

République Algérienne Démocratique et Populaire
Université Abou Bakr Belkaid– Tlemcen
Faculté des Sciences
Département d'Informatique

Mémoire de fin d'études

Pour l'obtention du diplôme de Master en Informatique

Option:

Réseaux et Systèmes et Distribués (R.S.D)

Thème

Algorithme d'optimisation multi-objectifs dans le Cloud computing

Réalisé par :

- BOURMAKI Imane
- MOHAMMEDI Meriem

Présenté le 6 Juillet 2019 devant le jury composé de :

- Mr LEHSAINI Mohamed (Président)
- Mr BENMAMMAR Badr (Encadrant)
- Mr BOUAFIA Zoheir (Co-encadrant)
- Mr BELHOUCINE Amin (Examineur)

Année universitaire : 2018-2019

Remerciement

Nous remercions Allah qui nous a aidé et donné le courage, la volonté pour la réalisation de ce PFE.

Tout d'abord, nous tenons à exprimer nos gratitude à Mr BENMAMMAR Badr en tant que encadrant de ce PFE et Mr BOUAFIA Zoheir en tant que co-encadrant. Ils nous ont permis de réaliser ce travail sous leurs directions et aussi pour leurs conseils, orientations durant toute la réalisation de ce travail .

Nous remercions les membres du jury pour avoir accepté d'examiner et évaluer notre PFE.

Nos remerciements également à tous nos enseignants durant les années de nos études.

Enfin, nous tenons à remercier toute personne qui nous ont aidé et encouragé le long de ce travail.

Dédicace

Nous dédions ce mémoire :

A Nos très chers parents qui ont toujours nous ont soutenus et encouragés

durant toutes nos années d'études,

A nos frères, nos sœurs,

A nos familles et tous nos amis,

A tous qui ont aider pour réaliser ce travail,

Je vous dis merci.

Sommaire

Remerciement.....	i
Dédicace	ii
Liste des figures	v
Liste des tableaux	vi
Liste des abréviations.....	vii
Introduction générale.....	1
CHAPITRE I: CLOUD COMPUTING.....	2
I.1. Introduction	3
I.2 Notions de base.....	3
I.2.1 Définition	3
I.2.2 Avantages et inconvénients	3
I.2.3 La virtualisation	4
I.2.4 Le DataCenter	4
I.2.5 Le service provider.....	4
I.2.6 Le SLA.....	4
I.2.7 La consolidation.....	5
I.2.8 Machine virtuelle	5
I.2.9 Cloudlet.....	5
I.2.10 Broker	5
I.3 Types de Cloud computing.....	5
I.3.1 Cloud privé	5
I.3.2 Cloud public.....	6
I.3.3 Cloud hybride.....	6
I.4 Différentes architectures de Cloud computing	6
I.4.1 Infrastructure as a Service (IaaS)	6
I.4.2 Platform as a Service (PaaS).....	7
I.4.3 Software as a Service (SaaS)	7
CHAPITRE II: Les métaheuristiques.....	8
II.1 Introduction.....	9
II.2 Heuristique et métaheuristique	9
II.2.1 Heuristique.....	9
II.2.2 Métaheuristique	9

II.3 Classification des métaheuristiques	9
II.3.1 Les métaheuristiques à solution unique.....	10
II.3.2 Les métaheuristiques à base de population	10
II.4 Les algorithmes d'intelligence en essaim	10
II.4.1 L'algorithme de colonies de fourmis	11
II.4.2 L'algorithme des chauves-souris.....	11
II.5 Le cuckoo search	11
II.5.1 Comportement de reproduction de coucou.....	11
II.5.2 Cuckoo search et Levy Flights	12
II.5.3 Description de l'algorithme	12
II.6 Conclusion	13
CHAPITRE III: Implémentation.....	14
III.1 Introduction	15
III.2 Outils utilisés et environnement de développment	15
III.2.1 Java.....	15
III.2.2 NetBeans	15
III.2.3 CloudSim.....	16
III.3 Architecture de CloudSim.....	16
III.4 Approche proposée	18
III.4.1 La fonction objectif	18
III.4.2 Phase de configuration.....	20
III.4.3 Phase de pré-simulation.....	20
III.4.4 Phase de simulation	20
III.5 IHM développée	20
III.6 Résultat obtenus.....	24
III.7 Conclusion.....	27
Conclusion générale	28
Bibliographie	29
Résumé	32

Liste des figures

Figure I.1 : Les différentes architectures de Cloud Computing [8].....	6
Figure II.1 : Classification des méta-heuristiques.....	10
Figure II.2 : Pseudo code de Cuckoo Search	13
Figure III.1 :Architecture de cloudSim [27]	17
Figure III.2 :Interface principale	21
Figure III.3 : Création de Datacenter	22
Figure III.4 : Création des Hosts	23
Figure III.5 : Création des machines virtuelles	23
Figure III.6 : Création des Cloudlets	24
Figure III.7 : Comparaison entre le CS et le RR en termes de temps de réponse.....	25

Liste des tableaux

Tableau III.1 : Les paramètres des hosts.....	24
Tableau III.2 : Les paramètres des VMS	25
Tableau III.3 : Les paramètres des Cloudlets.....	25
Tableau III.4 : Comparaison entre le Cs et le RR en termes de réponse.	25
Tableau III.5 : Comparaison entre le CS et le RR en termes de cout	26
Tableau III.6 : Comparaison de la consommation d'énergie entre CS et RR.....	27

Liste des abréviations

ACO	Ant Colony Optimization
CS	Cuckoo Search
IAAS	Infrastructure As A Service
JVM	Java Virtual Machine
OS	Operating System
PAAS	Platform As A Service
QoS	Quality of Service
RR	Round Robin
SAAS	Software As A Service
SLA	Service Level Agreement
VM	Virtual Machine
VPN	Virtual Private Network

Introduction générale

Le Cloud computing est une technologie moderne dans le domaine informatique. Il est apparu avec l'augmentation de la puissance des serveurs et la baisse de leurs coûts de fonctionnement ainsi qu'avec l'apparition de l'internet haut débit. Le Cloud computing permet de fournir plusieurs services flexibles et évolutifs permettant l'accès au réseau à la demande. Les ressources sont partagées et la puissance de calcul est configurable par rapport aux besoins des utilisateurs. Actuellement, beaucoup de fournisseurs comme Google, Amazon, IBM offrent une vaste gamme de services de Cloud Computing.

Le Cloud computing se caractérise par sa souplesse et par ses avantages parmi lesquels, l'automatisation de la maintenance des applications. Pas besoin d'acheter les nouvelles versions logicielles et de les installer sur toutes les machines de l'entreprise. L'installation des logiciels sur les différentes machines se fait automatiquement par le fournisseur de l'application et après l'installation il suffit de payer l'abonnement et les services désirés. Tous les utilisateurs bénéficient ensuite des nouveautés instantanément. Le Cloud computing permet également aux utilisateurs d'accéder à leurs fichiers pendant leurs déplacements, et ceci à partir de n'importe quel appareil relié à internet. En outre, il permet d'améliorer la sécurité des données de l'utilisateur.

Dans ce PFE, nous proposons un algorithme pour l'optimisation de plusieurs objectifs dans le Cloud computing. Cet algorithme est basé sur la métaheuristique cuckoo search permettant de trouver la solution optimale en un temps raisonnable. La problématique traitée concerne l'affectation des tâches dans différentes machines virtuelles.

Le simulateur CloudSim a été utilisé dans le cadre de ce PFE et les performances de notre algorithme sont évaluées à l'aide de trois critères qui sont : le temps de réponse, le coût et la consommation d'énergie.

Pour la validation de l'approche proposée, nous l'avons comparé avec un algorithme classique déjà implémenté dans CloudSim qui est le Round Robin.

Ce mémoire est organisé en trois chapitres :

Dans le premier chapitre, nous avons donné quelques notions de base sur le Cloud computing, les types de Cloud ainsi que les différentes architectures de Cloud.

Dans le deuxième chapitre, nous avons présenté les métaheuristiques en se focalisant en particulier sur l'algorithme cuckoo search qui a été utilisé dans la partie réalisation de ce PFE.

Notre contribution est détaillée dans le troisième chapitre. Ce dernier présente l'environnement de travail en particulier le simulateur CloudSim, l'IHM réalisée dans le cadre de ce PFE. Enfin, les résultats des différentes simulations réalisées avec CloudSim et aussi les comparaisons entre cuckoo search et Round Robin sont détaillées dans ce chapitre.

Le mémoire se termine par une conclusion générale et des perspectives.

CHAPITRE I: CLOUD COMPUTING

Chapitre I : Cloud computing

I.1. Introduction

Le concept de Cloud computing consiste à offrir des ressources informatiques (serveurs, stockage, réseau, applications...) en tant que services à la demande sur un réseau. Il s'agit d'une notion qui est très utilisée aujourd'hui par exemple le gmail ou la sauvegarde des données sur Internet en général. Le Cloud computing offre aussi la possibilité d'utiliser une puissance de calcul et une capacité de stockage offerte par des serveurs distants.

Ce chapitre sera consacré à la présentation des principales notions de Cloud Computing : sa définition, ses caractéristiques, les conditions qui encouragent à l'utiliser ainsi que les problèmes qui se posent avec son utilisation.

I.2 Notions de base

I.2.1 Définition

Le Cloud computing fournit un ensemble de matérielle (serveurs, réseaux) et de logicielle à la demande. Il permet d'accéder rapidement à des ressources informatique n'importe où dans le monde et avec n'importe quel appareil connecté. C'est une technologie qui permet de gérer la puissance de calcul, le stockage à travers des serveurs distants. Il est aussi caractérisé par sa souplesse permettent aux fournisseurs d'adapter une capacité de stockage, et une puissance de calcul selon les besoins des utilisateurs.

Le Cloud computing est basé sur la consommation des ressources, des matérielles et des logicielles. Des services qui sont gratuits ou payants selon l'utilisation sont les suivants :

- Une demande libre de service.
- Un accès en diffusé via un réseau.
- La mis en commun des ressources.
- L'élasticité rapide.
- Un service mesuré.

I.2.2 Avantages et inconvénients

▪ **Avantages** : le Cloud présente de nombreux avantages qui sont cités dans ce qui suit [1] :

- La flexibilité d'utilisation.
- La facilité de la récupération des données.
- La mise à jour logicielle qui est faite d'une manière automatique.
- L'absence de frais d'investissement pour les utilisateurs « paiement à l'utilisation ».
- Le contrôle des documents à distance.
- La sécurité assurée pour les services Cloud.

▪ **Inconvénients** : toutefois, le Cloud présente quelques inconvénients comme [2] :

Chapitre I : Cloud computing

- Le coût pour la mise en œuvre de son infrastructure.
- Le coût pour assurer l'optimisation des applications ainsi que les mises à jour.
- Le coût pour assurer sa sécurité (ex. piratage des données).
- Le coût pour avoir une connexion internet très performante car c'est primordiale pour les différents services Cloud.
- Le cadre légal : les données transférées dans le Cloud ne sont pas forcément présentes sur le territoire national : elles peuvent l'être, comme elles peuvent être dans un autre pays. Par conséquent, sauf mention contraire du prestataire de service, on ne sait pas précisément à quel endroit sont stockées les données. De plus, on n'a aucun accès physique à ces données.

I.2.3 La virtualisation

La virtualisation est un mécanisme informatique qui consiste à faire fonctionner plusieurs systèmes, serveurs ou applications, sur un même serveur physique. Il s'agit d'un composant technique clé dans le Cloud Computing. [3]

La virtualisation repose sur ce qui suit [4] :

- Un système d'exploitation principal (appelé « système hôte ») est installé sur un serveur physique unique. Ce système sert d'accueil à d'autres systèmes d'exploitation.
- Un logiciel de virtualisation est installé sur le système d'exploitation principal. Il permet la création d'environnements clos et indépendants sur lesquels seront installés d'autres systèmes d'exploitation (« systèmes invités »). Ces environnements sont des « machines virtuelles ».
- Un système invité est installé dans une machine virtuelle qui fonctionne indépendamment des autres systèmes invités dans d'autres machines virtuelles. Chaque machine virtuelle dispose d'un accès aux ressources du serveur physique (mémoire, espace disque...).

I.2.4 Le DataCenter

Un DataCenter ou centre de données, est une infrastructure composée d'un réseau d'ordinateurs et d'espaces de stockage. Cette infrastructure peut être utilisée par les entreprises pour organiser, traiter, stocker et entreposer de grandes quantités de données.

Un DataCenter est un ensemble d'éléments qui regroupe des serveurs, des sous-systèmes de stockage, des commutateurs de réseau, des routeurs, des firewalls. [5]

I.2.5 Le service provider

C'est une entreprise qui fournit des services au client, du matériel virtuelle, des logiciels et d'autre service loués et gérés par le fournisseur basées sur un accord de niveau de service (SLA) et une négociation entre le fournisseur de services et le consommateur.

I.2.6 Le SLA

Un contrat SLA (contrat de niveau de service) est un contrat entre un fournisseur de services de Cloud et un client qui garantit le maintien d'un niveau de service minimal. Il

Chapitre I : Cloud computing

garantit des niveaux de fiabilité, de disponibilité et de réactivité aux systèmes et aux applications, tout en spécifiant qui gouvernera en cas d'interruption de service. Autrement dit, il s'agit d'une clause contractuelle qui définit les objectifs précis et le niveau de service qu'est en droit d'attendre un client de la part du prestataire signataire.

Il permet de garantir aux clients certains niveaux de sécurité dans le stockage et la gestion de leurs données à caractère personnel.

I.2.7 La consolidation

La consolidation survient généralement pendant les fusions et les acquisitions d'entreprises. Lorsque l'entreprise majoritaire n'a pas besoin des centres de données utilisées par les firmes qu'elles rachètent, donc l'entreprise avec plusieurs DataCenter peut choisir de les consolider, en réduisant leur nombre pour minimiser les coûts des opérations. [5]

I.2.8 Machine virtuelle

Une machine virtuelle, ou VM, est un environnement d'application ou de système d'exploitation (OS) installé sur un logiciel qui imite un matériel dédié. Côté utilisateur final, l'interaction avec une machine virtuelle est la même qu'avec un matériel dédié. [6]

I.2.9 Cloudlet

Est une unité de base de la programmation contrôlée par le système d'exploitation. Selon la manière dont le système d'exploitation définit la tâche dans sa conception, cette unité de programmation peut être un programme sous licence ou tout appel réussi à un programme. Parce que le programme peut faire des demandes à d'autres utilitaires, les utilitaires peuvent également être considérés comme des tâches.

I.2.10 Broker

Le Broker est l'intermédiaire entre les VMs et les Cloudlets. Il est le responsable de la médiation des négociations entre l'utilisateur et les prestataires de service selon les conditions de QoS des utilisateurs.

I.3 Types de Cloud computing

Il existe trois types de Cloud computing :

I.3.1 Cloud privé

Un Cloud privé est utilisé par un groupe ou une organisation spécifique avec un accès limité, peut être géré par l'entreprise utilisatrice elle-même.

Le Cloud privé peut se déployer sous deux formes distinctes:

- Cloud privé interne: hébergé par l'entreprise elle-même, parfois partagé ou mutualisé en mode privatif avec les filiales. [7]
- Cloud privé externe: hébergé chez un tiers, il est entièrement dédié à l'entreprise et accessible via des réseaux sécurisés de type VPN. [7]

Chapitre I : Cloud computing

I.3.2 Cloud public

Un Cloud public repose sur le modèle standard de Cloud computing dans lequel le fournisseur de services rend des ressources accessibles à tout utilisateur qui a une connexion internet. Il offre aussi à ses utilisateurs l'évolutivité, la flexibilité et aussi des options supplémentaires de contrôle. Les services de Cloud public peuvent être gratuits ou facturés à l'utilisation.

I.3.3 Cloud hybride

Un Cloud hybride est un environnement informatique qui fait une combinaison de différents types de Cloud (public, privé) relié entre eux par des technologies.

I.4 Différentes architectures de Cloud computing

L'IAAS, le PAAS et le SAAS sont actuellement les 3 principaux modèles de services qui s'offrent aux entreprises désireuses de s'orienter vers le Cloud Computing.

La Figure I.1 représente les différentes architectures du Cloud computing de l'architecture la moins visible à la plus visible pour les utilisateurs.

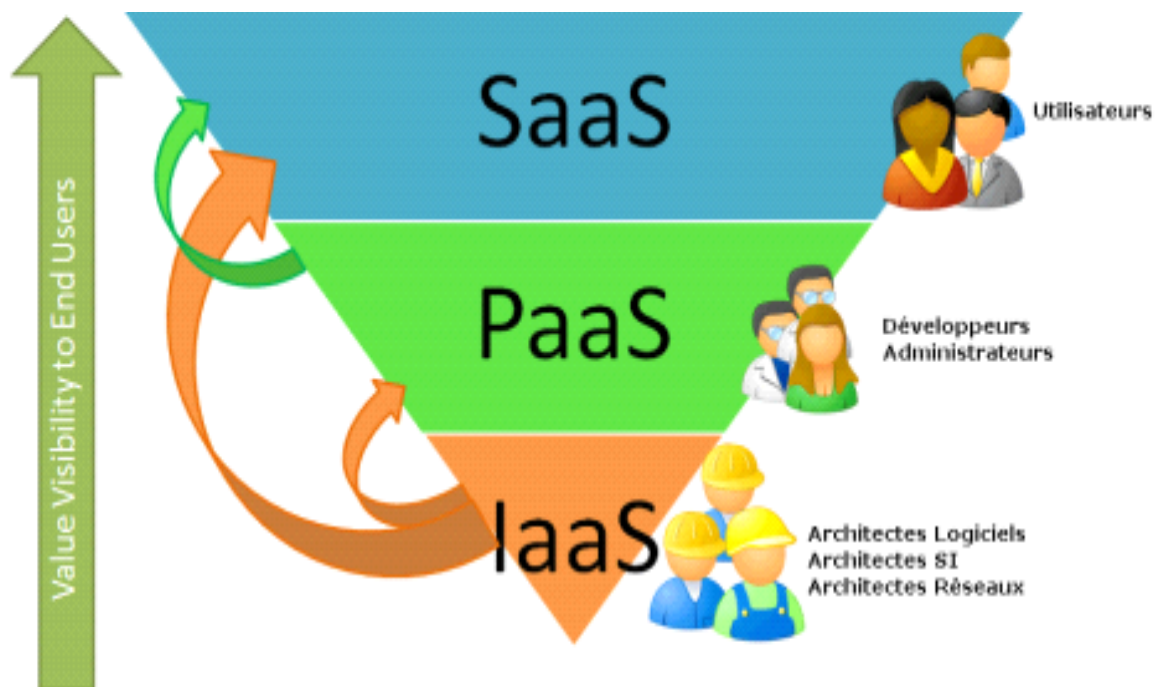


Figure I.1 : Les différentes architectures de Cloud Computing [8]

I.4.1 Infrastructure as a Service (IaaS)

L'Infrastructure As A Service (IAAS) signifie l'externalisation de l'infrastructure matérielle du service informatique (réseaux, stockage et serveurs) chez un fournisseur tiers. Les ressources informatiques sont hébergées sur des serveurs externes et les utilisateurs peuvent y accéder via une connexion internet. Ils conservent à leurs charges la gestion des

Chapitre I : Cloud computing

systèmes d'exploitation et du « middleware » : bases de données, intégration, environnement d'exécution, ainsi que les applications.

Avantage : Grande flexibilité, contrôle total des systèmes (administration à distance), et donc ainsi permettre d'installer tout type de logiciel métier. [9]

Inconvénient : Besoin d'administrateurs système comme pour les solutions de serveurs classiques sur site. [9]

I.4.2 Platform as a Service (Paas)

Platform as a Service permet aux utilisateurs de louer des serveurs virtualisés pour exécuter ou développer ses propres applications à l'aide du kit de développement fourni par le Cloud computing. Le PAAS facilite à travers des outils et des services le flux de travail lors de la conception, du développement, du test, du déploiement et de l'hébergement d'applications.

Avantage : Le déploiement est automatisé, pas de logiciel supplémentaire à acheter ou à installer. [9]

Inconvénient : Limitation à une ou deux technologies. Pas de contrôle de machines virtuelles sous jacentes. [9]

I.4.3 Software as a Service (Saas)

Le SAAS permet de disposer à distance d'une solution logicielle dans un environnement hébergé. Les applications et les données sont disponibles via un navigateur web.

Avantage : Plus d'installation, plus de mise à jour (elles sont continues chez le fournisseur), plus de migration de données etc. Paiement à l'usage. Test de nouveaux logiciels avec facilité. [9]

Inconvénient : Limitation par définition au logiciel proposé. Pas de contrôle sur le stockage et la sécurisation des données associées au logiciel. [9]

I.5 Conclusion

Ce chapitre a été consacré à la présentation du Cloud computing, le domaine qui représente le contexte de travail de notre PFE. Nous avons présenté donc les notions fondamentales et nécessaires à la compréhension de ce domaine, en particulier son architecture, ses types, ses avantages et ses inconvénients mais également quelques mots clé relatifs à sa compréhension comme les VMs, les Cloudlets, le SLA, le Datacenter, la virtualisation, ... etc.

Le deuxième chapitre de notre PFE sera consacré à la présentation des métaheuristiques.

CHAPITRE II:

Les métaheuristiques

Chapitre II : Métaheuristique

II.1 Introduction

Aujourd'hui les métaheuristique deviennent de plus en plus capable de résoudre des problèmes dans lesquels il est difficile de trouver des solutions efficaces avec des méthodes plus classique. La plus part de ces algorithmes sont inspirés de la nature à travers l'intelligence des animaux comme les poissons et les oiseaux.....etc.

Dans ce chapitre, nous commençons par donner la définition d'heuristique, de métaheuristique et de leurs caractéristiques. Nous présenteront par la suite quelques méta-heuristiques en s'intéressant à un algorithme en particulier. Il s'agit de l'algorithme de la recherche coucou (Cuckoo Search en anglais), un algorithme basé sur le comportement des coucous, leurs reproductions et leurs développements.

II.2 Heuristique et métaheuristique

II.2.1 Heuristique

Une heuristique est un algorithme approché qui fournit rapidement une solution réalisable, sans garanti d'optimalité, pour un problème d'optimisation spécifique. C'est une règle d'estimation, une stratégie, une méthode ou astuce utilisée pour améliorer l'efficacité d'un système qui tente de découvrir les solutions des problèmes complexes. [11]

II.2.2 Métaheuristique

Le mot metaheuristique est décomposé en deux mots grecs : meta, du grec signifiant « a un plus haut niveau » et heuristique qui signifie « je trouve ».

Une méta-heuristique est un algorithme plus complexe qu'une heuristique, adaptée pour différents types de problèmes. Il permet la résolution d'un grand nombre de problèmes d'optimisation pour trouver une solution optimale et de très bonne qualité dans un temps raisonnable.

II.3 Classification des métaheuristiques

Dans la littérature, les métaheuristiques sont généralement classées en deux sous catégories : les métaheuristiques à solution unique et les métaheuristiques à base de population.

La figure II.1 représente la classification des métaheuristiques.

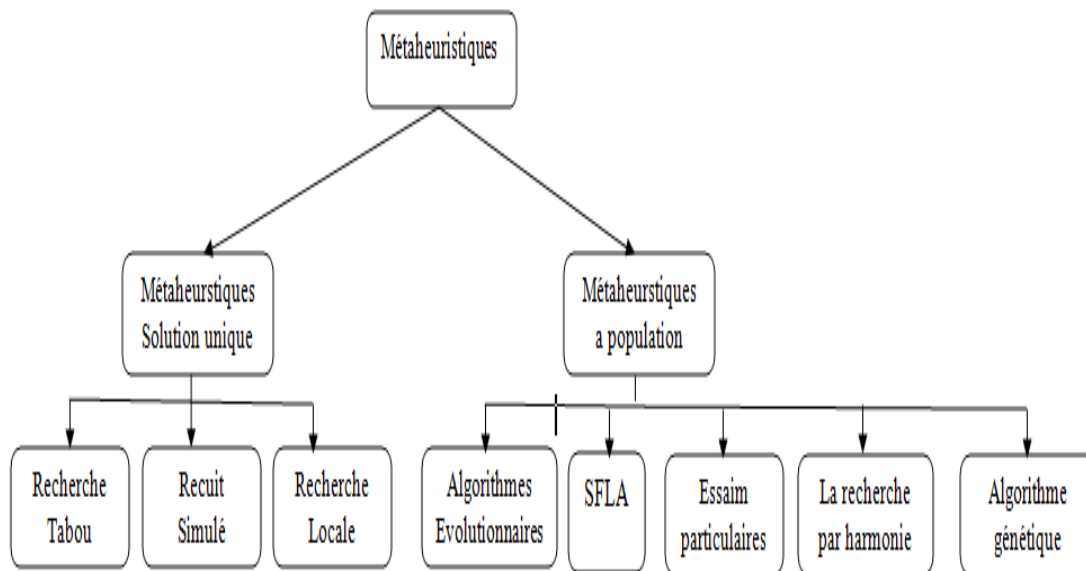


Figure II.1 : Classification des méta-heuristiques.

II.3.1 Les métaheuristiques à solution unique

Les métaheuristiques à base de solution unique, aussi appelées méthodes de trajectoire. Contrairement aux métaheuristiques à base de population, ces méthodes débutent la recherche avec une seule solution initiale. Elles se basent sur la notion du voisinage pour améliorer la qualité de la solution courante. En fait, la solution initiale subit une série de modifications en fonction de son voisinage. [11, 12]. De nombreuses méthodes ont été développées dans cette catégorie, parmi lesquelles nous citons les plus connues, à savoir : la méthode de descente [13], la méthode du recuit simulé [14], la recherche tabou [15].

II.3.2 Les métaheuristiques à base de population

Les métaheuristiques sont utilisées pour optimiser les problèmes complexes de grandes tailles. Contrairement aux méthodes exactes, les métaheuristiques à base de population de solutions améliorent une population de solutions, au fur et à mesure des itérations. [16]

Les métaheuristiques sont classées en deux grandes familles : les algorithmes évolutionnaires (comme les algorithmes génétiques) et les algorithmes basés sur l'intelligence en essaim. Dans ce qui suit, nous nous intéressons à cette deuxième catégorie.

II.4 Les algorithmes d'intelligence en essaim

Les algorithmes d'intelligence en essaim inspirés par les insectes sociaux sont de plus en plus étudiés en informatique pour résoudre des problèmes complexes.

Ces algorithmes ont fourni de puissantes méthodes pour concevoir des algorithmes d'optimisation et de contrôle distribué. Qui s'appliquent à toute une série de problèmes scientifiques et techniques.

Dans ce qui suit, nous allons présenter trois algorithmes de cette catégorie, à savoir : l'algorithme de colonies de fourmis (ACO), l'algorithme des chauves-souris (bat algorithm)

Chapitre II : Métaheuristique

et l'algorithme de la recherche coucou (Cuckoo search ou CS) . Nous donnons plus d'intérêt à ce dernier algorithme car c'est celui qui sera utilisé dans la partie réalisation de notre PFE.

II.4.1 L'algorithme de colonies de fourmis

Les algorithmes de colonies de fourmis ont été proposés par Marco Dorigo dans sa thèse de doctorat en 1992 [17]. L'idée de base imite le comportement collectif des fourmis lors de leur déplacement entre la fourmilière et la source de nourriture dans l'objectif de rechercher la nourriture en parcourant le plus court chemin, Quand une fourmi atteint la source de nourriture, elle retrace son chemin de retour vers le nid par une quantité de phéromone pour avertir les autres fourmis qu'il y a de la nourriture à la fin du chemin. Les fourmis les plus rapidement arrivées au nid après avoir visité la source de nourriture sont celles qui empruntent le chemin le plus court, aussi la quantité de phéromone présente sur le plus court chemin est plus importante que celle présente sur le chemin le plus long. Une piste qui présente une plus grande concentration en phéromone est plus attirante par les fourmis, elle a une probabilité plus grande d'être empruntée. La piste courte va alors être plus renforcée que la plus longue, et elle sera choisie par la quasi-totalité des fourmis. L'idée est de représenter le problème à résoudre sous la forme de la recherche d'un meilleur chemin dans un graphe, puis d'utiliser des fourmis artificielles pour rechercher de bons chemins dans ce graphe. [11]

II.4.2 L'algorithme des chauves-souris

L'algorithme des chauves souris a été proposé par Xin She Yang en 2010 [18]. L'optimisation par cette technique est une approche inspirée du comportement de chasse des chauvesouris qui utilisent l'écholocation et qui ont la capacité de déterminer l'environnement autour d'eux. Ils peuvent détecter la distance et faire la différence entre l'orientation de la cible (proie) et l'emplacement des obstacles.

La longueur d'onde et la fréquence des chauves-souris naturelles sont modifiées de temps en temps selon les cas suivants:

- Si aucune proie n'est dans le champ de vision, la valeur de la longueur d'onde est augmenté, et la fréquence est diminuée.
- Si une proie se trouve dans l'oscilloscope, la longueur d'onde est diminuée, et la fréquence est augmentée pour pouvoir suivre la proie précisément.

L'augmentation de la fréquence donne aux chauves-souris la possibilité de suivre la cible plus rapidement et surveiller chacun de ses mouvements. [19]

II.5 Le cuckoo search

II.5.1 Comportement de reproduction de coucou

Les coucous sont des oiseaux fascinants, non seulement à cause des beaux sons qu'ils peuvent faire, mais aussi à cause de leur stratégie de reproduction agressive. Certaines espèces comme les coucous ani et guira pondent leurs œufs dans des nids communs, bien qu'ils puissent retirer les œufs d'autrui pour augmenter la probabilité de leurs propres œufs. Il s'agit du parasitisme obligatoire couvé (parasitisme de ponte) qui consiste à pondre des œufs dans

Chapitre II : Métaheuristique

les nids d'autres oiseaux hôtes (souvent d'autres espèces). Certains oiseaux hôtes peuvent engager un conflit direct avec les coucous intrus. Si un oiseau hôte découvre que les œufs ne sont pas les leurs, ils vont soit jeter ces œufs exotiques ou tout simplement abandonner son nid et construire un nouveau nid ailleurs.

En outre, le moment de la ponte de certaines espèces est également incroyable. Les coucous parasites choisissent souvent un nid où un oiseau hôte vient de pondre ses propres œufs. En général, les œufs de coucou éclosent légèrement plus tôt que les œufs des hôtes. Une fois le premier poussin de coucou est éclos, la première action instinctive à prendre est d'expulser les œufs de son oiseau hôte en les propulsant aveuglément hors du nid, ce qui augmente la part de nourriture du poussin coucou fournie par son oiseau hôte. Des études montrent également qu'un poussin de coucou peut aussi imiter l'appel des poussins hôtes pour avoir accès à plus de nourriture. [20]

II.5.2 Cuckoo search et Levy Flights

Le vol Levy a été introduit par le mathématicien français Paul Levy en 1937. Il s'agit d'un algorithme de marche aléatoire dans lequel les étapes sont définies en termes de longueurs d'étapes qui suivent une certaine distribution de probabilité dans laquelle les directions des étapes doivent être aléatoires.

L'étude de Reynolds et Fyre [21] sur les vols Lévy indique que la plupart des animaux et des insectes explorent leurs paysages à l'aide d'une série de trajectoires rectilignes suivies d'un virage à 90°. Ce type de comportement s'appelle le modèle de recherche sans échelle intermittente de type Lévy-flight.

Le cuckoo search et à partir d'un emplacement bien connu, il générera une toute nouvelle génération à des distances réparties de manière aléatoire en fonction des vols Levy. Ensuite, la nouvelle génération sera évaluée pour sélectionner la plus prometteuse. Le processus est répété jusqu'à ce que les critères d'arrêt soient satisfaits [22].

II.5.3 Description de l'algorithme

Pour simplifier la description de la recherche de coucous, il y a trois règles idéalisées suivantes:

- Règle 1 : chaque coucou pond un œuf à la fois et le déverse dans un nid choisi au hasard.
- Règle 2 : les meilleurs nids avec des œufs de haute qualité seront conservés aux générations suivantes.
- Règle 3 : le nombre de nids hôtes disponibles est fixe et l'œuf pondu par un coucou est découvert par l'oiseau hôte avec une probabilité $p_a \in [0, 1]$. Dans ce cas, l'oiseau hôte peut jeter l'œuf ou abandonner le nid et construire un nid complètement nouveau.

Cette dernière hypothèse peut être approchée par la fraction p_a des n nids remplacés par de nouveaux nids (avec de nouvelles solutions aléatoires).

Chapitre II : Métaheuristique

De ce fait, un nid dans le cuckoo search est un individu de la population telle que le nombre de nids est fixé (égal à la taille de la population). Si le nid est abandonné, il faut le remplacer par un nouveau. Un œuf dans un nid représente une solution adoptée par un individu de la population. Chaque œuf du coucou est une nouvelle solution candidate pour une place dans la population

Pour un problème de maximisation, la qualité d'une solution peut simplement être proportionnelle à la valeur de la fonction objectif.(on dit toujours fonction objectif).

Sur la base de ces trois règles, les étapes de base de la recherche coucou peuvent être résumées sous la forme du pseudo-code suivant :

Initialiser une population de n nids d'accueil x_i ($i = 1, 2, \dots, n$)

Tant que ($t < \text{NbrMaxGénération}$) ou (critère d'arrêt) **faire**

 Créer un cuckoo (nommé i) aléatoirement par Levy Flight

 Evaluer sa qualité/fitness F_i

 Choisir un nid parmi n (nommé j) aléatoirement

Si ($F_i > F_j$) **alors**

 Remplacer j par la nouvelle solution

Fin Si

 Abandonner une fraction (p_a) des pires nids et construire des nouveaux à de nouveaux emplacements à travers Levy Flight

 Garder les meilleures solutions (ou nids avec des solutions de qualité)

 Classer les solutions et trouver le meilleur

Fin tant que

Figure II.2 : Pseudo code de Cuckoo Search

II.6 Conclusion

Dans ce chapitre, nous avons présenté les notions d'heuristique et de métaheuristique en se focalisant sur les deux sous classes des métaheuristicues, à savoir, celles à base de solution unique et celles à base de population de solution. Pour cette dernière classe, nous nous sommes focalisés sur les algorithmes basés sur l'intelligence en essaim et en particulier sur l'algorithme de la recherche coucou (cuckoo search).

Dans le dernier chapitre de ce PFE, nous allons présenter une adaptation de l'algorithme cuckoo search pour une optimisation multi-objectif dans le domaine du Cloud computing.

CHAPITRE III:

Implémentation

Chapitre III : implémentation

III.1 Introduction

L'ordonnancement des tâches dans un Cloud est un problème difficile. Ce problème est d'autant plus difficile lorsqu'il y a plusieurs facteurs à prendre en compte, à savoir, le temps d'exécution global des tâches (Makespan), le coût et la consommation d'énergie..., donc c'est un problème d'optimisation combinatoire, ou il est possible de trouver la solution optimale en utilisant des algorithmes ou des métaheuristiques simples.

Le problème d'ordonnancement des tâches est largement étudié dans de nombreux travaux. La majorité de ces travaux se sont concentrés uniquement sur l'optimisation d'une métrique de qualité de service, souvent, le temps d'exécution.

L'objectif de ce chapitre est de décrire notre contribution dans le cadre de ce PFE. Cette contribution consiste à utiliser la métaheuristique cuckoo search pour l'optimisation multi-objectifs dans le Cloud computing.

Dans ce qui suit, nous commençons par présenter les outils de simulations utilisés dans le cadre de ce PFE, à savoir, le langage java, l'IDE NetBeans et le simulateur CloudSim. Pour ce simulateur, nous nous intéressons à son architecture, ses classes fondamentales ainsi que les classes que nous avons intégrées pour la réalisation de notre travail.

Nous présentons également l'IHM que nous avons développée dans le cadre de ce PFE ainsi que les résultats obtenus de simulation.

III.2 Outils utilisés et environnement de développement

Dans ce PFE, nous avons utilisé le simulateur CloudSim (version 3.0.3) sous l'IDE NetBeans et avec le langage Java.

Pour la réalisation des différentes simulations, nous avons utilisé une machine sous Windows 7 de 64 bits, avec un processeur Intel(R) Core (TM) i5-4200U CPU @ 1.60GHz, et une capacité mémoire de 4 GB.

III.2.1 Java

Java est à la fois un langage de programmation informatique orienté objet et un environnement d'exécution portable. Il est créé par James Gosling et Patrick Naughton employés de Sun Microsystems avec le soutien de Bill Joy (cofondateur de Sun Microsystems en 1982), présenté officiellement le 23 mai 1995 au SunWorld [23].

Java est un langage compilé et interprété car le compilateur java produit un code intermédiaire qui sera interprété par une JVM, c'est la raison pour laquelle Java est un langage portable.

Tous les logiciels écrits en Java sont portables sur plusieurs systèmes d'exploitation tels que: Unix, Microsoft Windows, Mac OS ou Linux.

III.2.2 NetBeans

NetBeans est un environnement de développement intégré (EDI), placé en open source par Sun en Juin 2000. En plus de Java, NetBeans permet également de supporter différents autres

Chapitre III : implémentation

langages, comme Python, C, C++, JavaScript, XML, Ruby, PHP et HTML. Il comprend toutes les caractéristiques d'un IDE moderne (éditeur en couleur, éditeur graphique d'interfaces et de pages Web . . .).

Conçu en Java, NetBeans est disponible sous Windows, Linux, Mac OS X etc ou sous une version indépendante des systèmes d'exploitation (requérant une machine virtuelle Java).

Un environnement Java Développement Kit (JDK) est requis pour les développements en Java. NetBeans constitue par ailleurs une plateforme qui permet le développement d'applications spécifiques (bibliothèque Swing (Java)) et s'enrichit à l'aide de plugins. [24]

III.2.3 CloudSim

Le simulateur CloudSim a été développé au CLOUDS Laboratory de Melbourne en Australie [25].

Ce simulateur possède plusieurs fonctionnalités comme :

- Le support pour la modélisation et la simulation à grande échelle d'infrastructure de Cloud computing, y compris des centres de données sur un seul nœud physique.
- Une plateforme indépendante pour la modélisation des DataCenters, des brokers, de l'ordonnancement et des politiques d'allocation des ressources.

CloudSim est caractérisé par :

- La disponibilité de moteur de virtualisation, ce qui facilite la création et la gestion de services virtualisés multiple, indépendants et hébergés sur un nœud du Datacenter.
- La flexibilité pour commuter entre l'allocation en espace partagé et en temps partagé des cœurs de traitement au service virtualisés. [26]

CloudSim a deux politiques d'ordonnancement :

▪ Space shared :

Dans space shared, l'ordonnanceur donne une tâche à une machine virtuelle à un instant donné et après la terminaison de cette tâche une autre tâche est lancée sur cette machine virtuelle.

▪ Time shared :

Dans time shared, l'ordonnanceur donne toutes les tâches à une machine virtuelle en même temps. Il partage le temps entre toutes les tâches et les planifie simultanément sur la machine virtuelle.

III.3 Architecture de CloudSim

L'architecture de CloudSim est organisée sous formes de plusieurs couches. Les différentes couches sont représentées dans la figure suivante.

Chapitre III : implémentation

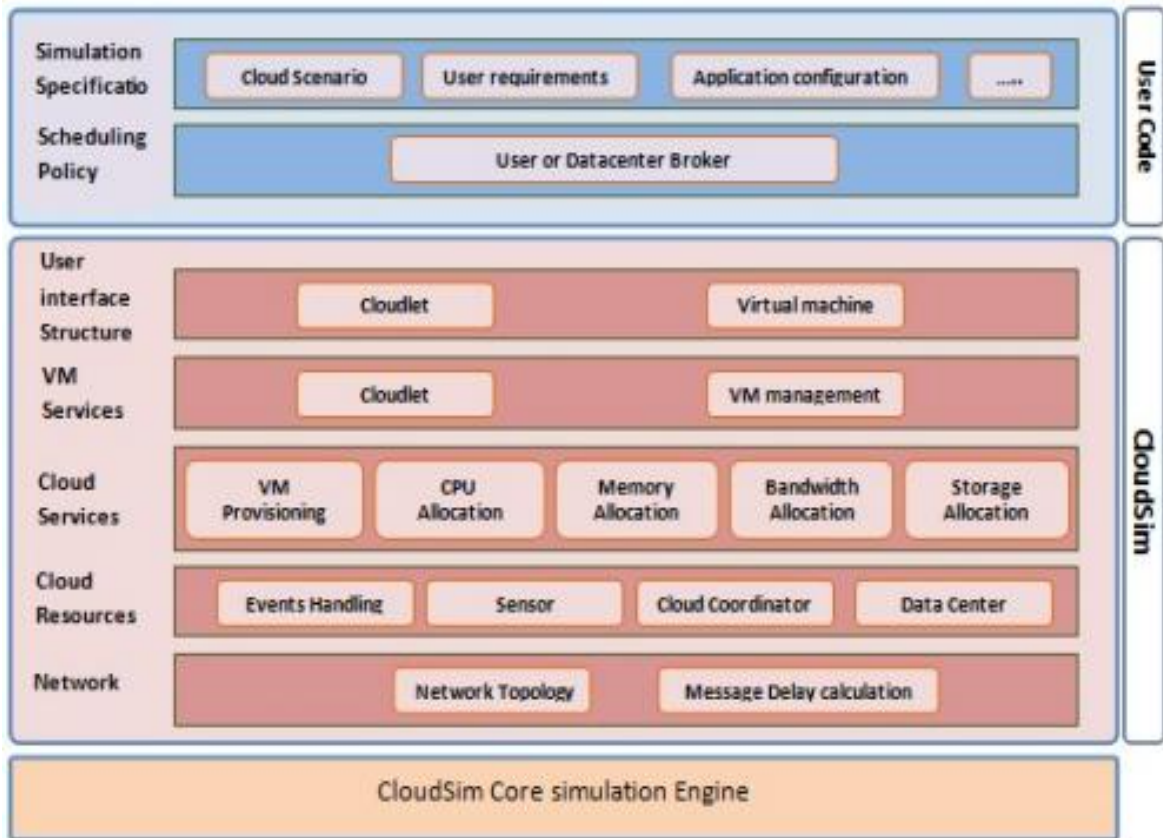


Figure III.1 : Architecture de CloudSim [27]

La première couche « User Code » contient les informations de base pour le fonctionnement des hôtes et des applications comme le nombre de Data Center, de machines physiques, de machines virtuelles et de tâches (Cloudlets).

Nous trouvons dans cette couche plusieurs paramètres liés aux :

- Caractéristiques des DataCenter : architecture, système d'exploitation, coûts de fonctionnement.
- Caractéristiques des hôtes : nombre de CPU, capacité des CPU, capacité mémoire, capacité de stockage, bande passante.
- Caractéristiques des VMs : capacité CPU, nombre de CPU, capacité mémoire, capacité de stockage.
- Caractéristiques des tâches : nombre d'instructions à exécuter, taille des fichiers d'entrée et de sortie, le choix des stratégies d'ordonnancement du broker, etc.

La deuxième couche « CloudSim » contient cinq couches comme mentionner dans la figure III.1.

Cette couche, permet d'allouer les Cloudlets aux machines virtuelles, de gérer les stratégies d'allocation des machines virtuelles et les topologies des différents réseaux, de mettre une communication possible entre différentes couches, de fournir l'interaction entre l'utilisateur et le simulateur et de fournir un support pour la modélisation des environnements dataCenters virtualisés avec des classes permettant la gestion et la configuration des machines virtuelles, mémoire, stockage, etc.

Chapitre III : implémentation

Les classes principales implémentées dans cette couches sont les suivantes: Datacenter, datacenterbroker, vm , cloudlet, host.

Datacenter: Elle permet de modéliser l'infrastructure du noyau du service (matériel, logiciel) dans le Cloud computing.

Il contient un ensemble de machines physiques appelées Hosts, et chaque Datacenter implémente un ensemble d'algorithmes pour l'allocation de la bande passante, la mémoire et le stockage aux différents hosts et machines virtuelles du Cloud.

DatacenterBroker : Elle permet de modéliser le Broker qui est le responsable de la médiation des négociations entre les utilisateurs et les prestataires de service selon les conditions de qualité de service des utilisateurs. Les négociations se font pour une allocation de ressources qui répond aux besoins de QoS des utilisateurs. Il permet aussi de déployer les tâches de service. Il déploie aussi les tâches de service à travers les Clouds.

Vm : Elle permet de modéliser une instance de machine virtuelle qui est gérée et hébergée par une machine physique. Chaque composant VM a accès à un composant qui stock plusieurs caractéristique liées à une VM telles que : la mémoire, le processeur et la capacité de stockage.

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

Host : Elle permet de modéliser une ressource physique comme le serveur de stockage ou de calcul. Elle encapsule des informations importantes telles que la quantité de mémoire et de stockage, le type de cœurs de traitement (pour représenter une machine multi-core), une politique d'allocation pour le partage de la puissance du traitement entre les machines virtuelles et les politiques d'attribution de mémoire et de bande passante aux machines virtuelles.

La dernière couche de l'architecture représente le moteur de simulation d'événements discrets javasim, qui prend plusieurs fonctionnalités de base telles que la mise en file d'attente et le traitement des événements, la création de Cloud entités système (services, hôte, Datacenter, broker, VMs)

III.4 Approche proposée

III.4.1 La fonction objectif

Dans ce PFE, nous nous sommes intéressés à l'optimisation de trois critères dans le domaine du Cloud computing, à savoir, le temps d'exécution (Makespan), le coût du traitement et la consommation d'énergie. Cette optimisation concerne l'affectation de tâches aux VMs.

Chapitre III : implémentation

Temps d'exécution :

Le temps d'exécution d'une Cloudlet représente sa durée d'exécution sur une machine virtuelle. Sachant qu'une tâche est exécutée sur un seul processeur élément (PE), nous utilisons cette formule

Temps d'exécution = Cloudletlength / VM mips.

Où :

Cloudletlength : la taille de la tâche en MIPS (million instructions par seconde).

VMmips : Vitesse de la machine virtuelle en MIPS.

Coût du traitement :

Le coût du traitement d'une cloudlet dans une machine est représenté par la formule suivante :

Coût = (Temps d'exécution * CostPerSec) + (getCloudletSize() * CostPerBw) + (getCloudletOutputSize() * CostPerBw).

Où:

CostPerSec : coût par seconde de l'exécution des instructions.

getCloudletSize() : la taille du fichier d'entrée de ce Cloudlet avant de le soumettre.

getCloudletOutputSize() : la taille de sortie de ce Cloudlet après la soumission et exécution.

CostPerBw : coût par octet de l'utilisation de la bande passante.

Consommation d'énergie :

La consommation d'énergie d'une Cloudlet représente l'énergie consommée pendant son exécution sur une machine.

Énergie = Cloudlet.getHost().getPowerModel().getPower(getUtilizationOfCpu(Temps de réponse)).

Cloudlet.getHost () : retourne le host dans le quel s'exécuté la Cloudlet.

getPowerModel().getPower() : retourne la consommation d'énergie de la cloudlet utilisant comme paramètre (getUtilizationOfCpu()) qui est le pourcentage d'utilisation de cpu de la cloudlet dans host.

Dans le cadre d'une optimisation multi-objectifs, la fonction objectif est défini comme étant la somme pondérée de ces trois critères. Les poids des différents critères seront représentés sous forme de nombres réels dans l'intervalle [0; 1] associés à chaque critère, et qui traduisent leurs importances relativement à l'exigence de l'utilisateur.

La somme de ces poids est égale à 1.

La sortie de la fonction objectif est normalisée dans l'intervalle [0; 1].

Dans ce PFE, nous avons considéré la fonction objectif suivante :

Fitness = Temps d'exécution * 0.7 + coût * 0.1 + Energie * 0.2.

Chapitre III : implémentation

III.4.2 Phase de configuration

Cette phase est consacrée à la configuration de l'environnement Cloud, à savoir le nombre des DataCenters et ses caractéristiques, le nombre des Hosts et ses caractéristiques, le nombre des VM et ses caractéristiques et enfin le nombre des Cloudlets et ses caractéristiques.

III.4.3 Phase de pré-simulation

Cette phase consiste à faire une pré-simulation afin de trouver les meilleures affectations des tâches aux machines virtuelles avant de les envoyer au Broker.

.Nous avons appliqué la métaheuristique cuckoo search pour l'affectation des tâches dans les différentes VMs. Le pseudo code de cette algorithme est présenté dans le chapitre précédent.

Le pseudo code suivant résume la phase de pré-simulation:

Pour chaque tâche (T_i) de la liste des cloudlets **faire**
 Pour chaque machine virtuelle (V_j) choisie aléatoirement **faire**
 Calculer la fonction objectif $F_j(T_i, V_j)$
 Fin pour
 Pour chaque machine virtuelle (V_k) Choisie par Levy flight (10 % des MV) **faire**
 Calculer la fonction objectif $F_k(T_i, V_k)$
 Choisir une machine virtuelle (V_j) parmi les 10 MV aléatoirement
 Si ($F_k < F_j$) **alors**
 Remplacer la machine(V_j) par la machine (V_k)
 Finsi
 Finpour
Trier les MV en fonction de F_j et trouver la meilleure solution pour la tâche (T_i)
Fin pour

III.4.4 Phase de simulation

Après avoir terminé la phase de pré-simulation, l'algorithme envoie l'affectation des tâches dans différentes machines virtuelles au broker afin de lancer la simulation et afficher les résultats contenant le temps d'exécution, le coût et la consommation d'énergie pour l'ensemble des tâches.

III.5 IHM développée

La version de CloudSim 3.0.3 n'a pas d'interface graphique, il utilise le mode console. Donc nous avons créé une IHM pour faciliter l'accès ainsi que la manipulation du simulateur.

Avant de lancer la simulation, nous devons configurer ses paramètres.

Les Figures suivantes montrent les différentes étapes pour la mise en œuvre de la simulation.

Chapitre III : implémentation

Interface Principale :

Le lancement de l'application donne droit à la page accueil suivante :



Figure III.2 : Interface principale

Configuration de Datacenter :

Pour configurer le Cloud, nous commençons par saisir les paramètres nécessaires à la topologie du réseau comme le nombre des dataCenters, le nom du DataCenter, le coût du traitement, le coût d'utilisation de la mémoire, le coût de stockage, et le coût de la bande passante comme indiquer dans la figure suivante.

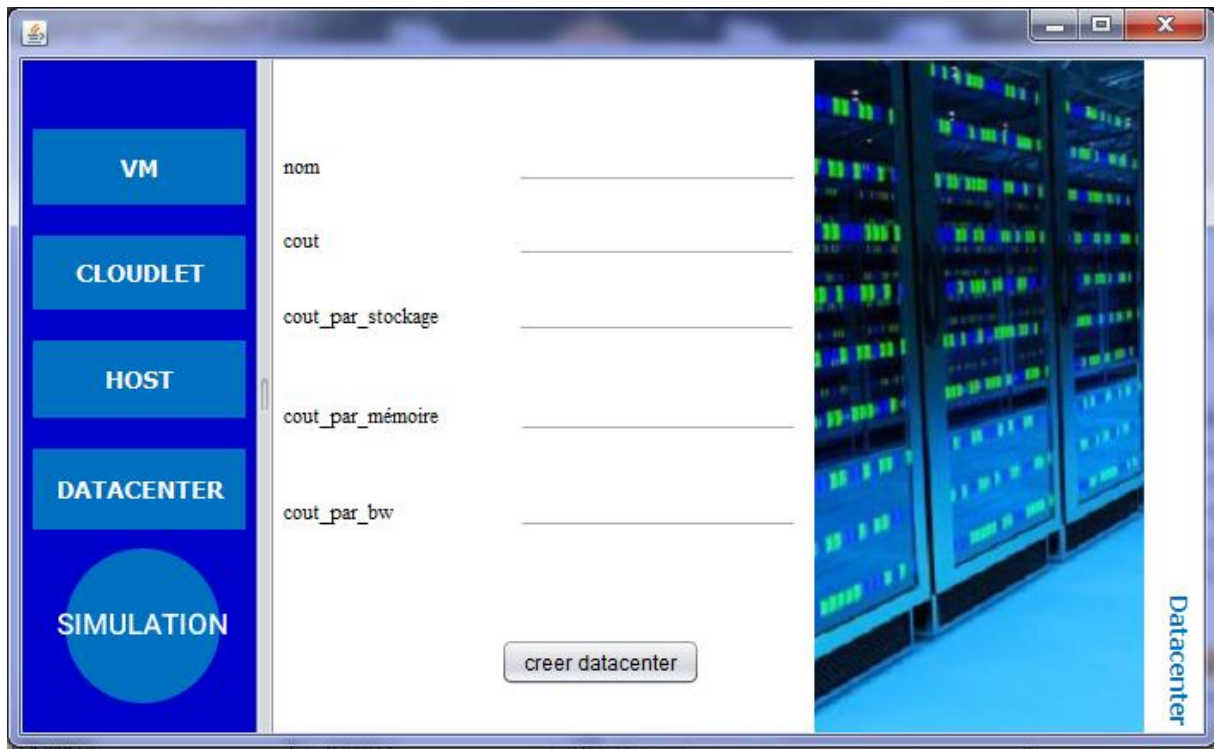
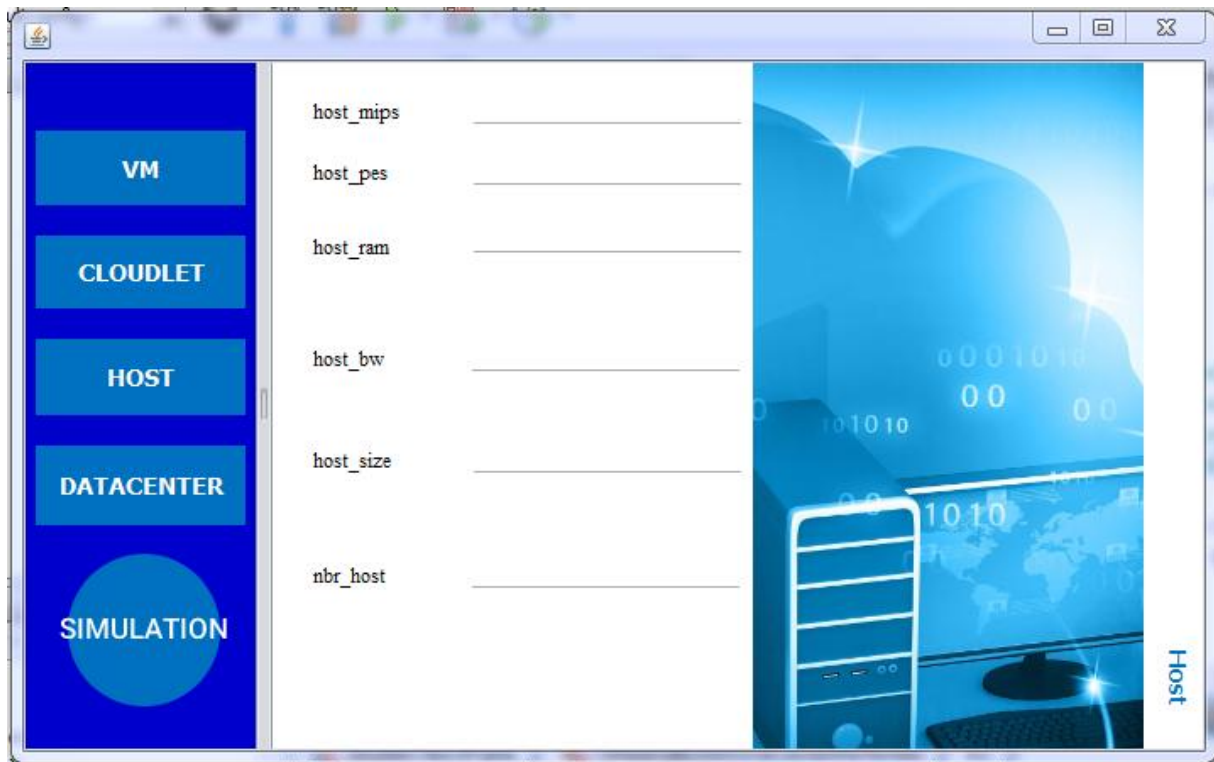


Figure III.3 : Création de Datacenter

Configuration du host :

Une fois nous avons configurés les différents DataCenter, nous devons configurer les hôtes qui se trouvent dans chaque DataCenter. L'utilisateur doit saisir les informations suivantes : le nombre de processeur (PES), la vitesse de chaque processeur (MIPS), la ram, la bande passante et aussi la capacité de stockage. On peut aussi définir le nombre d'hôte dans un DataCenter.

Chapitre III : implémentation



host_mips _____

host_pes _____

host_ram _____

host_bw _____

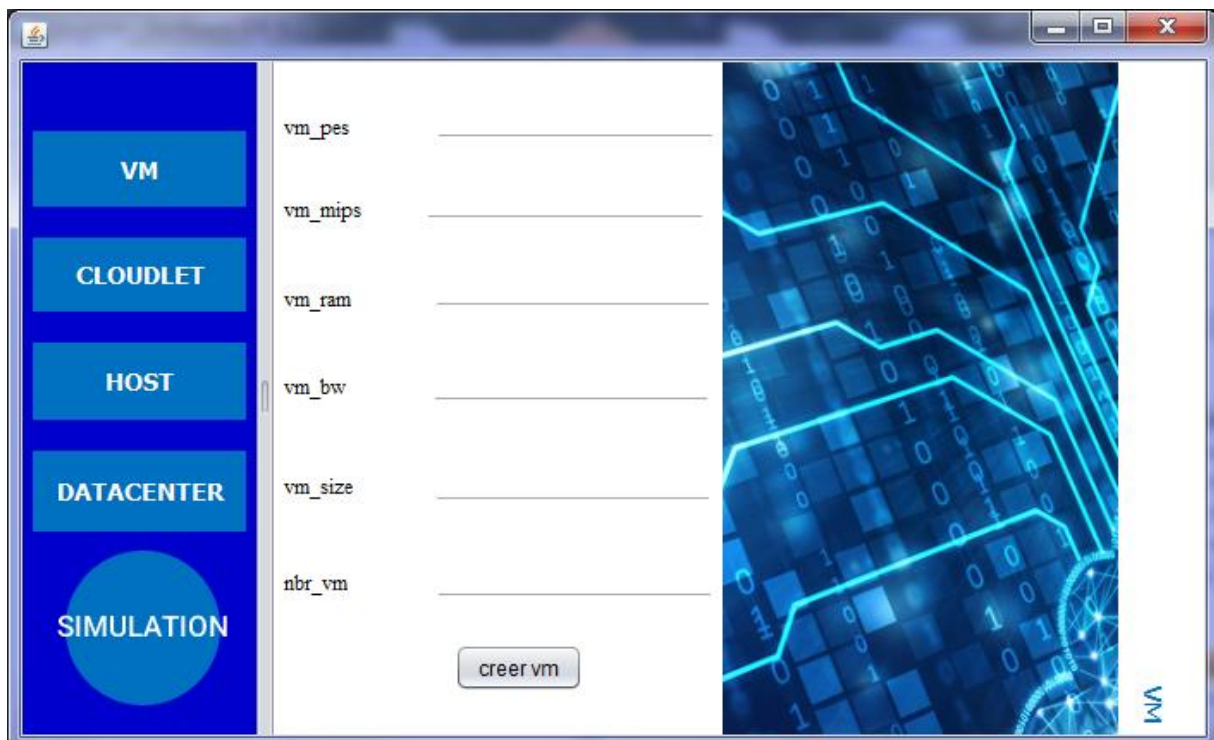
host_size _____

nbr_host _____

Figure III.4 : Création des Hosts

Configuration de la machine virtuelle :

Cette étape consiste à configurer les paramètres nécessaires à la création des machines virtuelles, de telle sorte que le VMs soit obligatoirement dans un hôte. Un hôte pouvant contenir une ou plusieurs machines virtuelles.



vm_pes _____

vm_mips _____

vm_ram _____

vm_bw _____

vm_size _____

nbr_vm _____

creer vm

Figure III.5 : Création des machines virtuelles

Chapitre III : implémentation

Configuration de Cloudlet :

Cette IHM nous permet de définir les caractéristiques de Cloudlet telle que la longueur de Cloudlet, le fichier d'entrée et de sortie et le nombre de Cloudlets.

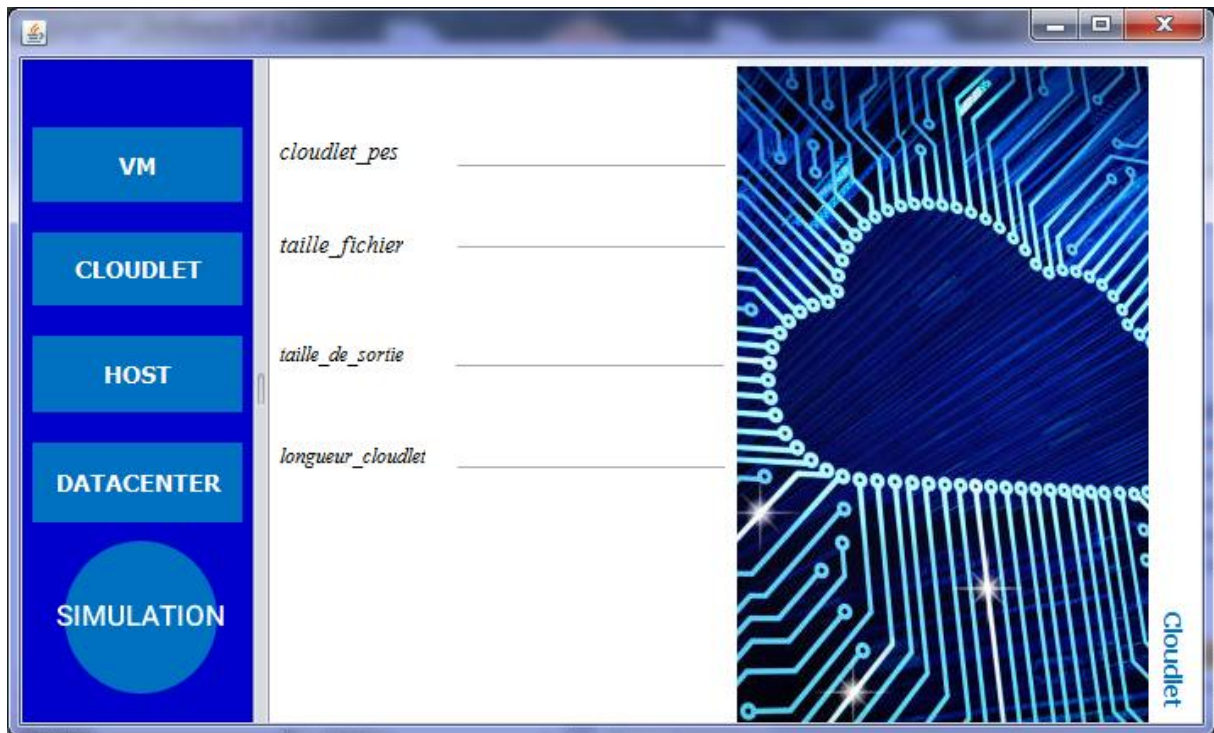


Figure III.6 : Création des Cloudlets

III.6 Résultat obtenus

Afin de valider notre approche, nous l'avons comparé avec l'algorithme RR (Round Robin). Ce dernier suit une stratégie simple et classique qui consiste à distribuer de manière équitable les tâches sur les machines virtuelles disponibles, c'est-à-dire que le nombre de tâches pour chaque machine virtuelle est à peu près le même.

Cet algorithme est implémenté dans le simulateur CloudSim.

Dans cette première simulation, nous avons mesuré le temps d'exécution, le coût et la consommation d'énergie.

Cette simulation a été réalisée avec les paramètres suivants : 4 DataCenters contenant 4 hosts chacun, pour chaque DataCenter, les hosts sont configurés comme indiqué dans le tableau III.1.

PES	Bande passante	RAM	Stockage
4	10000 Bit/s	8 GB	100 GB

Tableau III.1 : Les paramètres des hosts.

Avec une vitesse variante en MIPS (1860, 2660).

Le nombre de VM est fixé à 64, avec une vitesse variante en MIPS (100, 500, 1000 ou 1500).

Chapitre III : implémentation

Machine virtuelle	PES	Bande passante	RAM	Stockage
	1	10000 Bit/s	2 GB	20 GB

Tableau III.2 : Les paramètres des VMs

La simulation consiste à varier le nombre de Cloudlet (100, 500, 1000, 1500) avec les paramètres montrés dans le tableau suivant :

Cloudlet	Longueur	Taille du fichier d'entrée	Taille du fichier de sortie
	10000	300 byte	300 byte

Tableau III.3 : Les paramètres des Cloudlets.

Longueur : est le nombre des instructions à exécuter.

III.4.1 Comparaison par rapport au temps d'exécution

Le tableau suivant montre les résultats de comparaison entre le CS et le RR par rapport au temps d'exécution (le temps est mesuré en ms).

Tableau III.4 : Comparaison entre le CS et le RR en termes de réponse.

Approche \ Nombre de Cloudlets	CS	RR
100	20.4	20.4
500	120.4	160.4
1000	240.4	300.4
1500	300.4	300.4

La figure suivante représente les résultats obtenus.

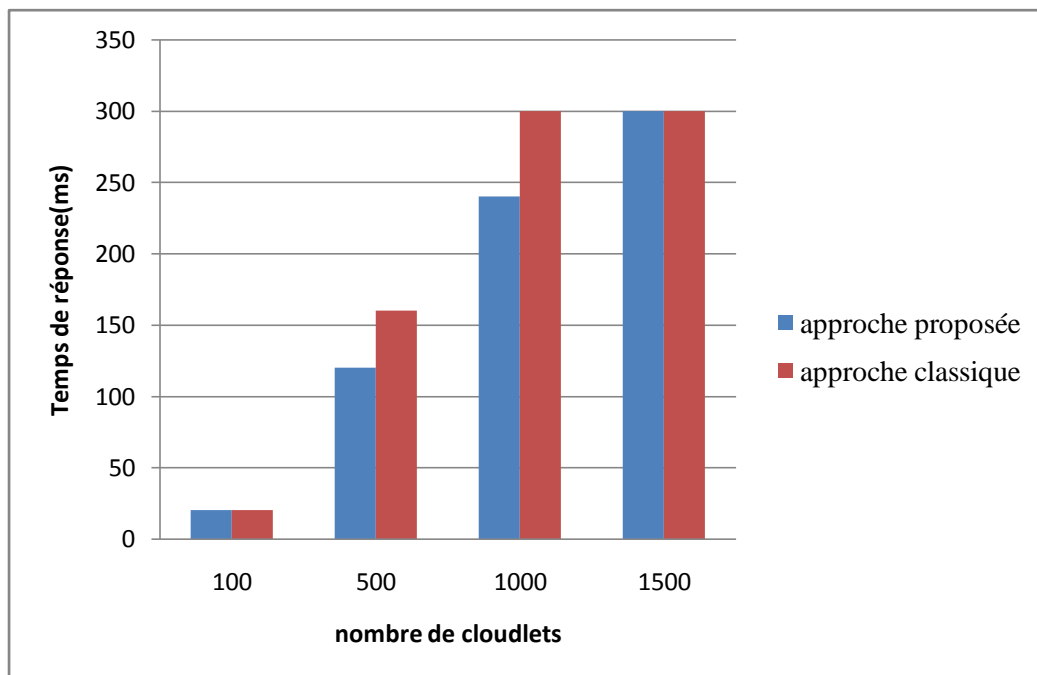


Figure III.7 : Comparaison entre le CS et le RR en termes de temps de réponse

Chapitre III : implémentation

Le nombre de cloudlets à un impact sur le temps d'exécution pour les deux approches par contre les performances du CS par rapport au RR sont meilleures, en particulier pour 500 et 1000 cloudlets.

III.4.2 Comparaison par rapport au coût

Le tableau suivant montre les résultats de comparaison entre le CS et le RR par rapport au coût.

Tableau III.5 : Comparaison entre le CS et le RR en termes de coût.

Approche \ Nombre de Cloudlets	CS	RR
100	3.61	6.1
500	144.08	179.2
1000	462.66	606.74
1500	582.98	1094.13

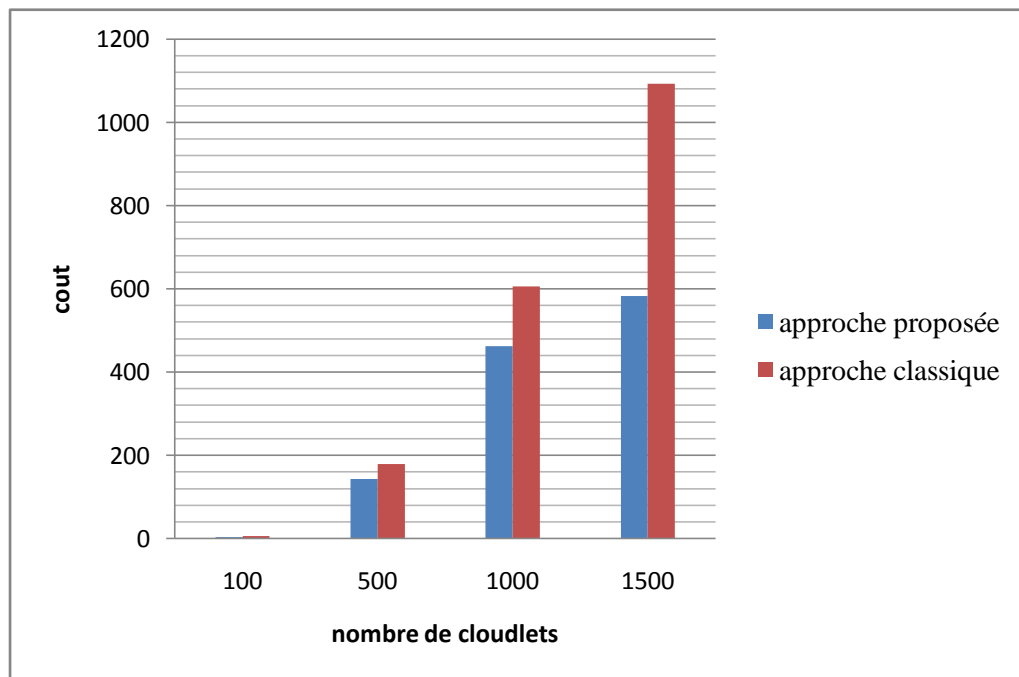


Figure III.8 : Comparaison entre le CS et le RR en termes de coût

Nous remarquons selon la figure III.8 que le CS a donné de meilleur résultat par rapport au RR en termes de coût.

III.4.2 Comparaison par rapport à l'énergie consommée

Le tableau suivant montre les résultats de comparaison entre le CS et le RR par rapport à l'énergie consommée.

Chapitre III : implémentation

Tableau III.6 : Comparaison de la consommation d'énergie entre CS et RR

Approche \ nombre de Cloudlet	CS	RR
100	24558.76	41153.04
500	919759.19	1019868.27
1000	3710125.2	3853693.6
1500	3.27E+06	6.89E+0.6

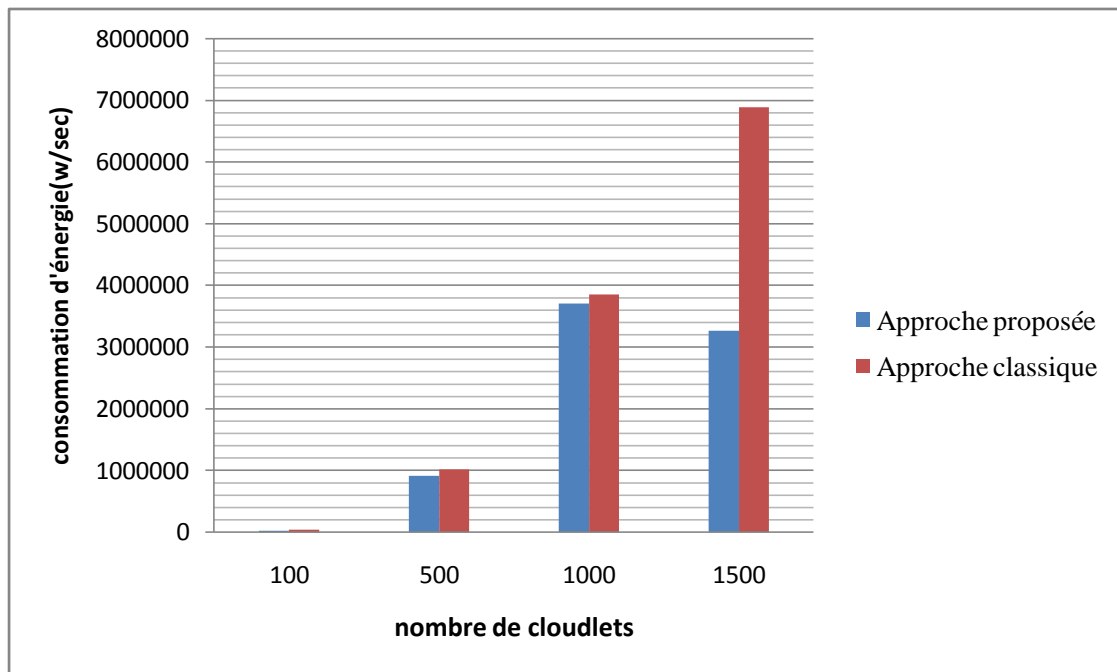


Figure III.9 : Comparaison de la consommation d'énergie entre CS et RR

Le nombre de cloudlets a un impact sur l'énergie consommée pour les deux approches par rapport aux performances du CS par rapport au RR sont meilleures, en particulier pour 1500 cloudlets.

III.7 Conclusion

Ce chapitre a été consacré à la présentation de notre contribution dans le cadre de ce PFE. Nous avons commencé par présenter les outils de simulations utilisés, à savoir, le langage JAVA, l'IDE NetBeans ainsi que le simulateur CloudSim.

Les résultats de simulation montrent l'efficacité de notre approche par rapport à l'approche classique en ce qui concerne le temps de réponse, le coût et l'énergie.

Conclusion générale

Le Cloud computing est une nouvelle technologie très tendance aujourd'hui dans le monde informatique. Il permet de fournir différents types de services offerts aux utilisateurs selon leurs besoins en termes de qualité de service.

Dans ce mémoire, nous avons proposé un algorithme pour l'optimisation de plusieurs objectifs dans le Cloud computing. Cet algorithme est basé sur la métaheuristique cuckoo search permettant de trouver la solution optimale en un temps raisonnable. La problématique traitée concerne l'affectation des tâches dans différentes machines virtuelles.

Pour implémenter l'approche proposée, nous avons utilisé le simulateur CloudSim qui a été choisi pour ses qualités de modélisation.

Les performances de notre algorithme sont évaluées à l'aide d'un ensemble de contraintes telles que : le temps de réponse, le cout et la consommation d'énergie.

Pour la validation de l'approche proposée dans le cadre de ce PFE, nous l'avons comparé avec un algorithme classique déjà implémenté dans CloudSim qui est le Round Robin.

Les simulations réalisées avec CloudSim montrent clairement l'efficacité de l'approche proposée par rapport à l'algorithme Round Robin. La métaheuristique cuckoo search a permis d'optimiser les trois critères utilisés pour les comparaisons à cause surtout de ses bonnes performances en termes d'exploitation de tout l'espace de recherche avec Levy Flight.

Il serait intéressant dans la suite de ce travail de l'étendre à d'autres contraintes tels que la fiabilité, l'équilibrage de charge et la tolérance aux pannes. Appliquer d'autres métaheuristiques serait aussi possible afin de se comparer avec le cuckoo search. Dans ce cas, des algorithmes comme l'algorithme des lucioles, les algorithmes de colonies d'abeilles ou l'algorithme de pollinisation de fleur seront des bons candidats dans ce contexte afin d'évaluer leurs performances.

Bibliographie

- [1] Les avantages du Cloud computing. Disponible sur: <https://missarte.wordpress.com/les-avantages-du-cloud-computing/>. Consulté le 26/04/2019.
- [2] Les inconvénient du Cloud computing. Disponible sur: <https://missarte.wordpress.com/les-inconvenients-du-cloud-computing/>. Consulté le 26/04/2019.
- [3] Qu'est-ce que la virtualisation? Disponible sur: <https://www.supinfo.com/articles/single/5969-qu-est-ce-que-virtualisation>. Consulté le 19/04/2019.
- [4] VIRTUALISATION – Définition – AB technologies. Disponible sur: <https://abtec.fr/virtualisation-definition/>. Consulté le 19/04/2019.
- [5] Définition Data Center : qu'est-ce qu'un centre de données ? Disponible sur: <https://www.lebigdata.fr/definition-data-center-centre-donnees>. Consulté le 17/04/2019.
- [6] Que signifie Machine Virtuelle (VM)? Disponible sur: <https://www.lemagit.fr/definition/Machine-Virtuelle-VM>. Consulté le 26/04/2019.
- [7] Esma Insaf Djebbar. « Optimisation d'ordonnancement et d'allocation de ressources dans les clouds computing ». Thèse de doctorat en informatique. Université d'Oran 1 Ahmed Ben Bella. 2016.
- [8] Elwessabi Ali Ahmed Yahya. Une approche basée agent mobile pour le cloud computing. Magister en Informatique. Université HADJ LAKHDAR – BATNA. 2014.
- [9] Meslem yamina, Debbas soumia ismahèn. Ordonnancement et réplification de données dans le Cloud Computing. Master en informatique. Université Dr Tahar Moulay SAIDA. 2017.
- [10] Slagle, James R. Artificial intelligence: The heuristic programming approach. McGraw-Hill, 1971.
- [11] Alaoui Abdiya, Application des techniques des métaheuristiques pour l'optimisation de la tâche de la classification de la fouille de données. Thèse de doctorat en informatique, Université des sciences et de la technologie d'Oran Mohamed Boudiaf, 2012.
- [12] Gherboudj, Amira. Méthodes de résolution de problèmes difficiles académiques. Thèse de doctorat en informatique. Université de Constantine 2. 2013.
- [13] Emile Aarts, Emile HL Aarts et Jan Karel Lenstra. Local search in combinatorial optimization. Princeton University Press, 2003 (cf. p. 42).

- [14] Scott Kirkpatrick. "Optimization by simulated annealing : Quantitative studies". In : Journal of statistical physics 34.5-6 (1984), p. 975-986 (cf. p. 43).
- [15] Fred Glover. "Future paths for integer programming and links to artificial intelligence". In : Computers & operations research 13.5 (1986), p. 533-549 (cf. p. 43).
- [16] Ilhem boussaïd, perfectionnement de métaheuristiques pour l'optimisation continue, Thèse de doctorat en informatique, Université des sciences et de la technologie houari Boumediene, 2013.
- [17] M. Dorigo, Optimization, Learning and Natural Algorithms, Thèse de doctorat, Politecnico di Milano, Italie, 1992.
- [18] Yang, Xin-She. A new metaheuristic bat-inspired algorithm." Nature inspired cooperative strategies for optimization (NCSO 2010). Springer, Berlin, Heidelberg, 2010. 65-74.
- [19] Rekaby, Amr. Directed artificial bat algorithm (daba)-A new bio-inspired algorithm. 2013 International Conference on Advances in Computing, Communications and Informatics (ICACCI). IEEE, 2013.
- [20] Yang, Xin-She, and Suash Deb. "Cuckoo search via Lévy flights." 2009 World Congress on Nature & Biologically Inspired Computing (NaBIC). IEEE, 2009.
- [21] Reynolds, Andy M., and Mark A. Frye. "Free-flight odor tracking in *Drosophila* is consistent with an optimal intermittent scale-free search." PloS one 2.4 (2007): e354.
- [22] Kamaruzaman, Anis Farhan, et al. "Levy flight algorithm for optimization problems-a literature review." Applied Mechanics and Materials. Vol. 421. Trans Tech Publications, 2013.
- [23] Deghbouch Hicham Nedjadi Lahcene. Une stratégie de réplication de données pour gérer la tolérance aux pannes et l'équilibrage de charge dans le cloud computing. MASTER OPTION : Réseaux Informatique et Systèmes Répartis. UNIVERSITE Dr. TAHAR MOULAY SAIDA 2018.
- [24] BECHAR Anissa SADELLI Salim. Conception et Implémentation d'une Application Mobile pour les Services d'Aide aux Etudiants sous Android. Master en Informatique. Université A. Mira- Béjaïa 2013.
- [25] CloudSim : <http://www.cloudbus.org/cloudsim/> consulté le 03/03/2019.
- [26] Bouamama Samah. Gestion des ressources dans le cloud computing a base des modèles économiques .magister en informatique. Université d'Oran 2010.
- [27] Architecture de cloudsim : <https://graal.ens-lyon.fr/~ecaron/m2rts/2015/blogbazm/> consulté le 02/06/2019.

Résumé

Le Cloud computing est une technologie qui est basée sur une meilleure qualité de service et une forte disponibilité de l'infrastructure permettant l'accès à des données et à des services disponibles sur des serveurs distants. Il offre en particulier aux sociétés l'opportunité de réduire les coûts d'exploitation des logiciels par leurs utilisations directement en ligne. L'objectif de ce travail est de réaliser une optimisation multi-objectifs dans le Cloud computing en utilisant la métaheuristique cuckoo search afin de satisfaire les objectifs suivants: la réduction du temps d'exécution, la réduction de la consommation d'énergie et la réduction du cout. Le problème traité concerne l'affectation des tâches dans différentes machines virtuelles. L'approche proposée a donné de meilleures performances par rapport à l'approche classique (Round Robin). Le simulateur CloudSim a été choisi pour la réalisation des différentes simulations et le langage Java a été utilisé pour la réalisation de l'IHM permettant ainsi les interactions entre l'utilisateur et le simulateur.

Mots clés: Cloud computing, CloudSim, Cuckoo Search, Round Robin.

Abstract

Cloud computing is a technology which is based on a better quality of service and a high availability of the infrastructure allowing data access and services available on remote servers. In particular, it offers companies the opportunity to reduce software operating costs by using them directly online. The objective of this work is to achieve a multi-objective optimization in Cloud computing using cuckoo search metaheuristics to satisfy the following objectives: the reduction of execution time, the reduction of energy consumption and the reduction of cost. The problem dealt with is the assignment of tasks in different virtual machines. The proposed approach gave better performance compared to the classical approach (Round Robin). The CloudSim simulator was chosen for the realization of the different simulations and the Java language was used for the realization of the HMI thus allowing the interactions between the user and the simulator.

Key words: Cloud computing, CloudSim, Cuckoo Search, Round Robin.

ملخص

الحوسبة السحابية هي تقنية تعتمد على أفضل جودة للخدمة و توافر كبير للبنية التحتية مما يتيح الوصول الى البيانات و الخدمات المتوفرة على الخوادم البعيدة. و يوفر للشركات فرصة لخفض تكاليف تشغيل البرامج عن طريق استخدامها مباشرة عبر الانترنت.

الهدف من هذا العمل هو تحسين خدمات متعددة الأهداف في الحوسبة السحابية باستخدام Métaheuristique Cuckoo Search

لتحقيق الاهداف التالية تقليل وقت التنفيذ، تقليل استهلاك الطاقة و تقليل التكلفة. المشكلة التي يتم التعامل معها هي تعيين المهام في اجهزة افتراضية مختلفة.

اعطى النهج المقترح أداء أفضل مقارنة مع النهج الكلاسيكي (Round Robin) .

تم استخدام محاكي CloudSim لتحقيق عمليات المحاكاة المختلفة و تم استخدام لغة البرمجة Java لتحقيق IHM مما سمح

بالتفاعلات بين المستخدم و المحاكي

الكلمات الرئيسية: Cloud computing, CloudSim, Cuckoo Search, Round Robin: