieurs systèmes d'exploitation (machines virtuelles), isolés les uns des autres. La croissance rapide de la demande de puissance de calcul par des entreprises, des applications web et des scientifiques a conduit à la création de nouveaux grands datacenters à grande échelle pour satisfaire la demande et fournir une bonne qualité de service. Cependant, ces datacenters consomment d'énormes quantités d'énergie électrique.

Nous avons présenté dans cette thèse, un algorithme de correspondance stable pour la migration des machines virtuelles dans une infrastructure Cloud. L'approche proposée a pour objectif d'améliorer l'économie d'énergie sans dégrader la qualité de services (QoS). Nous avons proposé également un algorithme pour le problème de polarisation de correspondance stable. En outre, nous avons utilisé le théorème de Coase pour réduire le coût de migration en déterminant le nombre de machines virtuelles à faire migrer et ainsi améliorer les performances du système. Les résultats montrent que notre approche agit indépendamment de la charge de travail. Aussi, les expériences montrent qu'avec un nombre adéquat de machines pysiques et de machines virtuelles, la stratégie proposée réduit considérablement la consommation d'énergie et le temps de réponse tout en minimisant la quantité des informations échangées dans le réseau.

Malgré les résultats prometteurs obtenus par notre contribution, il y a un certain nombre de défis de recherche ouverts qui peuvent être abordés afin d'affiner les stratégies de réduction de consommation d'énergie et d'amélioration de qualité de service. De ce fait, nous proposons :

1. d'affiner l'approche proposée en introduisant les contraintes du contrat SLA, ainsi que les pénalités résultantes de sa violation.
2. d'améliorer l'algorithme de polarisation de correspondance stable.
3. d'intégrer d'autres paramètres dans le calcul de la consommation d'énergie comme : l'utilisation des disques durs, la RAM, la bande passante, et les systèmes de refroidissement, pour des résultats plus pertinents.
4. d'étudier la question d'une approche hybride avec des heuristiques en les intégrant dans le processus de négociation, puisque le problème d'allocation des VMs est un problème d'affectation.
5. d'utiliser d'autres stratégies économiques basées sur des fonctions objectives pour quantifier les avantages et les inconvénients d'une situation donnée, plutôt que le processus de négociation utilisés par le "Stable Matching". Parmi ces stratégie, nous

étudions : CAPM (Capital Asset Pricing Model). CAPM permet de donner une évaluation de l'efficacité du coût d'un actif sur le marché, en fonction des risques, et en se basant sur cette rentabilité, nous déterminerons la valeur de l'actif. Nous pouvons considérer l'efficacité de l'actif comme la qualité de service et les risques comme la consommation d'énergie et d'appliquer une migration des VMs pour maintenir la rentabilité du système (actif).

6. d'implémenter notre approche dans une plate-forme de Cloud open source tels que Eucalyptus ou Open Stack.

Bibliographie

Accenture et WSP. Cloud computing and sustainability : The environmental benefits of moving to the cloud. Rapport technique, Accenture and "WSP Environment and Energy", 2010. http://www.accenture.com/sitecollectiondocuments/pdf/accenture_sustainability_cloud_computing_theenvironmentalbenefitsofmovingtothecloud.pdf. (Cité pages vii et 9.)

S. Akiyama, T. Hirofuchi, R. Takano, et S. Honiden. Miyakodori : A memory reusing mechanism for dynamic vm consolidation. Dans *In Proceedings of the 2012 IEEE Fifth International Conference on Cloud Computing (CLOUD'12)*, pages 606–613, Juin 2012. (Cité pages vii, 37, 38 et 43.)

ASHRAE. Real-time energy consumption measurements in data centres. Rapport technique, ASHRAE American Society of Heating, Refrigerating and Air-Conditioning Engineers, Janvier 2010. (Cité pages vii et 5.)

P. Barham, B. Dragovic, S. Hand K. Fraser, A. Ho T. Harris, R. Neugebauer, I. Pratt, et A. Warfield. Xen and the art of virtualization. Dans *in Proceedings of the 19th ACM Symposium on Operating Systems Principles (SOSP '03)*, volume 37, pages 164–177, New York, NY, USA, Décembre 2003. (Cité pages 1 et 18.)

L. André Barroso et U. Hölzle. The case for energy-proportional computing. *IEEE Computer*, 40 :33–27, Décembre 2007. (Cité page 11.)

A. Beloglazov. *Energy-Efficient Management of Virtual Machines in Data Centers for Cloud Computing*. PhD thesis, Department of Computing and Information System, THE UNIVERSITY OF MELBOURNE, Février 2013. (Cité pages vii, 7, 8, 9, 12, 13, 33 et 34.)

A. Beloglazov, J. Abawajy, et R. Buyya. Energy-aware resource allocation heuristics for efficient management of datacenters for cloud computing. *Future Generation Computer Systems*, 28(5) :755–768, Mai 2012. (Cité page 1.)

A. Beloglazov et R. Buyya. Energy efficient resource management in virtualized cloud datacenters. Dans *In Proceedings of the 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing (CCGRID '10)*, pages 826–831, Mai 2010. (Cité pages 39, 40 et 43.)

A. Beloglazov et R. Buyya. Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers. *Journal of Concurrency and Computation : Practice and Experience*, 24 (13) :1397–1420, Septembre 2012. (Cité pages 40 et 43.)

F. Berthoud. Environnement des serveurs : datacentres, Sep 2011. http://ecoinfo.cnrs.fr/IMG/pdf/Environnement_des_serveurs.pdf. (Cité pages vii et 6.)

S. BOSE, A. PASALA, D. RAMANUJAM, S. MURTHY, et G. MALAIYANDISAMY. Sla management in cloud computing : A service provider's perspective. Dans *Cloud Computing : Principles and Paradigms*, pages 427–431. John Wiley & Sons, Janvier 2011. (Cité pages 14 et 26.)

Rudi Bruchez. *Les bases de données NoSQL et le Big Data*. Eyrolles, Avril 2015. (Cité pages 28 et 29.)

R. Buyya, J. Broberg, et Andrzej M. Goscinski. *Cloud Computing : Principles and Paradigms*. Wiley Publishing, Mars 2011. (Cité page 27.)

R. Buyya, C.S. Yeo, S. Venugopal, J. Broberg, et I. Brandic. Cloud computing and emerging it platforms : Vision, hype, and reality for delivering computing as the 5th utility. *Future Generation Computer Systems*, 25(6) :599–616, Juin 2009. (Cité page 1.)

R. N. Calheiros, R. Ranjan, A. Beloglazov, César A. F. De Rose, et R. Buyya. A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Journal of Software-Practice and Experience*, 41(1) :23–50, Janvier 2011. (Cité pages viii, 74, 77, 78, 79 et 80.)

carms. Canadian resident matching service, Avril 2015. `http://www.carms.ca/`. (Cité page 47.)

C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul, C. Limpach, I. Pratt, , et A. Warfield. Live migration of virtual machines. Dans *In Proceedings of the annual conference on USENIX Annual Technical Conference (ATEC'05)*, pages 25–25, 2005a. (Cité pages vii, 26, 27, 30 et 51.)

C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul, C. Limpach, I. Pratt, , et A. Warfield. Live migration of virtual machines. Dans *In Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation (NSDI'05)*, volume 2, pages 273–286, 2005b. (Cité pages 26, 30 et 51.)

W. S. Cleveland. Robust locally weighted regression and smoothing scatterplots. *Journal of the American Statistical Association*, 74 :829–836, 1979. (Cité page 41.)

R. H. Coase. The problem of social cost. *Journal of Law And Economics, III*, pages 1–44, Octobre 1960. (Cité pages 3, 42 et 57.)

B. Colin et J. Desbureaux. Les différents types de virtualisation, 2009. `http://wapiti.telecom-lille1.eu/commun/ens/peda/options/st/rio/pub/exposes/exposesrio2009/COLIN-DESBUREAUX/La-Virtualisation-Diff%C3%A9rents-types-de-virtualisation.html`. (Cité pages vii et 22.)

Nobel Prize Committee. Alvin e. roth and lloyd s. shapley : Stable matching : Theory, evidence, and practical design. Rapport technique, Nobel Prize Committee, Oct 2012. `http://fr.slideshare.net/Corporate_Technologies/vmware-versus-hyperv1102`. (Cité pages 45, 46 et 47.)

G. COOK. How clean is your cloud ? Rapport technique, Greenpeace International, Apr 2012. `http://greenpeace.org`. (Cité page 2.)

P. DELFORGE. Americas data centers are wasting huge amounts of energy : Critical action needed to save money and cut pollution. Rapport technique, Natural Resources Defense Council, Aug 2014. `http://www.nrdc.org/energy/data-center-efficiency-assessment.asp`. (Cité page 1.)

A. Desai, R. Oza, P. Sharma, et B. Patel. Hypervisor : A survey on concepts and taxonomy. *International Journal of Innovative Technology and Exploring Engineering (IJITEE)*, 2(3) :222–225, Février 2013. (Cité page 22.)

S. Devadas et S. Malik. A survey of optimization techniques targeting low power vlsi circuits. Dans *In Proceedings of the 32nd annual ACM/IEEE Design Automation Conference (DAC'95)*, pages 242–247, 1995. (Cité page 33.)

V. Dinesh. Supporting service level agreements on ip networks. Dans *In Proceedings of IEEE/IFIP Network Operations and Management Symposium*, volume 29, pages 1382–1388, 2004. (Cité page 14.)

B. DuCharme. *The Operating System Handbook or, Fake Your Way Through Minis and Mainframes*. IBM, 2001. (Cité page 18.)

ExpertGlossary. hypervisor, 2015. `http://www.expertglossary.com/virtualization/definition/hypervisor`. (Cité page 22.)

X. Fan, W. D.Weber, , et L. A. Barroso. Power provisioning for a warehouse-sized computer. Dans *in Proceedings of the 34th Annual International Symposium on Computer Architecture (ISCA'07)*, volume 35, pages 13–23, Mai 2007a. (Cité pages vii, 9 et 10.)

X. Fan, W. D. Weber, et L. A. Barroso. Power provisioning for a warehouse sized computer. Dans *In Proceedings of the 34th annual international symposium on Computer architecture (ISCA '07)*, volume 35, pages 13–23, Mai 2007b. (Cité page 64.)

S. Frolund et J.O. Koistinen. A language for quality of service specification. Rapport technique, HP Labs, 1998. `http://www.hpl.hp.com/techreports/98/HPL-98-10.html`. (Cité page 14.)

D. Gale et L. S. Shapley. College admissions and the stability of marriage. *The American Mathematical Monthly*, 69(1) :9–15, 1962. (Cité pages 2, 41, 46, 48, 49, 50, 51, 52 et 54.)

P.B. Galvin. Vmware vsphere vs. microsoft hyper-v : A technical analysis. Rapport technique, CTI strategy, 2009. `http://fr.slideshare.net/Corporate_Technologies/vmware-versus-hyperv1102`. (Cité page 24.)

Inc Gartner. Gartner estimates ict industry accounts for 2 percent of global co2 emissions, Avril 2007. `http://www.gartner.com/newsroom/id/503867`. (Cité pages 5 et 13.)

Gurobi. Gurobi optimization, 2011. `http://www.gurobi.com`. (Cité page 26.)

D. Gusfield et R. W. Irving. *The Stable Marriage Problem : Structure and Algorithms*. MIT Press, 1989. (Cité pages 2, 41, 48, 52 et 56.)

M. M. Halldòrsson, R. W. Irving, K. Iwama, D. F. Manlove, S. Miyazaki, Y. Morita, , et S. Scott. Approximability results for stable marriage problems with ties. *Theoretical Computer Science*, 306 :431–447, Septembre 2003. (Cité pages 2 et 56.)

J. Hamiltoni. Cooperative expendable micro-slice servers (cems) : Low cost, low power servers for internet-scale services. Dans *in Proceedings of 4th Biennial Conference on Innovative Date Systems Research (CIDR), Asilomar, California, USA*, Janvier 2009. (Cité page 6.)

J. HORALEK, P. SUBA, et M. HATAS. Comparison of technologies for software virtualization. Dans *In Proceedings of AIASABEBI'11 Proceedings of the 11th WSEAS international conference on Applied informatics and communications, and Proceedings of the 4th WSEAS International conference on Biomedical electronics and biomedical informatics, and Proceedings of the international conference on Computational engineering in systems applications*, pages 160–165, 2011. (Cité page 24.)

P.J. Huber, J. Wiley, et W. InterScience. *Robust statistics*. Wiley New York, 1981. (Cité page 41.)

IPETI. Définition de langage java and java script, Janvier 2015. `http://ipeti.forumpro.fr/t21-definition-de-langage-java-java-script`. (Cité page 76.)

András Erik Csallner Iván Devosa. *Introduction to Java Programming Language*. JGYF Press, University of Szeged, 2010. (Cité page 76.)

K. Iwama, S. Miyazaki, , et H. Yanagisawa. Approximation algorithms for the sex-equal stable marriage problem. Dans *In Proceedings of the 10th International Workshop on Algorithms and Data Structures (WADS 2007)*, pages 201–213, Août 2007. (Cité pages 2 et 56.)

K. Iwama et S. Miyazaki. A survey of the stable marriage problem and its variants. Dans *In Proceedings of the International Conference on Informatics Education and Research for Knowledge-Circulating Society (icks 2008)*, pages 131–136, Janvier 2008. (Cité pages 47 et 56.)

jrmp. Japan residency matching program, Avril 2015. `http://www.jrmp.jp/`. (Cité page 47.)

G. Kecskemeti, P. Kacsuk, T. Delaitre, , et G. Terstyanszky. Virtual appliances : A way to provide automatic service deployment. Dans *Remote Instrumentation and Virtual Laboratories, Service Architecture and Networking*, pages 67–77. Springer US, 2010. (Cité page 18.)

G. Kecskemeti, G. Terstyanszky, P. Kacsuk, , et Z. Nemèth. An approach for virtual appliance distribution for service deployment. Dans *Future Generation Computer Systems*, volume 27, pages 280–289, Mar 2011. (Cité pages 18 et 29.)

A. Kella et G. Belalem. A stable matching algorithm for vm migration to improve energy consumption and qos in cloud infrastructures. *IJCAC : International Journal of Cloud Applications and Computing*, 4(2) :15–33, 2014. URL `http://dx.doi.org/10.4018/ijcac.2014040102`. (Cité page 43.)

Y. Kessaci, N. Melab, et E. Talbi. Optimisation multi-critère pour l'allocation de ressources sur clouds distribués avec prise en compte de l'énergie. Dans *Rencontres Scientifiques France Grilles 2011, inria*, Septembre 2011. (Cité pages 5 et 6.)

Richard Knight. Al roth : An economist who saves lives. *BBC Magazine*, Octobre 2012. (Cité page 47.)

J.G. Koomey. Estimating total power consumption by servers in the us and the world. Rapport technique, Lawrence Berkeley National Laboratory and Consulting Professor, Stanford University, Février 2007. (Cité page 5.)

J.G. Koomey. Growth in data center electricity use 2005 to 2010. Rapport technique, Analytics Press, Stanford University, Août 2011. `http://www.analyticspress.com/datacenters.html`. (Cité page 6.)

Y. Kouki, T. Ledoux, D. Serrano, P. Sens, J. Lejeune, L. Arantes, et S. Bouchenak. Sla et qualité de service pour le cloud computing. Dans *Conférence d'informatique en Parallélisme, Architecture et Système (ComPAS 2013)*, pages 1–11, Janvier 2013. (Cité page 13.)

D. Kusic, Jeffrey O. Kephart, J. E. Hanson, N. Kandasamy, et G. Jiang. Power and performance management of virtualized computing environments via look ahead control. Dans *In Proceedings of International Conference on Autonomic Computing(ICAC '2008)*, volume 35, pages 3–12, Juin 2008. (Cité page 64.)

L. T. Lee, K. Y. Liu, et H. Y. Dynamic resource management for energy saving in the cloud computing environment. Dans *In Proceedings of the International Conference on Information Science and Industrial Applications*, pages 176–181, Mai 2012. (Cité pages 34, 39 et 43.)

K. LEFEVRE. La virtualisation, Avril 2014. `http://blog.vsense.fr/2014/04/la-virtualisation/`. (Cité pages vii, 20, 21 et 30.)

B. Li, J. Li, J. Huai, T. Wo, Q. Li, et L. Zhong. Enacloud : An energy saving application live placement approach for cloud computing environment. Dans *In Proceedings of IEEE International Conference on Cloud Computing (CLOUD'09)*, pages 17–24, Septembre 2009. (Cité pages vii, 37, 38 et 43.)

Keqin Li. Scheduling precedence constrained tasks with reduced processor energy on multiprocessor computers. *TRANSACTIONS ON COMPUTERS*, 61(12) :1668–1681, Décembre 2012. (Cité page 5.)

W. Linlin et R. Buyya. Service level agreement (sla) in utility computing systems. *CoRR : Computing Research Repository*, Septembre 2010. (Cité pages ix et 14.)

H. Liu, H. Jin, X. Liao, Liting Hu, et C. Yu. live migration of virtual machine based on full system trace and replay. Dans *In Proceedings of the 18th ACM international symposium on High performance distributed computing (HPDC'09)*, pages 101–110, Juin 2009. (Cité pages vii, 35, 36 et 43.)

E. Mackaay et S. Rousseau. *Analyse économique du droit (2e édition)*. Éditions Thémis, Mars 2008. (Cité page 57.)

Pieter-Jan Maenhaut. User data management in multi-tenant cloud environments, Juin 2014. (Cité page 28.)

D. Marshall. The rise of virtual systems and virtualization (on-line), Juin 2011. `http://www.virtualizationdefrag.com/articles/history-of-virtualization/index.php`. (Cité pages 19 et 30.)

R. Martínez, J. Massó, A. Neme, et J. Oviedo. An algorithm to compute the full set of many-to-many stable matchings. *Mathematical Social Sciences*, 47(2) :187–2105, Mars 2004. (Cité pages 2, 3, 41, 42, 51, 56 et 70.)

D. G. McVitie et Leslie B. Wilson. The stable marriage problem. *Communications of the ACM*, 14(7) :486–490, Juillet 1971. (Cité page 56.)

R. H. Michael, D. Umesh, et G. Kartik. Post-copy live migration of virtual machines. *ACM SIGOPS Operating Systems Review*, 43(3) :101–110, Juillet 2009. (Cité pages vii, 36, 37 et 43.)

L. Minas et B. Ellison. *Energy Efficiency for Information Technology : How to Reduce Power Consumption in Servers and Data Centers*. Intel Press, 2009. ISBN 9781934053201. URL http://books.google.dz/books?id=qmM7QQAACAAJ. (Cité page 8.)

Keisuke Muda. A strategy for virtual machine migration based on resource utilization. Master's thesis, School of Media and Governance, Keio University, 2010. (Cité pages 24, 28, 29 et 30.)

Newsroom. Cloud service level agreement : Standardisation guidelines. Rapport technique, Newsroom Editor, Juin 2014. http://ec.europa.eu/information_society/newsroom/cf/dae/document.cfm?action=display&doc_id=6138. (Cité page 14.)

Nobelprize.org. Ronald h. coase : The sveriges riksbank prize in economic sciences in memory of alfred nobel 1991, Mai 2015. http://www.nobelprize.org/nobel_prizes/economic-sciences/laureates/1991/. (Cité page 57.)

courses Northeastern University. The stable matching problem. Rapport technique, Northeastern University, College of Computer and Information Science, jan 2015. http://www.ccs.neu.edu/course/cs1800/15S/handouts/StableMatching/stable.pdf. (Cité page 53.)

PLANETLAB. Comon, Avr 2015a. http://comon.cs.princeton.edu. (Cité page 11.)

PLANETLAB. Planetlab, Avr 2015b. http://www.planet-lab.org/. (Cité page 11.)

J. Pouwelse, Langendoen K., et Sips H. Dynamic voltage on a low-power microprocessor system. Dans *In Proceedings of the 7th annual international conference on Mobile computing and networking (MobiCom' 01)*, pages 251–259, 2001. (Cité pages 8, 34 et 39.)

H.D. Richa et N. Purohit. Energy efficient dynamic integration of thresholds for migration at cloud data centers. *Special Issue of International Journal of Computer Applications (0975 - 8887) on Communication and Networks*, comnetcn(1) :44–49, Décembre 2011. (Cité page 64.)

S. Ron et P. Aliko. Service level agreements. Rapport technique, Internet NG project, 2001. http://ing.ctit.utwente.nl/WU2/. (Cité page 14.)

Alvin E Roth. Stability and polarization of interests in job matching. *Econometrica*, 52(1) : 47–57, Janvier 1984. (Cité pages 2, 41, 46 et 47.)

Alvin E. Roth et E. Peranson. The redesign of the matching market for american physicians : Some engineering aspects of economic design. *American Economic Review*, 89(4) : 748–780, Septembre 1999. (Cité page 46.)

Alvin E. Roth et M. Sotomayor. The college admissions problem revisited. *Econometrica*, 57 :559–570, 1989. (Cité page 46.)

Alvin E. Roth et John H. Vande Vate. Random paths to stability in two-sided matching. *Econometrica*, 58(6) :1475–1480, Novembre 1990. (Cité page 46.)

Andrew L. Schlafly. The coase theorem : The greatest economic insight of the 20th century. *Journal of American Physicians and Surgeons*, 12(2), Juin 2007. (Cité page 57.)

B. Shappl et F. X. Durandy. Efficacité énergétique des technologies et infrastructures dans les datacentres et salles serveurs. Rapport technique, consortium projet PrimeEnergyIT, Jul 2011. www.efficient-datacenters.eu. (Cité pages vii et 23.)

P. Singh, B. Kumar Pandey, et H. L. Mandoria. Review of energy aware policies for cloud computing environment. *OPEN JOURNAL OF MOBILE COMPUTING AND CLOUD COMPUTING*, 1(1) :26–34, Août 2014. (Cité pages vii et 33.)

SPO. Scottish prho allocations, Avril 2015. http://www.dcs.gla.ac.uk/%CB%9Crwi/SPA.html. (Cité page 47.)

V. Venkatachalam et M. Franz. Power reduction techniques for microprocessor systems. *ACM Computing Surveys (CSUR)*, 37(3) :195–237, Septembre 2005. (Cité pages 5, 8 et 33.)

Thayer Watkins. Illustration du théorème de coase, Mai 2015. http://www.applet-magic.com/coasetheoremf.htm. (Cité page 57.)

Y. Wu et M. Zhao. Performance modeling of virtual machine live migration. Dans *in Proceedings of the 2011 IEEE 4th International Conference on Cloud Computing (CLOUD '11)*, volume 46, pages 492–499, Washington, USA, Juillet 2011. (Cité page 18.)

Li Wubin. Virtual machine placement in cloud environments, Mai 2012. (Cité page 30.)

Xen. How does xen work ? v 1.0. Rapport technique, Xen, Décembre 2009. http://fr.slideshare.net/Corporate_Technologies/vmware-versus-hyperv1102. (Cité page 24.)

H. Xu et B. Li. Egalitarian stable matching for vm migration in cloud computing. Dans *Computer Communications Workshops (INFOCOM WKSHPS), 2011 IEEE Conference*, pages 631–636, Shanghai, Avril 2011. (Cité pages 41 et 43.)

Liste des travaux

1- Communications et publications scientifiques

Communications :

- Journées Doctorales du Laboratoire d'informatique d'Oran, 2013 :

Approche économique pour la migration de machines virtuelles afin d'améliorer la consommation d'énergie et la qualité de service dans le cloud computing

KELLA Abdelaziz
Département d'informatique
Université d'Oran
Oran, Algérie
Kaa.kella.abdelaziz@gmail.com

GHALEM Belalem
Département d'informatique
Université d'Oran
Oran, Algérie
ghalem.belalem@univ-oran.dz

Résumé— Le cloud computing est un nouveau paradigme qui consiste à proposer des ressources informatiques sous forme de services à la demande accessibles de n'importe où, n'importe quand et par n'importe qui. Ce paradigme s'appuie principalement sur la virtualisation qui permet de faire fonctionner sur une seule machine physique (PM) plusieurs systèmes d'exploitation (ou machines virtuelles VM), isolés les uns des autres. La croissance rapide de la demande de puissance de calcul par des entreprises, des applications web et des scientifiques a conduit à la création de nouveaux grands datacenters à grande échelle consommant d'énormes quantités d'énergie électrique. Nous proposons dans cet article une stratégie de migration à chaud des VMs, basée sur une superposition de deux théorèmes économiques. Le théorème de mariage stable cherche à trouver une correspondance stable et équitable entre les exigences des utilisateurs (qualité de service, temps de réponse, SLA ...etc.) et celles des fournisseurs (Consommation d'énergie etcon l'équilibrage de charge, et la charge de migration), le théorème de Coase choisira, par la suite, la distribution optimale des VMs sur les PMs selon des critères spécifiés au préalable. La stratégie proposée a pour objectif de réduire la consommation d'énergie dans une infrastructure cloud sans affecter la qualité de service.

Mot clé— Cloud computing, virtualisation, migration à chaud, machine virtuelle, correspondance stable, problème de mariage stable, théorème de Coase, consommation d'énergie, qualité de service.

I. INTRODUCTION

Cloud computing est considéré comme un nouveau paradigme pour l'approvisionnement dynamique de ressources informatiques pris en charge par les datacenters qui emploient habituellement des machines virtuelles (VMs) à des fins de consolidation et d'isolation [1] [2]. Le Cloud computing offre une infrastructure, plate-forme, et des logiciels (applications) comme des services qui sont mis à la disposition des utilisateurs sous un modèle pay-as-you-go. Ces services sont désignés, respectivement, comme Infrastructure as a Service (IaaS), Platform as a Service (PaaS), et Software as a Service (SaaS). Ces services [3] ne sont pas seulement utilisés mais aussi installés, déployés ou répliqués à l'aide de la virtualisation. La virtualisation est une technique qui permet à plusieurs systèmes d'exploitation de fonctionner simultanément sur une seule machine physique. Elle est devenue l'un des aspects essentiels dans les serveurs et datacenters modernes en raison de plusieurs avantages, tels que le partage souple et efficace des ressources, la tolérance aux pannes, la portabilité et la rentabilité [1]. En outre, les plates-formes de virtualisation prennent en charge le redimensionnement des capacités des machines virtuelles VMs, en temps réel, pour s'adapter à l'évolution des charges de travail ainsi que la capacité de migrer des machines virtuelles d'une machine physique à une autre de manière transparentes [23]

La migration à chaud des machines virtuelles est l'une des principaux caractéristiques de la virtualisation, qui permet de migrer des services et des applications d'une machine physique (PM) à une autre de manière transparente. La migration à chaud permet, en outre, de migrer des VMs presque sans interruption de service. Cette technique est un moyen important pour la gestion des services dans les grands systèmes virtualisés tel que le Cloud Computing. Elle permet aussi de déplacer dynamiquement des services et des applications de façon dynamique et transparente pour améliorer les performances et la fiabilité [4]

L'utilisation de la virtualisation combinée à la migration à chaud des machines virtuelles, en conformité avec les exigences des utilisateurs, peut résoudre plusieurs problèmes de calculs et de gestions de ressources. Cependant, La croissance rapidede la demande depuissance de calculpar desentreprises, des applications web et des scientifiques, conduit les fournisseurs du cloud àla création de nouveaux grands datacenters à grande échelleconsommant d'énormes quantités d'énergie électrique pour répondre aux besoins de leurs utilisateurs. Pour résoudre ce problème, de nombreux travaux ont été proposées pour réduire la consommation d'énergie des datacenters. Ces recherches ont porté sur la consolidation des ressources, l'exploitation complètes des machines physiques (PMs), et éteindre les PMs inactif. Les travaux proposés ont apporté des avantages considérables dans le contexte de la consommation d'énergie, mais de l'autre côté, ils affectent, négativement, les performances des systèmes. Par conséquent, la façon d'économiser l'énergie sans affecter les performances des systèmes (qualité de service) est l'un des problèmes les plus courants

- The 4^{th} International Conference on Cloud Computing and Services Science, CLOSER 2014 :

VM Live Migration Algorithm Based on Stable Matching Model to Improve Energy Consumption and Quality of Service

Abdelaziz Kella and Ghalem Belalem
Department of Computer Science, University of Oran, Oran, Algeria
kaa.kella.abdelaziz@gmail.com, ghalem1.dz@gmail.com

Keywords: Energy Efficiency, Quality of Service, Cloud Computing, Datacenter, Energy Consumption, Stable Matching Problem, the Coase Theorem, VM Migration, Live Migration.

Abstract: Cloud Infrastructure is a newly emerged computing platform that builds on the latest achievements of diverse research areas, such as Grid computing, Service-oriented computing, business process management and virtualization. The use of server virtualization techniques for such platforms provides great flexibility with the ability to allocate users and applications per virtual machine, ability to consolidate several virtual machines (VMs) on the same physical machine (PM), to resize a virtual machine as needed and being able to support migration of virtual machines across physical machines (PMs) to meet users and system requirements. However, datacenters hosting Cloud applications consume huge amounts of electrical energy, contributing to high operational costs for the providers' services as well as for the users' services. This paper presents a live migration strategy based on economic model named "stable marriage problem" to find a fair stable matching between VMs' requirements and PMs' requirements. The proposed approach aims to improve the quality of service while reducing the energy consumption. We refine our approach by integrating the Coase Theorem to reduce the number of migrating VMs and maximize the system's performances. Preliminary experiments show that, with a suitable number of PMs and VMs, the strategy can reduce the energy consumption of Physical machines and datacenter while improving the quality of service.

1 INTRODUCTION

Cloud computing (Richa et al., 2011) (Buyya et al., 2011) is modeled to provide utility-based IT services. Services like computation, software, data access and storage are provided to its users without its knowledge about physical location and configuration of the server that is providing the services. To guarantee these specifications, large-scale datacenters are established. These datacenters usually employ Virtual Machines (VMs) technologies for consolidation and environment isolation purposes (Beloglazov et al., 2012) (Barham et al., 2003). The Cloud services are, generally, referred to Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS) respectively.These services (Buyya et al., 2011) are not only used but also installed, deployed or replicated with the help of virtualization.

Virtualization is a key technology that has enabled several operating systems to run simultaneously on a single physical machine.Hardware virtualization provides a logical separation between applications and the underlying physical machine resources, thereby enabling a flexible mapping of virtual machines to Physical Machines (PMs) in a datacenter. Further, virtual machine platforms support resizing of VM containers to accommodate changing workloads as well as the ability to live-migrate virtual machines from one physical machine to another without incurring application downtimes (Wood et al., 2011).

VM migration is one of the important capabilities of system virtualization, which allows applications to be transparently migrated along with their execution environments across PMs. Live migration further allows the VM to be migrated almost without any interrupt to its application's execution. VM migration is an important means for managing applications and resources in a large virtualized system. It enables resource usage to be dynamically balanced in the entire virtualized system across physical host boundaries. It, also, allows applications to be dynamically relocated to

Publications :

- International Journal of Cloud Applications and Computing, IJCAC 2014 :

International Journal of Cloud Applications and Computing, 4(2), 15-33, April-June 2014 15

A Stable Matching Algorithm for VM Migration to Improve Energy Consumption and QOS in Cloud Infrastructures

Abdelaziz Kella, Department of Computer Science, Faculty of Exact and Applied Sciences, University of Oran, Oran, Algeria

Ghalem Belalem, Department of Computer Science, Faculty of Exact and Applied Sciences, University of Oran, Oran, Algeria

ABSTRACT

Cloud Computing is one of the fast spreading technologies for providing utility-based IT services to its users. Large-scale virtualized datacenters are established in order to provide these services. Based on a pay-as-you-go model, it enables hosting of pervasive applications from consumer, scientific, and business domains. However, datacenters hosting Cloud applications consume huge amounts of electrical energy, contributing to high operational cost for the service providers as well as for the service users. Energy consumption can be reduced by live migration of virtual machines (VM) as required and by switching off idle physical machines (PM). Therefore, we propose an approach that finds a stable matching fair to both VMs and PMs, to improve the energy consumption without affecting the quality of service, instead of favoring either side because of a deferred acceptance procedure. The approach presumes two dynamics thresholds, and prepares those virtual machines on the physical machines that the load is over one of the two presumed values to be migrated. Before migrating all those VMs, we use the Coase theorem to determine the number of VMs to migrate for optimal costs. Our approach aims to improve energy consumption of the datacenters, while delivering an expected Quality of Service.

Keywords: *Cloud Computing, Coase Theorem, Datacenter, Energy Consumption, Energy Efficiency, Live Migration, Quality of Service, Stable Matching Problem, VM Migration*

1. INTRODUCTION

Cloud computing can be classified as a new paradigm for the dynamic provisioning of computing services supported by datacenters that usually employ Virtual Machine (VM) technologies for consolidation and environment isolation purposes (Barham et al., 2003; Beloglazov et al., 2012). Cloud computing delivers an infrastructure, platform, and software (applications) as services that are made available to users in a pay-as-you-go model. These services are re-

DOI: 10.4018/ijcac.2014040102

2- Encadrement

الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et Populaire
وزارة التعليم العالي و البحث العلمي
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
جامعة وهران Université d'Oran
كلية العلوم الدقيقة و التطبيقية
Faculté des Sciences Exactes et Appliquées
قسم الإعلام الآلي
Département d'Informatique

Mémoire de Fin d'Etudes

Pour l'Obtention du Diplôme de
Master en Informatique

Présenté par :
GUELLIL CHAHRAZED
BELALIA SOUMIA

Domaine : Mathématiques & Informatique
Spécialité : *Technologies et Systèmes Informatiques*

Session Juin 2013

THEME

VERS UNE GESTION DE MIGRATION DES DONNEES DANS LE CLOUD COMPUTING POUR AMELIORER LA QUALITE DE SERVICE

Encadré par : **G.Belalem**
Co-encadré par : **A.Kella**

Jury

Président : **L.Loukil**
Examinatrice : **A.Deba**
Examinatrice : **N.Hadi**

Code Master : 03-TSI/2013

Promotion 2012/2013