



Approche prédictive de l'efficacité énergétique dans les Clouds Datacenters

Fréjus A. Roméo Gbaguidi

► To cite this version:

Fréjus A. Roméo Gbaguidi. Approche prédictive de l'efficacité énergétique dans les Clouds Datacenters. Réseau de neurones [cs.NE]. Conservatoire national des arts et métiers - CNAM; Université d'Abomey-Calavi UAC (Bénin), 2017. Français. ⟨NNT : 2017CNAM1163⟩. ⟨tel-01830891⟩

HAL Id: tel-01830891

<https://theses.hal.science/tel-01830891v1>

Submitted on 5 Jul 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



HAL Authorization



le cnam

Ecole doctorale Informatique, Télécommunications et Electronique (Paris)

Centre d'études et de Recherche en Informatique et Communications

Institut de Mathématiques et de Sciences Physiques (Dangbo)

Laboratoire de Réseautique et de Systèmes d'information

THESE DE DOCTORAT

présentée par : Fréjus A. Roméo GBAGUIDI

soutenue le : **22 Décembre 2017**

*pour obtenir le grade de : Docteur du Conservatoire National des Arts et Métiers et
de l'Université d'Abomey-Calavi*

Spécialité : **Informatique**

Approche Prédictive de l'Efficacité Energétique dans les Clouds
Datacenters

THESE dirigée par

BOUMERDASSI Selma
EZIN C. Eugène

*Maitre de Conférences, HDR, Cnam Paris
Maitre de Conférences, UAC*

RAPPORTEURS

HAMAMACHE Kheddouci
KORA Ahmed

Professeur des Universités, Université de Lyon 1
Maitre de Conférences, ESMT, Dakar

PRESIDENT

OTHMAN Jalel Ben

Professeur des Universités, Université Paris 13

EXAMINATEUR

MILOCCO Ruben

Professeur des Universités, Université Nationale de Comahue

INVITE

RENAULT Eric

Maitre de Conférences, HDR, Télécom Sud Paris

Abstract

With the democratization of digital technologies, the construction of a globalized cyberspace insidiously transforms our lifestyle. Connect more than 4 billion people at high speed, requires the design of new concepts of service provision and traffic management that are capable to face the challenges. For that purpose, cloud computing have been set up to enable Datacenters to provide part or total IT components needed by companies for timely services delivering with performance that meets the requirements of their clients. Furthermore, the proliferation of datacenters around the world has brought to light the worrying question about the amount of energy needed for their function and the resulting difficulty for the humanity, whose current reserves are not extensible indefinitely. It was therefore necessary to develop techniques that reduce the power consumption of datacenters by minimizing the energy loss orchestrated on servers where each wasted watt results in a chain effect on a substantial increase in the overall bill of datacenters. Our work consisted first in making a review of the literature on the subject and then testing the ability of some prediction tools to improve the anticipation of the risks of energy loss caused by the misallocation of virtual equipment on servers.

This study focused particularly on the AutoRegressive Moving Average (ARMA) tools and neural networks which in the literature have produced interesting results in related fields. After this step, it appeared to us that ARMA tools, although having less performance than neural networks in our context, runs faster and are best suited to be implemented in cloud computing environments. Thus, we used the results of this method to improve the decision-making process, notably for the proactive re-allocation of virtual equipment before it leads to under-consumption of resources on physical servers or over-consumption

ABSTRACT

inducing breaches of Service Level Agreements (SLA). Based on our simulations and results, this approach enabled us to reduce energy consumption on a firm of 800 servers over a period of one day by more than 5Kwh. This gain could be significant when considering the enormous size of modern datacenters and projected over a relatively long period of time. It would be even more interesting to deepen this research in order to generalize the integration of this predictive approach into existing techniques in order to significantly optimize the energy consumption within datacenters while preserving performance and quality of service which are key requirements in the concept of Cloud Computing

Keywords : Cloud Computing, Energy, Prediction, Neural networks, ARMA.

Résumé

Avec la démocratisation des technologies du numérique, la construction d'un cyberspace globalisé s'est faite insidieusement, transformant littéralement notre mode de vie et notre vécu quotidien. Faire communiquer plus de 4 milliards d'individus à une vitesse devenue incontrôlable, nécessite la mise au point de nouveaux concepts pour la production des services informatiques capable de s'adapter à ce défi. Le cloud computing, dans cette optique permet de fournir à travers des datacenters, une partie ou la totalité des composants nécessaires aux entreprises pour la délivrance de leurs services dans les délais et avec des performances conformes aux exigences de leurs clients. Dès lors, la prolifération conséquente des datacenters dans le monde a mis au jour la préoccupante question de la quantité d'énergie nécessaire pour leur fonctionnement et la difficulté qui en résulte pour l'humanité dont les réserves actuelles ne sont pas extensibles à l'infini. Ainsi, il est apparu nécessaire de développer des techniques permettant de réduire la consommation électrique des datacenters en minimisant les pertes d'énergie orchestrées sur les serveurs dont le moindre watt gaspillé entraîne par effet de cascade une augmentation substantielle de la facture globale des datacenters.

Notre travail présente dans un premier temps une revue de la littérature sur le sujet, et à tester la capacité de quelques outils de prédiction afin d'améliorer l'anticipation des risques de pertes d'énergie engendrés par la mauvaise allocation des machines virtuelles sur les serveurs. Cette étude s'est focalisée notamment sur les outils ARMA (AutoRegressive Moving Average) et les réseaux de neurones qui dans la littérature ont produit des résultats intéressants dans des domaines proches. Après cette étape, il nous est apparu que les outils ARMA bien qu'ayant des performances inférieures aux réseaux de neurones dans notre

contexte, s'exécute dans plus rapidement et sont les plus adaptés pour être implémenté dans les environnements de *cloud computing*. Ainsi, nous avons utilisé les résultats de cette méthode pour améliorer le processus de prise de décision, notamment pour la re-allocation proactive des machines virtuelles avant qu'il n'entraîne des sous-consommations des ressources sur les serveurs physiques ou des surconsommations pouvant induire des violations des accords de niveaux de service. Cette démarche a permis sur la base de nos simulations de réduire de plus de 5Kwh la consommation d'énergie dans une ferme de 800 serveurs et sur une durée d'une journée. Ce gain pourrait se révéler important lorsque l'on considère la taille énorme des datacenters modernes et que l'on se projette dans une durée relativement longue. Il serait encore plus intéressant d'approfondir cette recherche afin de généraliser l'intégration de cette approche prédictive dans les techniques existantes afin d'optimiser de façon significative les consommations d'énergie au sein des datacenters tout en préservant les performances et la qualité de service indispensable dans le concept de cloud computing.

Mots clés : Cloud Computing, Energie, Prédiction, Réseaux de neurones, ARMA

Remerciements

A l'issue de cette belle aventure je tiens avant tout à rendre grâce à Dieu, le Tout Puissant par qui tout ceci fût possible et implorer son soutien pour d'autres défis toujours plus grands

Je ne trouverai jamais assez de mots pour remercier **Mme Selma BOUMERDASSI**, ma Directrice de thèse qui au delà de ce rôle déjà très difficile m'a servi de repère personnel, sachez que je n'oublierai jamais cette marque d'affection

A **M. Eugène C. EZIN**, mon co-Directeur de thèse, je voudrais témoigner toute ma reconnaissance pour sa disponibilité et son soutien inconditionnel

A tous **les membres du jury** qui ont accepté de consacrer une partie de leur précieux temps pour apporter un regard critique en vue de l'amélioration de ce travail, je tiens à vous dire sincèrement merci

A toi ma chère épouse, **Carine Ruthe ADJA**, et à toute ma famille, qui ont dû consentir d'énormes sacrifices pendant ces longues nuits et ces weekends interminables de travail. Ce succès est d'abord le vôtre

A mon feu père **Constant C. GBAGUIDI**, à ma mère **Victoire SOUDE** et à mes frères **Hubert** et **Wenceslas**, à qui j'ai fait la promesse de ne jamais m'arrêter, sachez

REMERCIEMENTS

que vous resterez à jamais mes premières sources d'inspiration

Je m'en voudrais de ne pas témoigner ma gratitude à **Mme Samia BOUZZE-FRANE** et **M. Eric RENAULT** pour leur soutien inconditionnel à chaque fois qu'il a été nécessaire

A tous mes amis et à toutes les personnes formidables qui m'ont accompagné par des mots d'encouragement, tout au long de cette aventure, sachez que tout cela m'a touché énormément.

Table des matières

INTRODUCTION GENERALE	21
I PARTIE 1 : ETAT DE L'ART	29
Résumé de la partie 1	31
1 TECHNIQUE D'OPTIMISATION DE L'ENERGIE AU SEIN DES DATACENTERS	33
Introduction	35
1.1 Approche globale d'optimisation énergétique	38
1.2 Module d'analyse	38
1.2.1 Les méthodes réactives	39
1.2.2 Les méthodes proactives	43
1.3 Module de contrôle des actions	45
1.3.1 Techniques matérielles	45
1.3.2 Techniques logicielles	47
Conclusion	49
2 CHOIX DES OUTILS ET DES DONNEES	51

TABLE DES MATIÈRES

Introduction	53
2.1 Choix des outils	54
2.1.1 Démarche d'évaluation des outils	54
2.1.2 Outils choisis	55
2.1.3 Outils de simulation des environnements de Clouds	56
2.2 Jeux de données de trafic au sein des datacenters	60
2.2.1 Données synthétiques	61
2.2.2 Traces réelles	61
2.3 Caractéristiques de Google cluster data 2011	66
2.3.1 Dataset description	67
2.3.2 Analyse des traces	69
Conclusion	75
 II PARTIE 2 : PREDICTIBILITE DES BESOINS EN RESSOURCES AU SEIN DES DATACENTERS	 77
 Résumé de la partie 2	 79
 3 PREDICTION DES BESOINS EN RESSOURCES BASEE SUR LES OUTILS STATISTIQUES	 81
Introduction	83
3.1 Modèle ARMA	84
3.1.1 Généralités	85
3.1.2 Les processus Autoregressifs (AR)	86
3.1.3 Les processus Moving Average (MA)	86

TABLE DES MATIÈRES

3.1.4	Les processus ARMA	86
3.1.5	Détermination des modèles ARMA	88
3.1.6	Evaluation des prédictions	90
3.2	Prédiction ARMA des consommations de ressources sur les serveurs	91
3.2.1	Présentation du problème	91
3.2.2	Détermination des paramètres du modèle ARMA des consommations énergétiques	92
3.2.3	Fenêtre de prédiction	93
3.3	Evaluation des performances	93
3.3.1	Simulations	93
3.3.2	Fiabilité des prédictions statistiques	95
Conclusion		99
4 PREDICTION BASEE SUR L'APPRENTISSAGE DES BESOINS EN RESSOURCES AU SEIN DES DATACENTERS		101
Introduction		103
4.1	Aperçu des réseaux de neurones	104
4.1.1	Perceptron multi-couches	106
4.2	Prédictions basées sur les réseaux de neurones	107
4.2.1	Méthodologie	107
4.2.2	Objectifs	107
4.3	Evaluation de la prédiction des ressources par les réseaux de neurones . . .	108
4.3.1	Jeux de données	108
4.3.2	Expérimentations	109
4.3.3	Choix de l'architecture des réseaux	112

TABLE DES MATIÈRES

4.3.4	Fiabilité des prédictions basées sur les réseaux de neurones	114
4.3.5	Analyse des résultats	116
4.4	Analyse comparative des prédictions statistiques et de l'apprentissage par ordinateur	118
4.4.1	Cadre de l'analyse	118
4.4.2	Résultats	120
Conclusion		123
 III PARTIE 3 : AMELIORATION DE L'ALLOCATION DES RESSOURCES AUX MACHINES VIRTUELLES		 125
Résumé de la partie 3		127
 5 PLACEMENT DES MACHINES DANS UN CONTEXTE D'EFFICACITE ENERGETIQUE		 129
Introduction		131
5.1	Problème de Bin Packing	133
5.1.1	Différents cas	133
5.1.2	Différents types de complexité	133
5.1.3	Différentes solutions	134
5.2	Placement initial des machines virtuelles dans le cloud computing	135
5.2.1	Problème de placement des machines virtuelles	135
5.2.2	Principe du placement initial	137
5.2.3	Approche de résolution	139
5.3	Simulation et résultats	140

TABLE DES MATIÈRES

5.3.1	Expérimentation	140
5.3.2	Résultats	141
Conclusion		143
6 CONSOLIDATION PREDICTIVE DES MACHINES VIRTUELLES		145
Introduction		147
6.1	Introduction	147
6.2	Principe de la consolidation des machines virtuelles	149
6.2.1	Cadre de l'étude	149
6.2.2	Contraintes de la consolidation	150
6.2.3	Etapas de consolidation	153
6.3	Intégration de la prédiction dans le processus de consolidation de VM . . .	155
6.3.1	Architecture du modèle	155
6.4	Application du modèle	157
6.4.1	Paramètres d'expérimentation	157
6.4.2	Résultats	158
Conclusion		163
CONCLUSION GENERALE		165
Annexes		177
A Annexe 1 :State of art of traffic prediction methods		177
A.1	Introduction	177
A.2	Generic prediction process scheme	178

TABLE DES MATIÈRES

A.3	Classification	179
A.3.1	Statistics methods	179
A.3.2	Computer learning methods	182
A.3.3	Hybrid methods	184
A.4	Type of data and measurement	185
A.5	Conclusion	185
B	Annexe 2 :Environnement Neurals Networks dans MATLAB	187
B.1	Présentation de l'environnement	187
B.2	Fonctions principales	188
B.3	188
C	Annexe 3 :Algorithme de prédiction ARMA	189
C.1	Code traitement ARMA dans MATLAB	189
D	Annexe 4 :Fichiers de politiques d'allocation de VM dans Cloudsim	191
D.1	PowerVmAllocationPolicyMigrationAbstract	191
D.2	PowerVmAllocationPolicyMigrationLocalRegression	214
D.3	PowerVmAllocationPolicyMigrationLocalRegression	220
	Index	225

Liste des tableaux

2.1	Taille des fichiers dans Google cluster	67
2.2	Taux de consommation sur six serveurs très sollicités du cluster	73
3.1	Erreurs de prédiction	96
4.1	Erreurs de prédiction	121
5.1	Rangement des serveurs par groupe de compatibilité	141
5.2	Performance des deux méthodes	142

LISTE DES TABLEAUX

Table des figures

1.1	Architecture en couche des systèmes d'optimisation énergétique	39
1.2	Classification des approches d'optimisation d'énergie dans les datacenters .	40
2.1	Architecture des couches dans CloudSim	58
2.2	Classes dans Cloudsim	59
2.3	Nombre de serveur par CPU et RAM	70
2.4	Serveur par CPU	71
2.5	Serveur par RAM	71
2.6	Task per event	72
2.7	Task per scheduling	72
2.8	Task per priority	73
2.9	Usage of processors	74
3.1	Schéma de glissement de fenêtre de prédiction	93
3.2	Prédiction des consommations de processeurs	94
3.3	Histogramme des erreurs de prédiction	94
3.4	Histogramme des erreurs de prédiction sur la fenêtre de 100 valeurs	95
3.5	Graphe de corrélation entre les valeurs de consommation de processeurs . .	96
4.1	Représentation d'un Neurone	105

TABLE DES FIGURES

4.2	Perceptron Multicouche	106
4.3	Consommation de CPU sur le serveur 1	108
4.4	Consommation de CPU sur le serveur 2	109
4.5	Consommation de CPU sur le serveur 3	110
4.6	Consommation de CPU sur le serveur 4	110
4.7	Consommation de CPU sur le serveur 5	111
4.8	Exemple de fichier d'entrée	111
4.9	Exemple de fichier de sortie	111
4.10	Algorithme de traitement des fichiers en entrée et en sortie des réseaux . .	112
4.11	Réseaux de neurones à 4 entrées a) avec une couche cachée b) avec deux couches cachées	113
4.12	Réseaux de neurones à 8 entrées a) avec une couche cachée b) avec deux couches cachées	113
4.13	Réseaux de neurones à 12 entrées a) avec une couche cachée b) avec deux couches cachées	114
4.14	Graphe de regression pour les réseaux à 4 entrées a) avec une couche cachée, b) avec deux couches cachées	114
4.15	Graphe de regression pour les réseaux à 8 entrées a) avec une couche cachée, b) avec deux couches cachées	115
4.16	Graphe de regression pour les réseaux à 12 entrées a) avec une couche cachée, b) avec deux couches cachées	115
4.17	MSE pour les réseaux à 4 entrées a) avec une couche cachée, b) avec deux couches cachées	116
4.18	MSE pour les réseaux à 8 entrées a) avec une couche cachée, b) avec deux couches cachées	117

TABLE DES FIGURES

4.19	MSE pour les réseaux à 12 entrées a) avec une couche cachée, b) avec deux couches cachées	117
5.1	Problème de placement de machines virtuelles	135
5.2	Notre approche de placement	137
5.3	Algorithme de placement	138
6.1	Scenario de sous consommation de ressources des serveurs	150
6.2	Scenario de surconsommation de ressources des serveurs	151
6.3	Processus classique de la migration de VM	155
6.4	Processus de migration de VM basée sur la prédiction	157
6.5	Energie consommée	159
6.6	Nombre de migration	160
6.7	Nombre de serveurs éteints	160
6.8	Taux de violation des SLA	161
A.1	Prediction process scheme	179
A.2	prediction methods classification	180
B.1	Initialisation de Matlab	187

TABLE DES FIGURES

Introduction

La marche du monde vers le tout numérique ne s'arrête pas. Tous les secteurs vitaux, de l'économie au sport en passant par la santé se métamorphosent jour après jour sur le socle commun des technologies informatiques. L'avènement de l'internet 2.0 et des terminaux mobiles ont déclenché une course effrénée vers l'information dont la ligne de fin semble désormais sans limite. A en croire le rapport 2017 du site We Are Social (wearesocial.com) [are social 2016], le taux de pénétration mondiale de l'accès à Internet a franchi la barre des 50% avec plus de 46% de personnes qui utilisent Internet à partir d'un terminal mobile soit plus de 3,4 milliards d'internautes mobile de par le monde. La progression est aussi spectaculaire d'une région à une autre [are social 2016] avec un essor remarquable sur le continent africain qui totaliserait maintenant plus de 362 millions d'utilisateurs d'internet sur une population dépassant les 1,2 milliards d'habitants et qui compte 7 des 10 pays où ce nombre augmente le plus rapidement. Le facteur qui sert de levier à ces prouesses reste la mobilité qui elle-même se nourrit des évolutions récentes des réseaux sociaux et des services en ligne. L'internet social en effet opère en parallèle sa petite révolution. Chaque minute dans le monde, ce sont 400 heures de vidéos qui sont téléchargées sur Youtube, 216 millions de photos « likées » sur Facebook, 830 000 fichiers téléchargés sur Dropbox et la liste des petites applications de partage et de socialisation sur la toile ne s'arrête pas. Au-delà des réseaux sociaux, tous les secteurs de l'économie poursuivent leur transformation digitale qui se manifeste par l'introduction des technologies dans les processus métiers pour l'amélioration de leur productivité. Le commerce électronique quant à lui connaît une assise confortable et tend à remplacer les modes traditionnels d'échange de biens et services. Selon les statistiques 2017 de l'Organisation Mondiale du Commerce,

le chiffre d'affaire du e-commerce a passé la barre des 2200 milliards de dollars américain et pourrait atteindre les 4500 milliards à l'horizon 2021 [OMC 2017] (eurostat, OMC). La télémédecine permet depuis plusieurs années maintenant, de mutualiser des expertises à travers le monde pour sauver des vies humaines en brisant les barrières de la présence physique du chirurgien qui peut diriger une opération médicale en Afrique depuis son bureau situé à l'autre bout du monde. L'internet des objets, les jeux vidéo et l'impression en 3D méritent également d'être cités parmi ces usages au sein d'une liste interminable, qui font de notre planète un endroit de plus en plus digital. La synergie sociale créée sur Internet a généré un écosystème de partage dans lequel la quantité de donnée produite est devenue effrayante. On estime maintenant à 44 milliards de TeraOctets, la quantité de données qu'il va falloir traiter d'ici à 2020 [Kellner 1995].

Afin de contenir cette évolution effrénée, les technologies de virtualisation et le Cloud Computing sont apparus et se sont adaptés au fil du temps pour faire face à la vélocité dans les échanges, aux quantités volumineuses des données produites et à la complexité de plus en plus grande des technologies à maîtriser pour avoir des systèmes d'informations à la fois performants et sécurisés. Partant jadis de l'informatique d'entreprise dans laquelle de nombreux serveurs et baies de stockage sont déployés pour répondre au besoin spécifique du business, le cloud computing introduit de nouveaux modes d'hébergement des systèmes d'informations en se basant sur la virtualisation de tous les composants et la rationalisation des ressources de traitement. En français "informatique dans les nuages", le concept [Plouin 2011] fait référence à la fourniture de tout ou partie des composants d'un système d'informatique afin de masquer pour les entreprises, la complexité liée à leur implémentation et la rationalisation des moyens nécessaires à leur mise en œuvre. Les trois modèles les plus populaires sont :

- l'Infrastructure as a Service (IaaS) qui consiste à fournir aux clients l'ensemble des composants physiques du SI tels que le réseau informatique, les serveurs, le stockage, etc.
- Platform as a Service (PaaS) qui revient à fournir en plus de l'infrastructure physique, un environnement système composé des systèmes d'exploitations, des

environnements de développement et d'hébergement des services

- Software as a Service (SaaS) qui revient à fournir en plus des deux précédents composants, les applications prêtes à l'emploi auxquelles peuvent accéder directement les clients. Dans ce même modèle, d'autres types de composants peuvent être fournis tels que la communication, l'impression, etc qui sont dans ce cas, sollicités à la demande

La fourniture des services cloud se base sur des datacenters, des salles serveurs géantes dans lesquelles l'on concentre des capacités d'hébergement relativement grandes ainsi que les conditions environnementales favorables au bon fonctionnement des équipements informatiques et par conséquent à la disponibilité des services finaux fournis aux utilisateurs. En concentrant dans des Datacenters uniques l'informatique de nombreuses entreprises des plus petites et moins sensibles aux plus grandes et plus exigeantes, le cloud computing s'impose désormais comme les usines de prédilection pour la production de l'informatique d'entreprise et la fourniture des services en ligne devenu le centre névralgique des affaires dans le domaine du numérique et au-delà.

Puisque chaque médaille a son revers, les nouveaux modes de production de l'informatique ne manquent pas d'effrayer de par les quantités astronomiques d'énergie électrique qu'ils nécessitent pour leur fonctionnement. Lorsque l'on considère que l'énergie consommée pour une minute de vidéo sur un smartphone est de 1wh, les 830 000 vidéos simultanées qui sont visionnées sur Youtube à travers le monde portent à plus de 1000 Mwh la facture d'électricité journalière correspondante supérieure à toute la consommation annuelle d'une ville de 200 000 habitants. La consommation des datacenters aux Etats Unis dépasse 90 Twh [ene] et pourrait rivaliser d'ici à 2020 avec l'énorme besoin de l'industrie aéronautique [Kellner 1995]. Cette évolution fait craindre à court terme de nombreuses difficultés énergétiques au niveau mondial si les mesures les plus idoines ne sont pas prises au plus tôt. Pour faire face à cet état de chose, différentes techniques sont développées pour soit rendre possible l'accès à de nouvelles sources d'énergie ou optimiser l'utilisation de l'existant. Dans le domaine du cloud computing, cette deuxième solution est la plus privilégiée notamment en développant des méthodes de fonctionnement des datacenters au juste coût énergétique. D'abord l'adaptation des composants électroniques des serveurs, l'efficacité

électrique de la climatisation et autres composants électroniques et ensuite des techniques logicielles tendant à optimiser de façon dynamique, le dimensionnement des ressources IT aux besoins de traitement du datacenter. Notre thèse se focalise sur ces techniques notamment en intégrant une approche anticipative de cette optimisation basée sur des outils de prédiction.

L'efficacité énergétique dans le contexte du cloud computer est l'ensemble des techniques permettant de tirer de chaque unité d'énergie fournie, l'équivalent en terme de traitement informatique. Autrement dit, elle consiste à minimiser la proportion non productive de l'énergie fournie à un serveur. Le concept prend tout son sens dans la démonstration des taux d'occupation généralement bas des ressources physiques des serveurs tandis qu'il mobilise une part importante de l'énergie qui leur est destinée dans un mode de fonctionnement maximal. Il est établi [Meisner et al. 2009], [Barroso and Hölzle 2007] qu'à partir de 20% de fonctionnement, les serveurs peuvent consommer jusqu'à 70% de leur pic d'énergie entraînant de facto 50% de perte d'énergie qui à l'échelle d'un Datacenters de plusieurs milliers de serveurs pourraient représenter une facture substantielle sur les charges d'exploitation et surtout une mauvaise nouvelle pour l'épuisement supposé ou réel des réserves énergétiques de l'humanité. L'efficacité énergétique est donc un ratio calculé entre la quantité d'énergie fournie à une unité de production (serveurs par exemple) et celle ayant effectivement servie pour exécuter un traitement utile. L'objectif étant de maximiser ce taux, de nombreuses recherches ont porté sur les moyens de son optimisation sur chaque composants des datacenters. Il est établi que chaque watt que l'on réussit à économiser sur un serveur fait baisser d'un watt l'énergie nécessaire pour le refroidissement, de 0,5 watt la consommation des composants télécoms et par effet de cascade, c'est l'équivalent de 3,5 watt par serveur qui pourrait être économisé à l'échelle d'un datacenter ce qui représente une énorme quantité dans notre contexte actuel de cloud computing. C'est donc un enjeu important que d'intensifier la recherche des techniques d'efficacité énergétique et de façon particulière même quelques petits watts que l'on réussira à soustraire de la facture électrique d'un serveur sera un pas considérable dans l'inversion de la tendance effrayante de surconsommation actuelle par les datacenters. Nous avons choisi d'apporter une modeste contribution à cet effort scientifique en nous focalisant sur l'impact que pourrait avoir

l'approche prédictive sur l'atteinte de l'objectif d'efficacité. Pour ce faire, nous utiliserons plusieurs outils afin de mesurer leur degré de performance et surtout quelles peuvent être les contraintes de leur implémentation en situation réelle. La pertinence de cette approche s'inscrit dans le postulat qu'une meilleure connaissance des dynamiques de consommation des unités de production des Datacenters pourrait améliorer le placement des machines virtuelles sur les serveurs physiques de façon à en tirer le meilleur rendement et par conséquent amoindrir les pertes énergétiques potentielles. Notre travail participe donc à élargir le champ de la recherche sur les moyens de l'efficacité énergétique en vérifiant l'apport de la prédiction dans l'amélioration des techniques courantes dans le contexte particulier et préoccupant du Cloud computing.

Avant d'aborder la question du cloud computing et de l'efficacité énergétique, il nous est paru nécessaire dans un premier temps de parcourir l'état de la littérature et de réaliser un positionnement de notre travail. Ainsi, nous avons fait un tour d'horizon des travaux précédents le nôtre qui révèle une abondance des recherches sur les sujets et un intérêt avéré de la communauté aux problématiques liées à l'énergie et à l'empreinte des technologies du numérique. Cette étude nous a permis d'identifier les différents aspects de la problématique tels que abordés par les scientifiques notamment l'amélioration apportée aux composants techniques dans les datacenters, la rationalisation de l'utilisation des serveurs, la maîtrise des types de trafic et l'adaptation des techniques d'ordonnancement des requêtes dans le contexte du cloud computing. Il nous a été donné de classer les techniques d'optimisation de l'énergie en deux groupes à savoir, les techniques réactives faisant référence à celles qui implémentent des méthodes déterministes avec peu d'adaptabilité à la dynamique des trafics au sein des Datacenters modernes et celles proactives qui rajoutent une couche d'intelligence basée sur l'apprentissage du trafic pour une meilleure contextualisation des méthodes d'efficacité énergétique. Notre étude nous a révélé en outre la nécessité de disposer de plus de données issues de traces réelles afin d'améliorer la compréhension des traitements dans les Datacenters et rendre les outils élaborés plus portable d'un environnement à l'autre. A ce sujet, les questions de protection des données à caractère personnel et de l'intelligence économique restreignent considérablement la quantité de traces réelles laissées dans le domaine public. A l'exception de rares collections

d'une pertinence significative comme le « google cluster trace » qui a servi de base à notre étude, la plupart des recherches se contente de générer des données de simulation à partir de modèles mathématiques réduisant ainsi l'immuabilité des résultats obtenus. Il ressort de tout ce qui précède que notre travail introduit dans la recherche non seulement la notion d'apprentissage préalable pour l'amélioration des techniques d'optimisation des ressources mais aussi l'utilisation des traces réelles pour la validité des résultats.

La seconde étape de notre thèse a consisté à établir la prédictibilité des traitements et des quantités de ressources à utiliser au sein des Datacenters. Etant donné la dimension prédictive que nous souhaitons approfondir en matière d'optimisation de la consommation énergétique des Datacenters, différents outils ont été testé notamment pour établir d'une part leur adéquation avec les types de données à analyser et d'autres parts la fiabilité de leur résultat par rapport aux techniques plus ou moins manuelles rencontrées dans la littérature. Ainsi, nous avons effectué deux séries d'études portant sur les méthodes statistiques de type ARMA et les méthodes d'intelligence artificielle notamment les réseaux de neurones et avons essayé de comparer les résultats obtenus. L'utilisation des méthodes statistiques, révèle qu'il est possible de prédire à court terme, avec une fiabilité de 30% comparé à une méthode de type naïve, les besoins futures de ressources sur les serveurs dans les Datacenters. En outre, le recours aux outils d'apprentissages tels que les réseaux de neurones fournissent des performances similaires lorsque les paramètres adéquats sont sélectionnés avec les fenêtres de données correspondantes. Les réseaux de neurones s'adaptent parfaitement au caractère dynamique des traitements sur les serveurs étant donné que pour la multitude de service fournis au sein des cloud Datacenters, il ne pourrait avoir de méthodes déterministes qui permettraient de modéliser tous les types de besoin en ressources. L'apprentissage fourni donc en plus des performances, la capacité de modélisation des besoins en ressources en rapport avec la dynamique des traitements eux-mêmes soumis à la diversité des services qui co-existent dans les environnements de cloud computing. De notre comparaison des deux méthodes, il ressort que le recours à la prédiction des besoins en consommation de ressources au sein des Datacenters pourrait faciliter l'optimisation de leur planification pour en tirer le meilleur rendement. Les méthodes statistiques de type ARMA, même si elles produisent des résultats satisfaisants en terme

de performance présentent des limites quant à la dynamicité des traitements comparées aux réseaux de neurones qui s'ajustent efficacement sur les changements brusques des besoins en ressources.

Dans une troisième phase, cette étude se focalise sur l'amélioration des techniques de consolidation des ressources virtuelles des serveurs en se basant d'une part sur la classification préalable des serveurs en fonction de leur profil de consommation et d'autres parts sur les prédictions des besoins en ressources issues des étapes d'analyses préalables. L'intérêt principal de cette thèse réside dans la démonstration que nous faisons de la possibilité d'optimiser la consommation de ressources virtuelles sur les serveurs en combinant plusieurs facteurs et en s'adaptant aux contraintes très variables des Datacenters. Nous abordons deux notions essentielles à savoir : la première concerne le placement initial des machines virtuelles sur les serveurs en considérant la nécessité de les classer préalablement suivant leur profil de consommation et afin de regrouper les machines virtuelles dont les besoins en ressources varieront de manière similaire au cours du temps. Nous implémentons une technique de placement initial basé sur l'algorithme du *Bin packing* et plus précisément le *First Fit Decreasing* permettant d'optimiser au départ l'occupation des serveurs physiques. Cet outil très utilisé pour des problèmes de placement d'objets dans d'autres domaines tels que le transport, la logistique et le commerce ont produit des résultats satisfaisants pour le placement des machines virtuelles en termes de gain de temps et d'efficacité en comparaison aux méthodes manuelles qui sont soumises au jugement personnel des ingénieurs. La deuxième notion abordée est la consolidation des machines virtuelles qui vise à minimiser à tout moment le nombre total de ressources virtuelles engagées au regard des traitements en cours au sein des Datacenters. La technique développée intègre les résultats de la prédiction basée sur les outils statistiques comme input de l'algorithme de migration des machines virtuelles en cas de surconsommation ou de sous-consommation de ressources. Les résultats obtenus sont satisfaisants. Ils montrent une économie d'énergie pouvant dépasser les 5 Kwh pour un parc de 800 serveurs avec des niveaux de violation des SLA plutôt acceptable et sur un temps de simulation de moins d'une journée. Ceci confirme que le recours aux approches prédictives est une piste très intéressante qu'il est pertinent d'explorer afin d'atteindre un niveau de rationalisation plus important de

l'énergie engagée dans le fonctionnement de l'informatique moderne.

A la fin de ce projet de thèse, nous sommes restés convaincu que la recherche dans le domaine de l'efficacité énergétique, spécifiquement dans le domaine du Cloud Computing est une nécessité absolue et qu'il faudra explorer encore plus de piste dans l'espoir d'inverser à court terme la progression dangereuse de la facture énergétique de la révolution digitale.

Première partie

PARTIE 1 : ETAT DE L'ART

Résumé de la partie 1

Cette partie est consacrée à la synthèse de notre revue de littérature sur les questions en rapport avec l'efficacité énergétique dans le contexte du cloud computing. Cette étape nous a permis d'affiner notre compréhension de la logique des techniques élaborées à travers les précédents travaux de recherche sur le sujet, de maîtriser les outils utilisés, de nous familiariser avec les types de données manipulées. Cette étude nous a permis d'obtenir les résultats ci-après :

L'architecture classique des méthodes d'optimisation de ressources se compose principalement d'un module d'analyse qui utilise des paramètres prédéfinis ou des données d'historiques afin de déterminer si une action correctrice peut être nécessaire, d'un module d'exécution effectif des actions correctrices en cas de nécessité. Ce dernier module peut être matériel ou logiciel.

Classification des méthodes d'optimisation des ressources : nous avons retenu un classement en deux catégories à savoir

- i) les méthodes réactives qui se compose des techniques qui surveille l'effectivité de la surconsommation ou de la sous-consommation des ressources avant de décider d'exécuter une action correctrice en opposition
- ii) aux méthodes proactives qui intègre la prédiction des situations de débordement de l'intervalle de consommation prédéfini et qui anticipe sur les actions correctrices. Cette dernière catégorie produit des résultats plus fiable. Notre étude s'inscrira dans ce cadre.

Les outils de prédiction : notre étude nous a permis d'identifier deux catégories à savoir les outils statistiques et ceux basés sur l'apprentissage par ordinateur. Dans ces deux catégories, nous avons retenus des critères suivant lesquels nous avons étalonné différents

outils parmi les plus répandus dans la littérature pour en arriver à retenir deux que sont les outils ARMA et les réseaux de neurones. Pour les premiers, ce sont leur popularité et leur capacité de modélisation de problème indépendamment des domaines d'application qui ont principalement permis ce choix. Pour les seconds, en plus de leur réputation à résoudre des problèmes certains sous des contraintes d'incertitudes, ils sont les plus fiables et les plus complets de leur catégorie. D'autres outils auraient pu être choisis mais nous nous limiterons à ces deux dans le cadre de ce projet de recherche.

Les environnements de simulation qui commencent à émerger dans le domaine du cloud computing. Cloudsim se démarque nettement étant donné qu'il fournit un framework complet de fonctionnalité qu'il peut être pertinent de tester dans les études sur le cloud computing.

Les types de données : ils sont de deux types : les données synthétiques qui sont les plus utilisées en raison de l'indisponibilité de traces représentatives pour la validation des modèles. Depuis quelques années quelques traces commencent à se démocratiser à l'instar du plus célèbre Google Cluster data publiée dans le domaine public dès 2012 et qui contient près de 300 Go de données portant sur le fonctionnement de plus de 12 000 serveurs sur une période de 29 jours. C'est sur ce jeu de données que nous avons principalement axé notre travail. En outre nous avons validé nos expériences avec un autre jeu de données, PlanetLab, identique sur le plan de la structure et qui provient d'une plateforme participative de mise en ligne de traces pour la communauté scientifique.

Chapitre 1

TECHNIQUE D'OPTIMISATION DE L'ENERGIE AU SEIN DES DATACENTERS

Introduction

Lors de la conception des datacenters, les architectes ont tendance à prévoir les besoins en énergie des serveurs sur la base de leur consommation maximale. Cependant, ces niveaux de consommation très élevés sont rarement atteints et pendant de longues périodes de fonctionnement, les serveurs restent dans un état de charge minimale tout en consommant une grande partie de l'énergie qui leur est fournie. Ce constat préoccupant a fait l'objet de nombreuses études à travers lesquelles, plusieurs mécanismes permettant de rationaliser l'utilisation de l'énergie fournie aux serveurs dans les datacenters en maximisant le fonctionnement de leurs ressources physiques ont été élaborés. Il ne paraît pas évident de chercher à approfondir cette question sans préalablement faire une revue de toutes ces techniques qui s'imposent progressivement dans l'ingénierie et dans l'industrie des datacenters. Nous proposons ici un angle d'attaque de la question basé sur l'analyse croisée des méthodes utilisées par les travaux précédents et des approches de leur implémentation afin de permettre d'une part un positionnement de la notre et de fournir à toutes nouvelles initiatives un état de l'art.

Les systèmes d'optimisation de l'énergie électrique poursuivent généralement plusieurs objectifs. Les plus importants sont la réduction des coûts d'exploitation des datacenters, le respect des exigences de disponibilité des services et la réduction de l'empreinte carbone. Les démarches employées peuvent se rivaliser principalement sur les techniques choisies mais également sur les types de données ayant permis de valider les résultats et sur les outils de simulation utilisés.

Les techniques d'optimisation de l'énergie électrique les plus populaires consistent en la consolidation dynamique des serveurs virtuels ou des applications sur un nombre stricte de

serveurs physiques nécessaires ou la réallocation dynamique des ressources physiques des serveurs en fonction de la charge ponctuelle supportée [Beloglazov et al. 2011]. Ces techniques permettent de ne garder fonctionnel qu’un nombre optimal de serveurs physiques en allumant ou en éteignant certains relativement au trafic du datacenter au cours du temps. Toutefois, les processus d’allumage et d’extinction de serveurs pourraient entraîner la mise en attente ou la perte de certaines requêtes rendant ainsi indisponible certaines ressources et résultant en une violation des exigences de qualité de service inscrit dans les accords de niveau de service (SLA). Les travaux de [Verma et al. 2008] ont permis d’établir une relation étroite entre la volumétrie du trafic généré par les applications et la consommation de ressources énergétiques et de conclure qu’il existe un niveau d’utilisation des ressources physiques des serveurs qui garantit un rendement optimal de l’énergie. Pour ce faire, toutes les décisions d’optimisation ci-dessus énumérées se basent préalablement sur un mécanisme d’analyse de la variabilité et des caractéristiques des requêtes envoyées vers le datacenter au cours du temps. Un moteur d’analyse du trafic du datacenter est donc nécessaire dans la plupart des travaux précédents. Il peut consister en une méthode d’analyse réactive dans les cas où les critères d’optimisation sont fixés à l’avance sans tenir compte du caractère dynamique du trafic ou en une méthode proactive pour désigner les critères qui s’ajustent en fonction de la variabilité du trafic [F Gbaguidi 2014].

Cette partie de la revue de littérature se focalisera essentiellement sur les différents aspects de la recherche de nouvelle technique d’optimisation énergétique ci-dessus exposés et sera consacrée principalement aux différentes techniques employées tant pour l’analyse des besoins en énergie des datacenters que pour les actions de contrôle des systèmes d’approvisionnement électrique. Lors de la conception des datacenters, les architectes ont tendance à prévoir les besoins en énergie des serveurs sur la base de leur consommation maximale. Cependant, ces niveaux de consommation très élevés sont rarement atteints et pendant de longues périodes de fonctionnement, les serveurs restent dans un état de charge minimale tout en consommant une grande partie de l’énergie qui leur est fournie. Ce constat préoccupant a fait l’objet de nombreuses études à travers lesquelles, plusieurs

mécanismes permettant de rationaliser l'utilisation de l'énergie fournie aux serveurs dans les datacenters en maximisant le fonctionnement de leurs ressources physiques ont été élaborés. Il ne paraît pas évident de chercher à approfondir cette question sans préalablement faire une revue de toutes ces techniques qui s'imposent progressivement dans l'ingénierie et dans l'industrie des datacenters. Nous proposons ici un angle d'attaque de la question basé sur l'analyse croisée des méthodes utilisées par les travaux précédents et des approches de leur implémentation afin de permettre d'une part un positionnement de la notre et de fournir à toutes nouvelles initiatives un état de l'art.

Les systèmes d'optimisation de l'énergie électrique poursuivent généralement plusieurs objectifs. Les plus importants sont la réduction des coûts d'exploitation des datacenters, le respect des exigences de disponibilité des services et la réduction de l'empreinte carbone. Les démarches employées peuvent se rivaliser principalement sur les techniques choisies mais également sur les types de données ayant permis de valider les résultats et sur les outils de simulation utilisés.

Les techniques d'optimisation de l'énergie électrique les plus populaires consistent en la consolidation dynamique des serveurs virtuels ou des applications sur un nombre stricte de serveurs physiques nécessaires ou la réallocation dynamique des ressources physiques des serveurs en fonction de la charge ponctuelle supportée [Beloglazov et al. 2011]. Ces techniques permettent de ne garder fonctionnel qu'un nombre optimal de serveurs physiques en allumant ou en éteignant certains relativement au trafic du datacenter au cours du temps. Toutefois, les processus d'allumage et d'extinction de serveurs pourraient entraîner la mise en attente ou la perte de certaines requêtes rendant ainsi indisponible certaines ressources et résultant en une violation des exigences de qualité de service inscrit dans les accords de niveau de service (SLA). Les travaux de ont permis d'établir une relation étroite entre la volumétrie du trafic généré par les applications et la consommation de ressources énergétiques et de conclure qu'il existe un niveau d'utilisation des ressources physiques des serveurs qui garantit un rendement optimal de l'énergie. Pour ce faire, toutes les décisions d'optimisation ci-dessus énumérées se basent préalablement sur un mécanisme d'analyse de la variabilité et des caractéristiques des requêtes envoyées vers le datacenter au cours du temps. Un moteur d'analyse du trafic du datacenter est donc nécessaire dans la plupart

des travaux précédents. Il peut consister en une méthode d’analyse réactive dans les cas où les critères d’optimisation sont fixés à l’avance sans tenir compte du caractère dynamique du trafic ou en une méthode proactive pour désigner les critères qui s’ajustent en fonction de la variabilité du trafic [F Gbaguidi 2014].

Cette partie de la revue de littérature se focalisera essentiellement sur les différents aspects de la recherche de nouvelle technique d’optimisation énergétique ci-dessus exposés et sera consacrée principalement aux différentes techniques employées tant pour l’analyse des besoins en énergie des datacenters que pour les actions de contrôle des systèmes d’approvisionnement électrique

1.1 Approche globale d’optimisation énergétique

En se référant à la classification ci-dessus figure 1.1, les approches d’économie d’énergie dans les datacenters se composent d’un module d’analyse et d’un module de contrôle de consommation.

1.2 Module d’analyse

Le but principal de ce module est l’analyse des caractéristiques du trafic entrant et sortant du datacenter et de l’impact de ces caractéristiques sur la consommation énergétique [F Gbaguidi 2014]. Plusieurs contraintes techniques et environnementales guident la gestion des Clouds datacenters notamment les SLA (Service Level Agreement) qui stipulent les exigences des utilisateurs et les pénalités éventuelles en cas de non satisfaction, les budgets de l’énergie qui se représentent un poste très important de l’OPEX et l’empreinte carbone résultante dont la diminution est un enjeu environnemental majeur. Le challenge au niveau du module d’analyse est donc de planifier les actions de contrôle de l’énergie qui présente le meilleur compromis entre les performances et la consommation énergétique totale du datacenter. A ce niveau, les travaux de recherches antérieures ont tenté de répondre à certains questionnements tels que : Quelle est la meilleure distribution

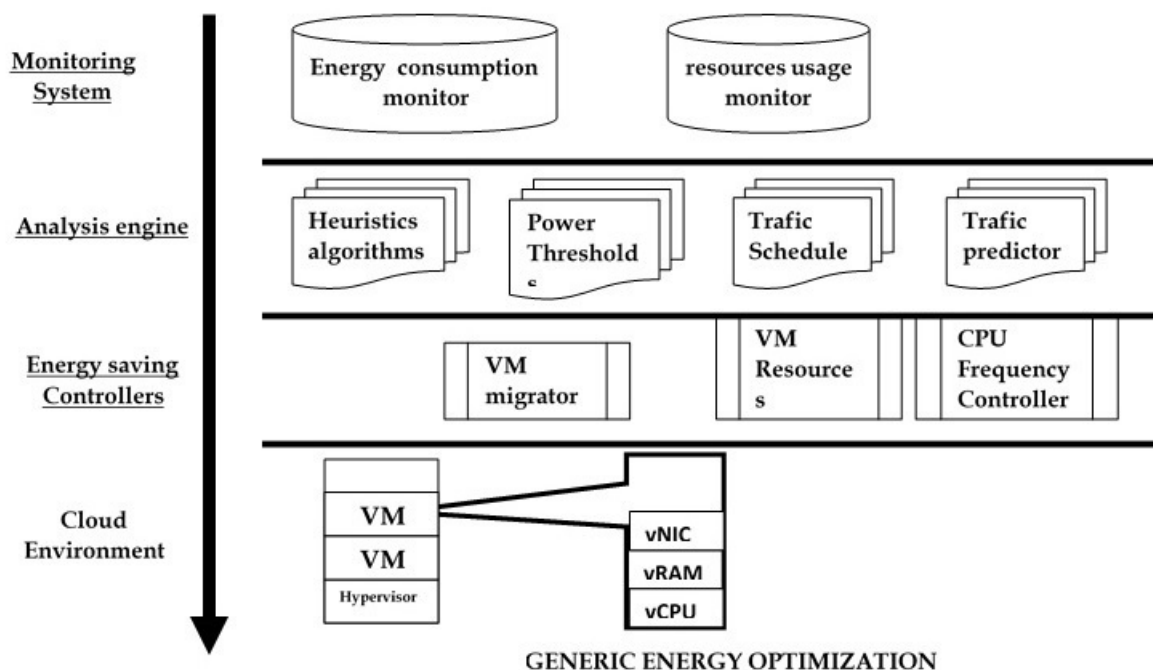


FIGURE 1.1 – Architecture en couche des systèmes d'optimisation énergétique

de la charge de trafic sur l'ensemble des serveurs du datacenter ? Quels sont les serveurs qui peuvent être éteints à un moment précis ? À quel moment faut-il rallumer certains serveurs pour répondre à la montée du trafic ? Comment doit-on répartir les ressources physiques (CPU, RAM, NIC, HDD etc.) sur les serveurs virtuels pour maximiser leur utilisation ?

Au nombre des approches élaborés dans la littérature, nous retenons deux grandes catégories que sont les méthodes que nous qualifions de «réactives» et les méthodes dites «proactives» en fonction de leur prise en compte dynamique ou non de l'état futur du trafic dans la planification des mesures de contrôle de la consommation énergétique comme illustré sur la figure 1.2

1.2.1 Les méthodes réactives

Les mécanismes adoptés ici consistent à évaluer de façon heuristique les caractéristiques du trafic dans le datacenter ou à fixer des seuils d'utilisation des ressources ou de consom-

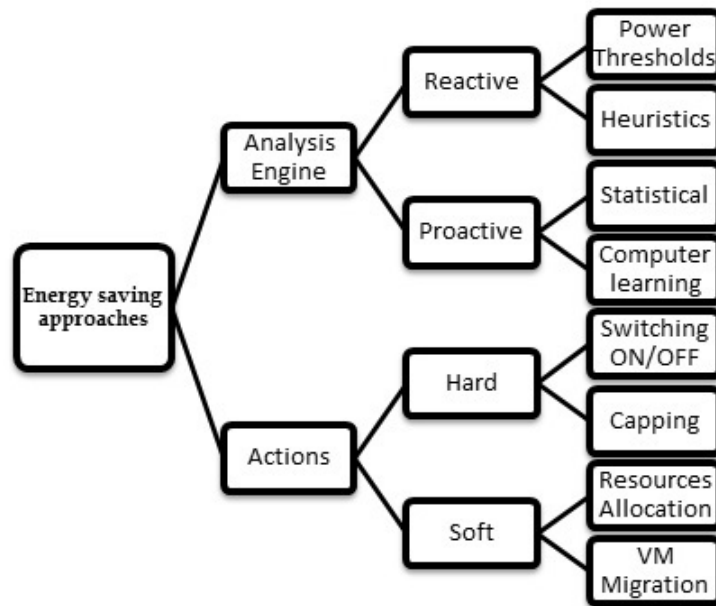


FIGURE 1.2 – Classification des approches d’optimisation d’énergie dans les datacenters

mation électrique. Les mesures de contrôle de l’énergie qui en résultent peuvent manquer de précision et donc doivent être continuellement réajustées afin d’assurer la performance et la disponibilité des ressources. Un système de monitoring fiable est donc nécessaire pour cette adaptation continue des contraintes. On distingue dans la littérature :

1.2.1.1 Les approches basées sur les seuils de référence de la consommation énergétique

Ces approches développées par les auteurs [Beloglazov and Buyya 2010a], [Beloglazov and Buyya 2010b] et [Buyya et al. 2010] consistent à fixer des valeurs hautes et des valeurs basses d’utilisation des ressources physiques (notamment le CPU) au delà desquelles une action de contrôle d’énergie doit être déclenchée. Etant donné que même lorsqu’ils ne sont pas sollicités (c’est-à-dire lorsque le processeur est dans un mode sommeil), les serveurs consomment plus de 50% de leur puissance maximale, la fixation de ces seuils permet soit d’éteindre les serveurs lorsqu’ils se retrouvent en dessous du seuil bas ou de désengorger leur trafic lorsqu’ils se retrouvent au delà du seuil haut. Les auteurs [[Beloglazov and Buyya 2010a], [Beloglazov and Buyya 2010b] et [Buyya et al. 2010] utilisent les statistiques

antérieures du trafic pour ajuster les seuils haut et bas à partir desquels une action est déclenchée. Par exemple, lorsqu'un seuil bas d'utilisation du CPU est atteint, toutes les machines virtuelles de ce serveur seront systématiquement déplacées et ce serveur sera éteint tandis qu'en cas de dépassement du seuil haut, certaines machines seront déplacées de sorte à maintenir l'utilisation en dessous de ce seuil. Cette technique montre des résultats satisfaisants sur la réduction de la consommation électrique même si leur efficacité dépend des valeurs de seuil choisies. En effet, en cas de mauvais ajustage des seuils ou des faux positifs, il pourrait en résulter une instabilité des mesures de contrôle de consommation telles qu'une extinction suivi immédiatement d'un rallumage dû à une montée soudaine du trafic. Les auteurs [Verma et al. 2009] conçoivent un système modulaire dans lequel un module indépendant de monitoring et un module d'ajustage des seuils sont contrôlés par un module d'arbitrage qui décide d'une action en corrélant les données et les suggestions venant des différentes composantes du système. D'une manière similaire, les travaux présentés dans [Govindan et al. 2009] caractérisent la consommation énergétique en termes de pic et de vallée. Ici l'optimisation de l'énergie électrique se fait de façon granulaire de l'échelle la plus élevée (datacenter) à la plus faible (Machine Virtuelle) en passant par les niveaux de châssis, de clusters et de rack. La rationalisation de l'énergie consiste donc à fixer des pics et des vallées pour chaque niveau afin de maintenir la consommation aux niveaux supérieurs au sein d'un intervalle donné.

Les approches basées sur les seuils de référence sont d'une grande simplicité de mise en œuvre. Il requiert néanmoins une bonne supervision du trafic et peuvent engendrer certaines instabilités.

1.2.1.2 Les approches heuristiques

Les méthodes heuristiques se rapportent à une démarche de résolution de problèmes consistant à compléter les algorithmes mathématiques par des solutions empiriques afin de réduire leur complexité et d'accélérer leur résolution. Dans le contexte de réduction de l'énergie consommée par les datacenters, les auteurs de [Kumar et al. 2009] ont combiné un algorithme d'optimisation et une méthode heuristique consistant à

associer à un vecteur $d = [req1, req2, req3]$ des requêtes enregistrées pour les applications 1, 2 et 3, une configuration de répartition sur les serveurs physiques ainsi que la fréquence de CPU qui leur sera alloué. Cette configuration prend la forme $conf = [(S1, f1, APP1), (S1, f2, APP2), (S1, f3, APP3)]$ indiquant pour cet exemple (ce vecteur de requête) que les applications 1, 2 et 3 seront hébergées sur le serveur S1 et il leur sera alloué respectivement les fréquences $f1$, $f2$ et $f3$ de processeur. Dans cette approche, lorsque le vecteur des requêtes excède la capacité du CPU, une application sera migrée vers un autre serveur avec une fréquence donnée afin de garantir des niveaux de performance convenables. Une technique similaire est utilisée par les chercheurs [Kumar and Sahoo 2014] qui proposent un algorithme dérivé du « greedy heuristics » pour déterminer la meilleure séquence de traitement des requêtes en fonction de la disponibilité des ressources physiques des serveurs. La démarche consiste à sélectionner de façon itérative les lots de requête à exécuter selon que ces dernières maximisent l’utilisation des temps processeur disponibles sans dépasser les 100%. Le but de cette technique est d’éviter de choisir des tâches qui maintiennent les processeurs des serveurs dans un mode de fonctionnement qui n’optimise pas l’utilisation de l’énergie provisionnée. L’algorithme élaboré **MaxMaxUtil** permet mécaniquement de rationaliser l’utilisation des ressources serveurs. Cependant, un critère de priorisation des flux devra être intégré afin d’éviter que le bon fonctionnement de certaines applications qui ne tolèrent pas les temps d’attente ne soit perturbé. Un autre exemple développé dans [Buyya et al. 2010] utilise un ensemble de 4 algorithmes heuristiques afin de choisir dynamiquement les VM à consolider. Ces algorithmes consistent à :

- fixer arbitrairement un seuil haut et un seuil bas et de déclencher la migration des VM (Machine Virtuelle) en cas de violation tel que présenté dans la section ci-dessus
- règle de « Minimization of Migration (MM) » qui consiste à migrer à la fois une quantité de VM qui minimise la charge de migration
- Une règle de « Highest Potential Growth » qui permet de migrer les VM consommant

une quantité minimale du CPU de façon à limiter une potentielle augmentation de la charge et donc une violation du CPU

- l’algorithme dit « Random Choice » qui choisit les VM à migrer suivant une distribution de variable aléatoire. Ces différents algorithmes heuristiques présentent chacun leurs avantages et inconvénients.

1.2.2 Les méthodes proactives

Par opposition aux méthodes réactives, nous classons ici dans les méthodes proactives, celles qui font une estimation du trafic à venir et planifient préalablement les mesures de contrôle de l’énergie. L’analyse proactive des caractéristiques du trafic consiste à élaborer des modèles statistiques applicables aux différents types de trafic ou de recourir aux techniques d’apprentissage dynamique non supervisé par ordinateur pour améliorer la précision des analyses.

1.2.2.1 Les approches statistiques

Les méthodes statistiques permettent d’établir des modèles statistiques modélisant le plus fidèlement possible les caractéristiques de la volumétrie et du débit des flux applicatifs des datacenters. Ces modèles sont ensuite utilisés pour planifier les meilleures répartitions possibles des machines virtuelles sur les serveurs ou de faire les allocations de ressources les plus optimales. Dans les travaux exposés en [Verma et al. 2009], les auteurs élaborent deux modèles statistiques notamment la relation de corrélation entre les volumes de trafic entrant sur les VM et d’une part et la corrélation entre les pics de trafic entrant sur les VM d’autres parts. Ces modèles sont utilisés pour l’agrégation des VM ayant des volumes ou des pics de trafic similaires lors de la consolidation des serveurs. Ces fonctions de corrélation permettent de planifier à l’avance la distribution des VM et de garantir à tout moment une utilisation optimale des ressources physiques des serveurs. Les techniques d’élaboration du profil de trafic en se basant sur les statistiques a également été développé dans [Govindan et al. 2009]. Les auteurs de ce modèle sont partis du fait que tous les pics

n'interviennent pas au même moment. Par conséquent l'établissement du profil statistique du trafic permet de répartir les VM de façon à équilibrer la répartition des pics dans le temps d'une part et de les distribuer également dans l'architecture spatial (racks, chassis, serveurs) du datacenter d'autre part. Les résultats montrent une nette amélioration du ratio utilisation du processeur/énergie fournie.

1.2.2.2 Les approches d'apprentissage par ordinateur

Les approches d'apprentissage par ordinateur se révèlent être les plus adaptées aux propriétés très variables du trafic dans les datacenters. Elles s'appuient sur une utilisation des données antérieures de trafic en représentant leur volume et leur débit sous forme de séries temporelles afin de pouvoir en prédire les valeurs futures. Les méthodes de prédiction de trafic sont d'une grande précision nonobstant le coût de traitement (temps et ressources processeur consommés) parfois élevé qu'elles génèrent. Cette précision dans la prédiction de trafic favorisent une meilleure planification des besoins en ressources énergétiques et donc d'en rationaliser l'utilisation. Cependant, leur fiabilité et leur éligibilité aux environnements des datacenters dépendent fortement du choix d'une méthode qui s'adapte aux variations parfois brusques de trafic, à l'indisponibilité ou l'incomplétude des données historiques, aux contraintes de temps réel etc. Dans les travaux présentés en [Kusic et al. 2009], les prédicteur à filtres de Kalman ont été utilisés pour prédire le débit du trafic puis l'algorithme Limited LookAhead Control (LLC) pour calculer le risque inhérent à chaque opération de consolidation de serveur en termes de violation des SLA. La méthode consiste à évaluer en fonction du trafic à venir le nombre de serveurs à garder allumé, le nombre de VM nécessaires et les ressources physiques à provisionner. Un plan d'allocation est donc préalablement établi et l'algorithme LLC permet d'affiner cette planification en évaluant le risque potentiel qu'elle présente en termes d'interruption de service. Les résultats sont très satisfaisants en terme d'économie d'énergie même si la durée de traitement reste très élevé. Une approche similaire d'apprentissage par ordinateur a été également utilisée par un groupe de chercheurs en [Berral et al. 2010]. Ici, c'est l'algorithme du Dynamic BackFilling qui a été utilisé pour calculer de façon proactive

le niveau de satisfaction des utilisateurs (respect des SLA) résultant d'un schéma de répartition d'un ensemble de requête sur un ensemble de serveurs donné. De bons résultats ont été obtenus avec cette approche même si les incertitudes dans la variation du trafic n'ont pas été appréhendées et traitées. Dans la même perspective, les auteurs [Gandhi et al. 2011] utilisent l'algorithme Fast Fourier Transform (la Transformée de Fourier Rapide) pour établir un périodogramme à partir d'une série temporelle des données historiques du trafic. Le graphe résultant ressort clairement la rythmique et la périodicité du volume de trafic entrant pour les périodes à venir à partir desquelles un plan d'allocation des ressources pourra être appliqué. Tout comme les méthodes précédentes, ce système est optimisé par un algorithme de rétro-propagation des erreurs en utilisant un module réactif pour l'ajustage des prédictions.

1.3 Module de contrôle des actions

Ce module permet d'exécuter la commande correspondante à l'action choisie en vue d'optimiser la consommation électrique tout en maintenant un niveau de disponibilité et de performance convenable. Une fois que le module d'analyse planifie quelle action effectuer sur quel serveur et à quel moment, le module de contrôle s'occupera de comment exécuter la commande finale. Les actions peuvent consister à commander directement le matériel (alimentation, CPU, RAM, NIC, etc.) ou à dérouler un algorithme de consolidation des machines virtuelles ou d'allocation des ressources virtuelles adéquates.

1.3.1 Techniques matérielles

Ces techniques exercent une action directe sur le matériel soit en l'éteignant ou en le rallumant au besoin de sorte qu'aucun matériel non sollicité ne consomme inutilement de l'énergie au sein du datacenter.

1.3.1.1 Allumage et extinction des serveurs

L'extinction des serveurs est une solution finale appliquée après s'être assuré que toutes les autres actions de contrôle de l'énergie ont été exécutées et qu'il se dégage clairement

des serveurs non sollicités à un moment donné. Un serveur est ainsi mis hors service soit directement ou à travers une commande programmée après déménagement de toutes les machines virtuelles qu’il abrite ou leur placement dans un mode inactif comme décrit dans [Beloglazov and Buyya 2010a], [Kumar et al. 2009], [Petrucci et al. 2010], [Verma et al. 2009] et [Govindan et al. 2009]. Cette technique doit cependant être utilisée avec beaucoup de prudence car elle peut présenter un risque de violation de SLA au cas où la procédure de rallumage n’est pas convenablement évaluée. La commande d’arrêt des serveurs peut également être envoyée à distance en utilisant la fonctionnalité *Wake-On-Lan* embarquée sur certaines cartes réseau comme présenté dans [Kusic et al. 2009].

1.3.1.2 Ecrêtage de la tension ou de la fréquence

Voltage and Frequency Scaling est l’une des techniques qui consiste à maintenir un meilleur ratio entre la quantité d’énergie provisionnée et la fréquence du CPU réellement utilisée. L’objectif est de rationaliser chaque watt provisionné au serveur. Il existe plusieurs variantes dont entre autres le DFS (Dynamic Frequency Scaling), le DVFS (Dynamic Voltage and Frequency Scaling) qui présente des résultats très intéressants en terme d’économie d’énergie. Il permet d’ajuster de façon dynamique la quantité d’énergie consommée à la charge de traitement courante du CPU. Plusieurs travaux ont porté sur l’utilisation de cette fonction pour la réduction de la consommation électrique dans les datacenter. Dans [Verma et al. 2009] et [Govindan et al. 2009], les paramètres d’ecrêtage ont été calculés à partir des profils de trafic et d’une technique de multiplexage des files d’attente. Les différentes variantes DFS et DVFS ont été comparées dans [Gandhi et al. 2009], et montrent une relation plus ou moins linéaire entre la tension fournie et la fréquence consommée. Cette linéarité varie avec le DVFS ou une combinaison des 2 variantes. Une technique similaire a été développée dans PowerNAP [Meisner et al. 2009]. Ici, l’optimisation de l’énergie consiste à placer les serveurs dans un mode de consommation près de zéro et d’être capable de le basculer à nouveau dans un mode de consommation normale dès que le signal d’un traitement est reçu. Cette technique peut être très efficace pour des serveurs hébergeant des applications inactives pendant de

longues périodes et donc la variation du trafic est imprévisible. Dans [Petrucci et al. 2010], les auteurs exposent VirtualPower, une technique hybride d’écrêtage utilisant Virtual Power Management (VPM). Les mécanismes mis en œuvre consistent à :

- recevoir les requêtes d’écrêtage envoyées par les VM et à exécuter l’action correspondante au niveau du serveur physique,
- à utiliser l’hyperviseur pour ajuster de façon dynamique les demandes de CPU envoyées par les VM ou
- à consolider toutes les requêtes sur une ressources spécifiques (CPU par exemple) en mettant les autres dans un état éteint ou en veille. Les mécanismes utilisés dans *VirtualPower* prouvent une grande efficacité dans diverses conditions d’utilisation.

Une fonctionnalité de file d’attente est introduite dans la plupart des travaux afin de limiter les pertes de trafic pouvant être engendrées par un mauvais paramétrage de l’écrêtage. La technique d’écrêtage paraît simple et efficace dans certains scénarii mais elle peut présenter certaines insuffisances en cas de mauvais choix des paramètres. L’autre insuffisance est que ces technologies ne sont disponibles que sur une gamme très limitée de processeur parmi les plateformes serveurs.

1.3.2 Techniques logicielles

Ces techniques sont dites de haut niveau étant données qu’elles utilisent généralement un algorithme sophistiqué qui commande à la fois les ressources physiques et virtuelles.

1.3.2.1 Allocation des ressources matérielles

Connue sous le nom de *VM provisioning*, l’allocation de ressources matérielles consiste à répartir convenablement les ressources physiques d’un serveur à savoir les processeurs, les mémoires, les cartes réseau, les disques en fonction des besoins réels des machines

virtuelles. L’objectif est de provisionner les machines virtuelles avec les ressources qu’elles consommeront effectivement et dont l’énergie allouée ne sera pas gaspillée. Dans [Beloglazov and Buyya 2010a], par exemple, un gestionnaire local est chargé de provisionner les VM au fur et à mesure que le volume de trafic augmente tout en restant dans une limite maximale. Dans ce système le gestionnaire local est utilisé pour garantir une répartition de ressources selon les charges au niveau d’un serveur physique tandis qu’un gestionnaire global effectue le même contrôle au niveau des pools de serveurs. Il permet de maintenir une bonne répartition de l’énergie sur les serveurs physiques tout en respectant une limite maximale de consommation dans toute la ferme de serveurs ou le datacenter. Une technique d’allocation de ressources en fonction de la quantité de tâches à exécuter par le serveur a été présentée dans [Gandhi et al. 2009]. Les auteurs planifient tout simplement les tâches à allouer au serveur de façon à maximiser sa capacité de traitement. Ils introduisent un mécanisme de file d’attente afin de prévenir la perte de certaines requêtes.

1.3.2.2 Consolidation des machines virtuelles

La consolidation des machines virtuelles consiste à répartir de façon dynamique les machines virtuelles sur les serveurs physiques de façon à maximiser les capacités de traitement de ces derniers à un moment donné. Elle est traitée généralement comme un problème de *Bin Packing* ou remplissage de conteneur qui consiste à trouver le meilleur rangement possible d’objets différents dans un conteneur de forme non régulière. Très utilisée dans la littérature, cette technique est appliquée après une bonne analyse des caractéristiques du trafic en utilisant une approche réactive ou proactive. Dans les travaux [Beloglazov and Buyya 2010a], [Beloglazov and Buyya 2010b], [Buyya et al. 2010] et [Verma et al. 2009] par exemple, un module de migration de VM exécute un algorithme de Bin Packing une fois que la condition planifiée est remplie. L’algorithme du Best Fit Decreasing et ses variantes sont généralement utilisés ou améliorés pour résoudre le problème de Bin Packing. La logique générique est de trier les VM en fonction de leur utilisation de CPU et de les répartir sur les serveurs sur lesquels ils présentent la plus faible augmentation de la consommation électrique.

Conclusion

La consommation de l'énergie électrique des datacenters reste très élevée et les projections sur les prochaines années sont très alarmantes. La communauté scientifique s'est véritablement penchée sur la question de la réduction de la consommation électrique des datacenters au cours des dernières années. Les solutions élaborées se basent essentiellement sur la virtualisation des serveurs devenus excessivement puissants comme prédit dans la loi de Moore. Cependant, les techniques d'approches diffèrent et produisent des résultats très variables en fonction des environnements de test. Il est à retenir que l'objectif global est de maintenir le nombre strictement nécessaire de serveurs allumés et de rationaliser l'utilisation de l'énergie provisionnée sur ces serveurs. Une phase d'analyse des propriétés du trafic s'est révélée nécessaire dans la plupart des études et permet de planifier les meilleures répartitions possibles des volumes de trafic sur les ressources du datacenter. Cette analyse peut se baser sur des démarches simples, intuitives ou proactives et peut déboucher sur des techniques de contrôle de l'énergie des plus simples aux plus sophistiquées.

En outre, en dépit de la fiabilité et de la précision des méthodes employées, les environnements de datacenter peuvent être très variables en termes de taille, des caractéristiques de trafic, de plateforme matérielle et de contraintes de SLA. La plupart des méthodes développées restent donc d'une fiabilité relative à l'environnement et leur généralisation n'est pas possible pour l'heure. Par ailleurs, le choix des techniques doit s'adapter aux aspects temps-réel et donc les algorithmes exécutés doivent être les plus rapides possibles. La portabilité des outils développés est aussi une condition nécessaire étant donné les différences de plateforme.

Sans pouvoir désigner une meilleure méthode pour approcher la problématique de réduc-

tion de la consommation énergétique des datacenters, notre travail a consisté à décrire et à classer les différentes techniques les plus utilisées dans la littérature. Cette classification pourra servir de point de départ pour de nouvelles recherches dans ce domaine étant donnée l'urgence pour notre humanité de résoudre cette question de la rationalisation de l'énergie disponible afin de prévenir les désastreuses conséquences annoncées.

Chapitre 2

CHOIX DES OUTILS ET DES DONNEES

Introduction

La recherche dans le domaine des datacenters ne pourrait se dérouler dans les environnements de production en raison du caractère contraignant des obligations des opérateurs de cloud computing et de la sensibilité des traitements effectués. Dès lors, de nombreux simulateurs sont de plus en plus élaborés avec comme objectif d’adresser des problématique bien précises telles que l’amélioration des performances, [Calheiros et al. 2011] la sécurité des transactions [Wickremasinghe et al. 2010], le stockage, la journalisation [Sá et al. 2014] ou l’efficacité énergétique [Liu et al. 2009]. Dans cette panoplie d’outils, il est indispensable pour aborder un projet de recherche, d’identifier le plus adéquat en fonction de ses objectifs afin d’y implémenter les adaptations nécessaires aux développements de ces propres techniques et à leur validation. Dans l’optique du choix de notre environnements de recherche, nous avons comparé les simulateurs orientés vers l’optimisation énergétique suivant différents critères dont les techniques implémentées, la disponibilité du code source ou de leur facilité d’utilisation.

En outre, l’accès aux données de traces pouvant permettre de généraliser les algorithmes et techniques développés est un défi majeur étant donné les considérations de confidentialité y relatives et les variations significatives des caractéristiques de trafic lorsque l’on passe d’un datacenter à un autre. Quelques traces laissées récemment dans le domaine public permettent d’élaborer et de tester des algorithmes. Cependant, le choix des données dépend fortement du contexte de l’étude, des caractéristiques du datacenter à étudier et de la taille des échantillons pour une fiabilité des résultats. Face aux insuffisances de jeu de données représentatives, le recours aux générateurs de données synthétiques est une

alternative permettant de conclure sur certains scénarii de simulation. Dans le cadre de ce projet, nous réalisons une analyse exploratoire du Google cluster data publié en 2011 pour identifier notamment son éligibilité à la conduite de la recherche sur les aspects liés à la consommation de ressources au sein des datacenters.

Dans ce chapitre, nous nous focaliserons essentiellement sur la démarche de sélection des outils et des environnements de simulation orientés vers l'amélioration de l'efficacité énergétique des datacenters et la synthèse des types de données disponibles pour la validation des travaux de recherche dans ce domaine.

2.1 Choix des outils

2.1.1 Démarche d'évaluation des outils

Cette partie présente les outils les plus utilisés dans les projets de recherche pour la prédiction de trafic ou autres usages y afférents. Ceci nous a permis d'explorer une panoplie d'outils que nous avons classifiés en 3 groupes principaux à savoir : Les outils statistiques : ici, nous classons toutes les méthodes de prédiction basées sur le traitement d'un échantillon de donnée pour la détermination d'un modèle utilisable pour prédire à d'autres échelles de temps. Dans ce lot, nous avons retenu les outils de type ARMA, les filtres de Kalman, Holt Winters et Kriging. Ils se différencient ensuite par leur simplicité ou non, leur adaptabilité à des types de scénario bien défini et aux performances qu'ils ont fournies dans d'autres travaux similaires. les outils de l'intelligence artificielle : Ici nous mettons tous les outils basés sur un processus d'apprentissage préalable avec une rétropropagation des erreurs de prédiction par occurrence. Les réseaux de neurones se distinguent clairement dans cette catégorie qui compte aussi la logique floue, les systèmes experts, le Data mining et le SVM (Support vector Machine). les outils hybrides : Cette catégorie comprend les outils basés sur des algorithmes combinés des deux précédents ou sur une autre logique. Les variantes spécifiques des réseaux de neurones tels que les réseaux génétiques et les neuro-fuzzy (combinaison entre les réseaux de neurones et la

logique flou) ont souvent été utilisées dans la littérature. Les wavelets chaos grey qui combinent 3 algorithmes de l'intelligence artificielle se retrouvent dans cette catégorie. Le graphique présente un résumé de cette classification

Sur la base de cette classification, nous avons proposé un certain nombre de critères qui correspondent aux contraintes liées au champ de notre recherche. Les principaux critères que nous avons retenus se présentent comme suit Adaptabilité aux jeux de données de la consommation de ressources dans les datacenters : notre analyse à ce niveau a porté sur l'identification des méthodes statistiques ou basées sur l'apprentissage ayant produit des résultats dans la littérature proche de notre contexte d'étude ou sur des jeux de données similaires au nôtre. L'une des spécificités recherchées est aussi l'adaptabilité de l'outil au caractère volatile des trafics au sein des datacenters. Possibilité de traitement en ligne : l'aspect temps réels peut être requis dans le cadre de notre travail notamment lorsqu'il est nécessaire d'effectuer des prédictions à très court terme. Les temps de calcul des outils étant différents, nous avons sélectionné les méthodes qui présentent le meilleurs compromis sur l'ensemble des critères. fréquence d'utilisation dans la littérature : les outils les plus populaires sont ceux qui sont les plus intéressants car ils sont potentiellement rassurants pour l'obtention de nos résultats. En outre, ils présentent l'avantage de l'abondance de la documentation. Ceci est un atout non négligeable. Existence d'environnement de simulation : La plupart des outils que nous avons étudiés peuvent être implémentés dans des environnements de simulation populaire tels que R ou Matlab.

2.1.2 Outils choisis

A l'issue de notre étude nous avons retenu de baser notre travail sur les outils ARMA et sur les réseaux de neurones. Ces outils présentent les caractéristiques ci-après : ARMA : très populaire dans le domaine de la statistique, ARMA présente l'avantage de s'adapter à des jeux de données très variés. Ils traitent de nombreuses caractéristiques des données tels que la linéarité, la saisonnalité. La détermination des paramètres est cependant un préalable critique duquel dépend la fiabilité des expériences et des résultats. Le simulateur R contient une librairie très riche des outils ARMA pour le domaine de la statistique

notamment. Réseaux de neurones : l'incontournable boîte à outils des réseaux de neurones se révèle être le plus adapté à notre étude. Ce choix s'impose au regard de la popularité, de fiabilité démontrée dans différents cas et surtout de son adaptabilité à une large gamme de domaines d'application. Matlab implémente la neural toolbox qui présente de nombreuses fonctionnalités. Ces deux outils présentent en outre de nombreux autres avantages

2.1.3 Outils de simulation des environnements de Clouds

Dans une large mesure, la recherche dans le domaine du cloud computing ne peut se faire autrement que par simulation. Ceci du fait que les environnements réels sont des zones d'accès très restreints et surtout que la sensibilité des données ainsi que les obligations de confidentialité exigées par les clients ne permettent pas d'en faire des champs d'expérimentation a priori. Seuls les datacenters privés des centres de recherche seront exploitables et dans la plupart des cas uniquement pour des centres partenaires et donc inaccessibles à la recherche grand public. Pour pallier cette difficulté, de nombreux simulateurs ont vu le jour avec chacun des objectifs spécifiques notamment les tests de performance, l'efficacité énergétique, la sécurité, etc... Le champ de notre étude étant l'énergie et l'optimisation des ressources au sein des datacenters, nous présentons ci-après quelques simulateurs usuels orientés « efficacité énergétique » et plus particulièrement le très populaire simulateur CloudSim qui est la base de nos travaux.

2.1.3.1 GreenCloud

GreenCloud [Kliazovich et al. 2012], [Liu et al. 2009] est une extension du simulateur NS2 connu pour sa popularité dans le domaine des réseaux notamment. Il a été développé au sein d'un projet de Green IT avec comme principal objectif la mesure et la simulation de l'efficacité énergétique dans le domaine du cloud computing. Il comprend quatre composants essentiels à savoir :

- Le réseau qui comprend un ensemble d'équipement et d'éléments d'architecture permettant de connecter les utilisateurs au datacenter et de simuler les performances d'accessibilité aux services. L'architecture implémentée est en haute disponibilité

- en trois couches (coeur, aggrégation et accès) avec des liens haut débit ;
- les requêtes ou tâches qui sont affectées aux datacenters à travers un agent et un gestionnaire de l'allocation sur la ferme de serveurs.
- les computing servers composés de plusieurs modèles de serveurs physiques et de serveurs virtuels avec un scheduler pour la planification des placements et des migrations. Les serveurs sont également capable de simuler différents modes dont l'optimisation énergétique qui est le principal objectif de ce simulateur.
- les utilisateurs sont vus comme les générateurs de trafic et les propriétaires des SLA

2.1.3.2 Cloudsim

Cloudsim [Calheiros et al. 2011], [Goyal et al. 2012] est un outil de simulation des environnements de cloud computing élaboré par une équipe de l'Université de Melbourne en 2010. Il fournit des riches bibliothèques qui modélisent les datacenters, leurs flux de travail, les utilisateurs, les machines virtuelles, les serveurs, les applications, les composants de traitement, ainsi qu'un ensemble de règles de fonctionnement pouvant intégrer la plupart des différents angles d'étude courantes du domaine du cloud computing.

Les composants de CloudSim peuvent être combinés pour tester de nouvelles stratégies d'amélioration du fonctionnement du cloud computing que ce soit pour la fourniture des services, le respect des SLA, l'optimisation de la consommation énergétique, la planification des ressources, l'ordonnancement du trafic etc. Il est devenu l'outil de référence pour les projets du Green Cloud au cœur des problématiques de réduction de l'empreinte du cloud computing sur l'environnement. CloudSim est un outil totalement personnalisable en fonction des objectifs de l'étude. Les algorithmes peuvent être modifiés dans l'optique de simuler un scénario sous divers paramètres. Nativement le simulateur s'utilise dans un environnement Java tel que Eclipse et NetBeans et n'intègre pas d'interface graphique mais plusieurs extensions développées à titre de contribution par différents chercheurs permettent de disposer d'une version graphique orientée vers un objectif bien défini tel que Cloud Analyst et Cloud Report. Les principaux composants de CloudSim se présentent comme ci-après sur la figure 2.1

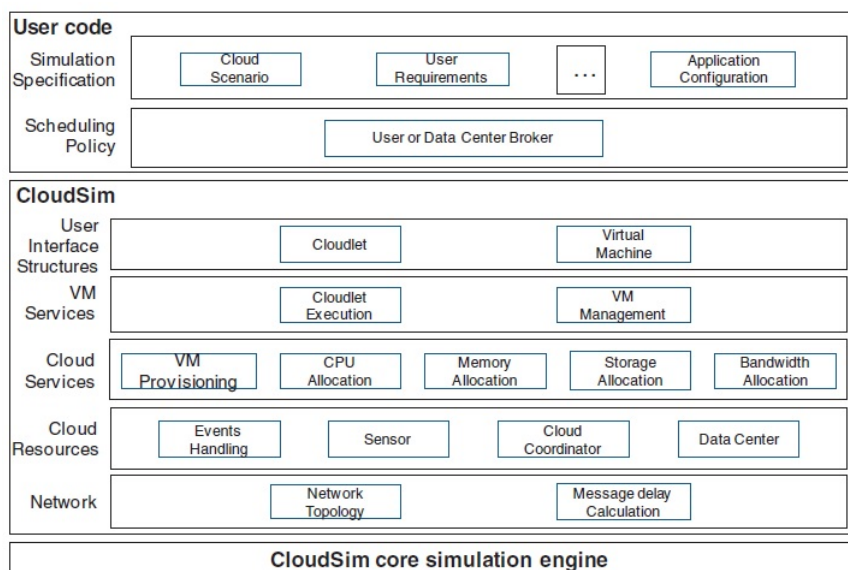


FIGURE 2.1 – Architecture des couches dans CloudSim

- *datacenter* : cette classe modélise les services de base de l'infrastructure (matériel) offerts par les fournisseurs Cloud (Amazon, Azure, App Engine). Il encapsule un ensemble d'hôtes de calcul pouvant être homogènes ou hétérogènes par rapport à leurs configurations matérielles (mémoire, noyaux, capacité et stockage). En outre, chaque composant datacenter crée une instance de provisionnement d'application généralisée qui implémente un ensemble de stratégies d'allocation de bande passante, de mémoire et de périphériques de stockage aux hôtes et VM ;
- *datacenterBroker* ou *Cloud Broker* : cette classe modélise un courtier, qui est responsable de la médiation des négociations entre les fournisseurs SaaS et Cloud ; et ces négociations sont axées sur les exigences de QoS. Le courtier agit au nom des fournisseurs de SaaS. Il découvre des fournisseurs de services cloud appropriés en interrogeant le contrat et entreprend des négociations en ligne pour l'allocation de ressources / services qui répondent aux besoins de QoS de l'application. Les chercheurs et les développeurs de systèmes doivent étendre cette classe pour évaluer et tester les politiques de courtage personnalisées. La différence entre le courtier et

CloudCoordinator est que le premier représente le client (c.-à-d. les décisions de ces composants sont réalisées afin d'augmenter les paramètres de performance liés à l'utilisateur), tandis que le dernier agit au nom du centre de données, c'est-à-dire qu'il essaie de maximiser les performances globales du centre de données, sans tenir compte des besoins de clients spécifiques.

- *Cloudlet* cette classe modélise les services d'application basés sur le Cloud (tels que la diffusion de contenu, les réseaux sociaux et le flux de travail des entreprises). CloudSim présente la complexité d'une application en fonction de ses besoins de calcul. Chaque service d'application dispose d'une longueur d'instruction pré-assignée et d'un transfert de données (pré et post-récupération) des frais généraux qu'il doit entreprendre pendant son cycle de vie. Cette classe peut également être étendue pour supporter la modélisation d'autres paramètres de performance et de composition pour des applications telles que des transactions dans des applications orientées vers la base de données. La figure 2.2 présente un diagramme des classes dans Cloudsim.

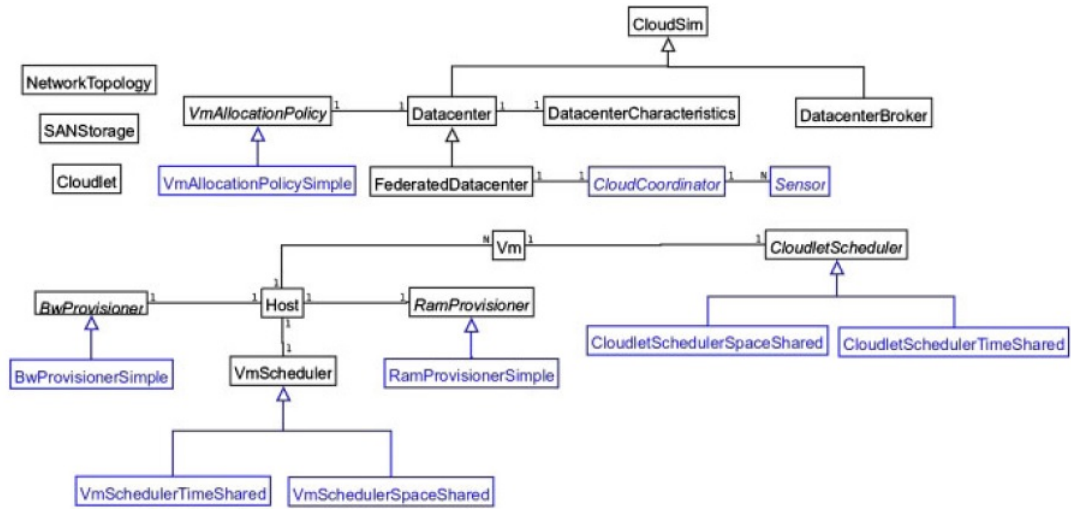


FIGURE 2.2 – Classes dans Cloudsim

- *CloudletScheduler* Cette classe abstraite est étendue par la mise en œuvre de différentes stratégies qui déterminent la part du pouvoir de traitement parmi les Cloudlets dans une machine virtuelle. Comme décrit précédemment, deux types de

stratégies de provisionnement sont offerts : espace partagé (*CloudletSchedulerSpaceShared*) et partagé par temps *CloudletSchedulerTimeShared*).

- *Vm* cette classe modélise une VM, qui est gérée et hébergée par un composant hôte Cloud. Chaque composant VM a accès à un composant qui stocke les caractéristiques suivantes liées à une machine virtuelle : mémoire accessible, processeur, taille de stockage et la stratégie de provisionnement interne de la machine virtuelle étendue à partir d'un composant abstrait appelé *CloudletScheduler*.
- *VmmAllocationPolicy* Cette classe abstraite représente une politique de provisionnement qu'un VM Monitor utilise pour attribuer des VM à des hôtes. La fonctionnalité principale de *VmmAllocationPolicy* est de sélectionner l'hôte disponible dans un centre de données qui répond aux exigences de mémoire, de stockage et de disponibilité pour un déploiement de VM.

2.2 Jeux de données de trafic au sein des datacenters

Dans la même logique que les environnements de simulation, les données réelles disponibles pour la recherche sont plutôt très rares. Les données traitées au sein des datacenter appartiennent avant tout à des clients exigeants sur la qualité de leur préservation et sur leur confidentialité. Des algorithmes mathématiques permettent de générer des données similaires à celles que l'on retrouve dans les datacenters. Ces données alimentent les simulateurs dans le but d'observer les résultats probables d'un choix technique. En dépit de leur utilité, les données synthétiques peuvent se révéler parfois non pertinente en raison de la grande variabilité des types de trafic d'un datacenter à un autre et par conséquent, les travaux de recherche y relatifs restent difficilement généralisables. Toutefois, il existe quelques collections de traces provenant de statistiques collectées sur les systèmes réels. Nous présenterons dans ce qui suit les caractéristiques de chaque type de données ainsi que les points forts et points faibles en ce qui concerne leur utilisation dans des projets de recherche.

2.2.1 Données synthétiques

Le recours aux traces synthétiques est une contrainte imposée à la recherche dans le domaine du cloud computing par l'insuffisance de traces et la variabilité de celles-ci d'un environnement à l'autre. En effet, les données collectées au sein d'un datacenters ont été pendant longtemps considérées comme étant sensibles en raison de la crainte d'un espionnage ou d'une trop grande connaissance des plateformes des opérateurs de cloud par leurs concurrents ou les hackers. L'ouverture des traces ayant donc été freinée par les risques supposés ou non qu'elle pourrait engendrée, les chercheurs ont développé des algorithmes pouvant simuler les caractéristiques des flux de travail traités dans les datacenters afin de générer sur les équipements des fonctionnements observables et dont les valeurs peuvent être stockées et traitées au même titre que celle issues de situations réelles. Ces algorithmes sont implémentés dans les simulateurs de cloud et dépendent des scénarii à tester. CloudSim, l'un des simulateurs les plus utilisés dans la littérature génère un volume de trafic correspondant à la durée de simulation souhaitée.

2.2.2 Traces réelles

2.2.2.1 Google Traces

Les données utilisées dans le cadre de ce travail proviennent des traces mise en ligne récemment par Google [Liu and Cho 2012], [Reiss et al. 2011]. Ces données concernent le trafic collecté pendant 29 jours dans un datacenter d'environ 12000 serveurs ayant des caractéristiques diverses. Ces traces rendues publiques constituent l'une des rares données d'étude sur le trafic dans les datacenters réels de taille significative pouvant permettre d'effectuer des analyses pertinentes et de tirer des conclusions crédibles. L'ensemble des données peut être scindé en trois catégories à savoir :

- les informations et les événements sur les serveurs incluant la création, la modification et la suppression des serveurs virtuels au sein du datacenter ;
- les données sur les requêtes et les tâches exécutées par les serveurs et

- les informations sur la consommation des ressources des serveurs tout au long de la période de collecte des données. C’est sur cette dernière catégorie de données que nous focaliserons notre analyse. Le dossier `task_usage` contient en effet une collecte des quantités de CPU (processeur), de mémoire RAM, d’espace disque etc consommées par les différentes tâches exécutées sur chaque serveur au sein du datacenter.

En raison du nombre de serveurs concerné, de la durée des collectes (chaque 5 minutes pendant 29 jours) et de l’importance du trafic traité dans le datacenter, le dossier `task_usage` se révèle très volumineux et donc requiert d’énormes puissances de calcul pour leur traitement et analyse. Pour cela, notre étude portera sur les 57 premiers fichiers de ce dossier (sur un total de 500 fichiers de 350 MegaOctet chacun) représentant environ 2 jours de collecte de données. Nous nous intéressons particulièrement aux quantités de processeur consommé sur les serveurs les plus sollicités du datacenter afin de donner une image pertinente et représentative de la sollicitation de ces serveurs. Nous procédons donc à une série d’extraction et de filtrage sur la collection de données afin d’obtenir un sous-ensemble correspondant à notre besoin. Certaines données ont été volontairement éliminées ou transformées par Google afin de garantir leur confidentialité. Bien que ces modifications n’empêchent pas notre travail, elles ne permettent pas d’obtenir les vraies valeurs des ressources CPU consommées sur les serveurs afin d’effectuer des calculs proches de la réalité. Nous nous contenterons donc d’obtenir des proportions de consommation de ressources susceptibles d’être confrontées avec des vraies valeurs de processeur pour tirer des conclusions.

2.2.2.2 PlanetLab dataset

PlanetLab est un jeu de donnée relatif au fonctionnement de multiples applications fourni de manière collaborative par un consortium d’acteur du domaine du numérique. Le concept de PlanetLab remplit trois objectifs principaux à savoir :

Une collection de machines réparties sur le globe

La plupart des machines sont hébergées par des instituts de recherche, bien que certains soient situés dans des centres de d'organisme indépendant. Toutes les machines sont connectées à Internet. L'objectif est que PlanetLab atteigne 1 000 nœuds largement distribués qui répondent à la majorité des backbones régionaux et internationaux d'Internet.

Un environnement logiciel uniforme

Toutes les machines PlanetLab exécutent un progiciel commun qui comprend un système d'exploitation basé sur Linux, un mécanisme d'orchestration des nœuds et de distribution des mises à jour logicielles, une collection d'outils de gestion qui surveillent l'état des nœuds, l'activité du système d'audit et les paramètres du système de contrôle ; et une installation pour la gestion des comptes utilisateurs et la distribution des clés. Ce logiciel est distribué sous forme de paquet, appelé MyPLC, que d'autres peuvent utiliser pour créer et déployer leurs propres «PlanetLabs privés».

L'objectif principal du logiciel est de soutenir la virtualisation distribuée c'est à dire la possibilité d'allouer une tranche de ressources matérielles à l'échelle du réseau PlanetLab à une application. Cela permet à une application de s'exécuter sur tous (ou certains) des machines réparties sur le globe, où, à tout moment, plusieurs applications peuvent être exécutées dans différentes tranches de PlanetLab.

Un banc d'essai de recouvrement

L'un des principaux objectifs de PlanetLab est de servir de banc d'essai pour les réseaux de recouvrement. Les groupes de recherche peuvent demander une tranche PlanetLab dans laquelle ils peuvent expérimenter divers services à l'échelle planétaire, y compris le partage de fichiers et le stockage en réseau, les réseaux de distribution de contenu, le routage et les superpositions de multidiffusion, les superpositions QoS, l'emplacement évolutif des objets, l'événement évolutif la propagation, les mécanismes de détection des anomalies et les outils de mesure du réseau. Il existe actuellement plus de 600 projets de recherche actifs sur PlanetLab.

L'avantage pour les chercheurs d'utiliser PlanetLab est qu'ils sont capables d'expérimenter de nouveaux services dans des conditions réelles et à grande échelle. Les exemples de services décrits ci-dessus bénéficient tous d'une large diffusion sur Internet : d'avoir des points de vue multiples à partir desquels les applications peuvent observer et réagir au comportement du réseau, être proches des nombreuses sources de données et des puits de données, et être distribués à travers plusieurs frontières administratives.

Une plate-forme de déploiement

En plus de soutenir des expériences à court terme, PlanetLab est également conçu pour prendre en charge les services à long terme qui prennent en charge une base de clients. C'est-à-dire plutôt que de voir PlanetLab strictement comme un banc d'essai, nous prenons la vision à long terme dans laquelle la superposition est à la fois un test de recherche et une plate-forme de déploiement, ce qui favorise la migration transparente d'une application à partir d'un prototype précoce, grâce à plusieurs itérations de conception, à un service populaire qui continue d'évoluer.

L'utilisation d'une superposition à la fois comme test de recherche et une plate-forme de déploiement est synergique. En tant que banc d'essai, la valeur de la superposition est de permettre aux chercheurs d'accéder à :

- Un large ensemble de machines géographiquement distribuées.
- Un substrat de réseau réaliste qui connaît des embouteillages, des défaillances et divers comportements de liaison.
- Le potentiel d'une charge de travail client réaliste.

Sa valeur en tant que plate-forme de déploiement est de fournir aux chercheurs un chemin de transfert de technologie direct pour les nouveaux services populaires et les utilisateurs ayant accès à ces nouveaux services. Nous croyons que le soutien de ces deux rôles est essentiel à la réussite du système.

Les services actuellement en cours d'exécution sur PlanetLab comprennent les CDN CoDeeN et Coral ; le service de mesure du réseau ScriptRoute ; l'accord et OpenDHT des services de localisation d'objets évolutifs ; et les services de surveillance du réseau PIER, Trumpet et CoMon.

Un microcosme de l'Internet suivant Non seulement les chercheurs évaluent et déploient des services utilisateurs finaux au-dessus de PlanetLab, mais nous nous attendons aussi à ce qu'ils développent des sous-services fondamentaux qui peuvent être repliés sur PlanetLab, améliorant ainsi l'installation pour les autres. Notre objectif à long terme est d'identifier les services de blocs de construction communs sur lesquels d'autres services et applications peuvent être construits, ou d'une autre manière, notre objectif est de comprendre comment Internet peut être conçu pour mieux supporter les superpositions.

Cette perspective est motivée par la question générale de savoir comment la communauté de recherche en réseau peut avoir un impact majeur sur l'Internet mondial. Malheureusement, le succès très commercial qui a alimenté notre dépendance accrue sur Internet a également réduit notre capacité à développer son architecture sous-jacente pour répondre aux nouvelles demandes et corriger les vulnérabilités émergentes. C'est parce que, comme l'a noté un récent rapport du Conseil national de recherches,

... les technologies réussies et largement adoptées sont sujettes à une ossification, ce qui rend difficile l'introduction de nouvelles fonctionnalités ou, si la technologie actuelle a fonctionné, de la remplacer par quelque chose de mieux. Les acteurs industriels existants ne sont généralement pas motivés à développer ou à déployer des technologies perturbatrices ...

Les réseaux de superposition offrent l'occasion d'introduire des technologies perturbatrices. Les nœuds de recouvrement peuvent être programmés pour supporter la nouvelle capacité ou fonctionnalité, puis dépendent des nœuds classiques pour fournir la connectivité sous-jacente. Au fil du temps, si l'idée déployée dans la superposition s'avère utile, il peut y avoir une motivation économique pour migrer la fonctionnalité dans le système de base, c'est-à-dire l'ajouter à l'ensemble de fonctionnalités des routeurs commerciaux. D'autre part, la fonctionnalité peut être suffisamment complexe pour qu'une couche de

recouvrement soit exactement là où elle appartient. Notre objectif global est de soutenir l'introduction de technologies perturbatrices sur Internet grâce à l'utilisation de réseaux de recouvrement. PlanetLab est un élément essentiel de cette vision.

Un consortium Le PlanetLab Consortium est une collection d'institutions universitaires, industrielles et gouvernementales qui coopèrent pour soutenir et améliorer le réseau de superposition PlanetLab. Il est responsable de la croissance à long terme de l'infrastructure matérielle PlanetLab ; concevoir et développer son architecture logicielle ; fournir un soutien opérationnel au jour le jour ; et définir des politiques qui régissent l'utilisation appropriée

plateforme libre pour le développement et le déploiement de scénarii de fourniture de services [Chun et al. 2003]

2.3 Caractéristiques de Google cluster data 2011

Cette section vise essentiellement à comprendre et interpréter le mode d'arrivé des trafics sur les serveurs au sein d'un datacenter. Cette compréhension des propriétés du trafic permettra à terme de proposer une meilleure approche de répartition des ressources matérielles du datacenter sur la charge globale de celui-ci dans l'optique de réduire les pertes d'énergie électrique. En effet, plusieurs travaux précédents ont établi que les serveurs dans les datacenters affichent un faible taux d'utilisation des processeurs tandis qu'ils consomment une grande quantité d'énergie. Une proportion importante de cette énergie est donc perdue. Pour pallier ce problème, il est nécessaire de trouver un mécanisme pour répartir suffisamment de charge aux processeurs des serveurs afin de rentabiliser sa consommation énergétique. Ce travail présenté dans [F. Gbaguidi and Ezin 2015] consiste à analyser les caractéristiques des traitements effectués sur les serveurs dans un datacenter réel de grande capacité et de mettre un focus sur :

- La proportion des demandes arrivant vers un serveur par rapport à ces ressources (processeur) disponibles
- La proportion des ressources processeurs effectivement consommées par rapport

TABLE 2.1 – Taille des fichiers dans Google cluster

Table	Number of files	total size
machine_events	1	3 MB
machine_attributes	1	1180 MB
job_events	500	315 MB
task_events	500	15000 MB
task_constraints	500	3000 MB
task_usage	500	160000 MB

aux ressources réservées par les requêtes

- La quantité approximative d'énergie fournie et consommée par un serveur au cours d'une période donnée
- La proportion des requêtes en fonction de la priorité des services
- La quantité d'énergie réservée et consommée par les requêtes non abouti

2.3.1 Dataset description

Les données de notre projet proviennent d'un datacenter de Google au sein duquel, différents types de statistiques ont été collectées pendant une période de 29 jours. Ce datacenter contient plus de 12 000 serveurs avec des caractéristiques variées et traitant des requêtes ayant des priorités différentes et une tolérance variable aux latences. Nous avons trouvé au sein de ce jeu de données de précieuses informations pour notre expérimentation même si elles présentent quelques insuffisances négligeables. Dès sa publication en 2011, plusieurs chercheurs ont investi ce jeu de données pour sortir des analyses statistiques aussi riches que variées. Nous adoptons donc ce jeu de données car comparé à d'autres traces publiées par Yahoo, et PlanetLab, il est très représentatif des types de datacenters les plus répandus et convient parfaitement à notre cadre d'expérimentation. Comme le montre le tableau 2.1, c'est une collection de données d'environ 40 Gigaoctets comprenant six dossiers libellés `job_events`, `machine_attributes`, `machine_events`, `task_constraints`, `task_events` et `task_usage` ainsi que trois fichiers de signature, un fichier `Readme` et un fichier `schema.csv` (structure des différents dossiers). Les collectes de données sont réalisées par intervalle de temps de 600 secondes normalisé pour tous les fichiers et portent sur

diverses caractéristiques du trafic au sein du cluster. Les fichiers de traces consistent en des fichiers .CSV archivés afin d'en réduire la taille et faciliter le téléchargement. Certaines données ont été transformées afin de préserver leur confidentialité notamment, les valeurs de capacité de CPU ou de mémoire ont été divisées par la valeur maximale afin de n'obtenir que des données entre 0 et 1 et les noms d'utilisateurs, spécifications des plateformes par exemple ont été chiffrés.

`machines_events` Cette table comprend 6 colonnes incluant le temps, l'identifiant, un type d'opération, les spécifications de la plateforme ainsi que les capacités de CPU et de mémoire. L'intérêt principal ici est de présenter les différents états des 12 000 machines tout au long de la période de collecte. Ainsi, les types d'opérations sont 0 pour les ajouts de machine dans le cluster, 1 pour les suppressions et 2 pour les modifications. Ces modifications peuvent porter sur les capacités de CPU et de mémoire. C'est un dossier comprenant un seul fichier de 3 Mégaoctets.

`machines_attributes` ce dossier contient un fichier qui présente à chaque instant et pour chaque serveur la ou les valeurs de certains attributs tels que la version du noyau, la vitesse de l'horloge, la présence d'une adresse Ip externe etc. Certains de ces attributs sont spécifiés comme contraintes sur certaines requêtes et déterminent donc la planification de ces derniers. `Machine_attributes` est un fichier .CSV unique de 1,18 Gigaoctet.

`jobs_events` les jobs représentent les requêtes envoyées vers les serveurs et peuvent contenir plusieurs tasks (instructions) qui sont collectés séparément dans la table `task_events`. Les jobs et les tasks sont collectés à divers moments de leur cycle de transition dont un aperçu est donné dans la figure 1 ci-après. La table `job_events`, comprend cinq cent fichiers .CSV zippé d'une taille totale d'environ 315 Mégaoctets. Chaque fichier .CSV contient principalement un identifiant, le libellé de la requête, le statut présenté à l'instant précis, le nom de l'utilisateur, la classe de planification à laquelle appartient la requête ainsi qu'un champ « missing info » qui est rajouté sur les requêtes pour lesquels l'information sur leur création ou leur suppression n'est pas retrouvée. L'attribut classe de planification

(scheduling class) représente le niveau de tolérance de la requête aux latences et intervient dans les décisions d’ordonnancement.

tasks_constraints certaines requêtes contiennent des contraintes spécifiques sur les caractéristiques des serveurs sur lesquels elles doivent s’exécuter. Ces contraintes sont collectées dans cinq cent fichiers .csv faisant une taille totale d’environ 3 Gigaoctets. Les contraintes sont libellées avec les opérateurs de comparaison appliqués à la valeur indiquée.

task_events le dossier contient cinq cent fichiers .csv faisant un total de 15 gigaoctets. Tout comme les fichiers de requêtes, les données sur les instructions exécutées (task) comprennent un identifiant d’instruction et l’identifiant du job auquel il appartient et celui du serveur sur lequel il est exécuté, le statut courant, le scheduler class, la priorité, le nom de l’utilisateur ainsi que les quantités de ressources nécessaires (CPU, mémoire et disque). Ici la priorité représente une classification du type de requête. Certaines tâches peuvent présenter une contrainte (different machine constraint) d’exécution sur une plateforme différente de celle sur laquelle la requête est exécutée.

tasks_usage c’est un grand dossier de cinq cent fichier d’une taille totale de près de 160 gigaoctets. Chacun des fichiers .csv présente près d’une heure et demi de données sur les niveaux de consommation de ressources de chaque instruction. Elle présente sur chaque collecte les identifiants de la requête, du task et du serveur ainsi que différentes informations sur la consommation du processeur, de la mémoire et du stockage.

2.3.2 Analyse des traces

Notre outil de travail est le logiciel libre de statistique R que nous avons installé dans un environnement Windows 8 version 64 bit. En raison des trop grosses tailles de données, nos traitements à l’étape actuelle se limiteront à des échantillons allant de quelques heures à quelques jours en fonction des tables manipulées. Ainsi, nous produisons les sorties ci-après

2.3.2.1 Quantités de ressources sur les serveurs

Les valeurs normalisées des capacités des processeurs et mémoire RAM retrouvées dans la population des 12583 serveurs appartenant au cluster 2.3se présentent comme suit : Les valeurs distinctes de la fréquence des processeurs sont 0.25, 0.5 et 1. Celles de la taille de la mémoire RAM sont également environ 0.25, 0.5, 0.75 et 1. Rappelons que ces valeurs ne sont pas les caractéristiques réelles des serveurs mais des valeurs volontairement normalisées par Google afin de préserver la confidentialité des spécifications techniques desdits serveurs.

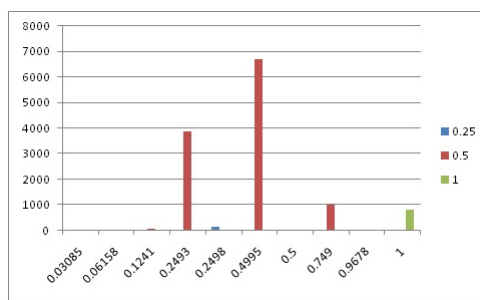


FIGURE 2.3 – Nombre de serveur par CPU et RAM

La plupart des serveurs disposent d'un processeur de 0.5 (10567, soit 84% du cluster) dont 6709 disposent d'une mémoire de 0.5 et 3858 une mémoire de 0.25. Il existe 4 plateformes distinctes correspondant principalement au type d'architecture de la carte mère et à la version du chipset à l'intérieur du serveur. Ci-après la répartition des configurations de processeurs et de mémoire par type de plateforme.

Les serveurs correspondants à l'architecture de la plateforme 3 disposent tous d'un processeur de 0.5 avec des valeurs de mémoire RAM différentes comme présenté sur les figures 2.4 et 2.5 Les « event type » (opérations) sur les serveurs correspondent soit à un ajout (0), une suppression (1) ou une reallocation des ressources (2) de ces derniers au sein du cluster. Au début de la trace, 12478 ont été inclus dans le cluster avant que d'autres opérations comme les suppressions et les réallocations de ressources ne démarrent. Rappelons que cette opération dénommée « consolidation » dans les clusters consiste à éteindre des serveurs lorsqu'ils affichent des taux d'utilisation très bas (suppression), de

leur diminuer ou augmenter les ressources à la demande (2) ou de les rallumer (0) en cas de nécessité.

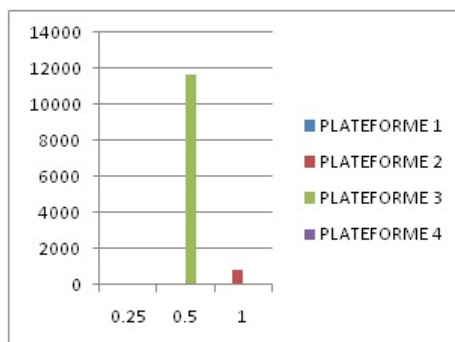


FIGURE 2.4 – Serveur par CPU

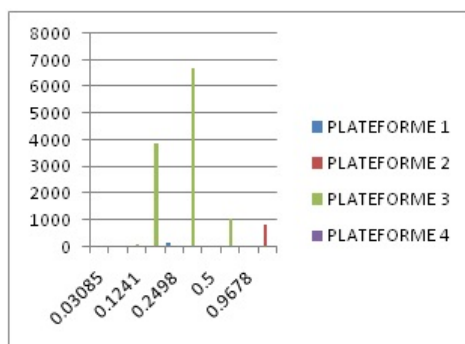


FIGURE 2.5 – Serveur par RAM

2.3.2.2 Caractéristiques des requêtes et tâches

Chaque requête contenue dans la trace peut être composée de plusieurs tâches et peut se présenter dans un des états présentés dans la figure 2.6 (champ « event type »). Les requêtes et les tâches sont classées suivant certaines catégories de sensibilité aux latences représentées de 0 à 3. Chaque tâche a un niveau de priorité qui va de faible (trafic moins important), à élevé (trafic de production). Chaque tâche est reliée à un ensemble de ressources sollicitées (processeur, mémoire et disque). En raison du trop grand volume des données relatives aux requêtes et tâches, nous travaillons juste sur un échantillon portant sur une période de 14 heures. Nous présentons ici quelques statistiques y relatif

La figure 2.7 montre que la plupart des tâches sont insensibles aux latences et donc que

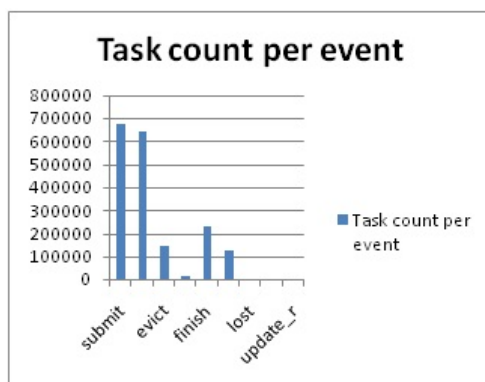


FIGURE 2.6 – Task per event

les traces présentent un nombre très faible de tâches sensibles aux latences (classe 2 et 3). D'un autre côté, on remarque qu'il existe dans les traces un grand nombre de tâches planifiées représentées par les événements de type submit (675721) et schedule (639725) tandis qu'une quantité faible d'entre elles sont réellement achevées (finish=230980). Ceci signifie que plus de 50% des tâches planifiées sur les serveurs échouent ou sont annulées. Le graphique 2.8 montre que plus de 90% des tâches exécutées sur les serveurs sont

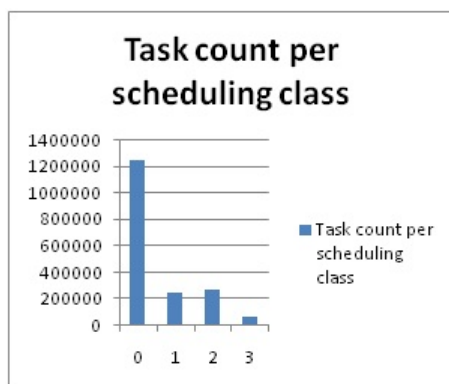


FIGURE 2.7 – Task per scheduling

d'une priorité faible. Ceci signifie que les tâches de production occupent une très faible proportion dans le trafic global du datacenter.

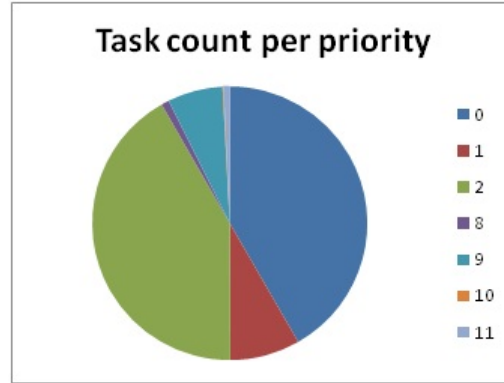


FIGURE 2.8 – Task per priority

TABLE 2.2 – Taux de consommation sur six serveurs très sollicités du cluster

Servers	Mid	CPU/RAM	NB of task	requested CPU	CPU used
Server 1	4802479866	0.5/0.5	443	27.610240	2.619854
Server 2	294880136	0.5/0.5	441	19.103591	1.108617
Server 3	4820183646	0.5/0.5	437	22.495116	2.800313
Server 4	4820508012	0.5/0.5	433	25.222442	2.761486
Server 5	2096989215	0.5/0.5	416	6.317939	1.137741
Server 6	1301865	0.5/0.5	372	10.363938	1.137741

2.3.2.3 Taux d'occupation des serveurs

Au sein de la table des taches 2.2, nous avons extrait six des serveurs qui exécutent le plus grand nombre de requête. ci-après quelques informations relatives à ces serveurs Ci-après sur la figure 2.9 une représentation graphique du taux de consommation de ressource processeur sur les serveurs les plus sollicités.

2.3.2.4 Tendence de consommation électrique identifiée

Plusieurs travaux ont établi une corrélation entre l'activité des processeurs d'un serveur et sa consommation électrique. Toutefois, ce ratio CPU/Energie varie considérablement en fonction des technologies de processeurs. L'estimation de cet indicateur à l'étape actuelle se basera juste sur des valeurs empiriques. Par ailleurs, la normalisation des valeurs de consommation de CPU effectuée sur la présente trace ne nous permet pas de connaitre avec précision la quantité de Mhz réelle à laquelle correspondent les valeurs indiquées. En

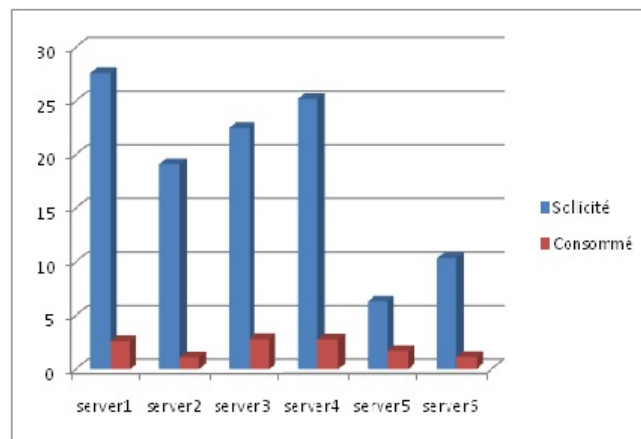


FIGURE 2.9 – Usage of processors

prenant les ratio publiés par l'un des constructeurs de processeurs le plus retrouvés dans les datacenters, il ressort clairement que les quantités de cycle processeur non utilisés dans la présente trace correspondent intuitivement à des quantités d'énergie perdu dans les mêmes proportions.

Conclusion

l'explosion du volume de trafic dans les datacenters pose un défi à la communauté scientifique qui doit trouver les meilleurs algorithmes pour faciliter les échanges de données en constante croissance sur les ressources technologiques du moment tout en limitant la quantité d'énergie nécessaire. Le défi est de taille et le chemin pour y arriver peut se révéler complexe. La recherche dans le domaine des datacenters et plus spécifiquement sur les questions de l'efficacité énergétique s'effectue principalement dans les environnements de simulation au regard de l'impossibilité de valider des théories sur de réels datacenters qui en production requiert la plus grande criticité. le choix des outils est donc crucial. Notre état de l'art nous a permis d'une part d'identifier quelques critères pertinents pour un choix judicieux de l'environnement de la recherche sur les datacenters en vue d'une analyse comparative des outils existants et d'autre part de collectionner et d'analyser les types de données les plus utilisés dans la littérature. Afin d'améliorer notre compréhension de ces données, nous avons étudié les caractéristiques de la collection de traces Google cluster data 2011 laissée dans le domaine public. Ceci nous a permis d'appréhender les spécifications technique des serveurs, les types de requêtes et de tâches, les contraintes de sensibilité à la latence des requêtes, les priorités dans le processus d'ordonnancement des tâches et les taux d'occupation d'environ 10% des ressources des serveurs. Ce dernier aspect étant une source potentielle d'inefficacité énergétique dans les datacenters, ouvre donc la voie à notre recherche au cours de laquelle nous tenterons de proposer une nouvelle approche de résolution de la question.

Deuxième partie

**PARTIE 2 : PREDICTIBILITE DES
BESOINS EN RESSOURCES AU
SEIN DES DATACENTERS**

Résumé de la partie 2

Cette partie est consacrée à la prédictabilité de la consommation des ressources sur les serveurs dans les datacenters. Notre objectif est d'établir le niveau de fiabilité des outils de prédiction afin de soutenir notre approche anticipative de l'efficacité énergétique des serveurs. Cette étude se compose de deux chapitres à savoir :

Etude des outils statistiques de prédiction : dans le premier chapitre de cette partie, nous avons présenté de façon synthétique les aspects théoriques de l'outil ARMA , la détermination des paramètres du modèles et la démarche de la prédiction. Ensuite nous avons présenté le cadre de notre étude qui consiste à étudier les performances des outils ARMA et de la méthode de prédiction dite naïve sur les consommations de ressources sur les serveurs sélectionnés au sein de notre jeu de données. Ces consommations ont été collectées sur une durée suffisante pour une modélisation efficace du phénomène. Il ressort de notre étude que les outils ARMA parviennent à établir des modèles d'évolution des valeurs des ressources avec des taux d'erreurs en dessous de 40%. En outre, l'analyse comparative avec la méthode naïve révèle une fiabilité supérieure d'environ 30% des outils ARMA dans le contexte de l'étude.

Etude des outils d'apprentissage par ordinateur : Ce chapitre se focalise sur l'adaptabilité des réseaux de neurones pour la prédiction des ressources nécessaires sur les serveurs à des horizons bien définis. Après une clarification conceptuelle des outils à utiliser, notre travail à consisté dans un premier temps à adapter les jeux de données aux besoins de notre expérience, notamment à travers la préparation des fichiers d'entrée des réseaux comportant les données de 4, 8 et 12h en vue d'une prédiction respectivement des

besoins de la 5ième, la 9ième et de la 13ième heure. Ensuite nous avons établi différentes architectures de réseaux pour répondre à l'objectif ci-dessus énoncé avec une possibilité de variation du nombre de couche intermédiaire. Ainsi avec les réseaux à 4 entrées, à 8 entrées et à 12 entrées, avec une ou deux couches intermédiaires, nous avons testé nos jeux de données et obtenus les résultats suivants : Les réseaux de neurones à 8 entrées et une couche intermédiaire ont produit les taux d'erreurs les plus faibles. Nous retenons au total qu'il est possible d'adapter les réseaux de neurones au contexte des données sur les ressources consommées sur les serveurs. Cependant, la phase de choix et d'entraînement des réseaux est déterminante pour l'obtention de résultats fiables.

A la fin de cette partie, nous avons effectué une analyse comparative des résultats obtenus avec les deux familles d'outils. Il en ressort que bien que l'analyse ne pourrait être généralisé à tous les cadre d'étude, les outils ARMA présentent une fiabilité supérieure à celle des réseaux de neurones dans une proportion de 30%. En outre, l'utilisation des réseaux de neurones peut se révéler chronophage dans des cas précis où le temps réel et les prédictions à très court terme sont requis.

Chapitre 3

PREDICTION DES BESOINS EN RESSOURCES BASEE SUR LES OUTILS STATISTIQUES

Introduction

Notre approche prédictive consiste à implémenter des méthodes dynamiques d'observation des tendances d'évolution des trafics arrivant sur les serveurs afin d'en définir les formules mathématiques correspondantes et de déterminer en temps opportun les plans de réaffectation nécessaires. Cette stratégie produit des résultats assez satisfaisants dans différents domaines de recherche et commence par être très explorée dans le domaine de l'informatique notamment pour la planification de ressources. Il s'agit dans le cadre de notre étude, de trouver sur la base d'une analyse des tendances de sollicitation des ressources matérielles des serveurs, les plans de répartition des machines virtuelles qui préviennent le mieux les phénomènes de sous-consommation susceptible d'engendrer des pertes d'énergies ou de sur-consommation pouvant entraîner une indisponibilité des services et une violation des SLA. Ce chapitre, aborde l'apport de la prédiction basée sur les outils statistiques. Plusieurs méthodes se concurrencent pour cette finalité. Leur choix dépend fortement du type de problème, des objectifs poursuivis ainsi que des types de données disponibles pour leur validation. Les méthodes de prédiction naïve et celle de "regression local" sont implémentées dans certains outils de simulation comme CloudSim et peuvent contribuer à anticiper sur les migrations de machines virtuelles et ainsi éviter les sur/sous consommation et préserver la continuité de service. Cependant, leur simplicité contraste avec la dynamique et la complexité des trafic traités au sein des datacenters. Etant donné que les données de consommation de ressources peuvent se présenter sous forme de série temporelle, nous avons estimé qu'au delà des méthodes simpliste d'anticipation des charges à venir, il est possible d'obtenir de meilleurs résultats en appliquant des méthodes de régression linéaire au regard de leur large adoption au niveau de la littérature.

Notre choix porte sur les méthodes ARMA (AutoRegression Moving Average) qui représentent d'excellent cadre de travail pour les séries temporelles et surtout pour leur adaptabilité à différents types de jeux de données. Ces méthodes très plébiscité en statistique, ont largement servi dans l'analyse prédictive notamment dans les domaines de l'analyse économique, de la gestion prédictive des flux de trafic dans les systèmes intelligents de transport (ITS) et dans de nombreux autres domaines et pour des problématiques très proches de la gestion prédictive de la consommation énergétique qui est l'objectif principal dans notre cas. Il s'agit alors de démontrer l'adaptabilité de cette méthode pour les environnements de Cloud et de formaliser la démarche permettant sa mise en œuvre. Ceci nous permet d'élaborer une technique fiable de prédiction des consommations futures des processeurs sur les serveurs au sein des datacenters et d'anticiper la décision d'ajustement de leur plan d'occupation pouvant rationaliser au mieux la consommation de l'énergie électrique approvisionnée.

Notre évaluation des méthodes statistiques pour la prédiction de consommation de ressources au sein des datacenters révèle une amélioration de la fiabilité des valeurs prédites de l'ordre de 30% en comparaison aux méthodes naïves. Ce résultat est une contribution à l'amélioration des algorithmes de simulation dans le domaine du Cloud étant donné que les modèles ARMA ont été que très peu testés sur la prédiction des consommation de ressources dans les datacenters et que les données à partir desquelles nous validons ce travail proviennent de traces réelles contrairement aux données synthétiques qui ont été souvent utilisées par ailleurs.

3.1 Modèle ARMA

Cette section aborde le aspects théoriques des modèles ARMA dans ces différentes composantes et rappelle brièvement la démarche de la détermination de ces paramètres.

3.1.1 Généralités

Telles qu'elles se retrouvent partout autour de nous, les séries temporelles représentent tout événement observable prenant des valeurs variables formant une séquence indexée au temps. Elles se présentent sous la forme $x_1, x_2, x_3, \dots, x_n$ avec x_1 , la première observation de l'événement, x_2 , la seconde ainsi de suite. Les modèles ARMA (AutoRegressive Moving Average) [Hibon and Makridakis 1997] aussi connu sous le nom d'approche de Box-Jenkins est une méthode d'estimation d'une série par association d'une composante moyenne mobile et d'une autoregression. On suppose que dans sa dynamique, les valeurs actuelles d'une série temporelle dépendent fortement des valeurs précédentes et par conséquent, une connaissance de l'historique de cette série peut permettre d'en prédire les valeurs futures. La forme de base des modèles autoregressifs est le ARMA. Il existe des variantes adaptées à différents types de données dont celles comportant une tendance et/ou une saisonnalité. Sa manipulation se déroule en trois phase à savoir :

- identification du modèle ;
- estimation des paramètres ;
- test du modèle.

La détermination des paramètres fiables du modèle ARMA est cependant un exercice délicat et nécessite une démarche rigoureuse ; un mauvais modèle pouvant aboutir à des prédictions totalement erronées. Les outils ARMA sont largement utilisés dans les prédictions statistiques et sont réputés pour la fiabilité des résultats produits sur des données des différents types. Nous choisissons donc d'effectuer une modélisation puis un test de fiabilité sur les prédictions de notre série temporelle d'utilisation des ressources processeur au sein du datacenter en utilisant cet outil.

Ces modèles sont applicables à la fois aux données saisonnières et aux données non saisonnières. La première étape de notre expérimentation consiste à porter un jugement sur les caractéristiques essentielles de notre série pour confirmer ou infirmer nos hypothèses

sur la pertinence des choix finaux de nos outils. Après cette étape nous évaluerons les aptitudes des différents outils à effectuer des prédictions fiables de notre série temporelle.

3.1.2 Les processus Autoregressifs (AR)

Un processus autorégressif est un processus où l'on écrit une observation au temps t comme une combinaison linéaire des observations passées plus un certain bruit blanc [Udny Yule 1927]. La suite $\{X_t : t \geq 0\}$ est un processus autorégressif d'ordre p avec ($p > 0$) s'il peut s'écrire sous la forme suivante :

$$X_t = \sum_{k=1}^p \theta_k X_{t-k} + \epsilon_t$$

Les $\theta_k (k = 1, 2, \dots, p)$ constituent les paramètres du modèle,

$$\epsilon_t = \phi(B)X_t$$

avec

$$\phi(B) = 1 - \phi_1 B - \phi_2 B^2 - \dots - \phi_p B^p$$

3.1.3 Les processus Moving Average (MA)

On dit que la suite $\{X_t : t \geq 0\}$ est un processus à moyenne mobile d'ordre q avec ($q > 0$) si celui-ci peut s'écrire sous la forme suivante :

$$X_t = \sum_{k=1}^p \theta_k \epsilon_{t-k} + \epsilon_t$$

où les $\theta_k (k = 1, 2, \dots, q)$ sont les paramètres du modèle. Dans ce cas, on note $\{X_t \sim MA(q)\}$. [Robinson 1977]

3.1.4 Les processus ARMA

Les processus ARMA [Hibon and Makridakis 1997] se composent d'une partie AR (AutoRegressive) et d'une partie MA (Moyenne Mobile) et s'écrit ARMA(p, q) avec p et q respectivement les ordres des composantes AR et MA. En matière de prédiction, les outils ARMA sont fréquemment utilisés notamment en raison de leur adaptabilité

à une large gamme de type de données, qu'elles soient stationnaires ou non, qu'elles soient saisonnières ou non [Hibon and Makridakis 1997], [Box 2012] et [Hoff 1983]. Définit comme AutoRegressive Moving Average et aussi connu sous le nom approche de Box-Jenkins ces outils ont servi à modéliser des phénomènes de différents types dans les domaines tels que l'économie, l'industrie, le commerce, les ITS, et l'énergie. Un processus est dit ARMA(p,q) s'il existe des suites réelles $\{\phi_k\}$ and $\{\epsilon_k\}$ tel que

$$X_t = \sum_{k=1}^p \phi_k X_{t-k} = \epsilon_t + \sum_{j=1}^q \theta_j \epsilon_{t-j}$$

avec $\{\epsilon_t\} \sim WN(0, \sigma_\epsilon^2)$. On peut aussi utiliser les polynômes $\phi(B)$ and $\theta(B)$ pour réécrire ce modèle sous la forme :

$$\phi(B)X_t = \theta(B)\epsilon_t$$

avec

$$\phi(B) = 1 - \phi_1 B - \phi_2 B^2 - \dots - \phi_p B^p$$

et

$$\theta(B) = 1 - \theta_1 B - \theta_2 B^2 - \dots - \theta_q B^q$$

On note

$$\{X_t \sim ARMA(p, q)\}$$

Remarque. On note certaines propriétés pour les modèles ARMA(p,q) :

1. Si $p = q = 0$, on a $X_t = \epsilon_t \sim WN(0, \sigma_\epsilon^2)$
2. Si $p = 0$ et $q \neq 0$, on a

$$\{X_t \sim MA(q)\}$$

3. Si $q = 0$ et $p \neq 0$, on a

$$\{X_t \sim AR(p)\}$$

3.1.5 Détermination des modèles ARMA

3.1.5.1 Méthode des critères d'erreurs

Le principe consiste à procéder dans un premier temps à l'identification d'un modèle correspondant le plus possible aux caractéristiques des données, puis à estimer dans un second temps les paramètres du modèle et enfin à tester ce modèle à l'épreuve du phénomène à modéliser. Cette démarche est répétée autant que nécessaire jusqu'à l'obtention des résultats escomptés. Autrement, cela revient en la détermination du couple (p, q) permettant d'obtenir les valeurs optimales des critères d'erreurs. Les critères d'erreurs les plus utilisés sont l'AIC (*Akaike's Information Criterion*) [Akaike 2011] et le BIC (Bayesian Information Criterion) [Weakliem 1999]. Les critères AIC et BIC pour un processus ARMA(p,q) prennent la forme suivante :

$$AIC(p, q) = \log(\hat{\sigma}_\epsilon^2) + 2\frac{p+q}{T}$$

et

$$BIC(p, q) = \log(\hat{\sigma}_\epsilon^2) + \frac{p+q}{T} \log(T)$$

Il revient alors à minimiser l'une de ces fonctions afin de déterminer les valeurs optimales de p et q . Ces méthodes sont rapidement très coûteuses en calcul car il faut effectuer l'estimation des coefficients de tous les modèles ARMA possibles (jusqu'à un certain ordre maximal a priori) et. Afin de minimiser ces fonctions, on a généralement recours à des algorithmes récursifs qui consiste à faire deux boucles itératives sur p et q pour tester tous les couples (p, q) jusqu'à certaines bornes $p < P$ et $q < Q$. À l'intérieur de ces boucles, on calcule d'abord les estimateurs ϕ , θ et $\hat{\sigma}_\epsilon^2$ utilisant par exemple les moindres carrés ou le maximum de vraisemblance, on calcule les critères AIC et BIC pour ces différents ordres et on trouve le minimum de ces quantités. On a donc les valeurs \hat{p} et \hat{q} qui minimisent l'AIC ou le BIC. Ensuite, on calcule des estimateurs efficaces des paramètres du modèle $ARMA(\hat{p}, \hat{q})$ utilisant la méthode du maximum de vraisemblance.

3.1.5.2 La méthode du maximum de ressemblance

En statistique, l'une des méthodes les plus utilisées pour l'estimation de paramètres est celle du maximum de vraisemblance. C'est Ronald Fisher [Fisher 1925] qui, en 1925, a publié sa méthode initialement appelée « Le critère absolu ». À l'époque, l'une des méthodes couramment utilisées était celle des moindres carrés. Soit l'échantillon $\{X_1, X_2, \dots, X_T\}$, la vraisemblance quantifie la probabilité que ces observations proviennent d'une loi paramétrique, par exemple une loi normale. La vraisemblance correspond donc au produit des fonctions de densité des X_i . On peut montrer que la vraisemblance, sous l'hypothèse de normalité, est donnée par l'expression :

$$\Gamma(R_T^X) = (2\pi^{-\frac{T}{2}} \cdot |\det(R_T^X)|^{\frac{1}{2}} \cdot e^{-1} 2X_T'(R_T^X)^{-1} X_T$$

où R_T^X est la matrice de covariance dont l'élément (i, j) est donné par $r^X(|i - j|)$. On écrit la vraisemblance en fonction des paramètres ϕ , θ et $\hat{\sigma}_\epsilon^2$ et on maximise la vraisemblance par rapport à ces paramètres. Cette méthode peut être utilisée pour n'importe quel modèle tandis que les équations de Yule-Walker sont plus reconnues pour les processus autorégressifs dû à la linéarité des équations [Weakliem 1999]. Si la série est stationnaire, causale et inversible, ces estimateurs auront la propriété d'être asymptotiquement normaux. La plupart des logiciels de statistique utilisent des algorithmes itératifs pour calculer ces estimateurs.

Le modèle ainsi obtenu peut être ensuite utilisé pour effectuer des prédictions.

3.1.5.3 Prédicteur pour un processus ARMA

On considère le processus ARMA(p, q) tel que

$$X_t = \phi_1 X_{t-1} + \dots + \phi_p X_{t-p} + \epsilon_t + \theta_1 \epsilon_{t-1} + \dots + \theta_q \epsilon_{t-q}$$

avec $(\theta_p, \phi_q) \in \mathbf{R}^{2*}$ et $\epsilon_t i.i.d(0, \sigma_\epsilon^2)$. Appliquons le théorème de Wold au processus $\{X_t, t \in \mathbf{Z}\}$ et considérons la forme $MA(\infty)$ correspondante

$$X_t = \sum_{j=0}^{\infty} \pi_j \epsilon_{t-j}$$

avec $\pi_0 = 1$.

Il s'en suit que la meilleure prédiction possible de X_{t+1} prend en compte toute l'information disponible à l'instant t noté $\hat{X}_t(1)$ donné par :

$$\begin{aligned}\hat{X}_t(1) &= E(X_{t+1}/X_t, X_{t-1}, X_{t-2}, \dots, X_0) \\ &= E(X_{t+1}/\epsilon_t, \epsilon_{t-1}, \epsilon_{t-2}, \dots, \epsilon_0) \\ &= \sum_{j=1}^{\infty} \pi_j \epsilon_{t+1-j}\end{aligned}$$

En conséquence, l'erreur de prédiction est donnée par la nouvelle information à l'instant $t + 1$ et qui n'était pas connu à l'instant t . Alors :

$$X_{t+1} - \hat{X}_t(1) = \epsilon_{t+1}$$

De façon générale, pour une prédiction à l'horizon k on a :

$$\begin{aligned}\hat{X}_t(k) &= \sum_{j=k}^{\infty} \pi_j \epsilon_{t-k-j} \\ X_{t+k} - \hat{X}_t(k) &= \sum_{j=k}^{k-1} \pi_j \epsilon_{t-k-j}\end{aligned}$$

3.1.6 Evaluation des prédictions

La réalisation d'une bonne prédiction suppose l'obtention des métriques d'erreurs les plus faibles. Dans le domaine statistique, plusieurs indicateurs de mesures sont utilisées, à savoir :

- **ME (Mean error)** : C'est une moyenne arithmétique des erreurs relevées entre une observation et celle qui la précède.
- **MSE : (Mean Square Error)** : c'est la moyenne arithmétique des carrés des écarts entre les prévisions et les observations. Elle représente l'indicateur d'erreurs la plus utilisée en raison de la simplicité de son calcul et de son adaptabilité à différentes sortes de processus.
- **RMSE** : racine carrée du MSE

- **MAE : (Mean Absolute Error)** : moyenne arithmétique des valeurs absolues des écarts
- **MAPE : (Mean Absolute Percentage Error)** : moyenne des écarts en valeur absolue par rapport aux valeurs observées. C'est donc un pourcentage et par conséquent un indicateur pratique de comparaison. Hélas, petit inconvénient, le MAPE ne peut s'appliquer qu'à des valeurs strictement positive
- **MASE** : proposé par [Hyndman and Koehler 2006] est une alternative au MAPE lorsqu'il existe dans la série, des données nulles ou très proches de zéro comme c'est le cas ici. Les valeurs de MASE inférieures à 1 indiquent que le modèle permet de réaliser de meilleures prédictions que les modèles dits « naïfs » et inversement, le modèle est inadapté lorsqu'elles sont supérieures à 1.

Afin de déterminer la fiabilité de nos prédictions, nous calculons à chaque occurrence, les valeurs de MSE afin de mesurer d'une part par rapport aux valeurs réelles obtenues, le niveau de biais introduit et d'autre part vérifier les performances de notre modèle par rapport à la méthode naïve.

3.2 Prédiction ARMA des consommations de ressources sur les serveurs

3.2.1 Présentation du problème

Au sein des datacenters, les constructeurs dotent désormais les serveurs physiques de d'importantes ressources processeurs, mémoires ou disques en vue de permettre de créer des quantités élevées de serveurs virtuels. Ce sont ces derniers qui reçoivent les nombreuses requêtes provenant des utilisateurs dont les besoins ne s'arrêtent pas. Les serveurs virtuels partagent de façon concurrente les ressources des machines physiques et il apparaît généralement que les valeurs réservées sont parfois soit sous-utilisées ou surutilisées.

La sous-utilisation des ressources entraîne la sous consommation de l'énergie électrique fournie à la machine physique étant donné que la plupart des alimentations des serveurs

fournissent leur puissance nominale quel que soit le taux d'utilisation tandis que la surutilisation pourrait causer des délais de latence très longs et entraîner une violation des accords de niveau de service souvent établis entre les opérateurs de Cloud et leurs clients. Notre objectif est donc de surveiller le taux d'utilisation des ressources par les serveurs virtuels en vue de s'assurer à tout moment qu'elles seront consommées de façon optimale. Pour ce faire, il est indispensable de prévoir à l'avance les taux d'occupation desdites ressources. Plusieurs méthodes existent. Nous essayons ici d'utiliser les méthodes statistiques basées sur les modèles ARMA pour prédire à un horizon déterminé les valeurs probables d'utilisation des ressources des serveurs physiques au sein des datacenters. Nous utilisons dans ce cadre, les traces historiques issues d'un datacenter de Google dont nous extrayons les valeurs de consommation de processeurs sur cinq (05) serveurs pendant 29 jours de collecte [Wilkes 2011]. Les graphes d'utilisation de processeur sur ces serveurs se présentent sur les graphes ci-après. Sur ces données que nous avons échantillonnées par heure d'utilisation, nous déterminons le modèle ARMA correspondant puis nous calculons une prédiction de la consommation au cours de l'heure suivante. Ce résultat est ensuite comparé avec celui obtenu sur les mêmes données en appliquant la méthode « naïve ».

3.2.2 Détermination des paramètres du modèle ARMA des consommations énergétiques

On remarque sur le graphe ci-dessus que nos 5 séries temporelles sont stationnaires et ne présente pas de tendance apparente. Il est donc aisé d'appliquer la démarche précédemment décrite pour déterminer le modèle ARMA le plus adapté à ce type de donnée. Nous rechercherons donc les valeurs de p et d permettant de minimiser les critères statistiques AIC et BIC en recourant à la méthode de la recherche du maximum de ressemblance.

Notre fichier de données comprenant un nombre élevé de valeurs de consommation de processeurs (plus de 8000 par serveur), nous faisons le choix de tester 15 paires de paramètres symétriques et obtenons les valeurs optimales pour les deux critères (AIC et BIC) pour le couple $p = q = 14$. Avec ces valeurs, nous appliquons dans une fenêtre glissante notre modèle au jeu de données pour obtenir une prédiction basée sur les n précédentes

observations à travers un algorithme récursif.

3.2.3 Fenêtre de prédiction

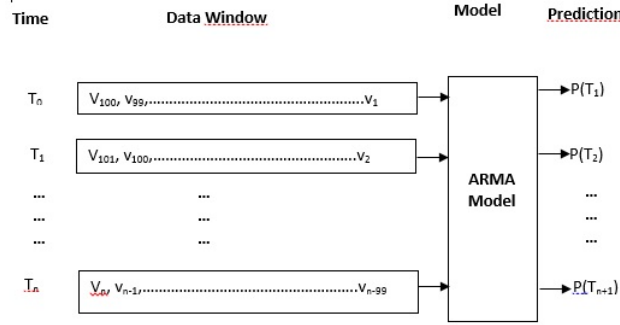


FIGURE 3.1 – Schéma de glissement de fenêtre de prédiction

Le choix de la fenêtre de prédiction est ici déterminant étant donné la taille élevée du jeu de donnée et l'horizon choisi pour nos prédictions comme illustré sur la figure 3.1. Cette taille de fenêtre doit rester en adéquation avec les valeurs de paramètres du modèle afin de garantir une stationnarité locale à chaque occurrence et de faciliter l'exécution de la routine en temps réel. Ceci est particulièrement important d'autant que la prédiction des valeurs doit être effectuée pour des multitudes de serveurs dans une durée permettant une prise de décision et une reconfiguration des systèmes à temps. Pour les besoins de notre projet, cette valeur est arbitrairement fixée à 100. Cette valeur est choisie pour respecter la propriété ci-après : $Wlength$ supérieur ou égale $2max(p, q) + p + q$

3.3 Evaluation des performances

3.3.1 Simulations

La figure 3.2 présente un aperçu de la précision des prédictions. Notre mesure des erreurs de prédiction s'inspire de la méthode du MSE pour calculer d'une part à chaque occurrence, la différence entre les 100 dernières réalisations et les 100 dernières prédictions et d'autres parts à calculer l'écart-type entre ces valeurs d'erreurs de prédiction à chaque date t et les valeurs réelles obtenues à $t + 1$

CHAPITRE 3. PREDICTION DES BESOINS EN RESSOURCES BASEE SUR LES OUTILS STATISTIQUES

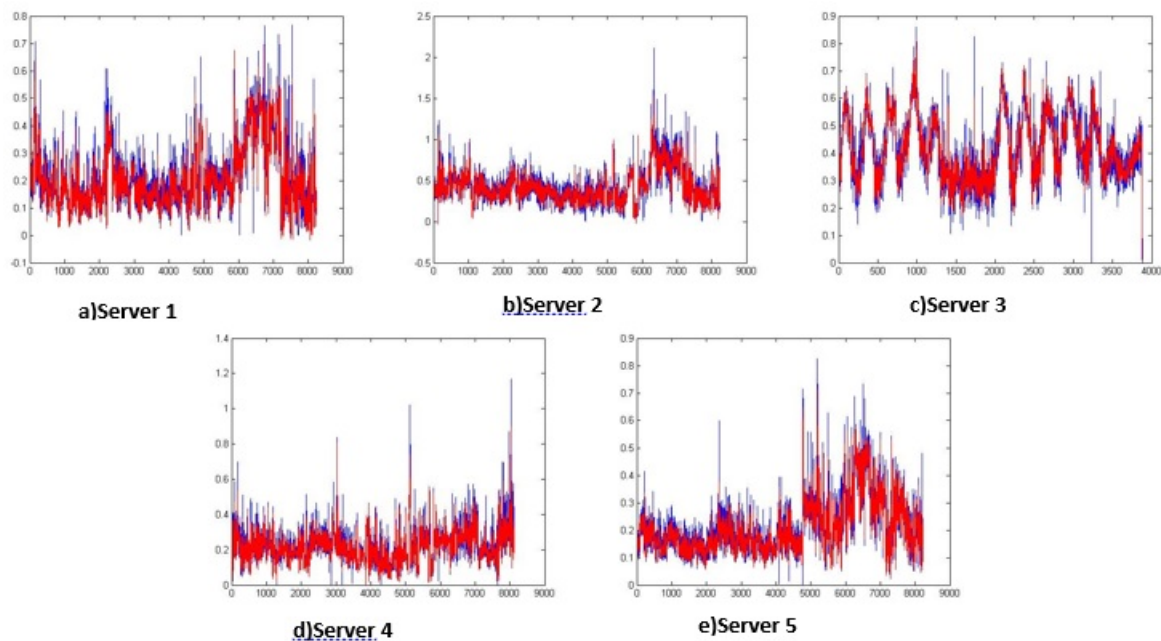


FIGURE 3.2 – Prédiction des consommations de processeurs

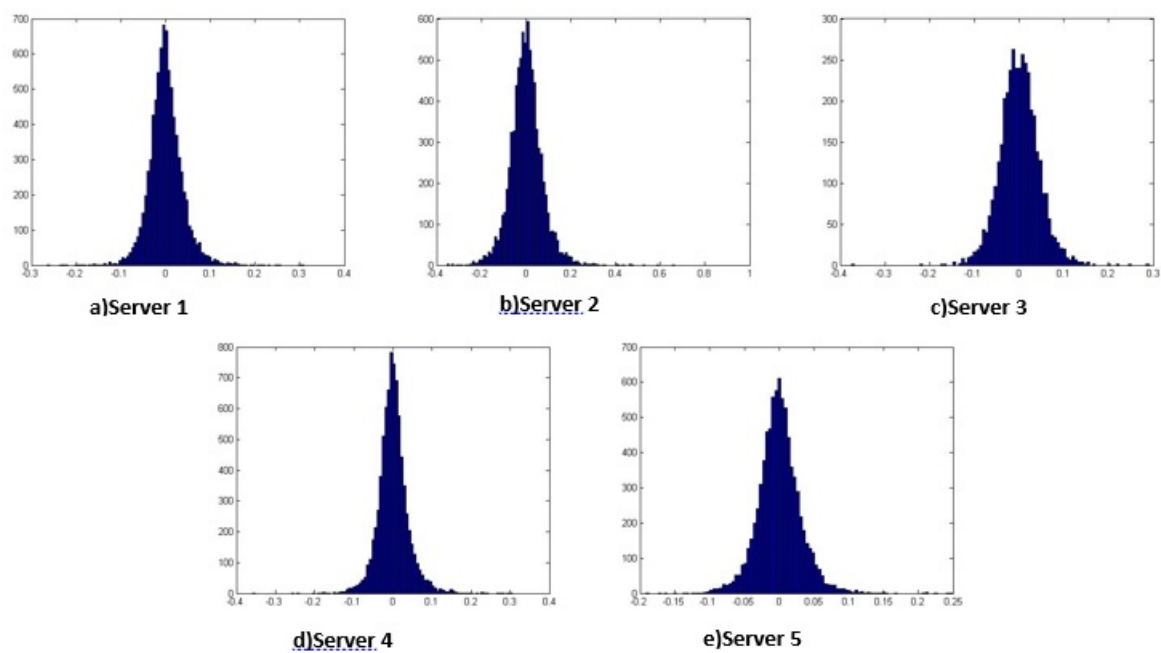


FIGURE 3.3 – Histogramme des erreurs de prédiction

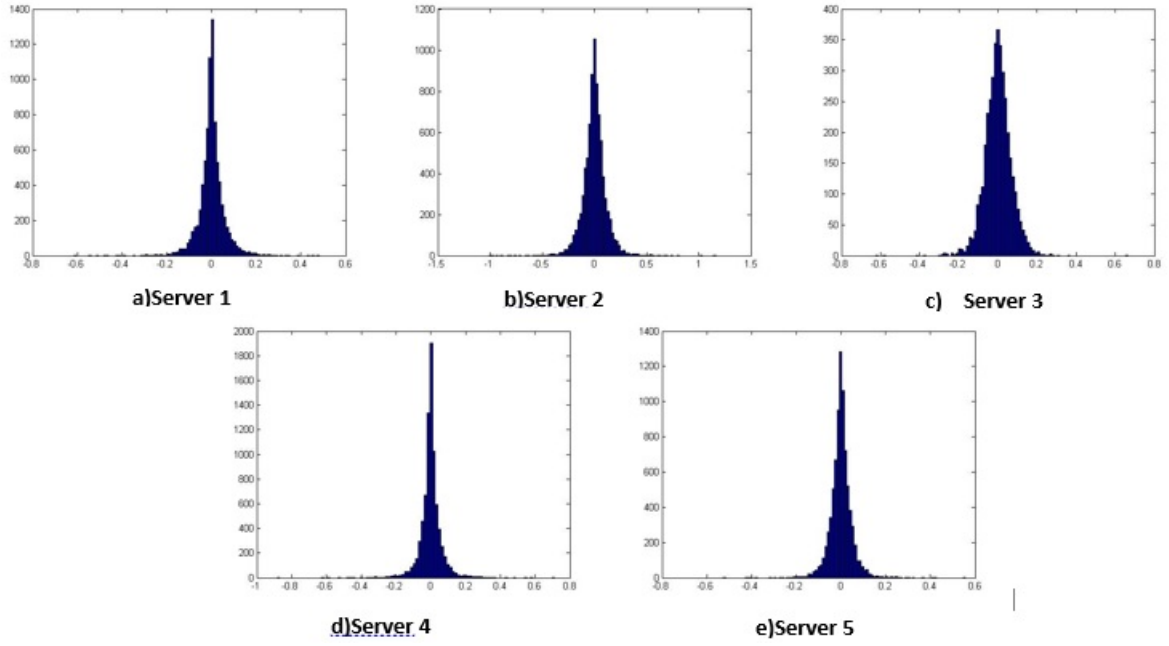


FIGURE 3.4 – Histogramme des erreurs de prédiction sur la fenêtre de 100 valeurs

Cette formule est donnée par :

$$E = \sigma(Y_t - \hat{Y}_{t+1}) / \sigma(Y_t)$$

Les graphes figure 3.3 et 3.4 présentent respectivement l’histogramme des erreurs entre 2 prédictions successives et l’histogramme des erreurs cumulées sur une fenêtre de prédiction de 100 valeurs. Elles montrent clairement que pour tous les serveurs, les mesures d’erreurs sont contenues dans un intervalle $[-0.2, 0.2]$ avec un centrage sur 0 et en élargissant la fenêtre, ces erreurs se resserrent davantage dans l’intervalle $[-0.1, 0.1]$. Par conséquent, nous pouvons affirmer que les prédictions de consommation sur les serveurs au sein de notre datacenter sont très fiables avec un choix adéquats des paramètres du modèle ARMA et un choix de fenêtre de prédiction convenable pour notre jeu de données.

3.3.2 Fiabilité des prédictions statistiques

L’autre finalité de ce travail est de prouver que notre modèle ARMA produit de meilleurs résultats que la méthode naïve dans la prédiction des consommations de processeurs au

CHAPITRE 3. PREDICTION DES BESOINS EN RESSOURCES BASEE SUR LES OUTILS STATISTIQUES

sein des datacenters. Les simulations montrent clairement qu'avec notre modèle ARMA(14, 14), les prédictions basées sur la méthode naïve produisent des taux d'erreurs compris entre 50 et 64% tandis que le processus ARMA des erreurs en dessous de 40%. Il en ressort que les processus ARMA sont environ 30% plus fiable que la méthode naïve sur l'ensemble des 5 serveurs de notre étude. Le tableau 3.1 présente les résultats comparatifs de prédiction avec les deux méthodes sur les 5 serveurs

Server	Méthode Naive	Méthode ARMA
Server1	0.64	0.40
Server2	0.59	0.37
Server3	0.58	0.36
Server4	0.69	0.43
Server5	0.50	0.31

TABLE 3.1 – Erreurs de prédiction

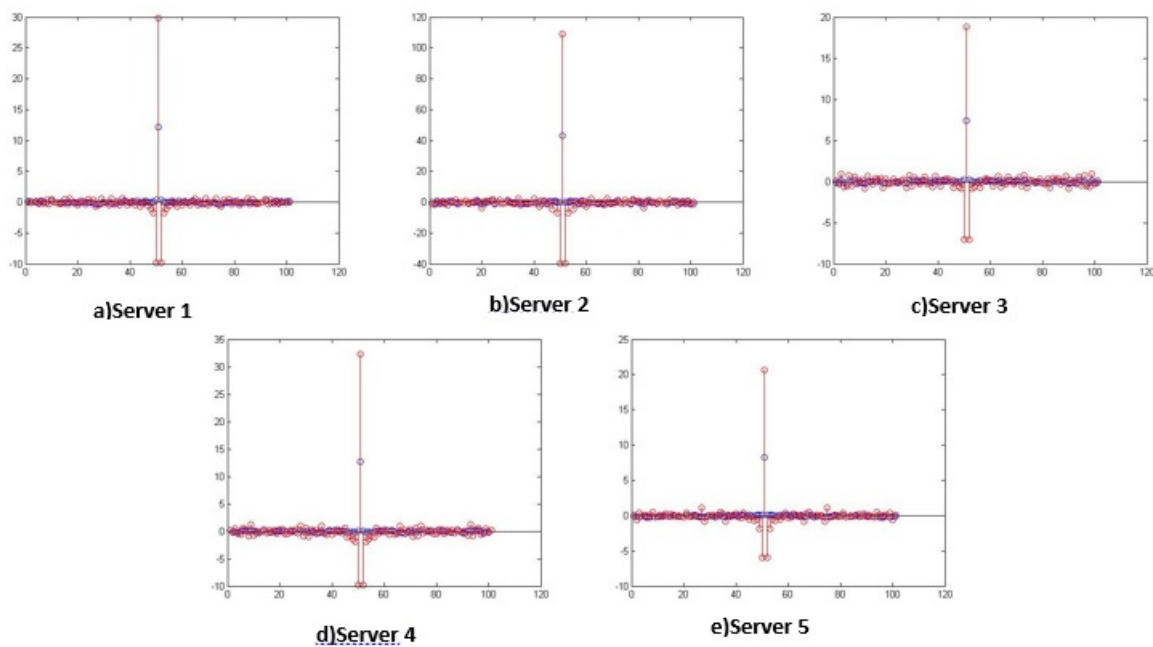


FIGURE 3.5 – Graphe de corrélation entre les valeurs de consommation de processeurs

La figure 3.5 présente les résultats de corrélation entre les observations successives des valeurs de consommations et celles des valeurs successives des prédictions. Elle montre

une faible variation des valeurs à des dates successives. Il n'existe pas de tendance claire dans la progression de la série et on n'aperçoit pas non plus d'autres caractéristiques particulières telles qu'une saisonnalité... La figure 3.2 est le graphe présentant en bleu la série initiale de consommation de ressources et en rouge la série prédite. On y remarque sur tous les serveurs une similitude qui prouve que notre modèle ARMA permet effectivement de fournir de meilleurs résultats sur notre jeu de données.

Conclusion

La recherche de solutions pour l'optimisation de l'utilisation des ressources au sein des datacenters mérite d'être approfondie étant entendu que les besoins en énergie de ces derniers ne décroissent pas et surtout que l'énergie n'est toujours pas une ressource extensible à l'infinie. Les techniques se rivalisent pour l'atteinte de cet objectif. Cependant, il reste un fort challenge que d'établir des modèles qui puissent s'appliquer à différents scénarii et qui s'adaptent aux contraintes de temps réel et de quantité énorme de données que nous imposent le cloud computing. Nous avons essayé à travers ce chapitre d'explorer la possibilité d'adopter une approche prédictive pour la maîtrise des besoins en ressource d'un datacenter au fil du temps en vue d'élaborer sur cette base les meilleurs plans d'approvisionnement de ces ressources. Notre travail a ainsi consisté à élaborer un modèle ARMA adapté au type de données que l'on traite au sein des datacenters relatives à la consommation des ressources processeurs et à réaliser des prédictions en temps réel des futures besoins. Pour ce faire, nous nous sommes appuyés sur une collection de données tirée d'un datacenter de Google au sein de laquelle nous avons extrait, sur 5 serveurs, les moyennes de consommation de processeurs sur 5 min sur une durée totale de 29 jours. Nous avons ensuite réalisé des fenêtres de 100 valeurs de nos données qui nous permettent à chaque fois de prédire les besoins en ressources processeurs à l'horizon d'une heure. Nos résultats montrent clairement que les modèles ARMA, lorsque les paramètres sont bien déterminés, constituent un moyen de prédire lesdites données. Les valeurs d'erreurs sont très faibles $[-0.1, 0.1]$. Ceci prouve que la plupart des prédictions sont très proches des valeurs réelles obtenues. L'autre contribution de ce travail a consisté à comparer nos prédictions basées sur le modèle ARMA avec celles réalisées en utilisant la méthode naïve.

Les résultats confirment une fois encore les performances de notre modèle ARMA dont la précision des prédictions est supérieure à celle de la méthode naïve dans une proportion d'environ 30%. Au total, les méthodes prédictives pourraient améliorer significativement l'utilisation des ressources matérielles au sein des datacenters et par conséquent favoriser l'optimisation des ressources énergétiques. L'obtention de ces résultats avec les modèles ARMA est une bonne ouverture vers l'atteinte de ce but d'autant qu'il pourrait s'intégrer désormais dans les algorithmes de gestion des placements ou des migrations des machines virtuelles permettant l'optimisation de l'énergie électrique approvisionnée.

Chapitre 4

PREDICTION BASEE SUR L'APPRENTISSAGE DES BESOINS EN RESSOURCES AU SEIN DES DATACENTERS

Introduction

La complexité et l'imprévisibilité des besoins des datacenters ne favorisent pas la généralisation des techniques statistiques et les différences entre les types de trafic et les périodes de pic de production ne tolèrent pas non plus une application des mêmes règles uniformes d'allocation de ressources sur les serveurs. En effet, les flux de travail peuvent varier de manière significative lorsqu'il s'agit d'un service de messagerie ou d'une application de production pour les entreprises ou encore selon que les clients, des quatre coins du globe, se retrouvent dans leur journée de travail ou dans la nuit. Pour pallier cette contrainte, nous proposons une approche encore plus proactive de la compréhension du trafic et des besoins en ressources au sein des datacenters. Cette technique consiste, à partir de utiliser des outils d'apprentissage par ordinateur pour la prédiction des besoins futurs de ressources en se basant sur les tendances précédentes de consommation de ressources. Notre étude sur les outils de prédiction de trafic présentée en annexe 1, propose un inventaire de divers outils pouvant satisfaire ce besoin. Nous avons choisi dans le cadre de ce chapitre de tester principalement les réseaux de neurones étant donné les résultats satisfaisants obtenus pour des objectifs similaires dans de nombreux autres domaines notamment en météorologie, en logistique, en économie etc..

Les réseaux de neurones sont des outils mathématiques, inspirés du fonctionnement des neurones biologiques humains qui sont utilisés pour la modélisation et la simulation de phénomènes très complexes qui ne sont pas gouvernés par des lois déterministes. Il s'agit principalement d'événements aléatoires dont l'issue est imprévisible ou dont la reproduction ne peut se faire autrement que par la simulation. Les fonctions importantes des réseaux de neurones sont l'apprentissage, la classification, la reconnaissance de motif,

le traitement de signal, la modélisation de fonctions complexes, la prédiction, etc. Les domaines d'utilisation sont tout autant variés et peuvent aller de la physique à l'économie en passant par la météorologie, la biologie, la robotique, les statistiques. Il existe différents algorithmes de mise en œuvre des réseaux de neurones qui s'adapte suivant le domaine d'utilisation et les situations à modéliser. Dans le cadre de notre travail, les réseaux de neurones pourraient apporter une plus grande souplesse dans la modélisation et la prédiction de la consommation des ressources en vue d'une meilleure efficacité de leur allocation.

La contribution de cette étude concerne d'une part l'établissement des meilleurs architectures de réseaux de neurones permettant de mieux modéliser les consommations de ressources au sein des datacenters [F. Gbaguidi and Ezin 2017] et d'autre part d'établir que les méthodes d'apprentissage par ordinateur, bien de produisant des prédictions plus précises de l'ordre de 25% des tendances futures de consommation peuvent coûter plus cher en temps de traitement de l'ordre de 50% comparés aux méthodes statistiques de type ARMA. Par conséquent, elles ne répondent pas forcément au besoin de prédiction en temps réel nécessaire dans certains scénarii mais pourraient se révéler plus adaptés pour faire des prédictions à moyen et long terme. De plus, nous avons établi que le choix du réseau est un facteur déterminant dans les capacités d'apprentissage et la fiabilité des prédictions.

Le présent chapitre s'articule ainsi qu'il suit : la première partie présente les généralités sur les réseaux de neurones et les différents choix que nous opérons pour la présente étude puis dans la partie 2 nous présentons les résultats de l'évaluation des différentes architectures de réseaux de neurones. Ensuite, nous faisons une comparaison des performances des résultats obtenus avec ceux obtenus en utilisant les méthodes statistiques de type ARMA ainsi que les commentaires qu'ils appellent avant de conclure le chapitre.

4.1 Aperçu des réseaux de neurones

Les réseaux de neurones constituent une modélisation mathématique de l'approche connexionniste des neurones du cerveau humain, introduit dans le domaine de l'intelligence

artificiel dans les années 40 par [McCulloch and Pitts 1943]. Ce concept a subi de nombreuses évolutions avant de s'imposer dans le domaine de l'informatique comme outil de résolution de problèmes complexes avec de nombreuses applications. Ils sont généralement représentés par un graphe composé de neurones reliés suivant certaines architectures se rapportant à leur nombre, les fonctions de leur activation, l'existence de retro-propagation et l'objectif visé par leur utilisation [Demuth and Beale 1993].

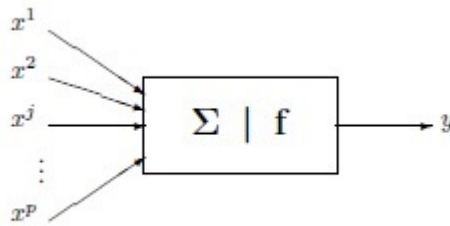


FIGURE 4.1 – Représentation d'un Neurone

Les réseaux de neurones figure 4.1 ont été très utilisés dans des problèmes complexes dont statistiques l'analyse de données la prévision la classification, etc.

-
- robotique : contrôle et guidage de robots ou de véhicules autonomes
- imagerie / reconnaissance de formes
- traitement du signal
- simulation de l'apprentissage

Par analogie à un neurone biologique du cerveau humain, les réseaux de neurone se caractérisent par un état interne $\in S$ des signaux d'entrée $[x_1; :::; x_p]$ et une fonction d'activation h . Les entrées x du réseau sont affectés d'un vecteur de poids $[\alpha_1; :::; \alpha_p]$ calculé au cours de la phase d'apprentissage. La fonction d'apprentissage est utilisée pour opérer les transformations internes au sein du réseau en fonction de l'objectif poursuivit. On distingue notamment les fonctions ci-après : linéaire f est la fonction identité,

- Sigmoides : $f(x) = \frac{1}{1+\exp(x)}$,
- Seuil $f(x) = 1[0; 1[(x)$,

- radial $f(x) = p1 = 2_{exp}(x2 = 2)$,
- stochastiques $f(x) = 1 + exp(x)$,

Les modèles linéaires et sigmoïdaux sont bien adaptés aux algorithmes d'apprentissage impliquant (cf. ci-dessous) une rétro-propagation du gradient car leur fonction d'activation est différentiable ; ce sont les plus utilisés.

4.1.1 Perceptron multi-couches

Le perceptron multi-couche [Demuth and Beale 1993] et [Haykin et al. 2009] est un réseau de neurones composé d'une ou plusieurs couches intermédiaires permettant d'optimiser l'apprentissage à travers un ajustage itératif du vecteur des poids comme le montre la figure 4.2. A chaque itération, l'erreur d'apprentissage relevé à la sortie est retro propagée aux entrées pour un réajustement des poids jusqu'à obtention d'une erreur minimale. Le choix des paramètres est cependant une étape déterminante dans l'efficacité

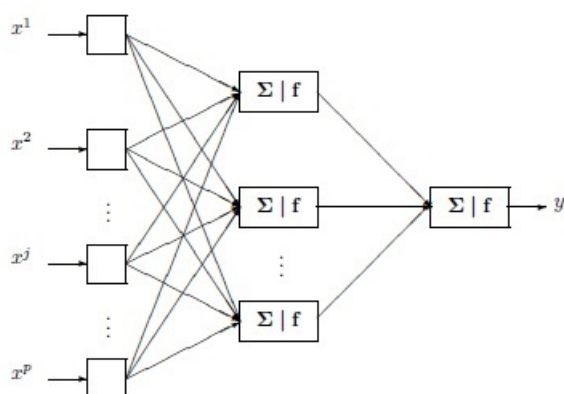


FIGURE 4.2 – Perceptron Multicouche

de l'algorithme du perceptron. Le nombre de couches, le nombre de neurones par couche, le nombre d'entrées du réseau et les poids initiaux affectés aux neurones doivent être choisis en adéquation avec le problème à résoudre. Il est recommandé de tester plusieurs jeux de paramètres et de simuler jusqu'à l'obtention du réseau fournissant les meilleurs résultats en terme d'apprentissage.

4.2 Prédictions basées sur les réseaux de neurones

4.2.1 Méthodologie

La réduction de la consommation énergétique des serveurs au sein des datacenters est l'un des objectifs principaux de notre travail de thèse. Elle consiste pour nous à élaborer les meilleurs modèles permettant de planifier les quantités optimales de processeurs et de mémoire RAM à affecter aux différentes machines virtuelles du datacenter en vue de rationaliser les sollicitations d'alimentation électrique nécessaires à leur fonctionnement. La réduction de la consommation énergétique des serveurs est d'autant plus cruciale qu'il entraîne d'autres économies d'énergie par effet de corrélation, sur les autres composants des datacenters (climatisations, onduleurs, télécoms, etc.). Pour y parvenir, nous formulons les hypothèses ci-après :

- l'estimation des besoins approximatifs des serveurs permettra de planifier judicieusement l'apport d'énergie qui leur est nécessaire
- la prévision de l'énergie nécessaire aux serveurs permettra d'éviter les problèmes de sur-approvisionnement rencontrés de manière classique au sein des datacenters
- l'apprentissage des habitudes de consommation énergétique antérieure sur les serveurs permettra de prévoir les besoins futurs

Les réseaux de neurones pourraient constituer un outil performant pour la prédiction des futures consommations des ressources à partir de l'apprentissage des quantités antérieurement consommées.

4.2.2 Objectifs

La présente étude vise principalement à évaluer les capacités des réseaux de neurones à prédire les consommations de ressources processeur sur les serveurs au sein des datacenters en se basant sur les niveaux consommations antérieures. Ce résultat nous permettra d'établir la prévisibilité des ressources à fournir aux serveurs au sein du datacenter de notre étude et de tester les possibilités de généralisation des méthodes utilisées. La qualification des techniques d'apprentissage à la prédiction de ressources nécessaires sera une étape

dans l'élaboration à terme d'un modèle optimal de planification de ressources (processeur, mémoire, etc.) au sein des datacenter pouvant permettre in fine d'éviter les pertes de ressources énergétiques. Ces pertes résultent généralement d'une sous-consommation des ressources réservées sur des machines physiques en se basant généralement sur des hypothèses surdimensionnées.

Les résultats de cette simulation seront également comparés à ceux obtenus avec des méthodes dites statistiques de types ARIMA et HoltWinters

4.3 Evaluation de la prédiction des ressources par les réseaux de neurones

4.3.1 Jeux de données

Dans le cadre de cette expérimentation, nous partons des traces de Google à partir desquelles nous nous intéressons à la colonne consommation de CPU sur les serveurs sélectionnés au cours des 29 jours de la collecte. Les valeurs ayant été normalisées entre 0 et 1, nous obtenons les graphes 4.3, 4.4, 4.5, 4.6 et 4.7 sur lesquels les valeurs de CPU sur les différents serveurs sont représentées.

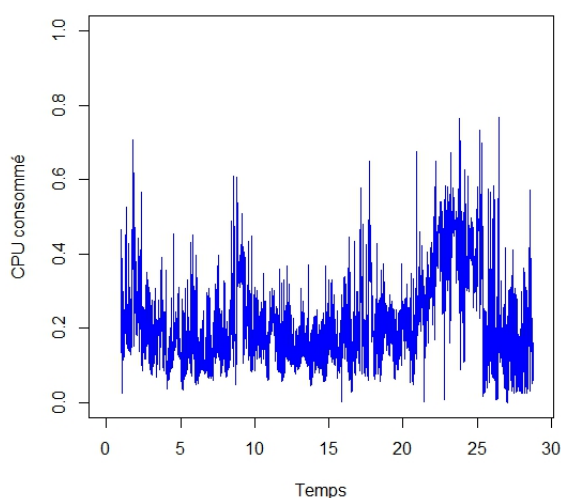


FIGURE 4.3 – Consommation de CPU sur le serveur 1

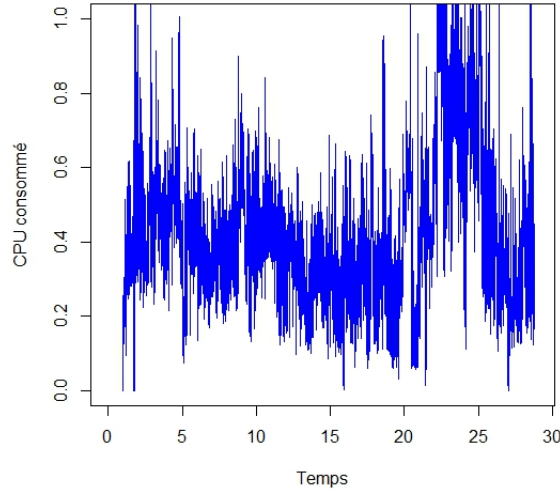


FIGURE 4.4 – Consommation de CPU sur le serveur 2

Sur chaque donnée, nous procédons à quelques traitements afin d'obtenir des groupes de données correspondants respectivement à 4h, 8h, et 12h successives de consommation pour servir de données d'entrée à nos réseaux. Ensuite nous extrayons les valeurs de l'heure suivante à chaque fois dans un autre fichier que nous appelons fichier cible et qui constitue les valeurs attendues des prédictions. Pour élaborer nos jeux de données, nous implémentons un algorithme figure 4.10 de groupage des données qui se présente comme ci-dessous

Où n représente la quantité de donnée en entrée des réseaux (4, 8 et 12). Les fichiers inputn et outputn représentent alors respectivement les fichiers d'entrée et les fichiers cibles de nos réseaux. les figures 4.8 4.9 présentent un aperçu des données obtenues avant et après exécution de l'algorithme figure 4.10 pour $n = 4$

4.3.2 Expérimentations

Pour vérifier les hypothèses ci-dessus énoncées, nous construisons différents types de réseaux de neurones en faisant varier notamment le nombre d'entrée (4, 8 et 12) et le nombre de couches intermédiaires. Ensuite, nous élaborons nos jeux de données pour

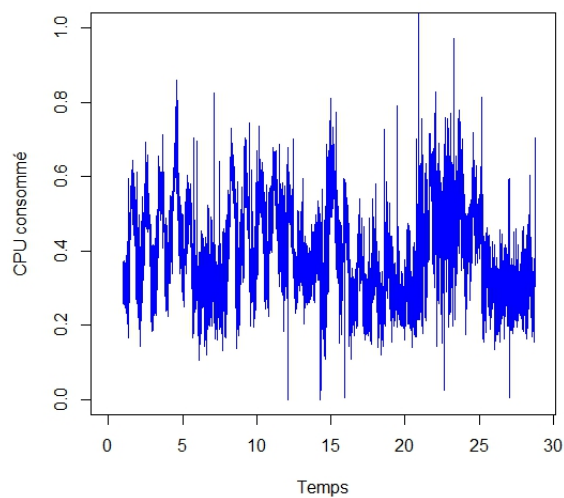


FIGURE 4.5 – Consommation de CPU sur le serveur 3

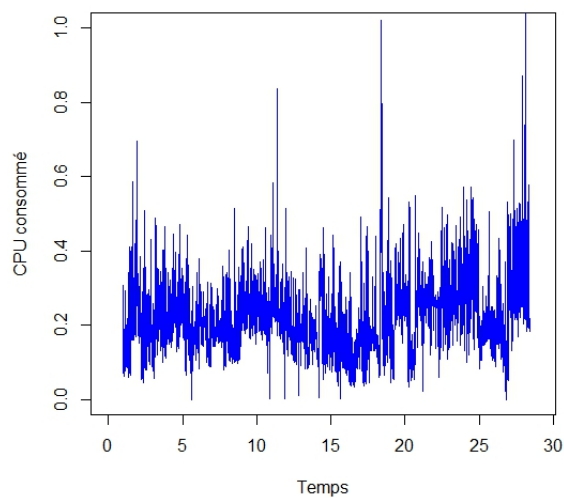


FIGURE 4.6 – Consommation de CPU sur le serveur 4

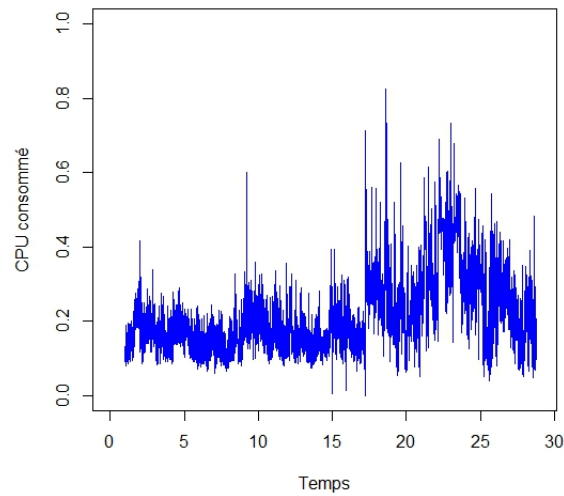


FIGURE 4.7 – Consommation de CPU sur le serveur 5

	1	2	3	4	5	6	7	8	9
1	0.0677	0.0698	0.0866	0.1219	0.2317	0.3309	0.2915	0.2171	0.1040
2	0.0698	0.0866	0.1219	0.2317	0.3309	0.2915	0.2171	0.1040	0.0662
3	0.0866	0.1219	0.2317	0.3309	0.2915	0.2171	0.1040	0.0662	0.0736
4	0.1219	0.2317	0.3309	0.2915	0.2171	0.1040	0.0662	0.0736	0.3519

FIGURE 4.8 – Exemple de fichier d'entrée

	1	2	3	4	5	6	7	8	9
1	0	0.2915	0.2171	0.1040	0.0662	0.0736	0.3519	0.3790	0.2032

FIGURE 4.9 – Exemple de fichier de sortie

correspondre au nombre d'entrée des différents réseaux en consolidant les données de consommation par heure et par demi-heure. Nous injectons les données dans les réseaux correspondants dans le but d'obtenir respectivement : La valeur probable de ressource nécessaire au bout de 5h si l'on dispose des consommations des 4 heures précédentes ; La valeur de la 9ième heure si l'on dispose de 8h de données ; La valeur au bout de 13h si l'on dispose des données des 12 heures précédentes. Les résultats obtenus sont consignés et comparés afin de déterminer le meilleur réseau capable de prédire ce type de données ainsi que le meilleur horizon auquel les prédictions sont les plus fiables

4.3.3 Choix de l'architecture des réseaux

Les réseaux

```
m=data
v=0
outputn=0
inputn=m(1:n)
for i=2:length(m)-(n+1)
    outputn=[outputn,m(i+(n+1))];
    v=m(i:i+(n-1));
    inputn=[inputn,v];
end
```

FIGURE 4.10 – Algorithme de traitement des fichiers en entrée et en sortie des réseaux

Les figures 4.11, 4.12 et 4.13 présentent les différents réseaux de neurones ayant fait l'objet de simulation. Il faut noter que chaque architecture, correspond à une variation du nombre d'entrée et du nombre de couche intermédiaire. Comme signalé plus haut, le choix de ces paramètres est déterminant quant à l'efficacité du réseau final choisi. Dans notre environnement de simulation, le choix des autres paramètres notamment le vecteur de poids initiaux est laissé à MATLAB. La fonction d'activation utilisée est la fonction sigmoïdale dont la formule générale est donnée plus haut.

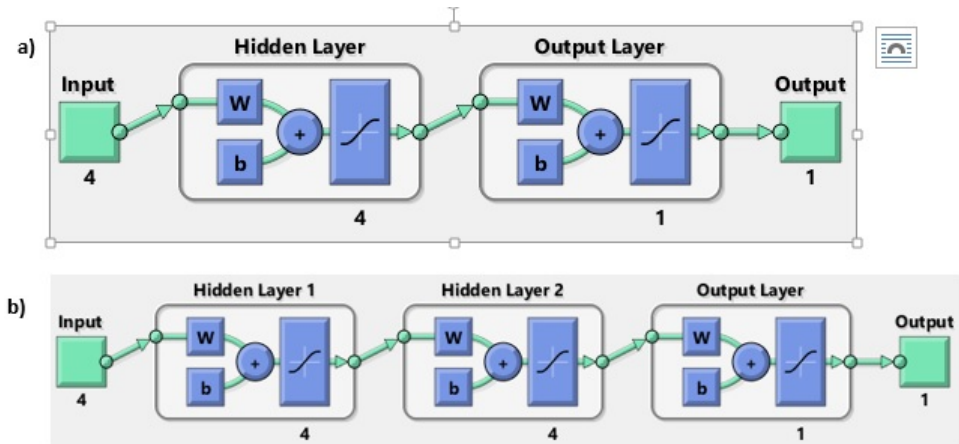


FIGURE 4.11 – Réseaux de neurones à 4 entrées a) avec une couche cachée b) avec deux couches cachées

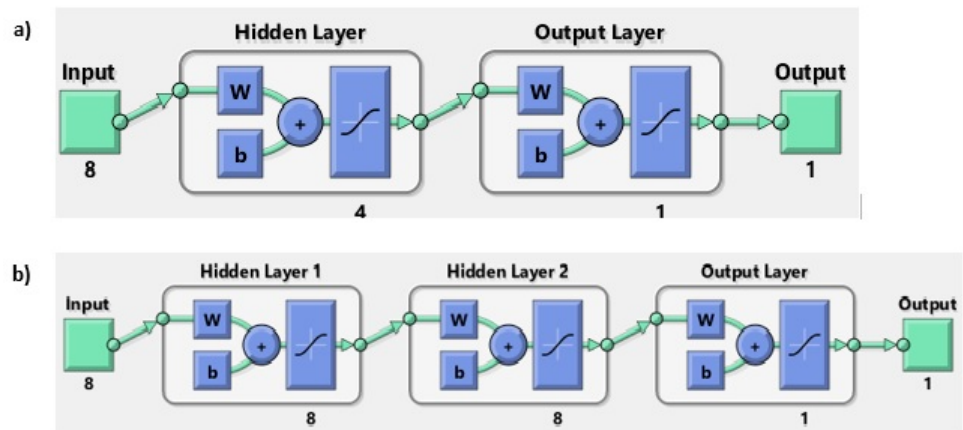


FIGURE 4.12 – Réseaux de neurones à 8 entrées a) avec une couche cachée b) avec deux couches cachées

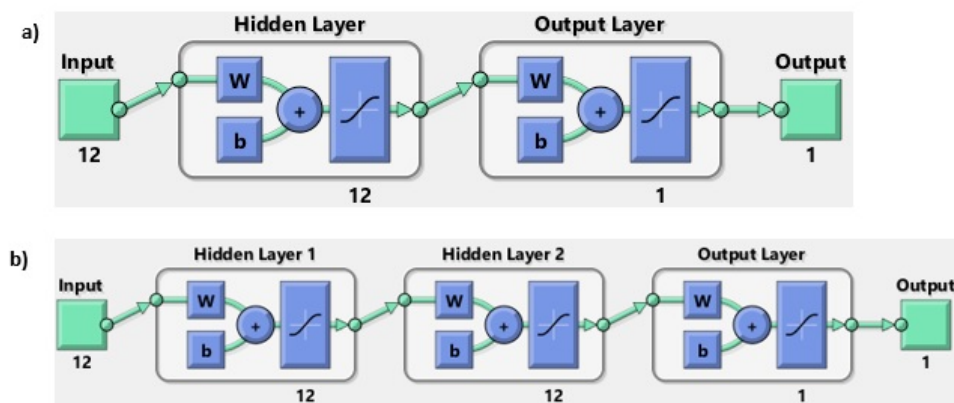


FIGURE 4.13 – Réseaux de neurones à 12 entrées a) avec une couche cachée b) avec deux couches cachées

4.3.4 Fiabilité des prédictions basées sur les réseaux de neurones

Les figures 4.14, 4.15 et 4.16 les graphes de régression des simulations pour les différentes architectures de réseaux de neurones considérés. Les graphes de régression montrent la correspondance entre les sorties calculées par le réseau et celles attendues à partir des fichiers output. Nous présentons donc pour chaque architecture, côte à côte, les résultats pour les réseaux avec une et deux couches intermédiaires.

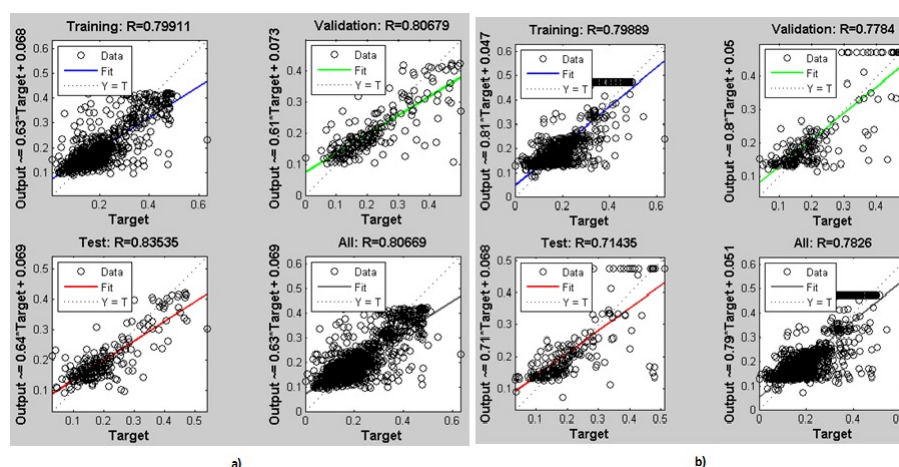


FIGURE 4.14 – Graphe de regression pour les réseaux à 4 entrées a) avec une couche cachée, b) avec deux couches cachées

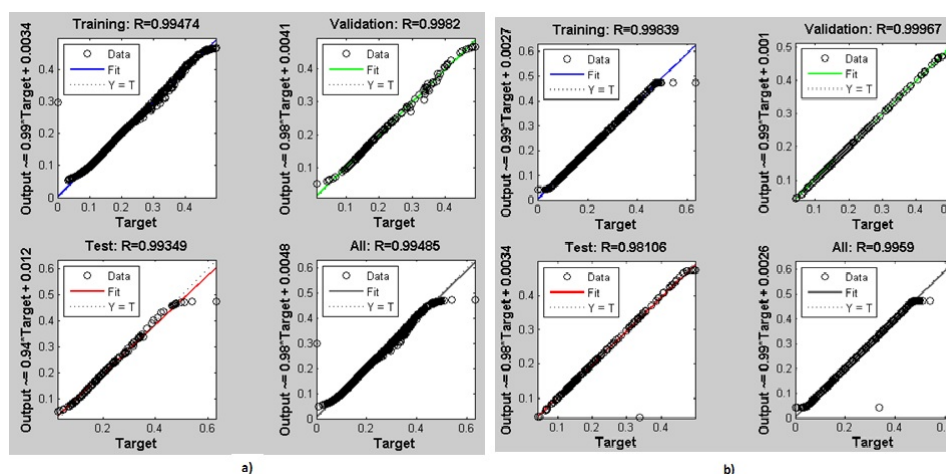


FIGURE 4.15 – Graphe de regression pour les réseaux à 8 entrées a) avec une couche cachée, b) avec deux couches cachées

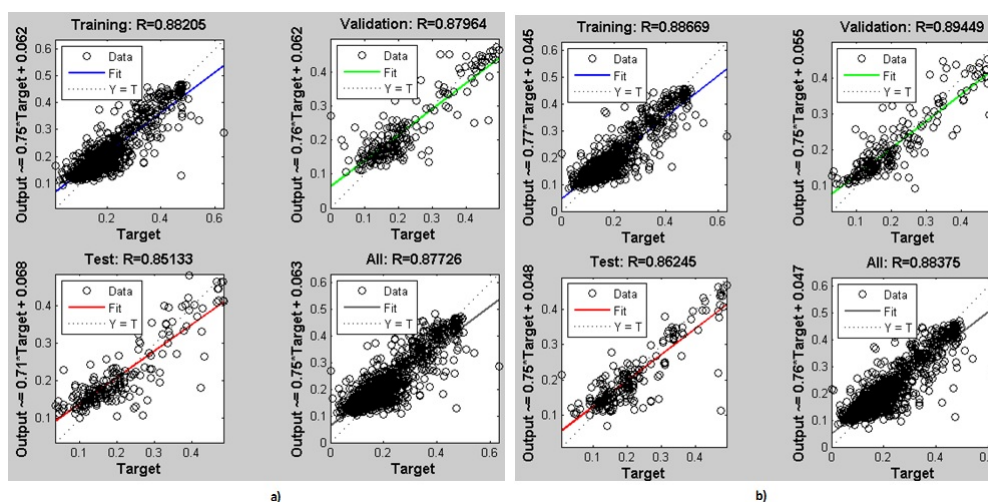


FIGURE 4.16 – Graphe de regression pour les réseaux à 12 entrées a) avec une couche cachée, b) avec deux couches cachées

4.3.5 Analyse des résultats

La valeur de R est une indication du niveau de conformité entre les sorties et les cibles. $R = 1$, indique qu'il existe une relation linéaire entre les résultats et les objectifs. Si R est proche de zéro, alors il n'y a pas de relation linéaire entre les sorties et les cibles. Ainsi que présenté sur les graphes de régression respectives, les lignes en pointillés représentent la meilleure conformité entre les cibles et les sorties obtenues à partir des réseaux (représentées en rond). Les lignes fermes quant à elles représentent la fonction de régression linéaire entre les sorties et les cibles. Le MSE (Mean Square Error) est le carrée des écarts moyens entre les sorties et les valeurs cibles. Il rend compte de la fiabilité des prédictions effectuées à partir des différents réseaux. Sur les graphes figure 4.17, 4.18 et 4.19, on peut remarquer les variations des MSE tout au long des itérations des simulations à la fois pour les phases d'apprentissage, de validation et de test. Les lignes en pointillés représentent les valeurs de MSE garantissant une meilleure fiabilité des paramètres de simulation. Plus les courbes se rapprochent de la ligne en pointillé, plus fiable est le réseau. En analysant les résultats de nos simulations, il ressort que : Les réseaux à 4 entrées

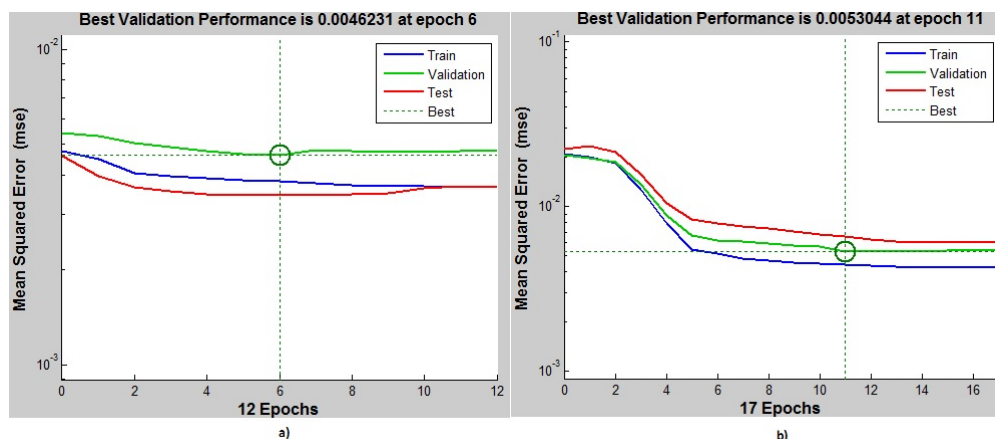


FIGURE 4.17 – MSE pour les réseaux à 4 entrées a) avec une couche cachée, b) avec deux couches cachées

présentent des valeurs de R inférieures à 0,9 même lorsque nous complétons une deuxième couche de neurones cachés. Généralement cette valeur doit être supérieure à 0.9 pour un réseau fiable. Cependant, les courbes des taux d'erreurs restent assez proches de la valeur

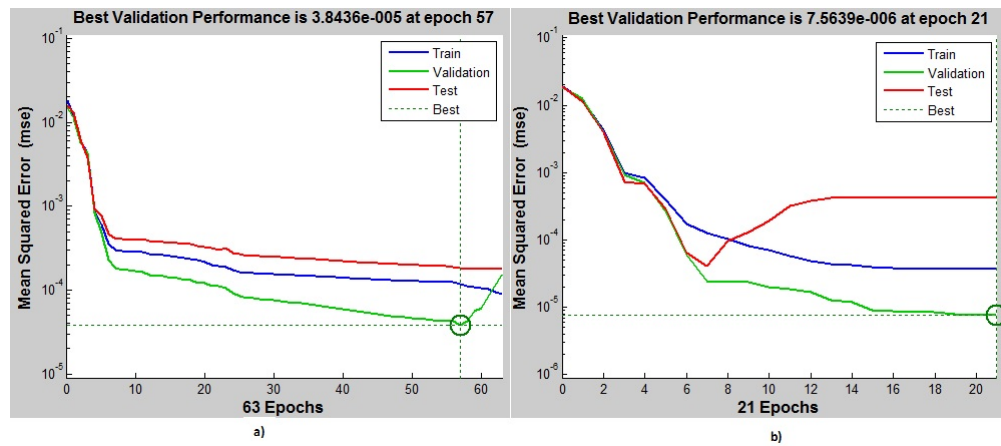


FIGURE 4.18 – MSE pour les réseaux à 8 entrées a) avec une couche cachée, b) avec deux couches cachées

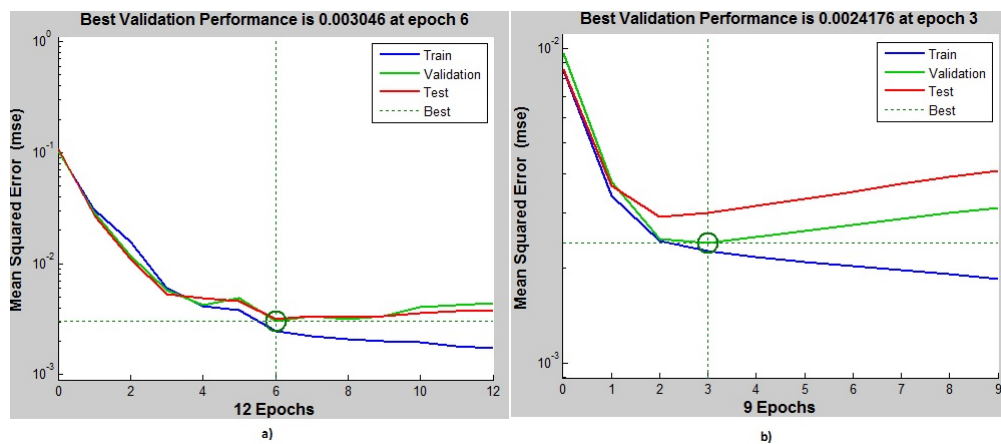


FIGURE 4.19 – MSE pour les réseaux à 12 entrées a) avec une couche cachée, b) avec deux couches cachées

optimale, ce qui montre que les paramètres de l'apprentissage sont adéquats. Au total, les réseaux de neurones à 4 entrées ne produisent pas des résultats très satisfaisants sur notre jeu de données. Les réseaux à 8 entrées quant à eux produisent des valeurs de R très proches de 1 et donc les valeurs prédites sont très proches des valeurs cibles. Egalement, les taux d'erreurs sont très faibles. Les courbes d'erreurs lors des phases de validation et de test ne remontent pas de façon significative après l'obtention de la meilleure performance. Il a été démontré qu'une deuxième couche cachée de neurones, permet d'obtenir également des résultats satisfaisants en dépit de la durée très faible de l'apprentissage. En conclusion, les réseaux de neurones à 8 entrées semblent être le meilleur réseau pour notre jeu de données.

Nous avons expérimenté les réseaux de neurones à 12 entrées qui nous donnent des valeurs de R inférieures à 0,9 ce qui montre une imprécision relative des prédictions de ressources. Les taux d'erreurs aux phases de validation et de test sont très proches des valeurs optimales du MSE. Cependant, les résultats de l'apprentissage sont peu satisfaisants même avec une couche supplémentaire de neurones.

4.4 Analyse comparative des prédictions statistiques et de l'apprentissage par ordinateur

4.4.1 Cadre de l'analyse

la phase de prédiction est l'importante étape avant la prise d'une action pour économiser des ressources sur les serveurs du datacenter. Elle est d'autant cruciale qu'une mauvaise appréciation des tendances de consommation future pourrait résulter en une action qui non seulement pénaliserait les accords de disponibilité de service signés avec les clients mais aussi produire un effet contraire c'est à dire entrainer une surconsommation de ressources au lieu de la réduire. Pour ce faire, l'objectif de cette analyse est de faire une analyse comparée des prédictions obtenues avec les méthodes statistiques et celles obtenues par l'apprentissage avec les réseaux de neurones afin d'interpréter au regard des différents contextes d'utilisation au sein des datacenters, laquelle des méthodes sera la mieux adaptée. Cette analyse porte sur :

- la fiabilité : le niveau de précision des prédictions est déterminant dans le choix d'un outil. lorsqu'il s'agit des ressources à provisionner sur les serveurs, il faut partir des observations antérieures sur une durée suffisante pour apprécier les profils de consommations de ces derniers. la fiabilité réside avant tout dans le choix des paramètres des modèles utilisés ou dans le choix des architectures de réseaux de neurones. Ensuite l'amélioration permanente de la précision en réévaluant continuellement sur la base des erreurs de prédiction constatées les modèles définis au départ doit être faite. Le choix des données de test est également un critère important. les flux de travail des datacenters étant très variables d'un environnement à un autre, l'utilisation de traces réelles pourraient fiabiliser les résultats obtenus et favoriser leur portabilité vers d'autres scénarii. Enfin les indicateurs d'erreurs choisis doivent être les plus objectifs possibles et s'adapter au contexte de l'étude. Pour cette analyse, nous retenons pour les deux outils, de mesurer le MSE obtenus après avoir réaliser les expériences sur les mêmes jeux de données
- le temps de traitement : La décision sur l'optimisation énergétique peut être de court, moyen ou long terme en fonction des caractéristiques des datacenters. Lorsque l'on se trouve dans un environnement ou le flux de travail est bien déterminé dans le temps, on peut planifier à l'avance les consolidations de serveurs sans se soucier des risques de variations brusques de trafic et de violation des SLA. C'est le cas des applications qui sont utilisées de manière saisonnière avec une temporalité connue. Les applications de gestion de paie ou d'établissement d'état financier sont sollicitées presque invariablement en fin de mois et en fin d'exercice. Dans ces cas de figure, la prédiction pourrait se réaliser pour le long terme. Le choix d'un outil performant mais chronophage pourrait se justifier. En revanche lorsqu'on se retrouve dans un environnement de volatilité du trafic la prise de décision peut être une question d'heure ou de minute. Il s'agit ici des trafics de messagerie, des réseaux sociaux, de CRM ou autre type de service de même caractéristique avec une variation aléatoire en fonction des événements qui surviennent. Un site de streaming vidéo comme Youtube peut exploser en l'espace de quelques minutes son audience en raison de

l'arrivée d'une vidéo à succès. La consolidation des flux de travail sur les serveurs devra se faire le plus rapidement possible avant qu'une indisponibilité du service n'intervienne. La prédiction à court-terme serait la plus adaptée à ces cas de figure et par conséquent les outils pouvant induire un temps trop élevé de prise de décision seront évités.

Nous retenons d'effectuer nos expérimentations sur la base du modèle ARMA que nous avons déterminé dans le chapitre précédent et des réseaux de neurones à 8 entrées et une couche en raison des résultats obtenus à partir de cette configuration.

4.4.2 Résultats

Notre analyse des résultats se déroule sur deux axes notamment la fiabilité et le temps de traitement des prédictions

- Fiabilité : On remarque clairement sur le tableau 4.1 que les taux d'erreurs obtenus avec les réseaux de neurones sont clairement supérieurs à ceux de la méthode ARMA dans une proportion d'environ 30%. Ceci s'explique par le fait que les réseaux de neurones restent dans un processus permanent d'apprentissage qui peut continuellement remettre en cause la stabilité du réseau et faire varier les taux d'erreurs tandis que les méthodes ARMA, une fois les paramètres déterminés, les taux d'erreurs peuvent rester stables tant que le jeu de donnée ne s'écarte pas trop des conditions initiales. Cependant, une conclusion définitive ne pourrait être tirée sur cette base car, dans un environnement de volatilité du trafic, les réseaux de neurones pourraient être d'une grande efficacité comparé aux méthodes ARMA.
- Temps de traitement : la prédiction basée sur les réseaux de neurones se révèle très coûteux en temps de traitement comparée aux modèles ARMA. Ceci s'explique par la nécessité d'ajuster continuellement les paramètres des réseaux par retro-propagation des erreurs successives dans les réseaux de neurones. Si dans certains environnements ceci pourrait représenter un handicap, ces temps peuvent se révéler plutôt satisfaisants pour des prédictions à moyen terme. Il reste cependant qu'un temps élevé de traitement signifie en lui même un coût en ressources et ne sera

tolérable que s'il cela se justifie par les objectifs visés par l'étude.

TABLE 4.1 – Erreurs de prédiction

Server	Critères	Réseaux de Neurones	ARMA
Server1	erreurs	0.64	0.40
Server1	temps de traitement	3.2ms	0.6ms
Server2	erreurs	0.59	0.37
Server2	temps de traitement	3.8ms	0.6ms
Server3	erreurs	0.58	0.36
Server3	temps de traitement	2.7ms	0.6ms
Server4	erreurs	0.69	0.43
Server4	temps de traitement	4.3ms	0.6ms
Server5	erreurs	0.50	0.31
Server5	temps de traitement	3.5ms	0.6ms

Conclusion

L'intégration de l'apprentissage pour l'amélioration de l'efficacité énergétique des datacenters est une contribution importante de ce chapitre. Le choix des réseaux de neurones apporte un éclairage sur la possibilité d'intégrer un outil réputé pour être gourmand en temps de calcul dans les algorithmes de consolidation de machines virtuelles et de flux de travail dans les datacenters. Nos tests sont basées sur des données provenant de traces réelles issues d'un datacenter très représentatif des environnements de Cloud.

Au terme de notre simulation, nous pouvons retenir que nos jeux de données fournissent des résultats différents sur différentes configurations de réseaux de neurones. Les réseaux de 4 et 12 entrées donnent des taux d'auto-régression situés entre 0,7 et 0,8 tandis que les réseaux à 8 entrées présentent des résultats d'auto-régression supérieur à 0,9. Nous en déduisons que pour obtenir de bonnes prédictions des futurs besoins de CPU, il faut fournir au réseau les données correspondantes à 8h de consommation afin de générer les besoins à l'horizon 9h. Un réseau de neurones à 8 entrées avec une couche intermédiaire est donc suffisant pour produire des résultats avec des temps de traitement courts.

L'introduction de l'apprentissage dans l'approche prédictive de la rationalisation des ressources est une contribution remarquable étant donné qu'elle permet non seulement de montrer les possibilités d'amélioration des techniques existantes mais aussi de favoriser la généralisation des méthodes d'optimisation de ressources. L'apprentissage permet donc de contourner la difficulté d'adaptation des outils aux types de données car il implique un processus de retropropagation pouvant contribuer à la minimisation des erreurs à la sortie. L'autre résultat important de cette partie est la comparaison des résultats produits par les réseaux de neurones avec ceux obtenus avec les méthodes de type ARMA. Cette analyse

nous permet de confirmer la fiabilité de l'approche basée sur l'apprentissage de l'ordre de 30% par rapport aux méthodes statistiques. Cependant, le temps de traitement du processus d'apprentissage est très élevé et pourrait ne pas convenir à une démarche de prise de décision rapide.

Au total, nous pouvons confirmer que les résultats obtenus avec l'une ou l'autre des méthodes contribuent à l'amélioration des techniques d'anticipation des ressources à provisionner au sein des datacenters. Dans la suite de nos travaux, nous retiendrons les méthodes ARMA qui offrent le meilleur compromis entre la fiabilité et les performances qui sont un critère important pour passer à l'implémentation de notre technique.

Troisième partie

**PARTIE 3 : AMELIORATION DE
L'ALLOCATION DES
RESSOURCES AUX MACHINES
VIRTUELLES**

Résumé de la partie 3

Dans cette partie, nous implémentons le mécanisme de l'amélioration de l'allocation de ressources dans une perspective d'efficacité énergétique au sein des datacenters. Nous abordons deux thématiques distinctes et liées à savoir : Le placement initial des machines virtuelles sur les serveurs physiques : l'enjeu à ce niveau concerne la répartition intelligente des applications sur les serveurs de sorte à ne garder qu'un nombre optimal de serveur allumé tout en respectant les exigences de disponibilité et de continuité de service des clients. La résolution de ce problème est ramené à un problème de bin packing qui consiste à effectuer des rangements optimaux d'objets dans des conteneurs avec quelques contraintes. Fort de la popularité de cet outil dans le domaine de la logistique notamment, nous avons essayé de l'adapter au contexte du placement de VM. Nous avons présenté les aspects théoriques de cet outils dans un premier temps, puis nous avons proposé un cadre opératoire, un algorithme adapté à notre étude ainsi qu'une démarche d'implémentation. De nos expérimentations, il ressort que les deux méthodes de résolution à savoir le Brute force et le First Fit decreasing (FFD) permettent une résolution du problème. Cependant, la méthode FFD présente l'avantage d'une résolution à la fois optimale et rapide de ce problème. La consolidation des machines virtuelles : cette opération consiste à mettre en place un mécanisme d'ajustage permanent du placement des machines au cours de leur fonctionnement sur la base de leur appel de ressources et des politiques d'allocation prédéfinies. Notre démarche a consisté à utiliser les prédictions ARMA des données historiques de consommation de ressources pour anticiper sur les opérations de migration des machines virtuelles avant même que l'insuffisance de ressources ne soit effective pour une VM. Nous avons recouru au célèbre simulateur Cloudsim qui dispose déjà de modules

adaptable et d'un jeu de données similaire à ceux ayant servi de base à notre modélisation et nos tests des outils. Ainsi, il ressort de nos expérimentations que la prédiction basée sur ARMA permet de mieux anticiper sur les opérations de migrations de VM dans une architecture de Cloud. De façon spécifique, il apparaît que la quantité totale d'énergie nécessaire pour le fonctionnement du datacenters a baissé de plus de 5 Kwh sur l'échelle de notre simulation. Ceci représente un gain substantiel qui pourrait être encore plus important si l'on se projette sur des environnements beaucoup plus dense comme on les rencontre dans des cas réels. L'efficacité de cette approche se matérialise également à travers une réduction du nombre total de machine nécessaire, du nombre de migration effectué, du taux de violation des SLA, etc. Cette contribution est très importante en ce qu'elle représente la finalité principale de notre travail et prouve que l'intégration de la prédiction dans les méthodes d'amélioration de l'efficacité énergétique des datacenters est une piste très intéressante qu'il est nécessaire d'approfondir.

Chapitre 5

PLACEMENT DES MACHINES DANS UN CONTEXTE D'EFFICACITE ENERGETIQUE

Introduction

Le placement des machines virtuelles est l'une des tâches les plus exécutées par les administrateurs système des opérateurs de cloud computing. Il consiste à trouver dans une ferme de serveurs dont la taille peut varier de quelques dizaines à plusieurs milliers, la meilleure répartition des machines virtuelles pouvant répondre à la fois aux exigences de performance des applications qu'elles hébergent et aux contraintes d'optimisation énergétique globale du datacenters. Il s'agit donc de remplir les objectifs de qualité de service pour les clients et réduire les coûts d'exploitation en minimisant notamment le nombre totale des serveurs allumés. Si cette tâche peut paraître simple pour un datacenter de quelques serveurs, elle peut tourner au cauchemar dans un environnement de grande densité dans lesquels, les services aux utilisateurs sont censés remplir des contrats très strictes. Dès lors, il est indispensable de disposer d'outil intelligent de placement de machine conformes aux exigences ci-dessus évoquées. La recherche de solution dans ce domaine avance très rapidement au cours des dernières années et plusieurs outils commencent à émerger. Des algorithmes ont alors été développés pour optimiser les plans de répartition des machines virtuelles sur les serveurs physiques. Globalement, ces algorithmes se basent sur la résolution d'un problème de Bin Packing en deux dimensions avec différentes variantes. Largement répandu dans le domaine de la logistique et du transport, le Bin packing est l'algorithme de référence pour le placement d'objets de taille variable dans un conteneur en tenant compte de diverses contraintes définies par le contexte du problème traité. Fort de cette réputation, cet outil commence à s'adapter dans d'autres domaines où des problèmes de placement représentent une symétrie avec les domaines déjà maîtrisés. Ainsi, nous tentons de ramener le problème de placement de machine virtuelle au sein des

datacenters à un problème de bin packing avec des contraintes d'incompatibilité. Nous essayons dans ce chapitre de confronter deux méthodes principales de résolution de ces types de problèmes NP-difficiles . D'un côté la méthode dite « de force brute » entraînant des temps de calcul très élevés et incompatibles avec le processus de prise de décision dans le domaine du cloud computing et d'un autre côté la méthode heuristique du Bin packing qui combine certaines logiques permettant de trouver la solution la plus réaliste dans un délai acceptable. Cependant, le problème de placement des machines virtuelles au sein des datacenters doit prendre en considérations certaines autres conditions telles que les incompatibilités, la conformité des valeurs combinées de processeurs de mémoire au minimum et le respect des contraintes de SLA.

Au total, cette tentative de résolution des problèmes de placement de machines virtuelles en utilisant l'algorithme du *Bin packing* doit être continuellement améliorée étant donné que dans ce cas spécifique, les contraintes peuvent varier d'un contexte à un autre avec entre autre les exigences de plateforme pour certaines machines virtuelles, les incompatibilités dans les environnements de haute disponibilité, etc. La contribution de ce chapitre est le développement d'un algorithme qui se base sur les méthodes heuristiques et qui produit un plan d'occupation des machines virtuelles au sein des datacenters qui combine les valeurs de processeurs minimales et pouvant optimiser à la fois la quantité totale de ressources déployées et par conséquent améliorer l'efficacité énergétique au sein des datacenters. Il favorise la détermination du nombre optimal de serveur nécessaire pour héberger l'ensemble des VM d'un datacenters. Nous proposons par étape une démarche de prise en charge de la question qui inclus le traitement des contraintes préalables. En comparaison avec les méthodes dites de brute force, les algorithmes heuristiques fournissent de meilleurs résultats. Toutes ces méthodes se révèlent plus efficaces que la méthode manuelle qui est une pratique répandue auprès des administrateurs systèmes en l'absence d'outils. Le chapitre s'articule en quatre parties à savoir les fondements théoriques du Bin packing, le problème de placement de machine virtuelle dans le contexte du cloud computing, notre approche de la résolution du problème et enfin les expérimentations.

5.1 Problème de Bin Packing

Etant donné un ensemble d'objets de formes rectangulaires de dimensions quelconques connues et étant donné un bin de forme rectangulaire plus large de dimensions connues, le problème de bin-packing en deux dimensions (2BP) consiste à déterminer le nombre minimum de bins nécessaires pour ranger sans chevauchement l'ensemble de ces objets (les objets ne débordent pas des bins et ne se chevauchent pas)[man Jr et al. 1996], [Korf 2002], [Lodi et al. 2002].

Plus formellement, le problème de bin-packing en deux dimensions (2BP) est défini de la façon suivante : étant donné un ensemble de n objets rectangulaires $A = a_1, \dots, a_n$ et un nombre illimité de rectangles identiques (les bins) de dimensions plus larges que celles des objets, le problème consiste à déterminer le nombre minimum de bins à utiliser pour ranger l'ensemble des objets sans chevauchement.

5.1.1 Différents cas

- 1BP : une dimension
- 2BP : deux dimensions

5.1.2 Différents types de complexité

- objets de formes homogènes ou non homogènes ;
- objets de tailles uniformes ou différentes ;
- objets déformables ou non déformables
- le nombre de dimensions du problème ;
- disposer d'un seul bin (problème de décision ou problème de maximisation) ;
- contraintes d'équilibre entre les objets ;
- contraintes sur l'ordre dans lequel les objets doivent être retirés du bin ; ;
- chercher à minimiser le nombre de bins à utiliser ;
- contraintes d'orientation d'un objet ;

- contraintes de poids (par exemple, le poids d'un bin complet ne peut pas excéder une limite donnée) ;
- contraintes de placement, certains objets très lourds doivent être placés en bas, d'autres fragiles doivent être placés en dessus ; ;
- l'orientation : les objets peuvent être à orientation fixe (on parle du cas orienté) ou bien ils peuvent être tournés de 90 degrés (le cas non orienté) ;
- La contrainte guillotine : Si elle est imposée, on doit avoir la possibilité de restituer les objets rangés par des coupes bout à bout parallèles aux dimensions de bins ;

5.1.3 Différentes solutions

1. Méthodes approchées : solution non optimal mais résolution rapide. Choix de l'algorithme en fonction du problème

1. Méthodes heuristiques

En une phase : ranger directement les objets dans le BIN

- Next Fit : affectation de l'objet courant à la boîte courante s'il y tient. Sinon, fermeture de la boîte courante, ouverture d'une nouvelle
- First Fit : affectation de l'objet courant à la première boîte disponible
- Best Fit : affectation de l'objet courant dans la meilleure boîte
- Worst Fit
- Any Fit
- En 2 phases : résolution préalable du problème de strip-packing (rangement de tous les objets dans un Bin sans limite de hauteur) en commençant par les ranger par ordre décroissant (Decreasing height) avant rangement dans le Bin ; et à appliquer l'une des méthodes 1BP ci-dessus. On parlera de NFDH, FFDH ou BFDH

1. Il existe d'autres méthodes de résolution tels que les Métaheuristiques, la recherche tabou, la méthodes des bornes inférieures

1. Méthodes exactes : temps de résolution trop long et coûteux. Algorithme inadapté selon les cas. Inexistence d'un algorithme universel. Utilise la programmation sous contrainte ou programmation linéaire pour la résolution des problèmes
2. PSE (Procédure par Séparation/Evaluation, Branch and Band)

5.2 Placement initial des machines virtuelles dans le cloud computing

5.2.1 Problème de placement des machines virtuelles

Répartir des machines virtuelles sur les serveurs physiques peut paraître à première vue à une opération de division des capacités des ressources du serveur physique par les quantités des mêmes ressources requises par les machines virtuelles comme le montre la figure 5.1. Nombre d'administrateurs de centre de données s'exerce à résoudre manuellement la question en tentant du mieux de leur ingéniosité d'allouer efficacement les ressources aux VM. Cependant, avec les datacenters de grande taille, il est indispensable de trouver les outils permettant d'une part d'automatiser cette tâche et de rationaliser l'allocation des ressources d'autre part. Pour ce faire, la répartition des ressources se ramène généralement

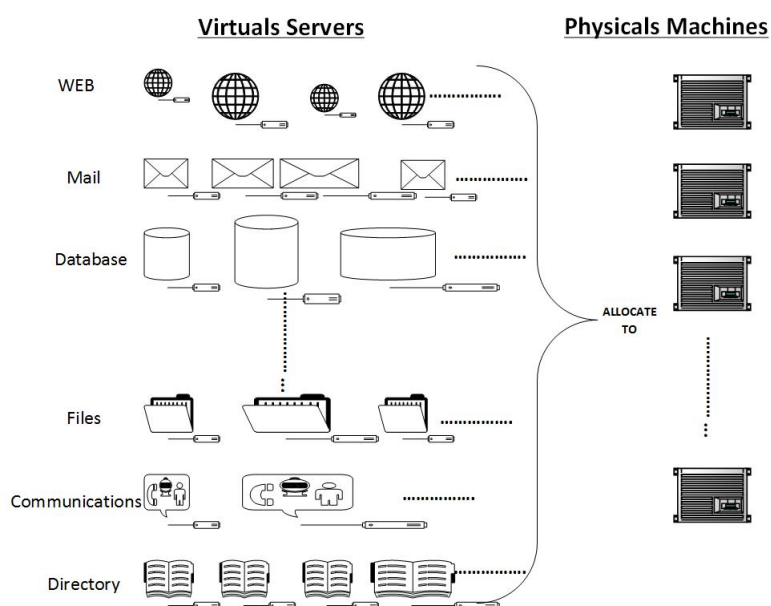


FIGURE 5.1 – Problème de placement de machines virtuelles

à un problème de Bin Packing à n dimensions dans lequel les quantités demandées pour chaque ressource constituent les objets à ranger pour cette dimension tandis que la taille de ladite ressource sur le serveur physique représente le bin de rangement. Les n dimensions pourraient être représentées par les ressources CPU, mémoire RAM, bande passante réseau ou disque dur. Certains travaux s'attèlent à résoudre le problème sur les n dimensions en une fois mais nous optons à une résolution par dimension. Ceci revient à trouver en fonction des demandes de CPU des serveurs virtuels, le nombre de serveur physique nécessaire pour les héberger. La forme standard d'expression mathématique du problème se présente comme suit.

Soit P l'ensemble des serveurs physiques

V l'ensemble des machines virtuelles à répartir

R_{ij} la quantité de la ressource i sur V_j

T_{ij} , la taille totale de la ressource i sur P_j

Il s'agit de trouver une matrice de correspondance avec $X_{ij} = 1$ si la VM i est placée sur la PM j et égale 0 sinon.

Cette matrice est réalisée avec les conditions ci-après :

- 1- La somme de la ressource pour toutes les VM placées sur un serveur donnée ne doit pas excéder la taille totale de la ressource sur ledit serveur
- 2- Toutes les VM doivent être placées. Ainsi, somme des $X_{ij} = 1$ quelque soit i
- 3- Répartir les VM sur le nombre optimal de PM. Ceci revient à trouver la fonction de minimisation qui donne le résultat le plus petit possible parmi un ensemble de solutions. Nous rajoutons à ce problème classique ainsi posé, une clause d'incompatibilité entre certaines VM. Concrètement, il s'agit de résoudre les contraintes techniques imposant une répartition des serveurs redondants au sein d'un cluster sur des machines physiques différentes de sorte qu'en cas de défaillance de l'un, la continuité du service soit assurée de façon automatique par l'autre. Pour ce faire nous implémenterons un prétraitement qui consistera à classer les serveurs en fonction de leur compatibilité.

5.2.2 Principe du placement initial

De nombreux algorithmes ont alors été développés pour optimiser les plans de répartition des machines virtuelles sur les serveurs physiques. Globalement, ces algorithmes se basent sur la résolution d'un problème de Bin Packing en deux dimensions avec différentes variantes. Nous présentons à la figure 5.2 les étapes de résolution du problème suivant notre approche. Deux méthodes principales de résolution de ces types de problèmes NP-difficiles s'affrontent.

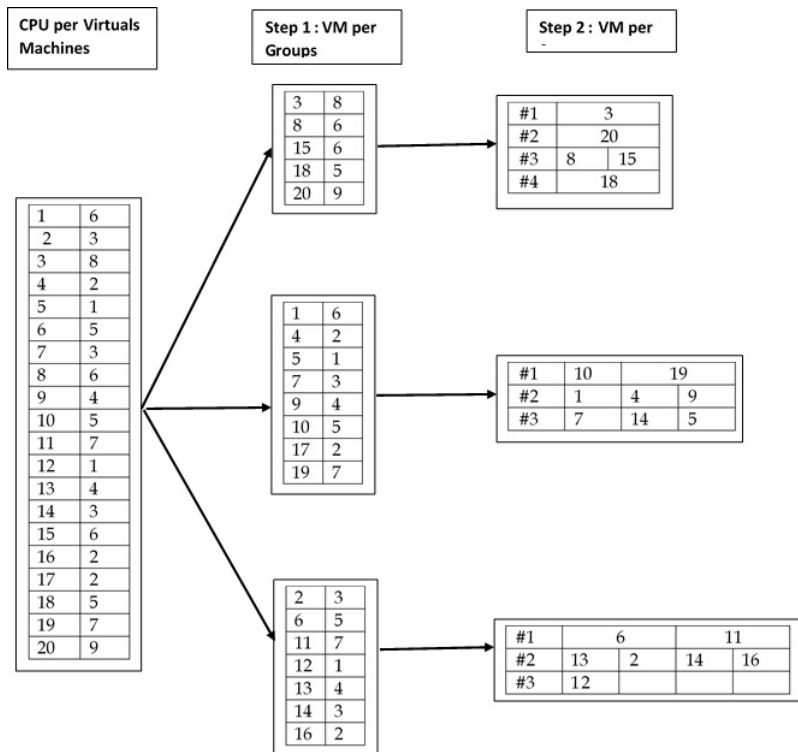


FIGURE 5.2 – Notre approche de placement

D'un côté la méthode dite « de force brute » entrainant des temps de calcul très élevés et incompatibles avec le processus de prise de décision dans le domaine du cloud computing et d'un autre côté la méthode heuristique qui combine certaines logiques permettant de trouver la solution la plus réaliste dans un délai acceptable. Cependant, le problème de placement des machines virtuelles au sein des datacenters doit prendre en considérations certaines autres conditions telles que les incompatibilités, la conformité des

valeurs combinées de processeurs de mémoire au minimum et le respect des contraintes de SLA. Nous proposons ici un nouvel algorithme figure 5.3 qui se base sur les méthodes

```
print("Entrez le nombre de Catégorie");
    scan (CAT) ;
    int cat = temp.nextInt();
    for(int j = 0 ; j < cat; j++){
        print("Entrez le nombre de VM");
        scan (n);
        print("Entrez la taille des processeurs de chaque VM");
        in = 0 ;
        for(int i = 0 ; i < n; i++){
            scan(val) ;
            in = concatenate(in, val);
        }
        print(in);
        print("Entrez la taille des processeurs des machines physiques");
        scan (p);
        bf = BinPackingBruteforce(in, p);
        ffd = FirstFitDecreasing(in, p);
    }
}

long startTime;
long estimatedTime;
startTime = System.currentTimeMillis();
print("needed bins (" + algoName + ") : " + algo.getResult());
algo.printBestBins();
estimatedTime = System.currentTimeMillis() - startTime;
print("in " + estimatedTime + " ms");
print("\n\n");
}
```

FIGURE 5.3 – Algorithme de placement

heuristiques et qui produit un plan d'occupation des machines virtuelles au sein des datacenters qui combine les valeurs de processeurs et de mémoire RAM minimales et pouvant optimiser à la fois la quantité totale de ressources déployées et par conséquent améliorer l'efficacité énergétique au sein des datacenters.

5.2.3 Approche de résolution

Les problèmes classiques de Bin Packing portent généralement sur les Bin de taille identiques. Nous considérons ici, par souci de simplicité, que tous les serveurs physiques disposent des mêmes caractéristiques techniques. Nous subdivisons le problème en deux parties que sont :

- la classification par groupe de compatibilité des serveurs virtuels

Cette étape est gérée de façon semi-automatique. Un opérateur devra inscrire manuellement les machines virtuelles dans un groupe déterminé. Il ne sera possible d'automatiser cette tâche qu'à travers un formulaire à remplir pour renseigner les services hébergés. Ensuite, un algorithme peut être implémenté pour effectuer des contrôles de compatibilité. Cependant, la décision de séparer les serveurs virtuels ne pourra être efficacement prise que par un ingénieur en toute connaissance des contraintes de sécurité, et de continuité de service nécessaire dans les environnements de datacenter afin d'honorer les engagements de SLA.

- la détermination du nombre de serveurs physiques nécessaires par groupe et les regroupements des serveurs virtuels par serveurs physiques

Notre démarche à ce niveau se base sur une adaptation de la méthode du *First Fit Decreasing* (FFD) à l'allocation de machine virtuelle sur les serveurs physiques. L'heuristique FFD se résout en deux phases à savoir : Une première phase consiste à ranger toutes les valeurs des ressources requises par les VM dans un ordre décroissant. Puis l'algorithme du rangement First Fit s'applique pour remplir successivement les serveurs physiques jusqu'à leur limite autorisée dans une seconde phase. Ainsi, les VM sont rangées sur les serveurs jusqu'à l'allocation globale de la liste des requêtes d'hébergement.

Les algorithmes du First Fit Decreasing et du Best Fit Decreasing se concurrencent fortement dans la résolution des problèmes similaires. Tandis que dans le BFD, un critère d'optimalité est rajouté pour le choix des Bins, le FFD procède au rangement Bin après Bin uniquement. En assumant que tous les Bin sont de tailles identiques, cette étape supplémentaire implémentée dans le BFD ne rajoute aucune efficacité. Pour ce faire, il

est utile de choisir le FFD pour économiser en terme de temps de traitement et donc de performance de l'opération. Etant donné les besoins d'opération en temps réel au sein des datacenters, ce gain en temps de traitement se justifie pleinement.

5.3 Simulation et résultats

5.3.1 Expérimentation

Nous essayons de démontrer à travers des simulations les performances de notre approche de résolution du problème de placement des machines virtuelles en comparant une implémentation du FFD avec la méthode dite de brute-force. Le problème de Bin packing étant NP-Hard, la résolution par la méthode de Brute-force consiste à parcourir à chaque occurrence de machine virtuelle, toutes les machines physiques afin de déterminer la possibilité de les allouer ou non.

Le cadre de notre étude est un datacenter institutionnel de l'Etat béninois dans lequel, les responsables des systèmes d'informations de plusieurs structures de l'administration expriment régulièrement des besoins d'hébergement de leurs applications. Ce datacenter contient 110 serveurs mais nous limiterons notre étude à 20 serveurs HP Blade BL 350 ayant des spécifications identiques et pouvant respecter les contraintes des problèmes de Bin Packing. Les caractéristiques des serveurs sont équivalentes à celles d'un HP proliant DL 380 Gen 98.

Afin de faciliter l'entrée des données dans notre programme, nous représenterons les performances des processeurs en MIPS (Millions d'instructions par seconde) de façon à ne pas avoir de nombres décimaux. Les machines virtuelles quant à elles sont sollicitées à travers un formulaire de demande d'hébergement précisant entre autres les quantités de MIPS requises. Ici également, nous nous limitons à une vingtaine de demande de machines virtuelles. Le récapitulatif des demandes de ressources se présente dans le tableau 5.1

Les machines sont virtualisées en utilisant l'hyperviseur VMWARE ESXI 5.5. Le but de notre simulation est de comparer le nombre de serveur actuellement utilisé au sein du

N	Demande de CPU	Rangement par groupe
1	6	1
2	3	1
3	8	1
4	2	1
5	1	1
6	5	1
7	3	1
8	6	2
9	4	2
10	5	2
11	7	2
12	1	2
13	4	2
14	3	3
15	6	3
16	2	3
17	2	3
18	5	3
19	7	3
20	9	3

TABLE 5.1 – Rangement des serveurs par groupe de compatibilité

datacenter en utilisant une méthode manuelle très proche du brute-force avec celui obtenu en utilisant l'algorithme modifié du *First Fit Decreasing*. La simulation consiste à entrer dans le programme développé à cet effet, les besoins des usagers du datacenter ainsi que la quantité disponible de la ressource concernée au niveau du serveur. L'algorithme fourni en sortie pour chaque méthode, le nombre de serveurs physiques nécessaires ainsi que le temps de traitement.

5.3.2 Résultats

Au regard des résultats ci-dessous présentés dans le tableau 5.2, on aperçoit clairement que la détermination du schéma de placement initial des VM est plus optimal par la méthode de Bin Packing. L'évaluation montre également que le temps de calcul par la méthode du Brute-force est très élevé par rapport au FFD d'un facteur de 10^3 . La répartition préalable des machines par groupe de compatibilité est une particularité de

Méthodes	Groupe 1		Groupe 2		Groupe 3	
•	Nb de serveurs	Durée	Nb de serveurs	Durée	Nb de serveurs	Durée
Manuel	4	NA	4	NA	4	NA
Brute Force	4	0	3	875	3	31
Bin packing FFD	4	0	3	0	3	0

TABLE 5.2 – Performance des deux méthodes

notre étude étant donné que cette réalité est souvent ignorée dans de nombreux travaux similaires alors même qu'elle constitue une règle de base du placement des VM au sein des datacenters

En rappelant que notre échantillon est de taille réduite par souci de simplicité, il est utile de remarquer que la durée de détermination du schéma de placement peut être multiplié par plusieurs lorsque l'on considère un datacenter de grande taille (plusieurs centaines ou milliers de serveurs). Le recours à une technique de type FFD est donc indispensable afin de répondre efficacement au flux des requêtes d'hébergement dans les datacenters.

La détermination d'un schéma optimale de placement des machines virtuelles au sein d'un datacenters ne représente cependant que le début des tâches de placement étant donné que les quantités de ressources sollicitées au départ sont consommées de manière très variable après la mise en production entraînant d'énormes pertes de ressources. Ces sous-consommations résultent en un gaspillage d'énergie électrique car une fois approvisionnée, les ressources des VM consomment presque les quantités d'énergie que lorsqu'elles sont en pleine charge. Pour pallier ce phénomène, une réallocation périodique des machines virtuelles sur les serveurs physiques est indispensable. Toutefois, même si les méthodes du Bin Packing peuvent être toujours utiles, d'autres algorithmes basés sur des critères adéquats sont nécessaires.

Conclusion

L'optimisation de la consommation énergétique au sein des datacenters est un domaine où la recherche est à ce jour, très féconde. La clé réside dans la virtualisation, l'élaboration de techniques d'amélioration de l'efficacité électrique des serveurs déployés et une meilleure allocation des serveurs virtuels. Avec les besoins en permanente croissance du cloud computing, le problème de placement des machines virtuelles sur les serveurs physiques déployés dans les datacenters est devenu central. Elle se subdivise en deux parties que sont : i) le placement initial et ii) la réallocation optimale en fonction des besoins réels des utilisateurs au fil du temps. Notre but au cours de ce travail a été de proposer une approche de solution à la question du placement initial à travers les méthodes du Bin Packing.

Fort des résultats produits dans d'autres domaines notamment la logistique, les algorithmes basés sur le Bin Packing sont porteurs de promesses dans le cadre de l'amélioration de la consommation des ressources sur les serveurs au sein des datacenters. En effet, nos tests sur un échantillon tiré d'un environnement réel montrent en comparaison à d'autres méthodes traditionnelles de placement des VM sur les serveurs que l'heuristique du First Fit Decreasing permet de réduire de façon sensible les gaspillages de ressources sur les serveurs physiques. A ceci s'ajoute l'optimisation du temps nécessaire pour la détermination du schéma de placement adéquat. Cette réduction du temps nécessaire aux administrateurs de datacenters à la prise de décision d'hébergement des besoins des clients est d'autant plus importante que la vitesse d'adoption des technologies du cloud computing exige plus de réactivité et plus d'efficacité pour une meilleure satisfaction des usagers.

Les résultats obtenus ne sont cependant qu'une piste de résolution des questions d'efficacité énergétique au sein des datacenters. La taille croissante de ces derniers, la variété des contraintes liées à la technologie et aux exigences des clients impliquent l'élaboration et la validation dans des environnements varié, d'autres techniques permettant de rendre dynamiques les solutions de placement en cours de production. Les méthodes de migration de machines virtuelles existantes pourraient être améliorées en les combinant avec les outils de *Bin Packing*.

Chapitre 6

CONSOLIDATION PREDICTIVE DES MACHINES VIRTUELLES

Introduction

6.1 Introduction

Le respect des accords de niveau de service est un principe fondamental des environnements de Cloud Computing. L'externalisation des services informatiques en dépit des doutes qu'elle suscite auprès de bien d'entreprises, présente l'avantage certain de la disponibilité des services et de la facilité de leur déploiement et exploitation. Ces exigences reposent essentiellement sur une bonne conception des infrastructures de datacenters et surtout sur les efforts quotidiennement fournis pour leur exploitation. Le challenge est énorme et requiert de la part des ingénieurs, de développer des techniques permettant de s'adapter de façon dynamique au flux de requêtes arrivant de façon totalement aléatoire vers les serveurs. Les constructeurs d'équipements, notamment les serveurs et le stockage pour datacenters mettent en œuvre des fonctionnalités natives d'optimisation des ressources des serveurs à travers la régulation dynamique de l'énergie consommée par le serveur en fonction de sa charge de travail. D'autres techniques encore moins sophistiquées consistent à détecter et à éteindre manuellement les machines en état d'inactivité et à les rallumer lorsque le besoin devient effectif dans le but d'économiser de l'énergie. Toutes ces méthodes peuvent se révéler inefficace tant les environnements de Cloud Computing peuvent être différents les uns des autres. En outre, priver les serveurs de ressources de manière hasardeuse peut se révéler fatale étant entendu que certains trafics prioritaires ou sensibles peuvent échouer et entraîner par conséquent des scénarii de violation de SLA parfois très onéreux pour les opérateurs.

Dès lors, il urge d'adopter des approches proactives de l'optimisation des ressources qui tiennent compte à la fois des besoins d'économies énergétiques et des exigences en termes

de disponibilité des serveurs. Nous combinons ici deux méthodes qui pourraient aider à cette fin. D'abord la migration des machines virtuelles encore appelée consolidation de machines virtuelles consiste à surveiller les VM en sous-consommation et en surconsommation et de les déplacer vers de nouveaux serveurs physiques pour respectivement éteindre leur serveurs initiaux et pour éviter des violations de SLA. Dans le premier cas, les sous-consommations de ressources sur les VM résultent en un gaspillage d'énergie car les processeurs et mémoires RAM continuent de fonctionner même lorsqu'il n'y a plus d'activité. Dans le second cas, les surconsommations de ressources entraîne une augmentation des temps de réponse y compris pour des requêtes prioritaires et donc un risque de lenteur qui souvent dans les contrats de SLA fait l'objet de pénalités énormes pour certains secteurs d'activités. Ensuite, il faut anticiper sur le moment de réalisation des migrations. La prédiction de l'arrivée des charges de travail au sein des datacenters peut être très utile dans la mesure où cela permet d'automatiser le moment opportun pour effectuer une migration de machine virtuelle et éventuellement une extinction de machine physique. L'objectif est de comprendre la rythmique et la périodicité des requêtes arrivant sur les serveurs et d'anticiper sur les pics et les vallées de celles-ci pouvant occasionner une sous consommation ou une surconsommation. En outre, sur la multitude d'outils de prédiction, nous avons retenu ARMA qui offre à la fois une meilleure fiabilité des résultats en comparaison avec les méthodes moins sophistiquées telles que les méthodes dites « naïves » et une simplicité de mise en œuvre par rapport aux méthodes plus sophistiquées notamment l'apprentissage par ordinateur.

Nous implémentons ici cette combinaison de ces deux techniques à travers le simulateur CloudSim qui permet de tester le fonctionnement des environnements de Cloud Computing dans des contextes très variés incluant le contrôle de la consommation énergétique. Nous modifions quelques modules pour y ajouter la prédiction ARMA. Les résultats révèlent une meilleure amélioration des performances

6.2 Principe de la consolidation des machines virtuelles

Nous présentons ci-après le cadre opératoire ainsi que la démarche de simulation de la consolidation de machines virtuelles dans un environnement de cloud

6.2.1 Cadre de l'étude

Considérons une ferme de cinq serveurs physique de grande capacité comportant chacun un nombre indéterminé de machine virtuelle hébergeant différentes applications remplissant toutes les conditions de cohabitation. En situation normal de fonctionnement, les machines virtuelles consomment chacune selon la charge de travail qu'elles exécutent. Considérons également que les serveurs physiques, pour répondre aux contraintes de SLA signées avec les clients devront maintenir leur taux d'utilisation de processeur entre un intervalle de 20% et 80%. La consolidation des machines virtuelles est l'opération qui consiste à effectuer des migrations de machines virtuelles d'un serveur physique à un autre de manière à maintenir le taux de consommation de ressources à l'intérieur de l'intervalle défini.

En d'autres termes, lorsqu'un serveur ne reçoit pas suffisamment de charge de travail pour consommer plus de 20% de sa capacité de traitement, il est considéré comme étant en sous-consommation (confère figure 6.1) et doit être éteint. Pour ce faire, toutes les machines virtuelles qu'il héberge devront être migrées vers un autre serveur dont le taux de consommation se trouve à l'intérieur de l'intervalle et qui est capable de reprendre toute la nouvelle charge sans dépasser la limite supérieure admise. De même, lorsqu'un serveur vient de dépasser la limite supérieure de consommation de ressource (confère figure 6.2), un désengorgement est initié. il consiste à identifier les machines virtuelles dont la migration permettra de ramener le taux de consommation à l'intérieur de l'intervalle fixé.

la consolidation des VM répond à une logique de proportionnalité entre l'énergie fournie au serveur et le volume de traitement exécuté. Lorsqu'ils opèrent à partir de 20% de leur capacité de traitement, les serveurs au sein des datacenters peuvent consommer plus de 70% de leur pic de consommation énergétique. Il en résulte une perte énorme de

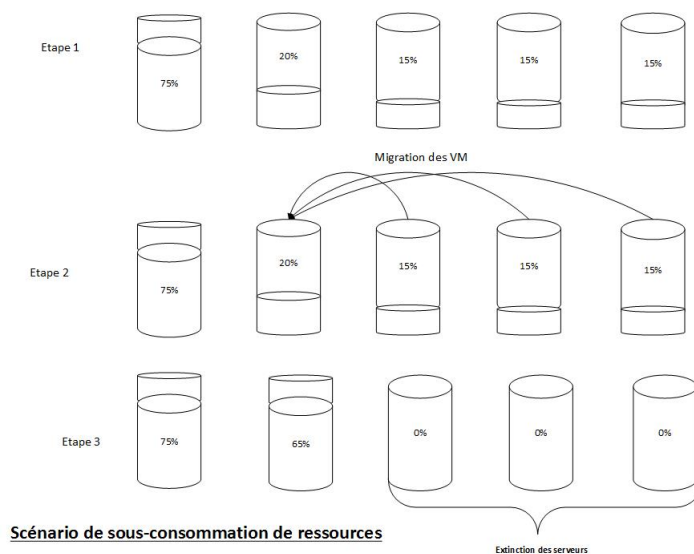


FIGURE 6.1 – Scenario de sous consommation de ressources des serveurs

l'énergie fournie mais qui n'a pas réellement servie à l'exécution d'une tâche productive. En considérant des datacenters hébergeant des milliers de serveurs, cette quantité perdue pourrait représenter des valeurs élevées en KWh occasionnant une augmentation des coûts opérationnels. la consolidation des VM permet de concentrer le maximum de tâches sur un nombre rationnel de serveurs physique et de garder hors tension tous les autres serveurs. Le gain en matière énergétique serait substantiel si cette pratique arrivait à être généralisée au sein des grands datacenters au niveau mondial.

6.2.2 Contraintes de la consolidation

La consolidation des machines virtuelles est une opération devant répondre à des contraintes bien défini au regard des objectifs de performance et de disponibilité des services des datacenters. Il faut comprendre avant tout que la consolidation n'implique que les serveurs éligibles. Dans une architecture d'IaaS (Infrastructure as a Service, les clients ont l'entière disponibilité de leur serveur et son chargé eux même de son administration. les opérations de consolidation ne pourront donc se faire que dans les contextes de PaaS (platform as a Service) ou de SaaS (Software as a Service) où l'opérateur rend entièrement transparente les couches inférieures aux clients. Ainsi, les serveurs sous contrat IaaS

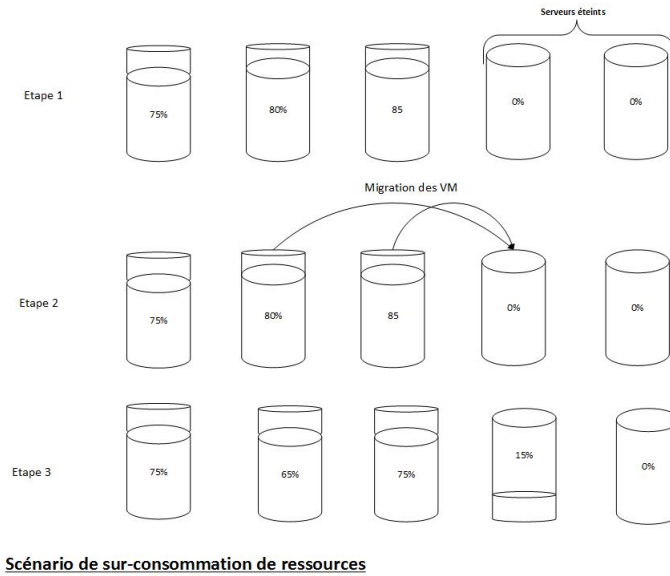


FIGURE 6.2 – Scenario de surconsommation de ressources des serveurs

ou de co-location ne font pas partie du périmètre de ces opérations. Ensuite, dans les architectures de haute disponibilité, les serveurs peuvent être en cluster ou en réplication afin d’assurer la sécurité des données et la continuité des services. Dans ces cas, les migrations de serveurs devront tenir compte de la bonne répartition et du maintien des machines virtuelles sur des serveurs séparés. Sur la base de ces exigences d’architecture, les opérations de consolidation des serveurs devront remplir les contraintes ci-dessous :

- le choix des serveurs cibles : il est important de s’assurer que les serveurs choisis pour héberger les VM migrées ne dépassent pas à leur tour les seuils supérieurs d’utilisation des ressources. En effet, le dépassement des seuils pourraient entrainer des temps de latence qui impacteraient négativement les Accords de niveau de service (SLA) signés avec les clients. il est donc nécessaire d’effectuer des calculs nécessaires afin de s’assurer du choix optimal de serveur cible. En outre, il faudra assurer qu’après la migration l’utilisation des ressources se trouverait dans un intervalle de confiance qui ne risque pas d’induire très rapidement un nouveau besoin de migration. Les besoins trop fréquents de migration dénote en effet d’un défaut de conception et résulte en une instabilité de l’architecture.

- le choix des VM à migrer : dans un scénario de sous consommation de ressource, ce problème ne se pose pas étant donné que toutes les VM sans exception seront migrées vers un nouveau serveur. En revanche, lorsqu'il s'agit d'un surconsommation, le but est de choisir une ou les VM dont la migration permettrait non seulement de ramener les taux d'utilisation de ressources des serveurs hôtes à l'intérieur de l'intervalle prédéfini mais aussi d'éviter qu'une nouvelle opération de migration soit nécessaire sur le même serveur à court terme. Ainsi, un algorithme permettrait d'élaborer une liste optimale des VM éligibles à la migration.
- le temps de migration : ce paramètre est très important. le temps de migration doit être calculé afin de s'assurer que l'opération n'excèdera pas la durée maximale d'indisponibilité admissible au regard des SLA. Ainsi, une VM dont le temps de migration respecterait ce critère sera plus susceptible d'être choisi plutôt qu'une autre. Ce paramètre est calculé suivant la formule ci-après : Soit T la taille totale de la VM, V , le débit du lien existant entre les deux serveurs source et destination. Nous noterons F_t le facteur de traitement des serveurs qui équivaut au taux d'occupation actuelle du serveur et donc à sa capacité à traiter une opération de copie ou de coller. le temps de migration TM est donné par la relation

$$TM = T * VM * Ft1 * Ft2$$

Le facteur de traitement peut être obtenu en faisant un ratio entre l'occupation actuelle du serveur et sa capacité de traitement total.

$$Ft = T / (Ct - O)$$

- le temps de remise en service : la remise en service une fois la migration effectuée doit être le plus rapide possible. Cee temps est à distinguer du temps de migration mais il compte pour le temps d'indisponibilité totale de la VM. Il est important de maintenir une relation entre le downtime admis pas les SLA, le temps de migration et le temps de remise en service. Cette relation peut s'écrire sous la forme

$$SLA(downtime) < TM + TRS$$

6.2.3 Etapes de consolidation

La décision de migration doit être précédé des étapes permettant de garantir les contraintes ci-dessus énumérées. La figure ci-dessous présente les différentes étapes précédant une migration, notamment :

6.2.3.1 Définition des intervalles de fonctionnement des ressources

Généralement fixé entre 20% et 80%. Le seuil inférieur de 20% est considéré comme la valeur à partir de laquelle le ratio entre les ressources qui se déploient et le taux d'occupation commence à être pertinent. Il en découle qu'il n'est pas justifié de laisser un serveur qui tourne à moins de 20% d'utilisation étant donné qu'il consommerait la même énergie que si l'on le faisait travailler à 70%. Avec les technologies de virtualisation il est devenu plus simple d'occuper les ressources des serveurs pour des besoins productifs. A partir de 80% d'utilisation, le temps de latence des serveurs devient élevé et il peut en résulter une baisse de performance. Il est nécessaire de penser à réguler. La plupart des outils de simulation tel que CloudSim utilise cet intervalle. Cependant, certains serveurs pourraient tolérer des seuils encore plus serrés. En dehors des méthodes de seuil, une migration peut être nécessaire lorsqu'il existe d'autres types de corrélations entre certains taux de consommation et les problèmes de sur ou de sous consommation.

6.2.3.2 Identification de la sous-consommation ou de la surconsommation de ressources sur les serveurs

Il est généralement fixé un seuil haut et un seuil bas de ressources utilisable sur les serveurs de sorte que les traitements envoyés à ceux-ci permettent effectivement d'absorber ces ressources et de ne pas les déborder. Leur utilisation efficiente réduit le risque de gaspillage d'énergie fournie pour les alimenter et par conséquent optimise le coût global de la facture électrique dans les datacenters. En outre, il est important de prévenir leur débordement car cela résulterait en temps de latence et en pertes de paquets incompatibles avec les applications de plus en plus exigeantes hébergées dans les Cloud datacenters.

6.2.3.3 Définition des critères d'éligibilité des VM à la migration

Lorsqu'une surconsommation est établie, une sélection des VM éligible à la migration est initiée. Cette sélection respecte les contraintes ci-dessus énumérées notamment la disponibilité des ressources au niveau des serveurs cibles, le maximum de corrélation, la conformité des temps de migration avec les SLA ou à défaut une sélection aléatoire sera faite.

6.2.3.4 Sélection des machines virtuelles à migrer

Les machines virtuelles à migrer sont celles selon le cas présentant l'une des caractéristiques suivantes :

- la plus grande probabilité d'entraîner un débordement du serveur,
- dont le temps de migration estimé est minimal,
- qui est le moins chargé possible ou
- qui est choisi de manière totalement aléatoire. Le choix du critère dépend fortement du scénario envisagé pour une simulation et des objectifs visés.

Autrement dit, pour une migration de VM, lorsqu'il se pose la contrainte de respect des SLA, celle qui ont les temps de migration les plus faibles seront les plus éligibles afin de garantir les downtimes les plus faibles.

6.2.3.5 Recherche de serveurs cibles

Au delà des serveurs exclus en raison des contraintes d'architecture, l'algorithme de migration des VM va scruter les serveurs en fonctionnement et vérifier s'ils sont capable d'héberger les nouvelles charges de VM à migrer sans entraîner une surconsommation. les serveurs cibles seront choisis parmi ceux qui sont les moins chargés ou à défaut, de nouveaux serveurs seront mis en marche. Dans les scénarii d'optimisation de l'énergie totale consommée dans un datacenter, la démarche serait de laisser éteint le maximum de serveurs et de ne déclencher un allumage qu'en cas de stricte nécessité. Le choix final du serveur tiendra compte de sa charge totale après la migration des VM candidates. Cette

charge doit se situer en dessous du seuil total fixé.

6.2.3.6 Migration

Une fois les VM et les hôtes sélectionnés, un plan d'allocation optimal est élaboré et un processus de migration est déclenché ou planifié. La figure 6.3 présente un processus classique de migration de VM dans le contexte des datacenters. Les paramètres de seuil sont également définis afin d'indiquer les taux maximal d'occupation des serveurs sélectionnés. L'opération de migration peut recommencer autant de fois que nécessaire jusqu'à l'obtention d'une configuration stable ou tant que les objectifs d'énergie totale consommée ou de respect des SLA ne sont pas atteints.

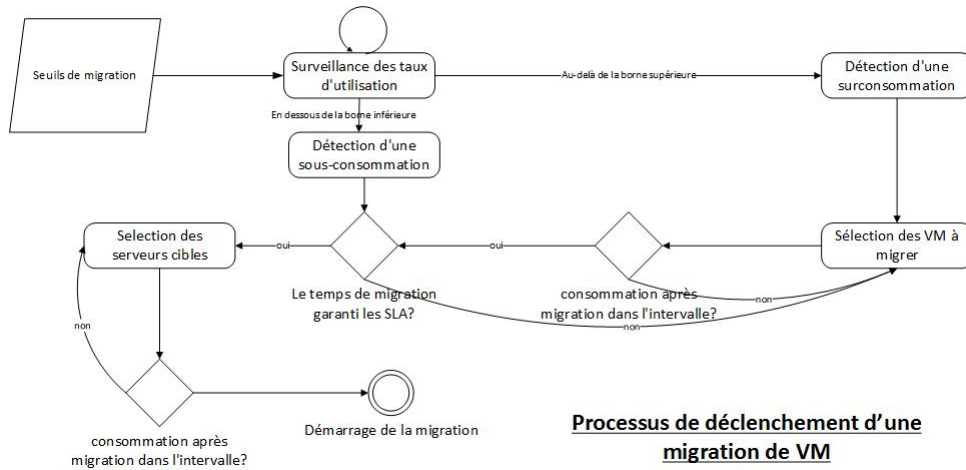


FIGURE 6.3 – Processus classique de la migration de VM

6.3 Intégration de la prédiction dans le processus de consolidation de VM

6.3.1 Architecture du modèle

Notre modèle de consolidation de VM que nous appelons "consolidation prédictive" (confère 6.4) est une amélioration des workflows classiques basée sur une évaluation des besoins futures et une planification des approvisionnements en ressources. Le processus de migration présenté ici découle du principe implémenté dans le simulateur CloudSim

et décrit ci-dessus auquel nous rajoutons en amont une étape de prédiction basée sur les outils ARMA. Nous partons donc de notre approche de prédiction présentée au chapitre 4 intitulé « prédiction basée sur les outils ARMA ». Pour rappel, notre démarche consiste à partir des données historiques que nous échantillonnons par fichier de X valeurs pour lesquelles nous prédisons la valeur $x + 1$ à chaque fois en faisant glisser la fenêtre de prédiction. Les valeurs prédites à court terme à permettent en les additionnant pour toutes les VM contenues sur un serveur, de déterminer l'occupation totale du serveur à $x + 1$ et d'anticiper la migration.

Les données utilisées sont celles intégrées au simulateur CloudSim et provenant des traces PlanetLab contenant des informations complètes de trafics traités dans des datacenters de taille moyenne. La description desdites données présentée en annexe reste très proche de celle de Google Cluster à partir desquelles nous avons élaboré nos modèles de prédiction. Ci-dessous le processus de migration. L'anticipation de la migration poursuit deux objectifs à savoir :

L'optimisation de la consommation des ressources : le processus ARMA permet d'évaluer la survenance d'un phénomène de sous-consommation à court-terme. Il est implémenté pour calculer sans cesse les futures valeurs de la consommation de ressources physiques sur les serveurs. A chaque fois d'une sous-consommation est détectée, une action de migration peut être enclenchée pour libérer le serveur concerné et l'éteindre éventuellement. Lorsque l'on projette cette opération sur un parc de serveur de taille moyenne à grande, le nombre de serveur à éteindre par cette technique peut devenir significatif et induire une réduction d'énergie conséquente.

La garantie des SLA : la prédiction des valeurs à court terme de la consommation de ressources sur les VM peut favoriser une détection anticipative de surconsommation en vue d'une planification des migrations qui garantisse la continuité des services. Cette technique permet une meilleure maîtrise du temps maximal de migration qui peut être un facteur de rupture de service si la migration n'a pas démarré à temps. Estimer le moment de la surconsommation est potentiellement une garantie d'assurer la migration des serveurs sans risque de rupture de service.

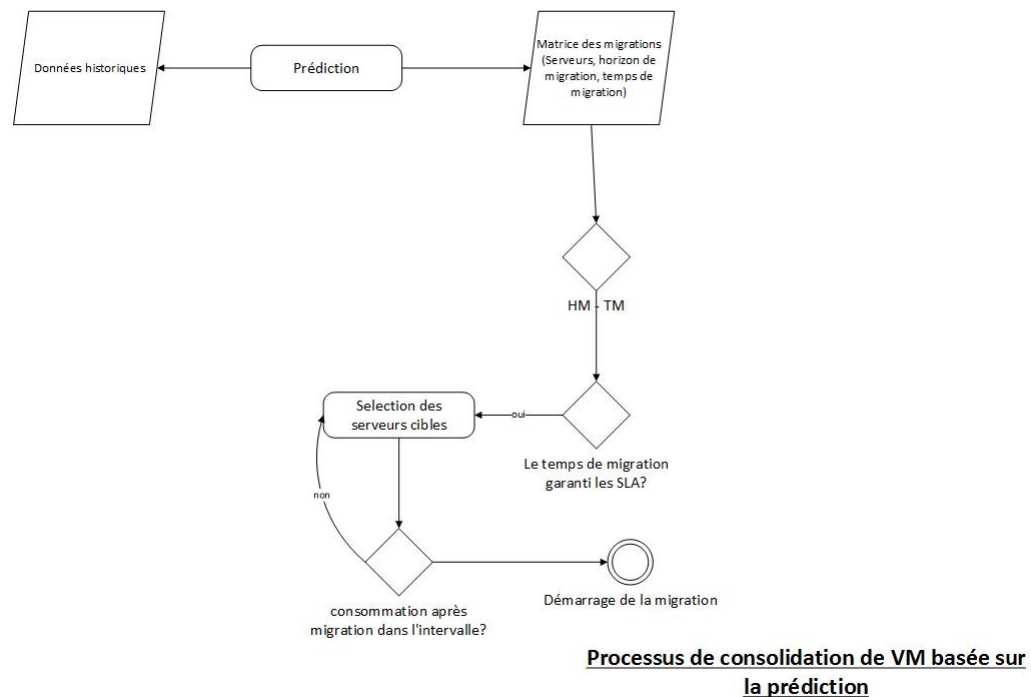


FIGURE 6.4 – Processus de migration de VM basée sur la prédiction

6.4 Application du modèle

6.4.1 Paramètres d'expérimentation

La simulation de notre approche d'optimisation de ressources sur les serveurs au sein des datacenters vise à évaluer l'impact de la prédiction basé sur ARMA sur les processus standards de migration de VM. Le cadre de travail est basé sur le simulateur Cloudsim 3.0 exécuté dans un environnement Java Eclipse. Les résultats des simulations de notre approche basée sur ARMA seront comparés avec deux stratégies de migration de VM implémentées dans Cloudsim à savoir :

Local Regression (LR) : la régression locale est une méthode qui consiste à déterminer, pour chaque point du jeu de données initial, les coefficients d'un polynôme de faible degré pour effectuer la régression d'un sous-ensemble des données, les valeurs des variables aléatoires étant proches du point pour lequel on effectue la régression, puis à calculer la valeur de ce polynôme pour le point considéré.

Local Regression Robust (LRR) : est une méthode pour lisser un diagramme de dispersion, (xi, yi) , $i = 1, \dots, n$, dans lequel la valeur ajustée à xk est la valeur d'un polynôme correspondant aux données en utilisant la méthode des moindres carrés pondérés où le poids pour (xi, yi) est grand si xi est proche de xk et petit s'il ne l'est pas.

Pour chacune des stratégies, nous appliquons les critères de :

Maximum Correlation (MC) : l'algorithme du maximum de corrélation calcule initialement à partir des données historiques un coefficient de corrélation (CC) de chaque VM candidate et la migration est décidée pour la VM avec la plus grande valeur.

Minimum Migration Time (MMT) : A ce niveau, une estimation du temps de migration est faite sur la base de la taille de cette dernière et du temps d'extinction et de rallumage afin de garantir l'intégrité des données. Le choix sera porté sur la VM dont l'estimation de cette valeur est la plus faible que possible. Cette stratégie peut permettre de minimiser lorsque cela s'avère inévitable, le temps d'interruption du service

Nous modifions ces modules pour implémenter la prédiction ARMA comme stratégie de migration. Le jeu de données PlanetLab intégré au simulateur Cloudsim se présente dans le même format que celui de Google cluster utilisé pour établir notre modèle ARMA. Ces données proviennent de traces réelles et se compose de 800 serveurs essentiellement des HP Proliant ML 110 G4 et G5 pour lesquels les quantités de CPU ont été présentées sous formes de MIPS. Les traces contenues dans la collection correspondent à 3 jours de fonctionnement du datacenter. La simulation consiste donc à exécuter les 3 stratégies de migration de VM en faisant varier les 2 critères ci-dessus et à comparer les résultats suivant les critères d'évaluation ci-après :

6.4.2 Résultats

L'implémentation de notre approche de réduction de la consommation des ressources dans les datacenters

6.4.2.1 Maximum Correlation VS Minimum Migration Time

D'abord, l'expérience révèle que l'hypothèse du « Maximum Correlation » produit des résultats encore plus satisfaisants que celle du « Minimum Migration Time ». Sur l'échelle de notre simulation l'énergie sauvée peut aller à 5Kwh comme le montre la figure 6.5. Lorsqu'on se projette dans un datacenter réel, ce gain pourrait être énorme sur une longue période.

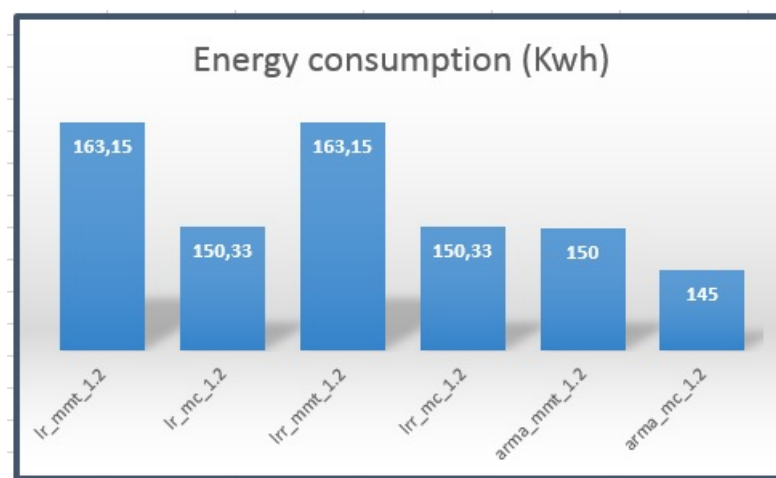


FIGURE 6.5 – Energie consommée

6.4.2.2 Impact sur le nombre de migration

Ensuite, nous remarquons sur la figure 6.6, que l'introduction de l'approche prédictive permet de réduire le nombre de processus de migration déclenchés. Avec notre technique, ce nombre a été réduit de près de 3000 migrations en comparaison avec les méthodes de régression locale tant dans l'hypothèse du MC que celle du MMT. La réduction du nombre de migration favorise la stabilité de l'architecture système du datacenters notamment à travers une réduction du recours à l'extinction des serveurs (confère figure 6.7). En corrélant la réduction totale de la consommation d'énergie consommée avec cette diminution du nombre de migration, on peut conclure que l'approche anticipative permet de trouver de façon stable l'allocation des VM la plus économique possible.

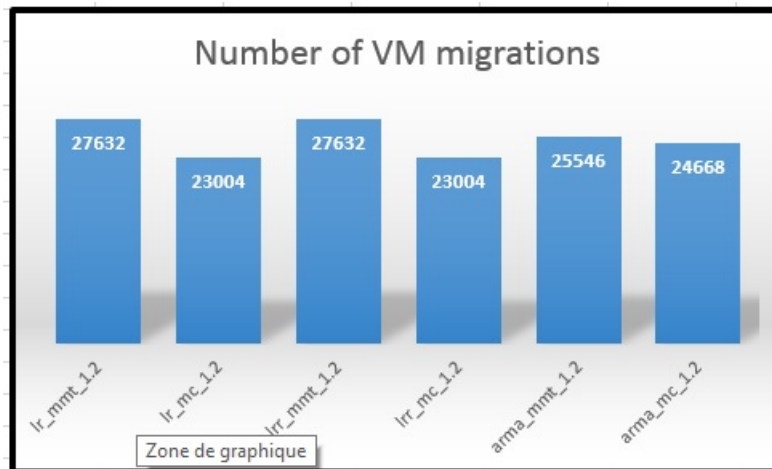


FIGURE 6.6 – Nombre de migration

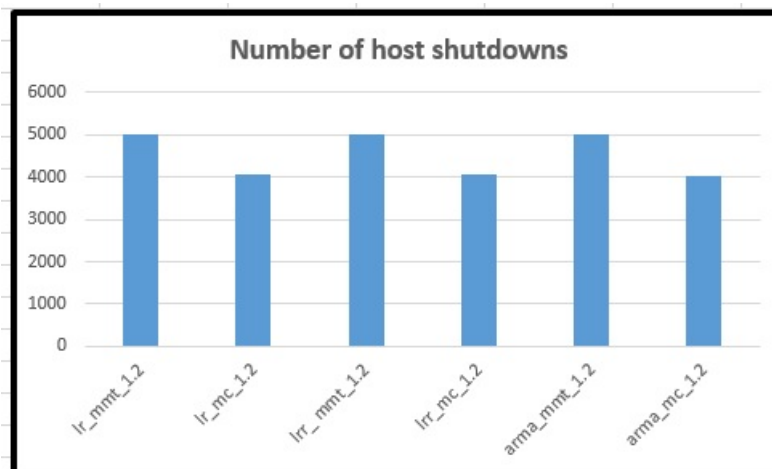


FIGURE 6.7 – Nombre de serveurs éteints

6.4.2.3 Impact sur le respect des SLA

Dans la configuration basée sur la prédiction, le respect des SLA est maintenu à un niveau stable autour de 9% en comparaison aux méthodes de régression locale. En dépit d'une légère baisse des violations enregistrées (confère figure 6.8), les trois méthodes permettent globalement de garantir la qualité de service tout en optimisant la consommation des ressources.

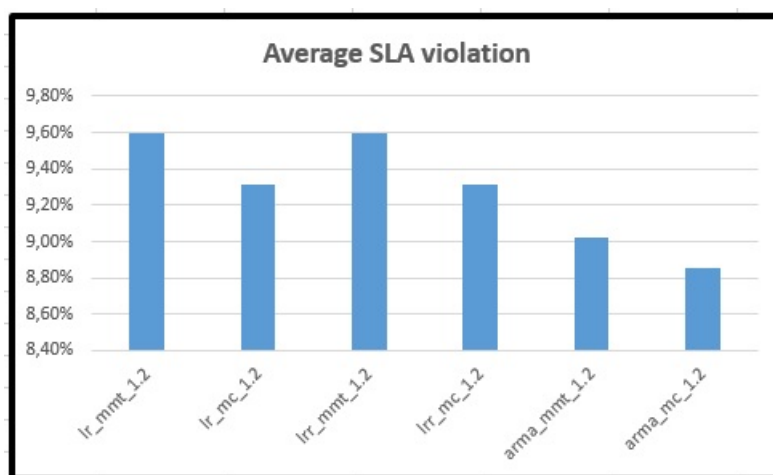


FIGURE 6.8 – Taux de violation des SLA

Conclusion

Réduire la consommation énergétique des datacenters tout en préservant les contraintes de qualité de service reste un challenge qu'il faudra absolument résoudre pour assurer l'avenir des technologies du numérique sans compromettre notre environnement. Au-delà de toutes les techniques déjà développées pour l'atteinte de l'objectif, nous estimons qu'il pourrait être efficace d'améliorer la compréhension des trafics au sein des datacenters afin de proposer une méthode anticipative. Ce projet a permis de mettre en exergue l'apport de la prédiction et de l'analyse anticipative sur la rationalisation des ressources sur les serveurs dans un contexte de fourniture de services cloud. Cet apport est relevé à différents niveaux.

En somme, l'approche prédictive présente un meilleur compromis entre la réduction de la consommation globale au sein des datacenters et le respect de la qualité de service aux utilisateurs. Ce résultat qui à l'échelle de notre simulation se situe entre 5 et 10 Kwh d'économie d'énergie pourrait représenter un gain substantiel lorsque l'on considère des datacenters de très grande taille ainsi qu'une durée plus longue. Ceci révèle enfin la nécessité d'approfondir cette recherche à travers des outils de prédiction plus sophistiqués et encore plus adaptés aux contextes de la fourniture de service dans des clouds datacenters de grande taille.

Conclusion

La problématique de l'efficacité énergétique des datacenters est un champ de recherche qui mérite une attention particulière étant donnée les enjeux liés aux deux thèmes qui la composent. D'abord les datacenters connaissent un niveau de prolifération record dans le monde avec toujours plus de services pour des utilisateurs dont le nombre est en constante ascendance depuis plusieurs années. Comme corollaire, les capacités de traitement des serveurs ainsi que les volumes de stockages augmentent continuellement en mettant en lumière la question préoccupante des quantités énormes d'énergie électrique nécessaire pour les alimenter de manière satisfaisante. Ensuite cette énergie étant une ressource critique pour l'informatique, il est opportun d'inventer les moyens d'optimisation des réserves actuelles de l'humanité sur lesquelles plusieurs études prospectives sont alarmantes quant à leur assèchement à moyen ou long terme. Dès lors, de nombreux travaux de recherche ont abordé la thématique et permis d'élaborer plusieurs techniques que nous nous sommes permis d'inventorier et de classer en deux groupes à savoir : les techniques réactives pour qualifier toutes celles qui consistent à surveiller les taux d'utilisations des ressources des serveurs et à prendre une action au cas lorsque ces derniers sortent de l'intervalle d'efficacité énergétique et les méthodes proactives qui surveillent les tendances de consommation électrique des serveurs et qui sont capable d'anticiper les éventuelles débordements et prendre une action avant qu'elles ne surviennent. Dans ces deux cas, outre les enjeux liés à la facture électrique, le respect des "Service Level Agreements (SLA)" sont également pris en charge. Ces techniques débouchent généralement sur une décision de migration des serveurs virtuels afin de leur garantir un confort de disponibilité des ressources et/ou une extinction des serveurs en sous-consommation afin de limiter le gaspillage d'énergie

qu'ils engendrent dans le Datacenter. L'objectif principal de ces actions est de réussir à sauvegarder le moindre watt d'énergie fourni aux serveurs et qui ne sert pas réellement au traitement d'une tâche donnée, ce qui pourrait à l'échelle des datacenters de grande taille constituer une diminution conséquente de leur facture énergétique.

Dans cette thèse nous avons introduit la notion de prédiction dans les techniques proactives d'optimisation des ressources des serveurs. Dans un premier temps nous avons évalué la prédictibilité des consommations de ressources en se basant sur les méthodes statistiques et les méthodes basées sur l'apprentissage par ordinateur. Avant d'aborder cette partie, nous avons fait le tour des données pertinentes pour notre étude et avons découvert qu'en raison de diverses contraintes dont la confidentialité, il n'existe que très peu de traces liées à la consommation des ressources des datacenters qui sont laissées dans le domaine public. Plusieurs projets de recherche pour contourner la difficulté se contenteront de générer des données synthétiques sur la base de formalisme mathématique adapté à leur contexte d'étude. Pour notre part, nous avons adopté l'une des rares traces pertinentes pour ce type d'étude, qui ont été rendus publiques par Google et qui concernent les statistiques de consommation de ressources de l'un de ces datacenters. Ensuite, à l'issue de notre survey sur la question, notre choix s'est porté sur les méthodes ARMA pour lesquelles, après une clarification conceptuelle, nous avons procédé à l'évaluation des performances sur les données de traces de Google. Il en ressort que les méthodes ARMA peuvent prédire les consommations de ressources à court terme avec un niveau de précision de plus 30% supérieur aux méthodes naïves. Egalement, nous avons procédé à une évaluation des méthodes d'apprentissage, notamment les réseaux de neurones sur lesquels, nous avons testé différentes configurations afin de détecter dans quelle mesure la variation du nombre de couches et du nombre d'entrée pourrait impacter les performances de la prédiction. Dans ce cadre, les réseaux de neurones à 4, 8 et 12 entrée consistant à fournir 4h, 8h ou 12h de données aux réseaux et à prédire respectivement la cinquième, la neuvième et la treizième heure ont été évalués avec à chaque fois une configuration avec une ou deux couches intermédiaires. Les résultats ici révèlent qu'avec 8 entrées, les

prédictions sont les plus fiables avec les taux de "residuals" les plus bas comparé aux autres configurations. Ces résultats ne varient que très peu lorsque l'on rajoute plus d'une couche intermédiaire. Nous avons donc conclu que dans notre cas les réseaux de neurones ont apporté un niveau de fiabilité satisfaisant sur les données issues des datacenters mais qu'il était important d'adapter l'architecture du réseau au type de données et au contexte de l'étude. Dans cette partie nous avons enfin établi en comparant les résultats obtenus sur les deux méthodes que les données relatives à la consommation de ressources dans les serveurs peuvent être prédites de manière relativement fiable. Cependant, les réseaux de neurones produisent de taux d'erreurs très inférieurs à ceux des méthodes ARMA et qu'à l'opposé, leurs délais de traitement peuvent être 3 fois supérieurs et non adaptés au contexte de prise de décision rapide imposée par le cloud computing. Pour la suite nous avons donc adopté la méthode ARMA comme la plus efficace pour notre étude.

La dernière partie de notre projet de recherche est consacrée l'utilisation de la prédiction pour la prise d'action permettant d'optimiser la consommation globale au sein du Data-center. Le principal challenge à ce niveau est d'une part d'opérer un placement initial des machines virtuelles sur les serveurs physiques qui respecte à la fois les contraintes d'optimisation de ressources et celles de la compatibilité entre les types de serveurs et d'autres part à implémenter la prédiction afin d'ajuster continuellement ce placement en procédant à la migration dynamique des machines virtuelles tout en respectant les enjeux d'optimisation de ressources et de SLA. En ce qui concerne le placement initial, nous avons évalué l'efficacité du Bin Packing qui est un outil mathématique très populaire dans le domaine de la logistique notamment pour le remplissage des conteneurs. En adaptant les algorithmes du First Fit Decreasing (FFD) au problème de placement de VM, il en ressort une efficacité de 25% comparée aux méthodes de Brute Force et aux méthodes manuelles. Le temps de traitement est presque nul tandis qu'il peut être de plusieurs secondes pour les autres méthodes. En considérant que nous sommes resté sur un échantillon très réduit pour cette étude, ces améliorations pourraient représenter un gain substantiel lorsque l'on se projette dans les datacenters de grande taille. Ensuite, nous nous sommes appuyé sur le simulateur CloudSim pour intégrer notre approche de prédiction dans le processus de migration des machines virtuelles. En effet, la librairie

org.cloudbus.cloudsim.allocationpolicies.power comprend un ensemble de module permettant de simuler les migrations suivant différents critères. Nous intégrons dans la logique de ces modules, l'algorithme ARMA que nous comparons plus tard aux méthodes similaires notamment celle de "local Regression" et de "Local Regression Robust". Les résultats nous confirment que l'intégration de la prédiction ARMA peut améliorer de manière significative l'optimisation des ressources sur les serveurs. En l'occurrence, cette approche permet de réduire de 10Kwh la consommation journalière sur un échantillon de petite taille (800 serveurs) comparé aux deux méthodes ci-dessus indiquées. Cette diminution s'accompagne d'une relative stabilité du Datacenter qui se matérialise par la réduction de nombre de migration ainsi qu'une diminution du taux de violation des SLA de près de 1%.

A l'issue de ce projet de thèse, il nous paraît évident qu'une des clés de la réduction de la facture énergétique réside dans la prédiction en dépit de toutes les autres techniques qui continuent d'être continuellement approfondies par la communauté. Notre travail a consisté à ouvrir quelques pistes qu'il faudra poursuivre afin d'apporter des réponses à d'autres problématiques tels que la classification intelligente des VM par profil de consommation de ressources afin de leur appliquer des règles conséquentes et la détermination des saisonnalités au sein des données en vue de l'établissement de plans de migration lointaine fiables. La résolution de ces questions nécessite en outre la disponibilité de plus de données de traces pertinentes et détaillées afin d'améliorer la compréhension du fonctionnement des datacenters dans le contexte très complexe de fourniture des services de cloud computing.

Bibliographie

Energie consommée par les data centers :. <https://www.planetoscope.com/electronique/230-energie-consommee-par-les-data-centers.html>. Accessed : 2017-10-20.

Hirotsugu Akaike. Akaike's information criterion. In *International Encyclopedia of Statistical Science*, pages 25–25. Springer, 2011.

We are social. We are social 2017. <https://wearesocial.com/special-reports/digital-in-2017-global-overview>, 2016. Accessed : 2017-10-20.

Luiz André Barroso and Urs Hölzle. The case for energy-proportional computing. *Computer*, 40(12), 2007.

Anton Beloglazov and Rajkumar Buyya. Adaptive threshold-based approach for energy-efficient consolidation of virtual machines in cloud data centers. In *MGC@ Middleware*, page 4, 2010a.

Anton Beloglazov and Rajkumar Buyya. Energy efficient resource management in virtualized cloud data centers. In *Proceedings of the 2010 10th IEEE/ACM international conference on cluster, cloud and grid computing*, pages 826–831. IEEE Computer Society, 2010b.

Anton Beloglazov, Rajkumar Buyya, Young Choon Lee, Albert Zomaya, et al. A taxonomy and survey of energy-efficient data centers and cloud computing systems. *Advances in computers*, 82(2) :47–111, 2011.

Josep Ll Berral, Íñigo Goiri, Ramón Nou, Ferran Julià, Jordi Guitart, Ricard Gavaldà, and

- Jordi Torres. Towards energy-aware scheduling in data centers using machine learning. In *Proceedings of the 1st International Conference on energy-Efficient Computing and Networking*, pages 215–224. ACM, 2010.
- George Box. Box and jenkins : Time series analysis forecasting and control. *A Very British Affair : Six Britons and the Development of Time Series Analysis during the 20th Century*, page 161, 2012.
- Rajkumar Buyya, Anton Beloglazov, and Jemal Abawajy. Energy-efficient management of data center resources for cloud computing : a vision, architectural elements, and open challenges. *arXiv preprint arXiv :1006.0308*, 2010.
- Rodrigo N Calheiros, Rajiv Ranjan, Anton Beloglazov, César AF De Rose, and Rajkumar Buyya. Cloudsim : a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Software : Practice and experience*, 41(1) :23–50, 2011.
- Brent Chun, David Culler, Timothy Roscoe, Andy Bavier, Larry Peterson, Mike Wawrzoniak, and Mic Bowman. Planetlab : an overlay testbed for broad-coverage services. *ACM SIGCOMM Computer Communication Review*, 33(3) :3–12, 2003.
- Howard Demuth and Mark Beale. Neural network toolbox for use with matlab. 1993.
- E. Renault F. Gbaguidi, S. Boumerdassi and E. Ezin. Characterizing servers workload in cloud datacenters. In *3rd IEEE International Conference on Future Internet of Things and Cloud (FiCloud), Roma, Italy*, pages 657–661. IEEE, 2015.
- S Boumerdassi F Gbaguidi. Taxonomy of energy efficiency methods for cloud datacenters. *Rapport de recherche 14-3129. Laboratoire CEDRIC, CNAM*, 2014.
- S. Boumerdassi F. Gbaguidi and E. Ezin. Predicting resource requirements of the servers into cloud datacenters using neural networks. In *15th International Conference on e-Society 2017 10 – 12 April, Budapest, Hungary*, pages 129–136. IEEE, 2017.
- Ronald Aylmer Fisher. *Statistical methods for research workers*. Genesis Publishing Pvt Ltd, 1925.

- Anshul Gandhi, Mor Harchol-Balter, Rajarshi Das, and Charles Lefurgy. Optimal power allocation in server farms. In *ACM SIGMETRICS Performance Evaluation Review*, volume 37, pages 157–168. ACM, 2009.
- Anshul Gandhi, Yuan Chen, Daniel Gmach, Martin Arlitt, and Manish Marwah. Minimizing data center sla violations and power consumption via hybrid resource provisioning. In *Green Computing Conference and Workshops (IGCC), 2011 International*, pages 1–8. IEEE, 2011.
- Sriram Govindan, Jeonghwan Choi, Bhuvan Urgaonkar, Anand Sivasubramaniam, and Andrea Baldini. Statistical profiling-based techniques for effective power provisioning in data centers. In *Proceedings of the 4th ACM European conference on Computer systems*, pages 317–330. ACM, 2009.
- Tarun Goyal, Ajit Singh, and Aakanksha Agrawal. Cloudsim : simulator for cloud computing infrastructure and modeling. *Procedia Engineering*, 38 :3566–3572, 2012.
- Simon S Haykin, Simon S Haykin, Simon S Haykin, and Simon S Haykin. *Neural networks and learning machines*, volume 3. Pearson Upper Saddle River, NJ, USA :, 2009.
- Michael Hibon and Spyros Makridakis. Arma models and the box-jenkins methodology. 1997.
- John C Hoff. *A practical guide to Box-Jenkins forecasting*. Lifetime Learning Publications, 1983.
- Rob J Hyndman and Anne B Koehler. Another look at measures of forecast accuracy. *International journal of forecasting*, 22(4) :679–688, 2006.
- Irwin L Kellner. Turn down the heat. *Journal of Business Strategy*, 16(6) :22–23, 1995.
- Dzmitry Kliazovich, Pascal Bouvry, and Samee Ullah Khan. Greencloud : a packet-level simulator of energy-aware cloud computing data centers. *The Journal of Supercomputing*, 62(3) :1263–1283, 2012.

- Richard E Korf. A new algorithm for optimal bin packing. In *AAAI/IAAI*, pages 731–736, 2002.
- Dilip Kumar and Bibhudatta Sahoo. Energy efficient heuristic resource allocation for cloud computing. 2014.
- Sanjay Kumar, Vanish Talwar, Vibhore Kumar, Parthasarathy Ranganathan, and Karsten Schwan. vmanage : loosely coupled platform and virtualization management in data centers. In *Proceedings of the 6th international conference on Autonomic computing*, pages 127–136. ACM, 2009.
- Dara Kusic, Jeffrey O Kephart, James E Hanson, Nagarajan Kandasamy, and Guofei Jiang. Power and performance management of virtualized computing environments via lookahead control. *Cluster computing*, 12(1) :1–15, 2009.
- Liang Liu, Hao Wang, Xue Liu, Xing Jin, Wen Bo He, Qing Bo Wang, and Ying Chen. Greencloud : a new architecture for green data center. In *Proceedings of the 6th international conference industry session on Autonomic computing and communications industry session*, pages 29–38. ACM, 2009.
- Zitao Liu and Sangyeun Cho. Characterizing machines and workloads on a google cluster. In *Parallel Processing Workshops (ICPPW), 2012 41st International Conference on*, pages 397–403. IEEE, 2012.
- Andrea Lodi, Silvano Martello, and Michele Monaci. Two-dimensional packing problems : A survey. *European journal of operational research*, 141(2) :241–252, 2002.
- EG Co man Jr, MR Garey, and DS Johnson. Approximation algorithms for bin packing : A survey. *Approximation Algorithms for NP-Hard Problems*, pages 46–93, 1996.
- Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4) :115–133, 1943.
- David Meisner, Brian T Gold, and Thomas F Wenisch. Powernap : eliminating server idle power. In *ACM Sigplan Notices*, volume 44, pages 205–216. ACM, 2009.

- OMC. Rapport annuel 2017 de l'omc. https://www.wto.org/french/res_f/publications_f/anrep17_f.htm, 2017. Accessed : 2017-10-20.
- Vinicius Petrucci, Orlando Loques, and Daniel Mossé. Dynamic optimization of power and performance for virtualized server clusters. In *Proceedings of the 2010 ACM Symposium on Applied Computing*, pages 263–264. ACM, 2010.
- Guillaume Plouin. *Cloud Computing-2e éd. : Une rupture décisive pour l'informatique d'entreprise*. Dunod, 2011.
- Charles Reiss, John Wilkes, and Joseph L Hellerstein. Google cluster-usage traces : format+ schema. *Google Inc., White Paper*, 2011.
- PM Robinson. The estimation of a nonlinear moving average model. *Stochastic Processes and their Applications*, 5(1) :81–90, 1977.
- Thiago Teixeira Sá, Rodrigo N Calheiros, and Danielo G Gomes. Cloudreports : An extensible simulation tool for energy-aware cloud computing environments. In *Cloud Computing*, pages 127–142. Springer, 2014.
- George Udny Yule. On a method of investigating periodicities in disturbed series, with special reference to wolfer's sunspot numbers. *Philosophical Transactions of the Royal Society of London Series A*, 226 :267–298, 1927.
- Akshat Verma, Puneet Ahuja, and Anindya Neogi. pmapper : power and migration cost aware application placement in virtualized systems. In *Proceedings of the 9th ACM/IFIP/USENIX International Conference on Middleware*, pages 243–264. Springer-Verlag New York, Inc., 2008.
- Akshat Verma, Gargi Dasgupta, Tapan Kumar Nayak, Pradipta De, and Ravi Kothari. Server workload analysis for power minimization using consolidation. In *Proceedings of the 2009 conference on USENIX Annual technical conference*, pages 28–28. USENIX Association, 2009.
- David L Weakliem. A critique of the bayesian information criterion for model selection. *Sociological Methods & Research*, 27(3) :359–397, 1999.

BIBLIOGRAPHIE

- Bhathiya Wickremasinghe, Rodrigo N Calheiros, and Rajkumar Buyya. Cloudanalyst : A cloudsims-based visual modeller for analysing cloud computing environments and applications. In *Advanced Information Networking and Applications (AINA), 2010 24th IEEE International Conference on*, pages 446–452. IEEE, 2010.
- John Wilkes. More google cluster data. *Google research blog*, Nov, 2011.

Annexes

Annexe A

Annexe 1 :State of art of traffic prediction methods

A.1 Introduction

With Web 2.0 and mobility technologies growth since last decade, network traffic significantly increased and then management tasks becomes a so hard. Networks administrators and designers need more efficient tools to dynamically and properly adjust traffic settings which must complies precise requirements. The demands are many : bandwidth allocation or planning, SLA maintaining, resources provisioning, suspicious behaviors detection, routing protocols enhancement, power efficiency optimization and so on. To achieve these goals, traffic pattern is an important research fields that helps to facilitate our understanding of both the regularity and the arrival packet size in a computers network. Many methods have been tested in the purpose to find the best way to forecast the data flow characteristics which can help to accurately take the best decision on management or planning networks resources. Network traffic data flow can show many patterns including self-similarity, regularity, trend, correlation at variable time scale, etc. and high dependencies between historical and future measurements. Otherwise, traffic analysis and prediction can also sound differently for engineering and management ends involving variables terms prediction, average or maximal packets size, specified type of service pattern. Forecasting techniques are widely used in many research domains such as Intelligent Transportation Systems, economy, political science, biology, etc. In the literature, prediction methods

are grouped by traditional and new model Zhou et al., 2011 or statistical and artificial intelligence Barimani et al., 2012 or single and combined by Han-Lin et al. 2009. After given an overview of what prediction mechanism stand for, we'll try to provide a novel classification which better clarify the field and is nearby network management issues. The remaining of this paper is as follow : In the first part, we'll present a generic forecasting scheme then we'll in part II present a synthetic classification of main methods which will be describing in part III. In part IV we'll talk about network data flow characteristics and we'll finally conclude the paper.

A.2 Generic prediction process scheme

The common architecture of prediction processing can be presented as follow : i) Training of model fitting step : After collecting data from real measurement of traffic or simulation output, the first step for most prediction process is the system learning step. The goal is the establishment of the prediction model for statistical methods or creating the best algorithm for self-prediction of futures values. That consists of injecting historical data on the training system and compares the output with the real's ones. The training step could be preceded by a pre-processing step which consists on filtering the data series, clustering them, isolating the learning sample, separate regular, random or noisy components or adapting them on the learning system format. ii) Testing or validation step : Once the system has learned from the historical data, the validation step must confirm the system output on a given time scale. This step concludes on the system accuracy on the specific case. The output values could be optimized given on the target reliability or expected output format. iii) Real use step : After these operations, the trained system or established model can be embedded on a framework for real utilization purpose.

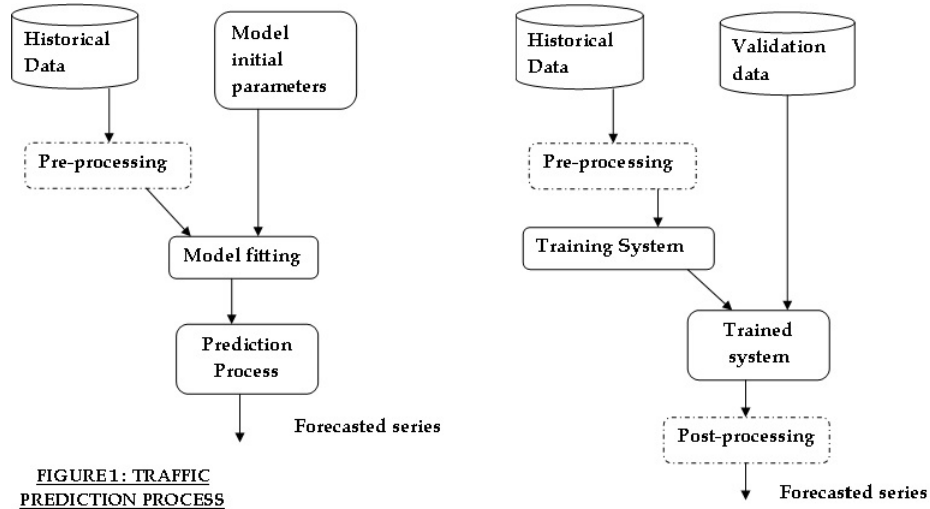


FIGURE A.1 – Prediction process scheme

A.3 Classification

Almost all of the forecasting models have context-based accuracy and must be considered as compliant for very specific cases. Forecasting can be real time or variable time scale (short, mid or long term). Methods reliability strongly depends on traffic data characteristics and prediction horizon. Han-Lin et al. 2009 assume that Network traffic prediction models can be grouped into two types, single models and combined ones since several methods are use in single mode or by combination with others. The classification below is based on the incoming mathematical field of the using methods. Then we classify then in statistical one group and artificial another group. The main idea is to give an inventory of main and most used prediction methods.

A.3.1 Statistics methods

A.3.1.1 ARMA, ARIMA, ARFIMA

ARMA models and its variants ARIMA and ARFIMA presented by Box and Jenkins [Box 2012] classical and so popular tools used in forecasting data series, at different times scales, based on historical values. With ARMA models, data are assumed to be stationary

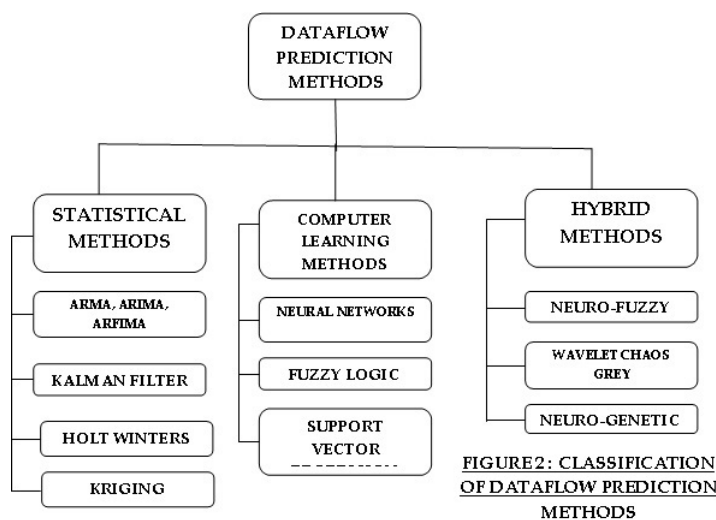


FIGURE A.2 – prediction methods classification

and univariate. In case of non-stationary, an integrated form of the models (ARIMA) consists on inserting a parameter d which represents the occurrence of variability into the series. In ARIMA models, the parameter d can take integer values. When data series shows seasonality, the FARIMA model with parameter d , set to discrete value, is used to isolate the seasonal and non seasonal parts. The ARMA models and its variants can be expressed on these equations :

Auto Regression part : Moving Average part : Where L is a Lag operator, d is the order of variability p and q are respectively AR and MA parameters. ϵ_t is an error factor which is assumed to be independent, identically distributed sampled on a normal distribution. These Auto Regressive models are widely used in prediction process and provides reliable results. Cortez et al 2012 after comparison of several prediction methods concludes that ARIMA and Neural Network method performs accurate results for real time and short terms prediction. However, ARIMA require high level resources. When considering fractional components (ARFIMA model) Zhou and al. 2011, the model also show lowest value of errors. Another autoregressive models is named Auto Regressive Conditional Heteroscedasticity model (ARCH) can be applied to forecast data for non linear series.

A.3.1.2 Holt winter

Holt Winter Makridakis et al., 1998 method is use for series which show some trend and seasonality patterns such as monthly, daily or hourly seasonality. The models is an enhancement of exponential smoothing method which representing the series as the weighted sum of its pasts values. The holt-winters method performs an exponential smoothing of three components shown in the series. They are defining as following equations Which is combination of the three components? the level or mean estimate , the trend estimate and , the seasonal estimate

The holt-winters models are generally applied to kind of traffic that show severe variation or presented some seasonality through time. Satisfied results are found by Cortez et al. 2012 by comparing Holt-Winters methods to ARIMA and Neural Networks with small data set (daily traffic of Internet Service Providers). Exponential Smoothing methods are also found by Iqbal and john, 2012 to shown low cost and low energy consumption when comparing to several method including ARIMA and ANN.

A.3.1.3 Kalman Filter

The Kalman Filter (KF) method, Welch, and Bishop, 1995 consists on using historical data which contain random variation and inaccuracies to predict data that tend to be closer to the true values. KF is preferred when applied to short-term prediction of traffic volume. The general form of Kalman Filter function is : $Z(k+1) = H(k) X(k) + v(k)$ where, $Z(k+1)$ is the predicted function, $H(k)$ the pasts values, $X(k)$ the combination coefficient and $v(k)$ a Gaussian white noise. Dorgbefu et al. 2013, using Kalman Filter model, performs an optimization on accuracy of UMTS network traffic short-term prediction. The validation of the model with r^2 concluded at 87 to 99% of correlation coefficient. KF has also been use on other research fields with good reliability.

A.3.1.4 Network Krigin

This statistical prediction method has been implemented in Chua et al, 2005. It's consisting on predicting traffic in large network links based on measurement on just few sample of traffic collecting on selected links. The linear algebra tools are used since network is designed as a graph of n vertices and m edges. The path selection algorithm is inspired from Golub and van Loan, 1989. The method experimentation results on very low value of Mean Relative Error on Network Average Path Delay estimation and when models parameters are well chosen, spike prediction can also be accurately predicted.

A.3.2 Computer learning methods

A.3.2.1 Neural network

Neural network (NN) is "...a computing system made up of a number of simple, highly interconnected processing elements, which process information by their dynamic state response to external inputs" Caudill, 1989. It can be seen as set of layered neurons connected by weighted connections through which, learning processing is performed. Learning processing consist on establishing a model by calculating the weighted sum of real data coming from the studying system by a chosen function called activation function. Once the system has achieved a good comprehension of the process, the next operations can be performing autonomously. The mains challenge with NN is the choice conditions of weight and activation function which are key parameters for the model. Neural networks rely on several principles, including gradient-based training, fuzzy logic, genetic algorithms, and Bayesian methods. Many architectures are use in prediction field. The more popular one is the Multi Layer Perceptron (MLP) with Back propagation (BP) algorithm. Others derived models are RRBFN (Recurrent radial Function Network) and ESN (echo state network) and Flexible neural tree In many Neural Network architectures, the processing times or resources needs could be real's limitations. For non-linear series prediction purpose, NN can provide accurate results. Despite the theoretical accuracy of Neural Networks models, they can cost very expensive in time since they are supervised process. In network traffic prediction, neural networks methods have been seen to provide successful results.

Gowrishankar, 2008 conclude an accuracy of 96.4% to 98.3% by combining RRBFN (Recurrent radial Function Network) and ESN (echo state network) while FARIMA was 78.5% to 80.2% accurate. Chen et al. lead to similar results on Flexible Neural Tree used to perform small term prediction on network traffic of Data Equipment Corporation. The tree parameters were optimized using Genetic Algorithm (GA) and the Particle Swarm Optimization (PSO) algorithm to improve the reliability of the tool. Akritas et al. 2001 experienced accuracy optimization of network traffic prediction using Artificial Neural Network and non-linear data from Dubna University LAN. Output leded shows that NN methods are also applicable on non-linear data with satisfying results.

A.3.2.2 Support Virtual Regression (SVR)

Support Vector Machines (SVM) Vapnik et al. 1997, Basak et al 2007, are learning machines implementing the structural risk minimization inductive principle to obtain good generalization on a limited number of learning patterns. Statistical Learning Theory has provided a very effective framework for classification and regression tasks involving features. SVM can be divided in two categories : Support Vector Classification (SVC) and Support Vector Regression (SVR). One of the main characteristics of Support Vector Regression (SVR) is that instead of minimizing the observed training error, SVR attempts to minimize the generalized error bound so as to achieve generalized performance. In the traffic prediction field, SVR have been use to apply nonlinear analysis method to network measurement data Qing-Fang et al 2009. The Adaptive Support Vector Regression (ASVR) has been combined with BIC-based neighboring point selection method for choosing the number of the nearest point of the local SVM model. Test on Berkeley Laboratory TCP data traffic (non-linear), result that the traffic distribution obtaining is very close to the real traffic and the errors distribution near to 0.

A.3.3 Hybrid methods

A.3.3.1 Adaptive Neuro-Fuzzy Inference System (ANFIS) (Jang, 1993)

A.3.3.2 WAVELET-CHAOS-GREY Model (WCG)

In Han-Lin et al. 2009, a novel prediction models was performed using first the decomposition feature of Wavelet à trous algorithm Dutilleux, 1989 to separate the data series into residual and burst part. Then, they apply the Grey model (GM) to predict the residual part and Chaos Model to predict the burst one. When comparing to others models such as the standalone GM(1,1), Chaos model, ARIMA and Wavelet ARIMA, it's shown that the WCG has a more accurate prediction in 5 minutes average traffic data. It's also seen that the combined methods accuracy closely depends on that of single methods used. Finally, network traffic prediction methods presented several features such as :

- Terms of prediction : whereas the computing load and storage requirement of traffic prediction, real time prediction is so challenged goal. Instead of that, short term prediction can greatly helps network administrators in performing of their management task. Prediction on clustering dataset at 5 minutes terms is for example sufficient information for many actions. Mean and long terms predictions (e.g. daily, weekly or monthly) could also be helpful for planning task or architecture improvement. Methods presented above could performs regarding to the clustering level of dataset. FNT (Short-terms)
- Accuracy : all of the methods are context-based accurate. Depending on the studying data namely on their spikes features, artificial methods shown good accuracy of prediction since they deal with both linear and non linear data series and also seasonal characteristics.
- Lightweight : unfortunately, despite their good accuracy artificial methods like Neural Networks are too heavy and complex to perform. Statistical methods could easily embedded cause of the simplicity of their models. However, they are better for just a few case of dataset and could experienced inaccuracy in some scenario.

A.4 Type of data and measurement

Network Data type is a key feature in traffic prediction field. The structure of a given dataset is very important on the choice of the prediction methods, the pre-processing process and the result interpretation. Many tools are available for network data measurement included SNMP protocol, CISCO NETFLOW, TCPDUMP or others sniffers, or SYSLOG. Many data trace libraries are also available on the research laboratories web sites and can be freely re-use for experimental purposes. The main characteristics extracted from network's traffic dataset are bytes count, packet count, packet size, packet source or destination address, packet source or destination port, transport layer protocol, application layer protocol or type of service, errors flags, ToS flags and traffic contents. Also clustered data can be computed by most of those tools such as total or per protocol packet count, total or per protocol bandwidth, errors rate, collisions count, broadcast, unicast or multicast packets count etc. Data collecting processes are resource consuming then in many case, real time traffic collection is recommended so it could considerably increase network traffic and create storage issues. Most of traffic measurement protocols are configured to perform collection task at precise period range from few seconds to a few minutes. It's also notice that clustered data are preferred instead of detail. Network traffic is usually presented in graphical form to better capture the mains characteristics and facilitate the understanding. When performing prediction task, the compared graphics were represented in purpose to visually show the correlation coefficient and other statistical measure as prediction errors.

A.5 Conclusion

Towards this study we attempts to exhibit how often mathematical methods can improve our knowledge of whether network traffic is delivery. This state of art is starting point for digging that field using novel methods or combining existing in the goal to perform a prediction framework which could respect the many rules that henceforth guide system implementations. The more accurate will be those methods, the more efficient will

be today's network management regarding to virtual environments, service availability, intelligent bandwidth management and green computing respect. We also seen that the prediction outcome closely depends on the type, temporality and quantity of data use to perform the prediction. Since network traffic collection is strongly resource consuming task, it will be more useful to resort to trace data for prediction system learning. Another important feature is the complexity of the mathematical algorithm using for prediction operation. Lightweight and portability are key requirement for all applications since they must be multi-platforms and embeddable on removable terminals.

Annexe B

Annexe 2 :Environnement Neurals Networks dans MATLAB

B.1 Présentation de l'environnement

MATrix LABoratory, MATLAB est un outil de calcul scientifique utilisé depuis la fin des années 70 dans de nombreux domaines de recherche et commercialisé par la société MathWorks. Il propose 3 fonctionnalités principales que sont le calcul numérique, l'analyse de données et la visualisation en utilisant une gamme très variée d'outils mathématique des statistiques à l'aérospatial en passant par la finance, la bioinformatique, le traitement des signaux, l'intelligence artificielle pour ne citer que ceux la. Sur l'espace de travail

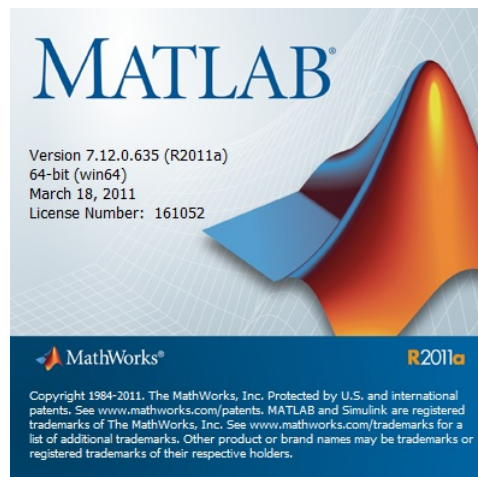


FIGURE B.1 – Initialisation de Matlab

B.2 Fonctions principales

B.3

Annexe C

Annexe 3 : Algorithme de prédiction ARMA

C.1 Code traitement ARMA dans MATLAB

```
clear all
clc
load Frejus
D=B1;
N=length(D);
M=100;
%Naive
Naive= std (D(M+1:N)-D(M:N-1))/std (D(M:N));

O=[1 0 Naive];
for nor=14 %ordre ARMA
    for i=M:N
        N-i
        data = iddata(D(i-M+1:i),[],1);
        orders = [nor nor];
        m = armax(data,orders);
        yp = predict(m,data,1)
        ypre(i-(M-1))=yp.OutputData(M);
    end
    r=D(M:N)-ypre(1:N-(M-1))';
    error=std(r)/std(D(M:N));
    O=[O;[orders error]];
end
O
figure(1);ww=xcorr(r,50);stem(ww);hold
ww=xcorr(D(M+1:N)-D(M:N-1),50);stem(ww,'r');hold
```

```
figure (10); hist (r ,100); figure (11); hist ((D(M+1:N)-D(M:N-1)) ,100);  
figure (2); plot (D(M:N)); hold; plot (ypre (1:N-(M-1)) ', 'r '); hold
```

Annexe D

Annexe 4 :Fichiers de politiques d'allocation de VM dans Cloudsim

D.1 PowerVmAllocationPolicyMigrationAbstract

```
/*
 * Title:          CloudSim Toolkit
 * Description:    CloudSim (Cloud Simulation) Toolkit for Modeling and Simulation o
 * Licence:       GPL - http://www.gnu.org/copyleft/gpl.html
 *
 * Copyright (c) 2009-2012, The University of Melbourne, Australia
 */

package org.cloudbus.cloudsim.power;

import java.util.ArrayList;
import java.util.HashMap;
import java.util.HashSet;
import java.util.LinkedList;
import java.util.List;
import java.util.Map;
import java.util.Set;
```

```
import org.cloudbus.cloudsim.Host;
import org.cloudbus.cloudsim.HostDynamicWorkload;
import org.cloudbus.cloudsim.Log;
import org.cloudbus.cloudsim.Vm;
import org.cloudbus.cloudsim.core.CloudSim;
import org.cloudbus.cloudsim.power.lists.PowerVmList;
import org.cloudbus.cloudsim.util.ExecutionTimeMeasurer;

/**
 * The class of an abstract power-aware VM allocation policy that dynamically optimizes
 * allocation using migration.
 *
 * If you are using any algorithms, policies or workload included in the power package,
 * the following paper:
 *
 * Anton Beloglazov, and Rajkumar Buyya, "Optimal Online Deterministic Algorithms and
 * Heuristics for Energy and Performance Efficient Dynamic Consolidation of Virtual M
 * Cloud Data Centers", Concurrency and Computation: Practice and Experience (CCPE), V
 * Issue 13, Pages: 1397-1420, John Wiley & Sons, Ltd, New York, USA, 2012
 *
 * @author Anton Beloglazov
 * @since CloudSim Toolkit 3.0
 */
public abstract class PowerVmAllocationPolicyMigrationAbstract extends PowerVmAllocationPolicyAbstract {

    /** The vm selection policy. */
    private PowerVmSelectionPolicy vmSelectionPolicy;

    /** The saved allocation. */
```

```
private final List<Map<String, Object>> savedAllocation = new ArrayList<Map<String, Object>>();

/** The utilization history. */
private final Map<Integer, List<Double>> utilizationHistory = new HashMap<Integer, List<Double>>();

/** The metric history. */
private final Map<Integer, List<Double>> metricHistory = new HashMap<Integer, List<Double>>();

/** The time history. */
private final Map<Integer, List<Double>> timeHistory = new HashMap<Integer, List<Double>>();

/** The execution time history vm selection. */
private final List<Double> executionTimeHistoryVmSelection = new LinkedList<Double>();

/** The execution time history host selection. */
private final List<Double> executionTimeHistoryHostSelection = new LinkedList<Double>();

/** The execution time history vm reallocation. */
private final List<Double> executionTimeHistoryVmReallocation = new LinkedList<Double>();

/** The execution time history total. */
private final List<Double> executionTimeHistoryTotal = new LinkedList<Double>();

/**
 * Instantiates a new power vm allocation policy migration abstract.
 *
 * @param hostList the host list
 * @param vmSelectionPolicy the vm selection policy
 */
public PowerVmAllocationPolicyMigrationAbstract(
```

```
List<? extends Host> hostList,
PowerVmSelectionPolicy vmSelectionPolicy) {
    super(hostList);
    setVmSelectionPolicy(vmSelectionPolicy);
}

/**
 * Optimize allocation of the VMs according to current utilization.
 *
 * @param vmList the vm list
 *
 * @return the array list< hash map< string, object>>
 */
@Override
public List<Map<String, Object>> optimizeAllocation(List<? extends Vm> vmList) {
    ExecutionTimeMeasurer.start("optimizeAllocationTotal");

    ExecutionTimeMeasurer.start("optimizeAllocationHostSelection");
    List<PowerHostUtilizationHistory> overUtilizedHosts = getOverUtilizedHosts();
    getExecutionTimeHistoryHostSelection().add(
        ExecutionTimeMeasurer.end("optimizeAllocationHostSelection"));

    printOverUtilizedHosts(overUtilizedHosts);

    saveAllocation();

    ExecutionTimeMeasurer.start("optimizeAllocationVmSelection");
    List<? extends Vm> vmsToMigrate = getVmsToMigrateFromHosts(overUtilizedHosts);
    getExecutionTimeHistoryVmSelection().add(ExecutionTimeMeasurer.end("optimizeAllocationVmSelection"));
```

```

Log.println("Reallocation of VMs from the over-utilized hosts:");
ExecutionTimeMeasurer.start("optimizeAllocationVmReallocation");
List<Map<String, Object>> migrationMap = getNewVmPlacement(vmsToMigrate, new HashS
overUtilizedHosts));
getExecutionTimeHistoryVmReallocation().add(
ExecutionTimeMeasurer.end("optimizeAllocationVmReallocation"));
Log.println();

migrationMap.addAll(getMigrationMapFromUnderUtilizedHosts(overUtilizedHosts));

restoreAllocation();

getExecutionTimeHistoryTotal().add(ExecutionTimeMeasurer.end("optimizeAllocationTo

return migrationMap;
}

/**
 * Gets the migration map from under utilized hosts.
 *
 * @param overUtilizedHosts the over utilized hosts
 * @return the migration map from under utilized hosts
 */
protected List<Map<String, Object>> getMigrationMapFromUnderUtilizedHosts(
List<PowerHostUtilizationHistory> overUtilizedHosts) {
List<Map<String, Object>> migrationMap = new LinkedList<Map<String, Object>>();
List<PowerHost> switchedOffHosts = getSwitchedOffHosts();

// over-utilized hosts + hosts that are selected to migrate VMs to from over-utili
Set<PowerHost> excludedHostsForFindingUnderUtilizedHost = new HashSet<PowerHost>()

```

```
excludedHostsForFindingUnderUtilizedHost.addAll(overUtilizedHosts);
excludedHostsForFindingUnderUtilizedHost.addAll(switchedOffHosts);
excludedHostsForFindingUnderUtilizedHost.addAll(extractHostListFromMigrationMap(migra

// over-utilized + under-utilized hosts
Set<PowerHost> excludedHostsForFindingNewVmPlacement = new HashSet<PowerHost>();
excludedHostsForFindingNewVmPlacement.addAll(overUtilizedHosts);
excludedHostsForFindingNewVmPlacement.addAll(switchedOffHosts);

int numberOfHosts = getHostList().size();

while (true) {
    if (numberOfHosts == excludedHostsForFindingUnderUtilizedHost.size()) {
        break;
    }

    PowerHost underUtilizedHost = getUnderUtilizedHost(excludedHostsForFindingUnderUtiliz
    if (underUtilizedHost == null) {
        break;
    }

    Log.println("Under-utilized host: host #" + underUtilizedHost.getId() + "\n");

    excludedHostsForFindingUnderUtilizedHost.add(underUtilizedHost);
    excludedHostsForFindingNewVmPlacement.add(underUtilizedHost);

    List<? extends Vm> vmsToMigrateFromUnderUtilizedHost = getVmsToMigrateFromUnderUtiliz
    if (vmsToMigrateFromUnderUtilizedHost.isEmpty()) {
        continue;
    }
}
```

```
Log.print("Reallocation of VMs from the under-utilized host: ");
if (!Log.isDisabled()) {
    for (Vm vm : vmsToMigrateFromUnderUtilizedHost) {
        Log.print(vm.getId() + " ");
    }
}
Log.println();

List<Map<String, Object>> newVmPlacement = getNewVmPlacementFromUnderUtilizedHost(
    vmsToMigrateFromUnderUtilizedHost,
    excludedHostsForFindingNewVmPlacement);

excludedHostsForFindingUnderUtilizedHost.addAll(extractHostListFromMigrationMap(newVmPlacement));

migrationMap.addAll(newVmPlacement);
Log.println();
}

return migrationMap;
}

/**
 * Prints the over utilized hosts.
 *
 * @param overUtilizedHosts the over utilized hosts
 */
protected void printOverUtilizedHosts(List<PowerHostUtilizationHistory> overUtilizedHosts) {
    if (!Log.isDisabled()) {
        Log.println("Over-utilized hosts:");
    }
}
```

```
for (PowerHostUtilizationHistory host : overUtilizedHosts) {
Log.println("Host #" + host.getId());
}
Log.println();
}
}

/**
 * Find host for vm.
 *
 * @param vm the vm
 * @param excludedHosts the excluded hosts
 * @return the power host
 */
public PowerHost findHostForVm(Vm vm, Set<? extends Host> excludedHosts) {
double minPower = Double.MAX_VALUE;
PowerHost allocatedHost = null;

for (PowerHost host : this.<PowerHost> getHostList()) {
if (excludedHosts.contains(host)) {
continue;
}
if (host.isSuitableForVm(vm)) {
if (getUtilizationOfCpuMips(host) != 0 && isHostOverUtilizedAfterAllocation(host, vm))
continue;
}

try {
double powerAfterAllocation = getPowerAfterAllocation(host, vm);
if (powerAfterAllocation != -1) {
```

```
double powerDiff = powerAfterAllocation - host.getPower();
if (powerDiff < minPower) {
    minPower = powerDiff;
    allocatedHost = host;
}
}
} catch (Exception e) {
}
}
}
return allocatedHost;
}

/**
 * Checks if is host over utilized after allocation.
 *
 * @param host the host
 * @param vm the vm
 * @return true, if is host over utilized after allocation
 */
protected boolean isHostOverUtilizedAfterAllocation(PowerHost host, Vm vm) {
    boolean isHostOverUtilizedAfterAllocation = true;
    if (host.vmCreate(vm)) {
        isHostOverUtilizedAfterAllocation = isHostOverUtilized(host);
        host.vmDestroy(vm);
    }
    return isHostOverUtilizedAfterAllocation;
}

/**
```

```
* Find host for vm.
*
* @param vm the vm
* @return the power host
*/
@Override
public PowerHost findHostForVm(Vm vm) {
    Set<Host> excludedHosts = new HashSet<Host>();
    if (vm.getHost() != null) {
        excludedHosts.add(vm.getHost());
    }
    return findHostForVm(vm, excludedHosts);
}

/**
 * Extract host list from migration map.
 *
 * @param migrationMap the migration map
 * @return the list
 */
protected List<PowerHost> extractHostListFromMigrationMap(List<Map<String, Object>> migrationMap) {
    List<PowerHost> hosts = new LinkedList<PowerHost>();
    for (Map<String, Object> map : migrationMap) {
        hosts.add((PowerHost) map.get("host"));
    }
    return hosts;
}

/**
 * Gets the new vm placement.
```

```
*
* @param vmsToMigrate the vms to migrate
* @param excludedHosts the excluded hosts
* @return the new vm placement
*/
protected List<Map<String, Object>> getNewVmPlacement(
List<? extends Vm> vmsToMigrate,
Set<? extends Host> excludedHosts) {
List<Map<String, Object>> migrationMap = new LinkedList<Map<String, Object>>();
PowerVmList.sortByCpuUtilization(vmsToMigrate);
for (Vm vm : vmsToMigrate) {
PowerHost allocatedHost = findHostForVm(vm, excludedHosts);
if (allocatedHost != null) {
allocatedHost.vmCreate(vm);
Log.println("VM #" + vm.getId() + " allocated to host #" + allocatedHost.getId())

Map<String, Object> migrate = new HashMap<String, Object>();
migrate.put("vm", vm);
migrate.put("host", allocatedHost);
migrationMap.add(migrate);
}
}
return migrationMap;
}

/**
* Gets the new vm placement from under utilized host.
*
* @param vmsToMigrate the vms to migrate
* @param excludedHosts the excluded hosts
```

```
* @return the new vm placement from under utilized host
*/
protected List<Map<String, Object>> getNewVmPlacementFromUnderUtilizedHost(
List<? extends Vm> vmsToMigrate,
Set<? extends Host> excludedHosts) {
List<Map<String, Object>> migrationMap = new LinkedList<Map<String, Object>>();
PowerVmList.sortByCpuUtilization(vmsToMigrate);
for (Vm vm : vmsToMigrate) {
PowerHost allocatedHost = findHostForVm(vm, excludedHosts);
if (allocatedHost != null) {
allocatedHost.vmCreate(vm);
Log.println("VM #" + vm.getId() + " allocated to host #" + allocatedHost.getId());

Map<String, Object> migrate = new HashMap<String, Object>();
migrate.put("vm", vm);
migrate.put("host", allocatedHost);
migrationMap.add(migrate);
} else {
Log.println("Not all VMs can be reallocated from the host, reallocation cancelled")
for (Map<String, Object> map : migrationMap) {
((Host) map.get("host")).vmDestroy((Vm) map.get("vm"));
}
migrationMap.clear();
break;
}
}
return migrationMap;
}

/**
```

```
* Gets the vms to migrate from hosts.
*
* @param overUtilizedHosts the over utilized hosts
* @return the vms to migrate from hosts
*/
protected
List<? extends Vm>
getVmsToMigrateFromHosts(List<PowerHostUtilizationHistory> overUtilizedHosts) {
    List<Vm> vmsToMigrate = new LinkedList<Vm>();
    for (PowerHostUtilizationHistory host : overUtilizedHosts) {
        while (true) {
            Vm vm = getVmSelectionPolicy().getVmToMigrate(host);
            if (vm == null) {
                break;
            }
            vmsToMigrate.add(vm);
            host.vmDestroy(vm);
            if (!isHostOverUtilized(host)) {
                break;
            }
        }
    }
    return vmsToMigrate;
}

/**
 * Gets the vms to migrate from under utilized host.
 *
 * @param host the host
 * @return the vms to migrate from under utilized host
 */
```

```
*/
protected List<? extends Vm> getVmsToMigrateFromUnderUtilizedHost(PowerHost host) {
    List<Vm> vmsToMigrate = new LinkedList<Vm>();
    for (Vm vm : host.getVmList()) {
        if (!vm.isInMigration()) {
            vmsToMigrate.add(vm);
        }
    }
    return vmsToMigrate;
}

/**
 * Gets the over utilized hosts.
 *
 * @return the over utilized hosts
 */
protected List<PowerHostUtilizationHistory> getOverUtilizedHosts() {
    List<PowerHostUtilizationHistory> overUtilizedHosts = new LinkedList<PowerHostUtilizationHistory>();
    for (PowerHostUtilizationHistory host : this.<PowerHostUtilizationHistory> getHostList()) {
        if (isHostOverUtilized(host)) {
            overUtilizedHosts.add(host);
        }
    }
    return overUtilizedHosts;
}

/**
 * Gets the switched off host.
 *
 * @return the switched off host
 */
```

```
*/
protected List<PowerHost> getSwitchedOffHosts() {
List<PowerHost> switchedOffHosts = new LinkedList<PowerHost>();
for (PowerHost host : this.<PowerHost> getHostList()) {
if (host.getUtilizationOfCpu() == 0) {
switchedOffHosts.add(host);
}
}
return switchedOffHosts;
}

/**
 * Gets the under utilized host.
 *
 * @param excludedHosts the excluded hosts
 * @return the under utilized host
 */
protected PowerHost getUnderUtilizedHost(Set<? extends Host> excludedHosts) {
double minUtilization = 1;
PowerHost underUtilizedHost = null;
for (PowerHost host : this.<PowerHost> getHostList()) {
if (excludedHosts.contains(host)) {
continue;
}
double utilization = host.getUtilizationOfCpu();
if (utilization > 0 && utilization < minUtilization
&& !areAllVmsMigratingOutOrAnyVmMigratingIn(host)) {
minUtilization = utilization;
underUtilizedHost = host;
}
}
```

```
}  
return underUtilizedHost;  
}  
  
/**  
 * Checks whether all vms are in migration.  
 *  
 * @param host the host  
 * @return true, if successful  
 */  
protected boolean areAllVmsMigratingOutOrAnyVmMigratingIn(PowerHost host) {  
    for (PowerVm vm : host.<PowerVm> getVmList()) {  
        if (!vm.isInMigration()) {  
            return false;  
        }  
        if (host.getVmsMigratingIn().contains(vm)) {  
            return true;  
        }  
    }  
    return true;  
}  
  
/**  
 * Checks if is host over utilized.  
 *  
 * @param host the host  
 * @return true, if is host over utilized  
 */  
protected abstract boolean isHostOverUtilized(PowerHost host);
```

```
/**
 * Adds the history value.
 *
 * @param host the host
 * @param metric the metric
 */
protected void addHistoryEntry(HostDynamicWorkload host, double metric) {
    int hostId = host.getId();
    if (!getTimeHistory().containsKey(hostId)) {
        getTimeHistory().put(hostId, new LinkedList<Double>());
    }
    if (!getUtilizationHistory().containsKey(hostId)) {
        getUtilizationHistory().put(hostId, new LinkedList<Double>());
    }
    if (!getMetricHistory().containsKey(hostId)) {
        getMetricHistory().put(hostId, new LinkedList<Double>());
    }
    if (!getTimeHistory().get(hostId).contains(CloudSim.clock())) {
        getTimeHistory().get(hostId).add(CloudSim.clock());
        getUtilizationHistory().get(hostId).add(host.getUtilizationOfCpu());
        getMetricHistory().get(hostId).add(metric);
    }
}

/**
 * Save allocation.
 */
protected void saveAllocation() {
    getSavedAllocation().clear();
    for (Host host : getHostList()) {
```

```
for (Vm vm : host.getVmList()) {
    if (host.getVmsMigratingIn().contains(vm)) {
        continue;
    }
    Map<String, Object> map = new HashMap<String, Object>();
    map.put("host", host);
    map.put("vm", vm);
    getSavedAllocation().add(map);
}
}
}

/**
 * Restore allocation.
 */
protected void restoreAllocation() {
    for (Host host : getHostList()) {
        host.vmDestroyAll();
        host.reallocateMigratingInVms();
    }
    for (Map<String, Object> map : getSavedAllocation()) {
        Vm vm = (Vm) map.get("vm");
        PowerHost host = (PowerHost) map.get("host");
        if (!host.vmCreate(vm)) {
            Log.println("Couldn't restore VM #" + vm.getId() + " on host #" + host.getId());
            System.exit(0);
        }
        getVmTable().put(vm.getUid(), host);
    }
}
```

```
/**
 * Gets the power after allocation.
 *
 * @param host the host
 * @param vm the vm
 *
 * @return the power after allocation
 */
protected double getPowerAfterAllocation(PowerHost host, Vm vm) {
    double power = 0;
    try {
        power = host.getPowerModel().getPower(getMaxUtilizationAfterAllocation(host, vm));
    } catch (Exception e) {
        e.printStackTrace();
        System.exit(0);
    }
    return power;
}

/**
 * Gets the power after allocation. We assume that load is balanced between PEs. T
 * restriction is: VM's max MIPS < PE's MIPS
 *
 * @param host the host
 * @param vm the vm
 *
 * @return the power after allocation
 */
protected double getMaxUtilizationAfterAllocation(PowerHost host, Vm vm) {
```

```

double requestedTotalMips = vm.getCurrentRequestedTotalMips();
double hostUtilizationMips = getUtilizationOfCpuMips(host);
double hostPotentialUtilizationMips = hostUtilizationMips + requestedTotalMips;
double pePotentialUtilization = hostPotentialUtilizationMips / host.getTotalMips();
return pePotentialUtilization;
}

/**
 * Gets the utilization of the CPU in MIPS for the current potentially allocated VMs.
 *
 * @param host the host
 *
 * @return the utilization of the CPU in MIPS
 */
protected double getUtilizationOfCpuMips(PowerHost host) {
    double hostUtilizationMips = 0;
    for (Vm vm2 : host.getVmList()) {
        if (host.getVmsMigratingIn().contains(vm2)) {
            // calculate additional potential CPU usage of a migrating in VM
            hostUtilizationMips += host.getTotalAllocatedMipsForVm(vm2) * 0.9 / 0.1;
        }
        hostUtilizationMips += host.getTotalAllocatedMipsForVm(vm2);
    }
    return hostUtilizationMips;
}

/**
 * Gets the saved allocation.
 *
 * @return the saved allocation

```

```
*/
protected List<Map<String, Object>> getSavedAllocation() {
return savedAllocation;
}

/**
 * Sets the vm selection policy.
 *
 * @param vmSelectionPolicy the new vm selection policy
 */
protected void setVmSelectionPolicy(PowerVmSelectionPolicy vmSelectionPolicy) {
this.vmSelectionPolicy = vmSelectionPolicy;
}

/**
 * Gets the vm selection policy.
 *
 * @return the vm selection policy
 */
protected PowerVmSelectionPolicy getVmSelectionPolicy() {
return vmSelectionPolicy;
}

/**
 * Gets the utilization history.
 *
 * @return the utilization history
 */
public Map<Integer, List<Double>> getUtilizationHistory() {
return utilizationHistory;
}
```

```
}

/**
 * Gets the metric history.
 *
 * @return the metric history
 */
public Map<Integer, List<Double>> getMetricHistory() {
    return metricHistory;
}

/**
 * Gets the time history.
 *
 * @return the time history
 */
public Map<Integer, List<Double>> getTimeHistory() {
    return timeHistory;
}

/**
 * Gets the execution time history vm selection.
 *
 * @return the execution time history vm selection
 */
public List<Double> getExecutionTimeHistoryVmSelection() {
    return executionTimeHistoryVmSelection;
}

/**
```

```
* Gets the execution time history host selection.
*
* @return the execution time history host selection
*/
public List<Double> getExecutionTimeHistoryHostSelection() {
return executionTimeHistoryHostSelection;
}

/**
* Gets the execution time history vm reallocation.
*
* @return the execution time history vm reallocation
*/
public List<Double> getExecutionTimeHistoryVmReallocation() {
return executionTimeHistoryVmReallocation;
}

/**
* Gets the execution time history total.
*
* @return the execution time history total
*/
public List<Double> getExecutionTimeHistoryTotal() {
return executionTimeHistoryTotal;
}

}
```

D.2 PowerVmAllocationPolicyMigrationLocalRegression

```
/*
 * Title:          CloudSim Toolkit
 * Description:    CloudSim (Cloud Simulation) Toolkit for Modeling and Simulation of C
 * Licence:       GPL - http://www.gnu.org/copyleft/gpl.html
 *
 * Copyright (c) 2009-2012, The University of Melbourne, Australia
 */

package org.cloudbus.cloudsim.power;

import java.util.List;

import org.cloudbus.cloudsim.Host;
import org.cloudbus.cloudsim.Vm;
import org.cloudbus.cloudsim.util.MathUtil;

/**
 * The Local Regression (LR) VM allocation policy.
 *
 * If you are using any algorithms, policies or workload included in the power packag
 * the following paper:
 *
 * Anton Beloglazov, and Rajkumar Buyya, "Optimal Online Deterministic Algorithms and
 * Heuristics for Energy and Performance Efficient Dynamic Consolidation of Virtual M
 * Cloud Data Centers", Concurrency and Computation: Practice and Experience (CCPE), V
 * Issue 13, Pages: 1397-1420, John Wiley & Sons, Ltd, New York, USA, 2012
 *
 * @author Anton Beloglazov

```

```
* @since CloudSim Toolkit 3.0
*/
public class PowerVmAllocationPolicyMigrationLocalRegression extends PowerVmAlloca

/** The scheduling interval. */
private double schedulingInterval;

/** The safety parameter. */
private double safetyParameter;

/** The fallback vm allocation policy. */
private PowerVmAllocationPolicyMigrationAbstract fallbackVmAllocationPolicy;

/**
 * Instantiates a new power vm allocation policy migration local regression.
 *
 * @param hostList the host list
 * @param vmSelectionPolicy the vm selection policy
 * @param schedulingInterval the scheduling interval
 * @param fallbackVmAllocationPolicy the fallback vm allocation policy
 * @param utilizationThreshold the utilization threshold
 */
public PowerVmAllocationPolicyMigrationLocalRegression(
List<? extends Host> hostList,
PowerVmSelectionPolicy vmSelectionPolicy,
double safetyParameter,
double schedulingInterval,
PowerVmAllocationPolicyMigrationAbstract fallbackVmAllocationPolicy,
double utilizationThreshold) {
super(hostList, vmSelectionPolicy);
```

```
setSafetyParameter(safetyParameter);
setSchedulingInterval(schedulingInterval);
setFallbackVmAllocationPolicy(fallbackVmAllocationPolicy);
}

/**
 * Instantiates a new power vm allocation policy migration local regression.
 *
 * @param hostList the host list
 * @param vmSelectionPolicy the vm selection policy
 * @param schedulingInterval the scheduling interval
 * @param fallbackVmAllocationPolicy the fallback vm allocation policy
 */
public PowerVmAllocationPolicyMigrationLocalRegression(
    List<? extends Host> hostList,
    PowerVmSelectionPolicy vmSelectionPolicy,
    double safetyParameter,
    double schedulingInterval,
    PowerVmAllocationPolicyMigrationAbstract fallbackVmAllocationPolicy) {
    super(hostList, vmSelectionPolicy);
    setSafetyParameter(safetyParameter);
    setSchedulingInterval(schedulingInterval);
    setFallbackVmAllocationPolicy(fallbackVmAllocationPolicy);
}

/**
 * Checks if is host over utilized.
 *
 * @param host the host
 * @return true, if is host over utilized

```

```
*/
@Override
protected boolean isHostOverUtilized(PowerHost host) {
    PowerHostUtilizationHistory _host = (PowerHostUtilizationHistory) host;
    double[] utilizationHistory = _host.getUtilizationHistory();
    int length = 10; // we use 10 to make the regression responsive enough to latest v
    if (utilizationHistory.length < length) {
        return getFallbackVmAllocationPolicy().isHostOverUtilized(host);
    }
    double[] utilizationHistoryReversed = new double[length];
    for (int i = 0; i < length; i++) {
        utilizationHistoryReversed[i] = utilizationHistory[length - i - 1];
    }
    double[] estimates = null;
    try {
        estimates = getParameterEstimates(utilizationHistoryReversed);
    } catch (IllegalArgumentException e) {
        return getFallbackVmAllocationPolicy().isHostOverUtilized(host);
    }
    double migrationIntervals = Math.ceil(getMaximumVmMigrationTime(_host) / getSchedu
    double predictedUtilization = estimates[0] + estimates[1] * (length + migrationInt
    predictedUtilization *= getSafetyParameter());

    addHistoryEntry(host, predictedUtilization);

    return predictedUtilization >= 1;
}

/**
 * Gets the parameter estimates.
```

```
*
* @param utilizationHistoryReversed the utilization history reversed
* @return the parameter estimates
*/
protected double[] getParameterEstimates(double[] utilizationHistoryReversed) {
return MathUtil.getLoessParameterEstimates(utilizationHistoryReversed);
}

/**
* Gets the maximum vm migration time.
*
* @param host the host
* @return the maximum vm migration time
*/
protected double getMaximumVmMigrationTime(PowerHost host) {
int maxRam = Integer.MIN_VALUE;
for (Vm vm : host.getVmList()) {
int ram = vm.getRam();
if (ram > maxRam) {
maxRam = ram;
}
}
return maxRam / ((double) host.getBw() / (2 * 8000));
}

/**
* Sets the scheduling interval.
*
* @param schedulingInterval the new scheduling interval
*/
```

```
protected void setSchedulingInterval(double schedulingInterval) {
    this.schedulingInterval = schedulingInterval;
}

/**
 * Gets the scheduling interval.
 *
 * @return the scheduling interval
 */
protected double getSchedulingInterval() {
    return schedulingInterval;
}

/**
 * Sets the fallback vm allocation policy.
 *
 * @param fallbackVmAllocationPolicy the new fallback vm allocation policy
 */
public void setFallbackVmAllocationPolicy(
    PowerVmAllocationPolicyMigrationAbstract fallbackVmAllocationPolicy) {
    this.fallbackVmAllocationPolicy = fallbackVmAllocationPolicy;
}

/**
 * Gets the fallback vm allocation policy.
 *
 * @return the fallback vm allocation policy
 */
public PowerVmAllocationPolicyMigrationAbstract getFallbackVmAllocationPolicy() {
    return fallbackVmAllocationPolicy;
}
```

```
}

public double getSafetyParameter() {
    return safetyParameter;
}

public void setSafetyParameter(double safetyParameter) {
    this.safetyParameter = safetyParameter;
}

}
```

D.3 PowerVmAllocationPolicyMigrationLocalRegression

```
/*
 * Title:          CloudSim Toolkit
 * Description:    CloudSim (Cloud Simulation) Toolkit for Modeling and Simulation of C
 * Licence:        GPL - http://www.gnu.org/copyleft/gpl.html
 *
 * Copyright (c) 2009-2012, The University of Melbourne, Australia
 */

package org.cloudbus.cloudsim.power;

import java.util.List;

import org.cloudbus.cloudsim.Host;
import org.cloudbus.cloudsim.util.MathUtil;
```

ANNEXE D. ANNEXE 4:FICHIERS DE POLITIQUES D'ALLOCATION DE VM DANS CLOUDSIM

```
/**
 * The Local Regression Robust (LRR) VM allocation policy.
 *
 * If you are using any algorithms, policies or workload included in the power pac
 * the following paper:
 *
 * Anton Beloglazov, and Rajkumar Buyya, "Optimal Online Deterministic Algorithms
 * Heuristics for Energy and Performance Efficient Dynamic Consolidation of Virtua
 * Cloud Data Centers", Concurrency and Computation: Practice and Experience (CCPE
 * Issue 13, Pages: 1397-1420, John Wiley & Sons, Ltd, New York, USA, 2012
 *
 * @author Anton Beloglazov
 * @since CloudSim Toolkit 3.0
 */
public class PowerVmAllocationPolicyMigrationLocalRegressionRobust extends
PowerVmAllocationPolicyMigrationLocalRegression {

/**
 * Instantiates a new power vm allocation policy migration local regression.
 *
 * @param hostList the host list
 * @param vmSelectionPolicy the vm selection policy
 * @param schedulingInterval the scheduling interval
 * @param fallbackVmAllocationPolicy the fallback vm allocation policy
 * @param utilizationThreshold the utilization threshold
 */
public PowerVmAllocationPolicyMigrationLocalRegressionRobust(
List<? extends Host> hostList,
PowerVmSelectionPolicy vmSelectionPolicy,
double safetyParameter,
```

```
double schedulingInterval,
PowerVmAllocationPolicyMigrationAbstract fallbackVmAllocationPolicy,
double utilizationThreshold) {
    super(
        hostList,
        vmSelectionPolicy,
        safetyParameter,
        schedulingInterval,
        fallbackVmAllocationPolicy,
        utilizationThreshold);
}

/**
 * Instantiates a new power vm allocation policy migration local regression.
 *
 * @param hostList the host list
 * @param vmSelectionPolicy the vm selection policy
 * @param schedulingInterval the scheduling interval
 * @param fallbackVmAllocationPolicy the fallback vm allocation policy
 */
public PowerVmAllocationPolicyMigrationLocalRegressionRobust(
    List<? extends Host> hostList,
    PowerVmSelectionPolicy vmSelectionPolicy,
    double safetyParameter,
    double schedulingInterval,
    PowerVmAllocationPolicyMigrationAbstract fallbackVmAllocationPolicy) {
    super(hostList, vmSelectionPolicy, safetyParameter, schedulingInterval, fallbackVmAll
}

/**
```

```
* Gets the parameter estimates.  
*  
* @param utilizationHistoryReversed the utilization history reversed  
* @return the parameter estimates  
*/  
@Override  
protected double[] getParameterEstimates(double[] utilizationHistoryReversed) {  
    return MathUtil.getRobustLoessParameterEstimates(utilizationHistoryReversed);  
}  
  
}
```




Fréjus A. Roméo GBAGUIDI

Approche Prédictive de l'Efficacité Energétique dans les Clouds Datacenters

le cnam

Abstract :

This study focused particularly on the AutoRegressive Moving Average (ARMA) tools and neural networks in improvement of It resources allocation and consequently power allocation into cloud computing environments. Based on our simulations and results, this approach enabled us to reduce energy consumption on a firm of 800 servers over a period of one day by more than 5Kwh. This gain could be significant when considering the enormous size of modern datacenters and projected over a relatively long period of time. It would be even more interesting to deepen this research in order to generalize the integration of this predictive approach into existing techniques in order to significantly optimize the energy consumption within datacenters while preserving performance and quality of service which are key requirements in the concept of Cloud Computing.

Keywords :

Cloud Computing, Energy, Prediction, Neural networks, ARMA.

Résumé :

Notre travail s'est focalisée notamment sur l'utilisation des outils ARMA (AutoRegressive Moving Average) et les réseaux de neurones pour l'amélioration des techniques d'allocation des ressources et par conséquent de l'énergie dans les environnements de cloud computing. Cette étude a permis sur la base de nos simulations de réduire de plus de 5Kwh la consommation d'énergie dans une ferme de 800 serveurs et sur une durée d'une journée. Ce gain pourrait se révéler important lorsque l'on considère la taille énorme des datacenters modernes et que l'on se projette dans une durée relativement longue. Il serait encore plus intéressant d'approfondir cette recherche afin de généraliser l'intégration de cette approche prédictive dans les techniques existantes afin d'optimiser de façon significative les consommations d'énergie au sein des datacenters tout en préservant les performances et la qualité de service indispensable dans le concept de cloud computing.

Mots clés :

Cloud Computing, Energie, Prédiction, Réseaux de neurones, ARMA.