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 Distribués (RSD)

Thème

Ordonnancement des tâches dans le Cloud computing avec SFLA : synthèse de quelques méthodes d'aide multicritères à la décision

Réalisé par :

-GHOMARI Mohammed El-Amine.

Présenté le 25 Juin 2023 devant le jury composé de :

- *Mr. BENMOUNA Youcef (*Président*).*
- *Mr. BENTAALLAH Amine (*Examinateur*).*
- *Mr. BENMAMMAR Badr (Encadrant).*
- *Mr. MALTI Arslan Nedhir (Co-Encadrant).*

Année Universitaire: 2022-2023

Remerciement

Tout d'abord, je remercie Allah Tout-Puissant de m'avoir donné la force de surmonter toutes les difficultés.

"الحمد لله والشكر لله "

Je tiens à remercier mon encadrant, Mr. BENMAMMAR Badr et mon co-encadrant, Mr. MALTI Arslan Nedhir. Je les remercie de m'avoir encadré, guidé, aidé et conseillé.

Je tiens également à remercier les membres du jury, Mr. BENMOUNA Youcef et Mr. BENTAALLAH Amine, qui ont eu la gentillesse d'examiner et de juger ce travail.

Enfin, je réserve des remerciements particuliers à mes chers parents pour leurs encouragements et leur soutien tout au long de ma vie, qu'Allah les protège.

Merci à tous...

Dédicaces

Je dédie ce modeste travail à :

Mes chers parents.
Mon frère.
Toute ma famille.
Mes amis et camarades.
Tous ceux qui ont cru en moi et ont prié pour mon succès.

Mohammed EbAmine Ghomari

Résumé

Le Cloud computing est la fourniture de différents services informatiques à la demande sur Internet. Dans ce contexte, une prise de décision multicritères pour l'ordonnancement des tâches est nécessaire pour déterminer la meilleure alternative parmi de nombreuses possibilités. Dans ce travail, nous avons évalué la métaheuristique SFLA en utilisant le simulateur CloudSim dans un contexte multi-objectif tenant compte de trois paramètres qui sont le makespan, le coût et l'utilisation des ressources. En plus de la somme pondérée, trois méthodes d'aide multicritères à la décision qui sont MARCOS, MAIRCA et EAMR ont été utilisées pour l'évaluation des individus. Les résultats obtenus sont très satisfaisants.

Mots clés : Cloud computing, SFLA, CloudSim, MARCOS, MAIRCA, EAMR.

Abstract

Cloud computing is the provision of various IT services on demand over the Internet. In this context, multi-criteria decision making for task scheduling is necessary to determine the best alternative among many possibilities. In this work, we evaluated the SFLA metaheuristics using the CloudSim simulator in a multi-objective context taking into account three parameters: makespan, cost and resource utilization. In addition to the weighted sum, three multi-criteria decision support methods, MARCOS, MAIRCA and EAMR, were used for the assessment of individuals. The results obtained are very satisfactory.

Keywords: Cloud computing, SFLA, CloudSim, MARCOS, MAIRCA, EAMR.

ملخص

الحوسبة السحابية هي توفير خدمات تكنولوجيا المعلومات المختلفة عند الطلب عبر الإنترنت. وفي هذا السياق، فإن اتخاذ قرارات متعددة المعايير لجدولة المهام أمر ضروري لتحديد أفضل بديل من بين إمكانيات كثيرة. في هذا العمل، قمنا بتقييم SFLA باستخدام محاكي CloudSim في سياق متعدد الأهداف مع الأخذ في الاعتبار ثلاثة معايير: وقت التنفيذ والتكلفة واستخدام الموارد. وبالإضافة إلى المتوسط المرجح، استُخدمت ثلاث طرائق متعددة المعايير لدعم اتخاذ القرارات، هي MARCOS، وMAIRCA، وEAMR، لتقييم الأفراد. النتائج التي تم الحصول عليها مرضية للغاية.

الكلمات الرئيسية: الحوسبة السحابية,SFLA, CloudSim, MARCOS, MAIRCA, EAMR .

Table des matières

Liste des figures

Liste des tableaux

Liste des abréviations

Acronyme	Signification
AWS	**A**mazone **W**eb **S**ervices
BI	**B**izness **I**ntelligence
CPU	**C**entral **P**rocessing **U**nit
CSP	**C**loud **S**ervice **P**rovider
EAMR	**E**valuation by an **A**rea-based **M**ethod of **R**anking
ECS	**E**lastic **C**ontainer **S**ervice
EFS	**E**lastic **F**ile **S**ystem
EKS	**E**lastic **K**ubernetes **S**ervice
GCP	**G**oogle **C**loud **P**latform
GPU	**G**raphics **P**rocessing **U**nit
MAIRCA	**M**ulti **A**ttributive **I**deal-**R**eal **C**omparative **A**nalysis
MARCOS	**M**easurement of **A**lternatives and **R**anking according to **C**ompromise **S**olution
MCDM	**M**ulti-**C**riteria **D**ecision-**M**aking
MIPS	**M**illion **I**nstructions **P**er **S**econd
OS	**O**perating **S**ystem
QoS	**Q**uality **o**f **S**ervices
RAM	**R**andom **A**ccess **M**emory
SFLA	**S**huffled **F**rog **L**eaping **A**lgorithm
S3	**S**imple **S**torage **S**ervice
TOPSIS	**T**echnic for **O**rder Performance by **S**imilarity **to** **I**deal **S**olution
VM	**V**irtuel **M**achine
VPN	**V**irtual **P**rivate **N**etwork
ZFS	**Z**ettabyte **F**ile **S**ystem

Introduction générale

Le Cloud computing est une technologie qui gagne en popularité en tant que nouvelle façon d'utiliser de manière transparente et efficace les services à la demande, tels que le calcul, le stockage et les ressources réseau. Les grands clients, tels que les entreprises ou les utilisateurs finaux, recherchent des ressources pour répondre à leurs besoins dans cet environnement dynamique.

Dans le domaine du Cloud computing, l'ordonnancement des tâches est devenue un sujet de recherche essentiel. Il vise à attribuer efficacement des tâches à des ressources disponibles, telles que des machines virtuelles, afin d'optimiser des objectifs spécifiques, tels que la minimisation du temps total de traitement ou la réduction des coûts ou la maximisation du taux d'utilisation des ressources.

L'ordonnancement des tâches est cependant un problème NP-complet. En conséquence, en raison de la nature combinatoire du problème, il n'existe pas d'algorithme efficace pour résoudre tous les cas en un temps polynomial. De nombreux chercheurs dans ce contexte se sont penchés sur ce problème et ont proposé diverses solutions. Dans notre PFE, On s'intéresse à l'utilisation de la métaheuristique SFLA pour l'ordonnancement des tâches dans le Cloud computing.

Pour l'évaluation de la fonction objectif, nous avons utilisé quelques méthodes d'aide à la décision multicritères afin d'évaluer trois métriques de QoS qui sont le makespan, le coût et le taux d'utilisation des ressources. Ces méthodes transforment le problème de l'ordonnancement des tâches multi-objectif en un problème mono-objectif.

Le simulateur CloudSim a été utilisé dans le cadre de ce PFE pour mener différentes simulations, et les résultats obtenus ont été satisfaisants.

Le reste de ce manuscrit est structuré en trois chapitres.

Le premier chapitre est consacré à la présentation du Cloud computing, qui constitue le contexte général de notre étude. Nous y abordons les principes fondamentaux et les modèles de classification du Cloud computing, ainsi que son architecture et sa sécurité.

Le deuxième chapitre se concentre sur la présentation des métaheuristiques, en mettant particulièrement l'accent sur l'algorithme SFLA que nous avons utilisé dans le cadre de notre

travail. Le chapitre présente également trois méthodes d’aide à la décision multicritères, à savoir : MARCOS, MAIRCA et EAMR.

Enfin, le troisième et dernier chapitre est dédié à notre contribution dans le cadre de ce projet de fin d'études. Nous présentons tout d'abord les outils de simulation et de développement que nous avons utilisé. Ensuite, nous décrivons en détail l'interface utilisateur que nous avons développée pour faciliter l'interaction avec notre système. Enfin, nous présentons les résultats des différentes simulations que nous avons réalisées, en mettant l'accent sur la comparaison entre les méthodes multicritères, en termes de makespan, de coût et de taux d’utilisation des ressources.

Chapitre 1 : Cloud computing

1.1 Introduction

Le Cloud computing est une technologie qui permet d'accéder à des services informatiques à distance via Internet. Ces services incluent le stockage de données, le traitement et l'analyse de données, le développement et l'exécution d'applications. En utilisant le Cloud computing, les utilisateurs peuvent éviter les coûts et la complexité de la gestion de leur propre infrastructure informatique, et peuvent plutôt louer des services à des fournisseurs de Cloud qui gèrent l'infrastructure à leur place.

L'objectif de chapitre est de présenter le Cloud computing. Nous allons aborder la classification des modèles de Cloud en couvrant l'aspect de la virtualisation. Nous expliquerons les principes fondamentaux du Cloud computing, décrirons les différents types de fournisseurs de services et parviendrons à une conclusion générale sur cette technologie.

1.2 Définition

Le Cloud computing est un moyen de fournir des services informatiques (Hardware et Software) via Internet. Ces services sont disponibles partout avec une connexion Internet et sont généralement fournis par des entreprises tierces qui gèrent l'infrastructure sous-jacente. [1]

1.3 Principes du Cloud computing

Les cinq principes du Cloud computing peuvent se résumer comme suit [2] :

- **La disponibilité :** le Cloud computing permet aux utilisateurs abonnés d'accéder à des ressources informatiques, telles que des serveurs, des espaces de stockage, des applications et des outils de développement, via Internet. Les fournisseurs de Cloud mettent à disposition ces ressources de manière dynamique, ce qui signifie que les utilisateurs peuvent augmenter ou diminuer leurs ressources en fonction de leurs besoins. Cela permet aux utilisateurs de payer uniquement pour ce qu'ils utilisent, évitant ainsi les coûts d'investissement initiaux et les coûts liés à la gestion de l'infrastructure.
- **La virtualisation** : est une technique qui permet de diviser un serveur physique en plusieurs machines virtuelles (VM). Chaque VM peut fonctionner de manière autonome avec son propre système d'exploitation, ses propres applications et ses propres ressources. La virtualisation permet de maximiser l'utilisation du matériel en permettant à plusieurs

machines virtuelles de fonctionner sur un seul serveur physique, ce qui réduit le nombre de serveurs nécessaires et les coûts associés.

- **La mise à l'échelle** : est une fonctionnalité clé du Cloud computing qui permet aux utilisateurs d'ajuster leurs ressources informatiques en fonction de leurs besoins. Par exemple, si un utilisateur a besoin de plus d'espace disque ou de bande passante, il peut facilement ajouter ces ressources à la demande en quelques clics de souris. Cela permet aux utilisateurs de gérer efficacement les pics de trafic et les variations de la demande sans avoir à surdimensionné leur infrastructure.
- **L'automatisation :** est une fonctionnalité clé du Cloud computing qui permet aux utilisateurs de gérer efficacement leurs ressources informatiques sans intervention manuelle. Les utilisateurs peuvent automatiser la construction, le déploiement, la configuration, la fourniture et la migration de leurs applications et de leurs données, ce qui réduit les erreurs humaines et accélère les processus informatiques.
- **La facturation** : est un élément clé du modèle commercial du Cloud computing. Les utilisateurs sont facturés en fonction de leur utilisation des ressources informatiques, ce qui signifie qu'ils ne paient que pour ce qu'ils utilisent. Les fournisseurs de Cloud peuvent facturer les utilisateurs à l'heure, à la journée, au mois ou à l'année, selon les besoins de l'utilisateur. Cela permet aux utilisateurs d'optimiser leurs coûts en payant uniquement pour ce qu'ils utilisent, plutôt que de devoir investir dans une infrastructure informatique coûteuse et difficile à gérer.

Ces principes permettent au Cloud computing d'apporter plus d'économies, d'automatisation et de flexibilité.

1.4 Classification du Cloud computing

L'institut national des normes et de la technologie a fondé une simple classification de Cloud Computing représenté dans [3] :

1.4.1 Modèles de service

Selon la nature de service qu'ils offrent aux usagers. Il existe trois types de fournisseurs de Cloud computing [4] :

- **IaaS** (Infrastructure as a service) : son principe est de fournir aux utilisateurs l'accès aux applications exécutées sur le Cloud Computing, ainsi que la possibilité de les utiliser depuis

différents périphériques grâce à une interface client, telle qu'un navigateur web ou une interface de programmation.

- **PaaS** (Platform as a service) : son objectif est de permettre aux utilisateurs de déployer des applications qu'ils ont créées eux-mêmes ou qu'ils ont acquises en utilisant des langages de programmation, des bibliothèques, des services et des outils pris en charge par le fournisseur. Il offre également un contrôle sur les applications déployées, ainsi que la possibilité de configurer les paramètres de l'environnement d'hébergement des applications.
- **SaaS** (Software as a service) : l'objectif est de fournir aux utilisateurs des ressources informatiques fondamentales telles que le traitement, le stockage, afin qu'ils puissent déployer et exécuter des logiciels de leur choix, y compris des systèmes et des applications d'exploitation.

La figure 1.1 montre la hiérarchie des types de services fournis dans le Cloud computing avec quelques exemples pour chaque type.

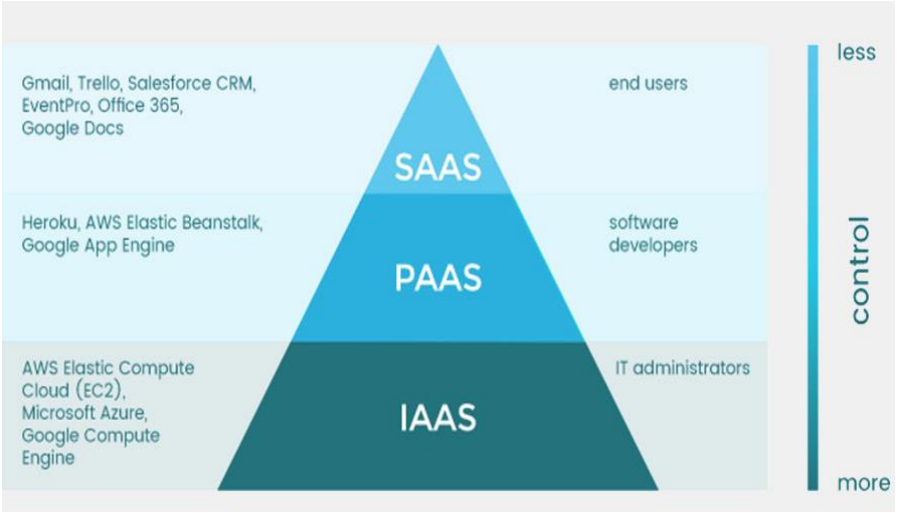

Figure 1.1 : les types de services [5].

1.4.2 Modèles de déploiement

Il s'agit de la classification du Cloud en fonction du type de client à qui est fourni le service. Il existe 4 modèles de déploiement de Cloud computing [6] :

- **Cloud public** : il consiste en une utilisation ouverte par le grand public.
- **Cloud privé** : il consiste à l'utilisation exclusive de différentes ressources par une seule organisation comprenant plusieurs consommateurs.
- **Cloud communautaire** : il s'agit d'une utilisation exclusive par une communauté de consommateurs ou d'organisations ayant des préoccupations communes (par exemple, le Cloud communautaire Canadien qui donnera aux hôpitaux l'accès à une application de stockage partagé).
- **Cloud hybride** : l'infrastructure d'un Cloud hybride se compose de deux ou plusieurs Cloud infrastructures (public, privé ou communautaire) qui sont liées par une technologie qui permet la portabilité des données.

Le tableau 1.1 montre une comparaison entre les types de déploiement en termes d'évolutivité, de fiabilité, de sécurité, de performances et de coût. De plus, quelques exemples sont présentés pour chaque type.

Paramètre	Cloud public	Cloud privé	Cloud communautaire	Cloud hybride
L'évolutivité	Plus élevée	Limité	Limité	Plus élevée
Fiabilité	Modérée	Plus élevée	Plus élevée	Moyenne à élevée
Sécurité	Dépend du fournisseur de service	Haut degré de sécurité	Sécurisé	sécurisé
Performance	Faible à moyenne	Bien	Très bien	Bien
Coût	Moins cher	Coût élevé	Cher	Cher
Exemples	AWS, IBM, Bluemix, Google Cloud, Windows Azure	Oracle, IBM, VMware, Amazon Virtual Private Cloud, Google Virtual Private Cloud	AWS Outposts, Azure Stack, Azure Arc, Google Anthos and VMware Cloud on AWS	Solas Community Cloud

Tableau 1.1 : comparaison entre Cloud public, privé, hybride et communautaire[7].

1.5 L'architecture du Cloud computing

L'architecture du Cloud computing se compose de deux parties : front end & back end.

- **Front end :** cette partie est spécifiée pour le client à travers lequel les interfaces utilisateur et les applications sont affichées, et l'accès et l'utilisation sont fournis via les différents serveurs web et les différents appareils disponibles.
- **Back end** : est destinée au fournisseur de service pour le management des différentes ressources du Cloud en général.

Une architecture du Cloud computing efficace doit inclure les composants suivants : l'infrastructure client, des applications, des services, l'environnement d'exécution, le stockage de données, le management, la sécurité et l'internet.

La figure 1.2 montre l'architecture du Cloud computing.

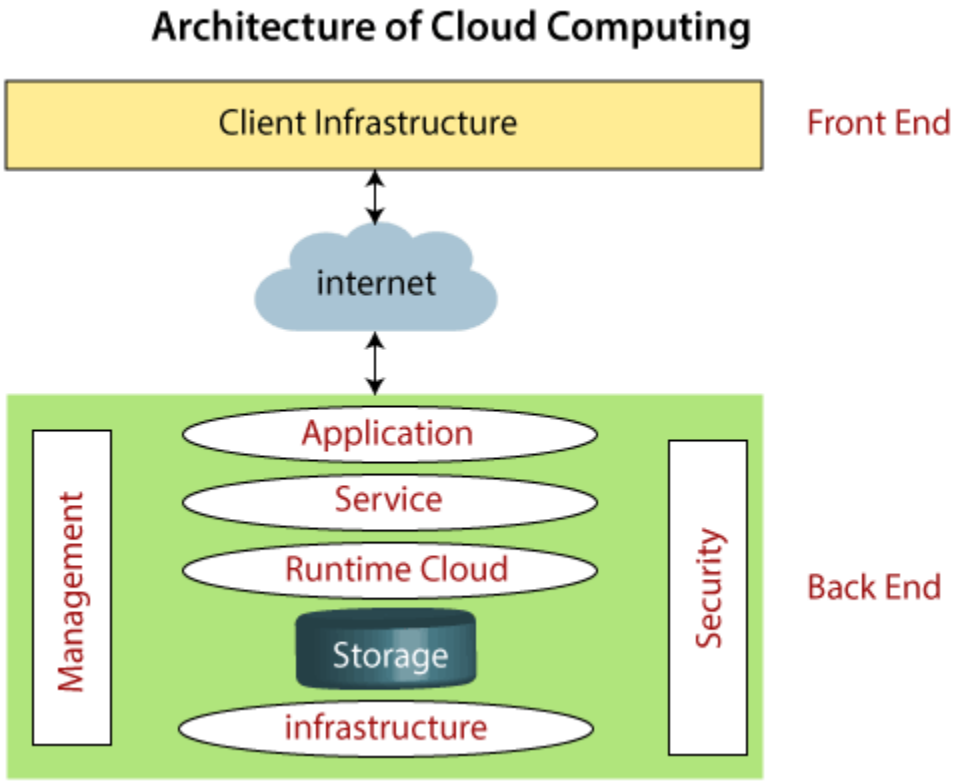

Figure 1.2 : l'architecture du Cloud Computing [8].

1.6 La virtualisation

La virtualisation est l'une de technologies connexes les plus utilisées dans le Cloud computing. Elle permet la segmentation d'un matériel physique en machines virtuelles. C'est ce qui crée de multiples taches et usages dans un même appareil selon différents systèmes d'exploitation [9].

La virtualisation est devenue l'une des technologies les plus demandées dans le monde IT en raison de ses avantages dont certains seront mentionnés ci-dessous [9].

- Réduire le cout du matériel : au lieu d'acheter de nombreuses machines comme des serveurs ou des centres de données, il suffit d'acheter une seule machine avec virtualisation.
- L'optimisation de ressources : qui permet de réduire et de facilité l'administration du système avec une efficacité plus élevée de management.
- L'utilisation optimale et maximale du matériel : elle permet d'augmenter l'utilisation de CPU de 5-15% à 60-80%.

1.6.1 L'importance de la virtualisation dans le Cloud computing

La virtualisation offre les avantages suivants pour le Cloud computing :

- La mise à l'échelle rapide des ressources.
- Elle rend le processus du Cloud computing plus facile et efficace.
- La virtualisation assure le concept de facturation « ne payez que ce que vous utilisez ».

1.6.2 Types de virtualisation

Il existe 6 types de virtualisation [10, 11] :

- **Virtualisation des applications :** elle permet au client d'avoir un accès à distance au diverses applications installées sur un serveur local via des instances d'applications sans avoir les installer localement dans la machine de l'utilisateur final. Elle permet également d'exécuter plusieurs fenêtres d'applications sans avoir un matériel de haute performance (CPU, GPU, RAM et disque dur plus élevés) car le traitement du processus est effectué dans le Cloud en temps réel.
- **Virtualisation du réseau :** elle permet de contrôler et d'exécuter les différents réseaux virtuels, chacun avec des plans de contrôle et de données distincts. La virtualisation de réseau offre la possibilité de créer et de déployer des réseaux virtuels, des commutateurs logiques, des routeurs, des pare-feu, des équilibreurs de charge, des réseaux privés virtuels (VPN) et de sécuriser la charge dans un temps raisonnable.
- **Virtualisation des postes de travail :** ce type ressemble au virtualisation des applications, mais en remplaçant les applications par des postes de travails entiers.
- **Virtualisation du stockage :** elle s'agit de sauvegarder les données dans plusieurs serveurs liés par un system de management de stockage par contre les données apparaissent à l'utilisateur regroupées dans un seul espace de stockage.
- **Virtualisation de serveur :** Elle permet l'utilisation et la division d'un serveur physique en plusieurs serveurs virtuels. Elle permet aussi d'améliorer les performances et de réduire les couts. On peut prendre l'utilisation de Virtual box pour travailler sur des projets qui nécessitent différents systèmes d'exploitation. Ce dernier est un environnement qui permet de créer plusieurs Vms avec un OS selon le besoin.
- **Virtualisation des données :** ce genre est principalement utilisé dans le cadre de l'intégration de données dans des domaines tels que la BI (Business Intelligence ou l'intelligence d'affaires). Il permet aux utilisateurs de visualiser collectivement des ensembles de données via une interface en temps réel.

La figure 1.3 représente les types de virtualisation.

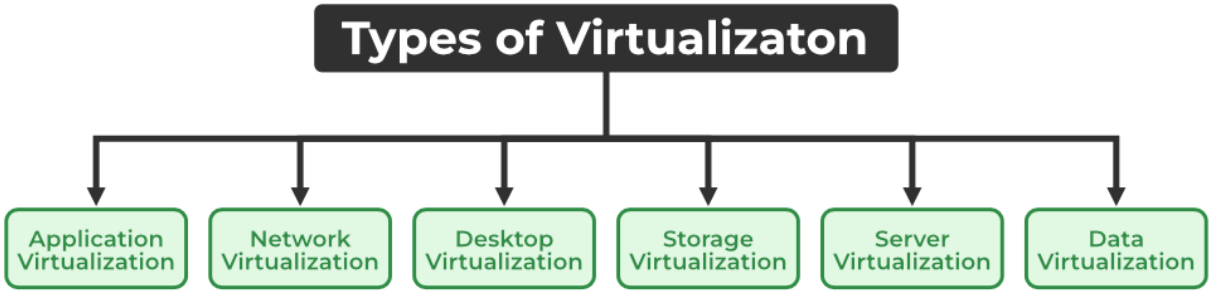

Figure 1.3 : les types de virtualisation [11].

1.7 Sécurité dans le Cloud computing

Le Cloud computing offre de nombreux avantages, mais il est également exposé à plusieurs menaces de sécurité. L'une des principales préoccupations est la confidentialité des données, car les informations sont stockées dans des serveurs appartenant à des tiers. Si ces serveurs sont compromis, des données confidentielles peuvent être divulguées.

La sécurité du Cloud est une responsabilité conjointe des fournisseurs de Cloud et des clients. Il faut assuré l'essentiel afin de prendre des mesures et des bonnes pratiques de sécurité pour réduire tout type de risques. Commencer par avoir une politique de sécurité claire, définir les rôles et les responsabilités de l'équipe de sécurité et utiliser par exemple l'authentification à deux facteurs pour renforcer la sécurité des comptes utilisateurs. Il est important de surveiller régulièrement l'environnement du Cloud computing pour détecter toute activité suspecte ou non autorisée. Des outils de surveillance automatisés peuvent être utilisés pour détecter les vulnérabilités et les attaques potentielles.

De plus, les données sensibles doivent être chiffrées en transit et au repos pour les protéger contre les écoutes clandestines et les accès non autorisés. Les fournisseurs de services Cloud proposent souvent des options de cryptage, mettre en place des sauvegardes régulières de données afin de pouvoir les restaurer en cas de perte ou de corruption de données. Les fournisseurs de services Cloud proposent souvent des options de sauvegarde.

Enfin, il est important de se tenir au courant des dernières menaces de sécurité et vulnérabilités du Cloud computing [12].

1.8 Cloud Service Provider

Les fournisseurs de services Cloud ou CSP (Cloud Service Provider) sont les entreprises qui offrent les différents services informatiques via la technologie de Cloud computing en les hébergeant sur des datas centers afin que les utilisateurs puissent accéder à ses services via internet. Trois grandes sociétés de fournisseurs de services Cloud sont à mentionner :

1.8.1 AWS

Amazon web services ou AWS : est un Cloud computing plate-forme fondée par Amazon en 2006 [13]. C'est l'une des plus performantes et populaires dans ce bizness. Elle offre plus de 200 services comme le calcul, les bases de données et le stockage et l'une des premières entreprises à introduire le paiement à l'utilisation.

1.8.2 Microsoft Azure

Microsoft azure est une public Cloud computing plate-forme qui offre les différents solutions (Saas, Paas, Iaas). Elle a été lancé en février 2010 [14], bien plus tard que son principal concurrent, AWS. Azure répartis dans le monde entier, ce qui représente le plus grand nombre de centres de données pour n'importe quelle plate-forme Cloud.

1.8.3 GCP

Google Cloud Platform (GCP) est une plateforme de Cloud computing fournie par Google. Les utilisateurs peuvent accéder à des services Cloud tels que le stockage de données, l'analyse de données, l'apprentissage automatique, la gestion de conteneurs, la mise en réseau et le développement d'applications.

Ces trois géants du Cloud sont engagés depuis longtemps dans une course féroce pour devenir un leader du marché [15].

La figure 1.4 montre une comparaison entre les troix CSPs précédents.

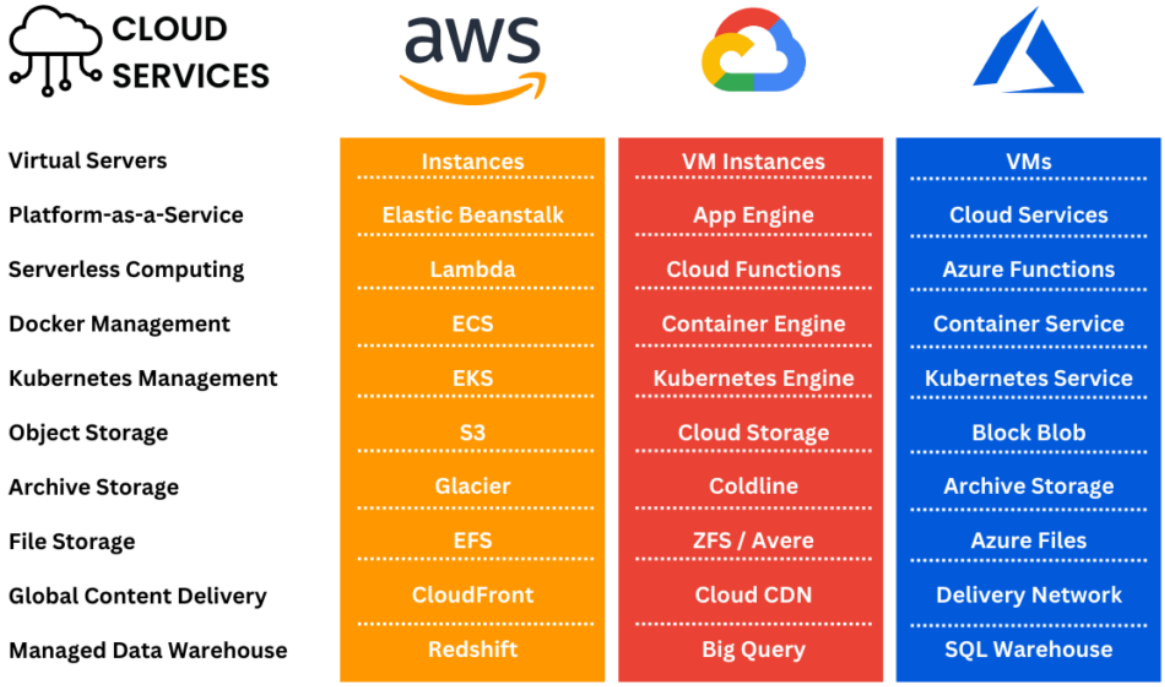

Figure 1.4 : comparaison entre AWS, Microsoft Azure et Google Cloud Platform [15].

1.9 Conclusion

Le Cloud computing est une technologie évolutive qui permet aux utilisateurs d'accéder à divers services informatiques sur Internet comme le stockage de données, l'analyse de données, l'apprentissage automatique, la gestion de conteneurs, etc. Les avantages du Cloud computing incluent des coûts réduits, une efficacité opérationnelle accrue et une flexibilité. Cependant, il est important de prendre en compte les problèmes de sécurité et de confidentialité lors de l'utilisation des services Cloud. Dans l'ensemble, le Cloud computing continue de transformer la façon dont les entreprises et les organisations gèrent leur infrastructure informatique, offrant des opportunités d'innovation future.

Dans le chapitre suivant, nous allons présenter les métaheuristiques et les méthodes d'aide à la décision multicritères.

Chapitre 2 : Les métaheuristiques

2.1 Introduction

L'optimisation des problèmes est l'art de comprendre et de transformer des problèmes réels en des modèles mathématiques, c'est une branche de l'informatique qui vise à trouver la meilleure solution possible pour un problème donné, en utilisant des algorithmes et des techniques d'optimisation. Il s'agit d'une discipline interdisciplinaire qui englobe des domaines tels que les mathématiques, l'informatique, l'ingénierie, la physique et l'économie.

L'optimisation des problèmes est un domaine qui utilise souvent les métaheuristiques pour trouver des solutions efficaces pour des problèmes complexes et difficiles à résoudre.

Le but de ce chapitre est d'introduire les concepts des méthodes d'optimisation ainsi que présenter les notions des métaheuristiques : ses caractéristiques, sa classification en se basant sur l'algorithme de Saut de Grenouilles ou "Shuffled Frog Leaping Algorithm" (SFLA) en anglais. Finalement, faire un passage sur les méthodes d'aide à la décision multicritères, en raison de son importance dans l'évaluation des critères en s'intéressant à trois méthodes en particulier, à savoir : MARCOS, MAIRCA et EAMR.

2.2 Méthodes de résolution

Les chercheurs ont développé des méthodes de résolution efficaces pour faire face à la diversité des problèmes rencontrés dans notre vie quotidienne. Ces méthodes visent à améliorer les performances en termes de temps de calcul nécessaire et/ou de qualité de la solution proposée.

Au fil des années, de nombreuses méthodes de résolution ont été proposées pour résoudre des problèmes de complexités diverses. Elles se distinguent par leurs principes, leurs stratégies et leurs performances.

Cette diversité a conduit à regrouper les méthodes de résolution des problèmes en deux classes principales : les méthodes exactes et les méthodes approchées.

2.2.1 Méthodes exactes

Les méthodes d'optimisation exactes (ou déterministes) offrent l'avantage de fournir des solutions garanties comme optimales. Ces méthodes fonctionnent généralement en explorant implicitement toutes les combinaisons possibles de l'espace de recherche. Elles sont spécifiquement conçues pour résoudre des problèmes avec des fonctions objectifs convexes, continues et dérivables [16]. Parmi les méthodes de résolution exactes les plus connues, on peut citer la méthode séparation et évaluation, la méthode de coupes planes et la méthode (Branch and Cut). Cependant, ces méthodes peuvent rapidement devenir inefficaces pour des problèmes de grande taille, car le nombre de solutions à explorer peut devenir astronomique.

2.2.2 Méthodes approchées

Une méthode approchée, bien que non exhaustive, est une méthode d'optimisation qui vise à trouver une solution faisable à la fonction objectif en un temps raisonnable, même sans garantie d'optimalité. La principale force de ces méthodes réside dans leur capacité à être appliquées à n'importe quelle classe de problèmes, qu'ils soient simples ou complexes. Cela est dû en partie à leur nature itérative et à leur capacité à explorer l'espace de recherche de manière efficace. Bien qu'il n'y ait pas de garantie quant à l'optimalité de la solution trouvée, les méthodes approchées offrent souvent des solutions suffisamment précises pour être utilisées dans la pratique. En fin de compte, l'efficacité des méthodes approchées dépendra de la complexité et de la nature du problème à résoudre, ainsi que des ressources disponibles pour effectuer les calculs nécessaires. Nous en distinguons deux classes : les heuristiques, et les métaheuristiques.

2.2.2.1Heuristiques

En général, les méthodes heuristiques sont spécifiques à un problème particulier, et sont conçues pour exploiter les caractéristiques et les particularités de ce problème spécifique [17]. Il existe trois principales catégories d'heuristiques utilisées pour résoudre des problèmes combinatoires [17] :

1. Les heuristiques de construction : ces méthodes permettent de construire progressivement une solution en suivant des règles et des stratégies spécifiques. Elles débutent généralement avec une solution partielle initiale et ajoutent de manière itérative des composants jusqu'à obtenir une solution complète.

2. Les heuristiques d'amélioration : ces approches visent à améliorer une solution existante en effectuant des modifications locales. Elles explorent l'espace des solutions voisines et apportent des modifications itératives pour trouver une solution de meilleure qualité.

3. Les heuristiques de recherche locale : ces techniques se concentrent sur l'exploration d'un voisinage autour d'une solution courante. Elles effectuent des mouvements locaux afin de trouver une solution optimale ou une solution se rapprochant de l'optimalité dans le voisinage immédiat.

2.2.2.2Métaheuristiques

Les métaheuristiques se sont des algorithmes qui donnent des solutions de bonne qualité en temps raisonnable pour des problèmes combinatoires réputés difficiles pour lesquels on ne connaît pas de méthode classique plus efficace. Ce sont essentiellement des algorithmes itératifs qui convergent vers une optimisation globale en évaluant une fonction objectif donnée. Ils se comportent comme algorithmes de recherche qui conduisent à la meilleure approximation de solution. [18]

2.3 Caractéristiques des métaheuristiques

Les métaheuristiques se caractérisent par [19] :

1. Ce sont des techniques qui dirigent la recherche d'une solution de manière stratégique.
2. Leur but est d'explorer de manière efficace l'espace de recherche afin d'identifier des points (presque) optimaux.
3. Les techniques qui composent les algorithmes de type métaheuristique varient de simples procédures de recherche locale à des processus d'apprentissage complexes.
4. En général, les métaheuristiques sont non-déterministes et ne fournissent aucune garantie d'optimalité.
5. Les métaheuristiques peuvent faire appel à des heuristiques qui tiennent compte de la spécificité du problème traité, mais ces heuristiques sont contrôlées par une stratégie de niveau supérieur.
6. Les métaheuristiques peuvent utiliser l'expérience accumulée lors de la recherche de l'optimum pour guider plus efficacement les étapes suivantes du processus de recherche.

2.4 Classification des métaheuristiques

Les métaheuristiques se subdivisent en deux classes : Les méthodes basées sur une solution (ou méthodes de trajectoire) et les méthodes basées sur population.

La figure 2.1 présente la classification des métaheuristiques [20].

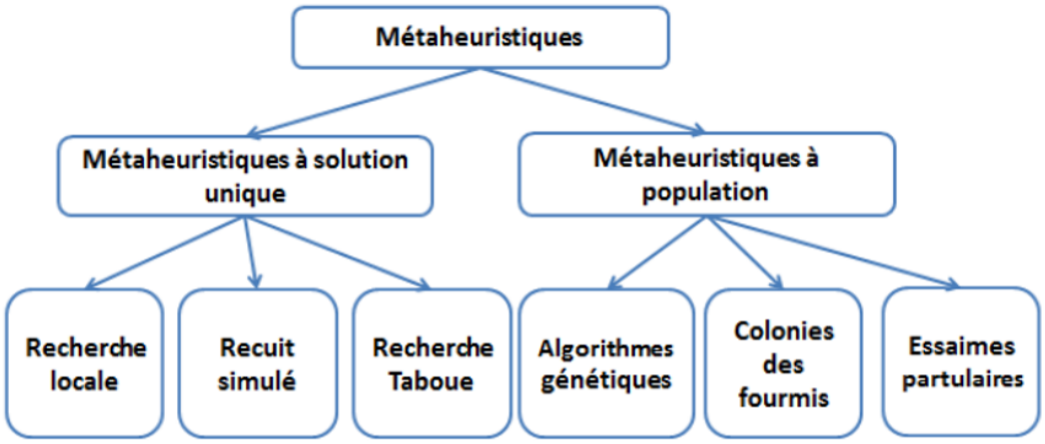

Figure 2.1 : classification des métaheuristiques [21].

- **Méthodes de trajectoire**

Ce sont des méthodes de recherche locale qui passent d'une solution à une autre dans l'espace des solutions jusqu'à trouver la solution optimale ou que le temps imparti soit dépassé. Ces méthodes reposent toutes sur un algorithme de recherche de voisinage. Elles commencent avec une solution initiale et l'améliorent progressivement en sélectionnant une nouvelle solution dans son voisinage à chaque étape. La méthode de recherche locale la plus élémentaire est la méthode de descente. [22]

La figure 2.2 montre le principe des méthodes de trajectoire.

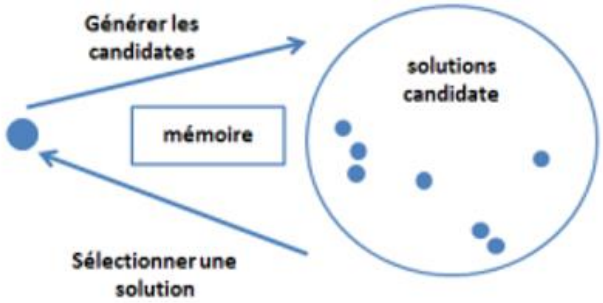

Figure 2.2 : principe des métaheuristiques à solution unique [21].

- **Méthodes basées sur une population des solutions**

Généralement ces méthodes sont inspirées de la nature. Les algorithmes génétiques, l'optimisation par essaim de particules et les algorithmes de colonies de fourmis présentent les exemples les plus connus des méthodes qui travaillent avec une population.

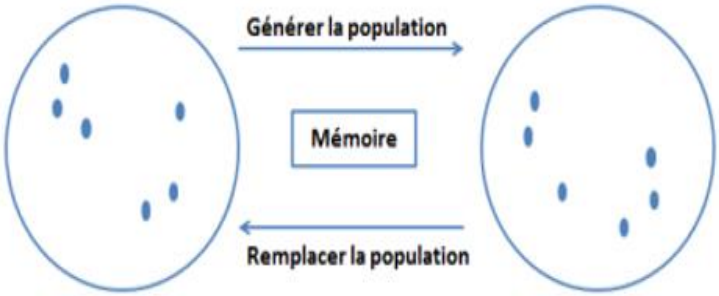

Figure 2.3 : principe des métaheuristiques à population [21].

2.5 Algorithme de saut de grenouille

L'algorithme de sauts de grenouilles, également connu sous le nom de Shuffled Frog Leaping Algorithm (SFLA), se compose d'un ensemble de populations virtuelles interactives de grenouilles, divisées en différents memeplexes. Le SFLA effectue simultanément une recherche locale indépendante dans chaque memeplex. Pour compléter la recherche locale, l'algorithme utilise une méthode d'optimisation par essaim de particules adaptée aux problèmes discrets. Pour assurer l'exploration globale, les grenouilles sont périodiquement mélangées et réorganisées en nouveaux memeplexes, et les informations entre différents memeplexes sont partagées à travers un processus de saut. Dans chaque memeplex, les grenouilles fournissent la meilleure solution Xb et la pire Xw, tandis que la grenouille donnant la meilleure solution dans toute la population est notée Xg. Lors de l'évolution d'un memeplex, c'est-à-dire lors de l'exploration locale, la grenouille la plus mauvaise saute vers la meilleure solution selon les équations suivantes : [23]

$$D = r \times (|Xb - Xw|) \quad (0 < r < 1) \tag{2.1}$$

$$Xw' = Xw + D. \tag{2.2}$$

Il existe plusieurs versions du Shuffled Frog-Leaping Algorithm (SFLA) qui ont été proposées par différents chercheurs pour résoudre différents problèmes d'optimisation. Voici quelques exemples de versions de SFLA :

1. SFLA classique : il s'agit de la version originale du SFLA proposée par Eusuff et Lansey en 2003 [24]. Cette version est utilisée pour résoudre des problèmes d'optimisation continus et discrets.
2. SFLA amélioré : cette version a été proposée par Wang et al. En 2009. Il s'agit d'une version améliorée du SFLA classique qui améliore la règle de saut en étendant correctement la taille du pas de saut et en ajoutant une composante d'inertie de saut pour tenir compte du comportement social. [25]
3. SFLA hybride : cette version a été déployé par Alireza Rahimi-Vahed, Ali Hossein Mirzaei en 2007. Il s'agit d'une version hybride multi-objectif basé sur l'algorithme de saut de grenouille mélangé (SFLA) et l'optimisation des bactéries (BO) [26].
4. SFLA multi-objectif : cette version a été proposée par Taher Niknam et ses associés en 2011. Il s'agit d'une version qui résoudre le problème multi-objectif de flux de puissance optimal. [27]
5. SFLA parallèle : cette version a été présenté par Chunlei Liu,Zhenzhou Ji,Yingsen Hong en 2013. Il s'agit d'une version parallèle du SFLA qui utilise la parallélisation pour accélérer le processus d'optimisation. [28]

   SFLA pour des problèmes spécifiques : d'autres versions du SFLA ont été proposées pour résoudre des problèmes spécifiques, tels que la planification de la production, la conception de réseaux de distribution, la sélection de caractéristiques et la classification de données.

Il convient de noter que ces versions du SFLA sont des exemples et qu'il existe probablement d'autres versions du SFLA qui ont été proposées dans la littérature scientifique.

Le pseudo code du SFLA est donne comme suit [29] :

Étape 1 : Définissez la taille F de la population, le nombre M des memeplexes et le nombre N des itérations

Étape 2 : Générer une population aléatoire de solutions F et évaluer chaque solution.

Étape 3 : Trier la population et déterminer la meilleure solution Xg.

Étape 4 : Diviser la population en M memeplexes.

Étape 5 : Recherche locale : pour chaque memplex, répétez pour les itérations N :

• Déterminer la meilleure solution Xb et la pire solution Xw.

• Calculer Xw′ à partir de Xb (appliquer les équations 2.1 et 2.2)

Si (Xw′ est meilleur que Xw) alors remplacer Xw par Xw′.

Sinon calculer Xw′ de Xg (appliquer les équations 2.1 et 2.2 avec le remplacement de Xb par Xg)

Si (Xw′ est meilleur que Xw) alors remplacer Xw par Xw′.

Sinon Générer aléatoirement Xw' et remplacer Xw par Xw'.

Fin si

Fin si

Étape 6 : Réunissez les memeplexes M pour reconstruire la population.

Étape 7 : Passez à l'étape 3 si le critère d'arrêt n'est pas atteint.

2.6 Méthodes d'aide à la décision multicritères

Les méthodes d'aide à la décision multicritères sont utilisées dans de nombreux domaines. Ces méthodes aident à comparer les alternatives et à trouver la meilleure. Il existe de nombreuses de ces méthodes comme MARCOS, MAIRCA, EAMR.

2.6.1 MARCOS

La méthode MARCOS (Measurement Alternatives and Ranking according to Compromise Solution) est un outil d'aide à la décision basé sur une analyse multicritère. Développé en 2019 par une équipe de chercheurs de l'Université de Novi Sad [30], en Serbie. Il est conçu pour être utilisé dans des scénarios décisionnels complexes où plusieurs critères et alternatives doivent être pris en compte.

La méthode MARCOS pour la prise de décision multicritère nécessite la mise en place d'une série d'étapes qui sont présentées ci-dessous [31] :

Étape 1 : Construire la matrice initiale en suivant l'équation suivante :

$$X= \begin{bmatrix} X_{11} & \dots & X_{1n} \\ X_{21} & \dots & X_{2n} \\ \vdots & \vdots & \vdots \\ X_{m1} & \dots & X_{mn} \end{bmatrix} \quad (2.3)$$

D'où m est le nombre d'options ; n est le nombre de critères, X_{mn} est la valeur du n-ième critère pour l'option m.

Étape 2 : Construction d'une matrice initiale qui s'étend en ajoutant une solution idéale (AI) et la solution opposée à la solution idéale (AAI) :

$$X= \begin{matrix} AAI \\ A1 \\ A2 \\ \vdots \\ Am \\ AI \end{matrix} \begin{bmatrix} X_{aa1} & \dots & X_{aan} \\ X_{11} & \dots & X_{1n} \\ X_{21} & \dots & X_{2n} \\ \vdots & \vdots & \vdots \\ X_{m1} & \dots & X_{mn} \\ X_{ai1} & \dots & X_{ain} \end{bmatrix} \quad (2.4)$$

Où **:**

AAI = min (X_{ij}) ; i = 1, 2, …, m ; j = 1, 2, …, n si j est un critère où plus grand est meilleur.

AAI = max (X_{ij}) ; i = 1, 2, …, m ; j = 1, 2, …, n si j est un critère où plus petit est meilleur.

AI = max (X_{ij}) ; i = 1, 2, …, m ; j = 1, 2, …, n si j est un critère où plus grand est meilleur.

AI = min (X_{ij}) ; i = 1, 2, …, m ; j = 1, 2, …, n si j est un critère où plus petit est meilleur.

Étape 3 : Calcul des valeurs normalisées selon les équations suivantes :

- $u_{ij} = \frac{X_{AI}}{X_{ij}}$ si j est un critère où plus petit est meilleur. (2.5)
- $u_{ij} = \frac{X_{ij}}{X_{AI}}$ si j est un critère où plus grand est meilleur. (2.6)

Étape 4 : Calcul des valeurs normalisées pondérées en utilisant l'équation suivante :

$c_{ij} = u_{ij} \,.\, w_j$, où w_j est le poids du critère j. (2.7)

Étape 5 : Calcul des coefficients K_i^+ et K_i^- à l'aide des équations suivantes :

$$K_i^- = \frac{S_i}{S_{AAI}} \tag{2.8}$$

$$K_i^+ = \frac{S_i}{S_{AI}} \tag{2.9}$$

Où S_i, SAAI et SAI sont la somme des valeurs de C_{ij}, XAAI et XAI, respectivement ;

avec i = 1, 2,..., m.

Étape 6 : Calcul des fonctions f(K_i^+) et f(K_i^-) selon :

$$f(K_i^-) = \frac{K_i^+}{K_i^+ + K_i^i} \tag{2.10}$$

$$f(K_i^+) = \frac{K_i^-}{K_i^+ + K_i^i} \tag{2.11}$$

Étape 7 : Calcul de la fonction f(Ki) selon l'équation suivante et classement des alternatives :

$$f(Ki) = \frac{K_i^+ + K_i^-}{\frac{1-f(K_i^+)}{f(K_i^+)} + \frac{1-f(K_i^-)}{f(K_i^-)}} \tag{2.12}$$

Le classement des solutions se fait en fonction de la meilleure solution, celle qui a la plus grande valeur de la fonction f(Ki).

2.6.2 MAIRCA

La méthode MAIRCA (Multi-Attributive Ideal–Real Comparative Analysis) a été introduite pour la première fois en 2018 [31]. L'avantage exceptionnel de cette méthode par rapport aux autres méthodes est que les objectifs peuvent être de types qualitatifs et quantitatifs. Plusieurs études ont appliqué cette méthode à la prise de décisions multicritères. Par exemple, pour déterminer le moment le plus efficace (année) pour les fusions et acquisitions d'entreprises en Turquie pendant la période 2015-2019 [32].

Les étapes pour implémenter la prise de décision multicritère selon la méthode MAIRCA sont les suivantes [31] :

Étape 1 : Similaire à l'étape 1 de la méthode MARCOS.

Étape 2 : Détermination de la priorité pour un indicateur. Lorsque le décideur est neutre, le rôle des indicateurs est le même (aucune priorité n'est donnée à l'un ou à l'autre). Ensuite, la priorité pour les critères est la même et est calculée comme suit :

$$P_{Aj} = \frac{1}{m}, \text{ j = 1, 2, …, n} \tag{2.13}$$

Étape 3 : Calcul des quantités tp_{ij} selon l'équation :

$$tp_{ij} = P_{Aj} \cdot w_j, \text{ i = 1, 2, …, m ; j = 1, 2,…,n} \tag{2.14}$$

D'où w_j est le poids du jème critère.

Étape 4 : Calcul des quantités tr_{ij}selon les équations :

$$tr_{ij} = tp_{ij} \cdot \left(\frac{x_{ij} - x_i^-}{x_i^+ - x_i^-} \right) \text{ si j est un critère où plus petit est meilleur.} \tag{2.15}$$

$$tr_{ij} = tp_{ij} \cdot \left(\frac{x_{ij} - x_i^+}{x_i^- - x_i^+} \right) \text{ si j est un critère où plus grand est meilleur.} \tag{2.16}$$

Étape 5 : Calcul des quantités g_{ij} selon l'équation :

$$g_{ij} = tp_{ij} - tr_{ij} \tag{2.17}$$

Étape 6 : Sommation des valeurs g_{ij} selon l'équation :

$$Q_i = \sum_{i=0}^{m} g_{ij} \tag{2.18}$$

Le classement des options est effectué selon le principe selon lequel celui qui a la plus petite valeur de Qi est le meilleur.

2.6.3 EAMR

La méthode EAMR (Evaluation by an Area-Based Method of Ranking) a été proposeé en 2016. Cette méthode a été utilisée pour un certain nombre d'études telles que la sélection de partenaires à embaucher pour la logistique, la sélection des types de contrat de services de santé, la décision de la quantité de commande pour chaque fournisseur pour garantir les critères environnementaux. Ci-dessous sont énumérées les différentes étapes à suivre pour mettre en place la prise de décision multicritère selon la méthode EAMR [31] :

Étape 1 : Construction d'une matrice de décision :

$$Xd = \begin{bmatrix} X_{11}^{d} & \dots & X_{1n}^{d} \\ X_{21}^{d} & \dots & X_{2n}^{d} \\ \vdots & \vdots & \vdots \\ X_{m1}^{d} & \dots & X_{mn}^{d} \end{bmatrix} \quad (2.19)$$

D'où $1 \leq d \leq k$, k est le nombre de décideurs ; d est l'indice représentant le décideur d.

Étape 2 : Calcul de la valeur moyenne de chaque alternative pour chaque critère selon l'équation:

$$X_{ij} = \frac{1}{k}(X_{ij}^{1} + X_{ij}^{2} + \cdots + X_{ij}^{k}) \quad (2.20)$$

Il convient de noter que k est l'indice du k-ème décideur, et non l'exposant.

Étape 3 : Détermination des pondérations pour les critères. À cette étape, chaque décideur peut choisir une méthode de pondération différente.

Étape 4 : Calcul de la valeur moyenne pondérée pour chaque critère selon l'équation :

$$w_j = \frac{1}{k}(w_j^{1} + w_j^{2} + \cdots + w_j^{k}) \quad (2.21)$$

Étape 5 : Calcul des valeurs N_{ij} selon l'équation :$N_{ij}=\frac{Xij}{Ej}$ (2.22)

Dans lequel, E_j est déterminé par l'équation : E_j= **max** $i \in \{1,\dots,m\}(X_{ij})$ (2.23)

Étape 6 : Calcul des valeurs de poids normalisées selon l'équation : $V_{ij}=N_{ij}.W_j$ (2.24)

Étape 7 : Calcul du score normalisé pour les critères :

- $G_i^+ = V_{i1}^+ + V_{i2}^+ + \ldots + V_{im}^+$ si j est le critère le plus grand est le mieux (2.25)
- $G_i^- = V_{i1}^- + V_{i2}^- + \ldots + V_{im}^-$ si j est le critère le plus petit est le mieux (2.26)

Étape 8 : Le rang de la valeur (RV) est trouvé en fonction de G_i^+ et G_i^-.

Étape 9 : Calcul du score d'évaluation pour les options selon l'équation :

$$Si = \frac{RV(G_i^+)}{RV(G_i^-)} \quad (2.27)$$

La solution avec le plus grand Si sera la meilleure, qui est le principe de classement de la méthode EAMR.

2.7 Conclusion

Dans ce chapitre, nous avons présenté les métaheuristiques et ces caractéristiques ainsi que les méthodes de résolution et ces différentes classifications. Le chapitre s'est intéressé en particulier à l'algorithme SFLA et aux méthodes d'aide à la décision multicritères, à savoir : MARCOS, MAIRCA et EAMR.

Dans le chapitre suivant, nous allons évaluer l'algorithme SFLA avec les méthodes MCDM citées précédemment pour l'ordonnancement des tâches dans le Cloud computing.

Chapitre 3 : Implémentation de l'application et évaluation des résultats obtenus

3.1 Introduction

Ce chapitre se concentre sur l'implémentation de l'algorithme SFLA avec les méthodes d'aide à la décision multicritères à savoir MAIRCA, MARCOS, EAMR et la somme pondérée afin de réaliser un ordonnancement des taches dans le Cloud computing. Trois critères ont été évalués qui sont : le makespan, le coût et le taux d'utilisation des ressources.

Le chapitre présente les outils de simulations et de développement utilisés dans le cadre de la réalisation de ce PFE à savoir, le langage Java, l'environnement de développement NetBeans, la bibliothèque JFreeCharts et le simulateur CloudSim.

Nous allons présenter en détail l'IHM que nous avons conçue pour faciliter l'interaction avec notre système. De plus, nous procédons à une évaluation détaillée des résultats obtenus, permettant ainsi une comparaison approfondie entre les méthodes Multi-Criteria Decision-Making (MCDM).

3.2 Environnement de développement et de simulation

Les outils de développement et de simulation suivants ont été utilisés pour développer l'algorithme SFLA et les méthodes MCDM :

3.2.1 Java

Java est un langage de programmation polyvalent inventé par James Gosling en 1991. Il a été développé en 1995 pour créer des applications web, serveur, desktop, mobiles et robotiques. Java fonctionne sur divers systèmes d'exploitation et est considéré comme l'un des langages les plus populaires. Sa simplicité et sa portabilité en font un choix courant pour les applications d'entreprise, les jeux, les applications web, la science des données, l'intelligence artificielle, et bien d'autres domaines [35].

3.2.2 NetBeans

NetBeans est un environnement de développement intégré (IDE) populaire utilisé pour programmer en Java et d'autres langages. Il a été converti en version open source par Sun en juin 2000, et en octobre 2016, il a été transféré à l'Apache Software Fondation. Il propose une gamme complète d'outils et de fonctionnalités pour faciliter le développement logiciel. Son interface conviviale comprend des fonctionnalités telles que la coloration syntaxique, l'achèvement automatique du code et la détection d'erreurs, ce qui simplifie l'écriture du code [36].

3.2.3 JFreechart

JFreeChart est une bibliothèque de graphiques 100% Java gratuite qui permet aux développeurs d'afficher des graphiques de qualité professionnelle dans leurs applications. JFreeChart offre une vaste gamme de fonctionnalités [37] :

- Une API cohérente et bien documentée prenant en charge de nombreux types de graphiques.
- Une conception flexible facile à étendre, adaptée aux applications côté serveur et côté client.
- La prise en charge de nombreux types de sortie, y compris des composants Swing et JavaFX, des fichiers image (tels que PNG et JPEG) et des formats de fichiers graphiques vectoriels (comme PDF, EPS et SVG).

3.2.4 CloudSim

CloudSim est un Framework open source qui permet la simulation de l'infrastructure et des services de Cloud computing. Il est développé par l'organisation CLOUDS Lab et est entièrement implémenté en Java. Son utilisation consiste à modéliser et simuler un environnement de Cloud computing pour évaluer des hypothèses avant le développement de logiciels, en reproduisant des tests et des résultats.

Le moteur de simulation de base de CloudSim Core fournit des interfaces pour gérer les ressources telles que les machines virtuelles (VM), la mémoire et la bande passante des datacenters virtualisés.

La couche CloudSim gère la création et l'exécution des entités essentielles telles que les VM, les Cloudlets et les hôtes. Elle prend également en charge l'exécution réseau, la fourniture de ressources et leur gestion.

La couche de code utilisateur est contrôlée par l'utilisateur, permettant au développeur de spécifier les exigences matérielles selon le scénario.

Certaines des classes les plus couramment utilisées lors de la simulation incluent :

- **Datacenter** pour modéliser un datacenter.

- **Host** pour la gestion des VMs.
- **VM** pour représenter une machine virtuelle.
- **Cloudlet** pour représenter une tâche exécutée sur une VM.
- **DatacenterBroker** pour agir au nom de l'utilisateur/client.
- **CloudSim** pour l'initialisation et la gestion de l'environnement de simulation [38].

La figure 3.1 montre la structure logicielle de CloudSim :

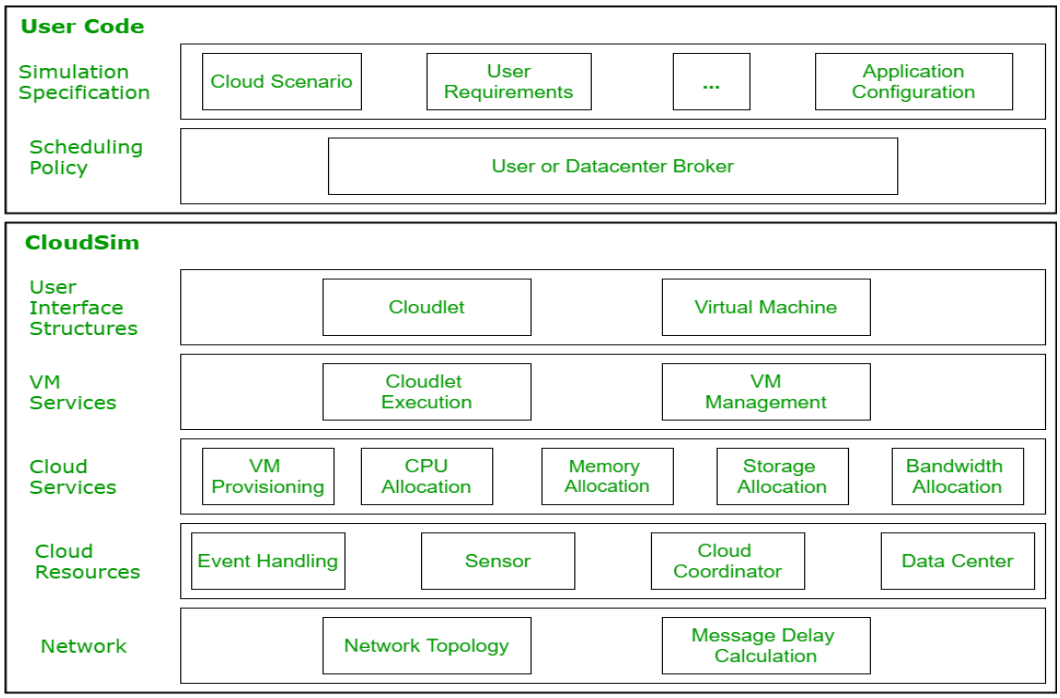

Figure 3.1 : l'architecture de CloudSim [38].

3.3 Critères d'évaluation

Les trois métriques de QoS suivantes ont été utilisées dans ce travail :

- **Makespan** : ou le temps de complétion, c'est l'un des mesures les plus utilisées dans l'évaluation des algorithmes d'ordonnancement. Il représente la durée écoulée entre la soumission du workflow et la réception des résultats. En d'autres termes, il correspond à la date de fin d'exécution de la dernière tâche [39].

  Le makespan est indiqué dans l'équation 3.1 :

$$Makespan = Max_{Ti\,\epsilon\,V}\{DF(T_i)\} \quad (3.1)$$

Où, DF est la date de fin d'exécution de la tâche T_i.

- **Coût** : la majorité des fournisseurs de services Cloud ont établi des tarifs pour l'utilisation de leurs services. Ils ont défini des prix pour le transfert d'une unité de données (par exemple, par mégaoctet) entre deux services, ainsi que des prix pour le traitement par unité de temps (par exemple, par heure).

  Dans le cadre de notre travail, le coût d'exécution d'une tâche spécifique dépend à la fois de sa date de fin d'exécution sur la machine VMj qui lui est assignée et du prix unitaire de cette machine (prixUj). Ainsi, il est donné par l'équation 3.2 [39] :

$$\text{Coût} = \sum_{i=1}^{n} DF(T_i).PRIXU_j \tag{3.2}$$

- **Utilisation de ressources** : l'optimisation multicritère vise à maximiser l'utilisation des ressources disponibles, y compris les machines virtuelles, afin de réaliser les tâches. Le taux moyen d'utilisation, une mesure de l'efficacité des ressources, est utilisé dans ce contexte pour représenter leur utilisation. Il est calculé selon les équations 3.3 et 3.4 [40] :

$$Avg\_RUR = (Avg\ Makespan\ /\ Cloud\ Makespan) \tag{3.3}$$

$$Avg\ Makespan = \frac{\sum MS(VM_x)}{n} \tag{3.4}$$

D'où, Avg_RuR : est le taux moyen d'utilisation des ressources.

Avg Makespan : est la moyenne du makespan de chaque VM.

Cloud Makespan : est le Makespan global.

n est le nombre de machines virtuelles.

MS(VM$_x$) est le makespan des tâches assignées à la machine virtuelle *x*.

Ces métriques de QoS sont classées selon les objectifs à atteindre et sont les suivantes :

- Minimiser le makespan.
- Minimiser le coût.
- Maximiser le taux d'utilisation de ressources.

Dans le cadre d'une optimisation multi-objectif, et en plus des méthodes MCDM que nous avons détaillées dans le chapitre précédent à savoir : MARCOS, MAIRCA et EAMR, nous avons aussi utilisé la méthode de la somme pondérée.

3.4 La méthode de la somme pondérée

La méthode de la somme pondérée, également connue sous le nom de Weight Sum Method (WSM), est une approche simple pour la prise de décision multicritère. Elle est également largement utilisée dans la vie quotidienne.

Pour appliquer cette méthode, les critères doivent être quantitatifs, avoir la même unité de mesure et s'étendre sur une même échelle de valeurs, ou ils doivent être tous normalisés.

La formule pour calculer la somme pondérée est la suivante [41] :

$$f(x) = \sum_{i=1}^{n} w_i . f_i \tag{3.4}$$

Pour utiliser la méthode de la somme pondérée, il est nécessaire de respecter deux contraintes importantes concernant les poids w1, w2, ..., wn :

- Les poids doivent être non négatifs, $0 \leq w_i \leq 1$ (3.5)
- La somme de tous les poids doit être égale à 1 , $\sum_{i=1}^{n} w_i$=1 (3.6)

Comme les 3 critères n'ont pas la même unité de mesure, donc il faut calculer les valeurs normalisées selon l'objectif à atteindre (maximiser ou minimiser) pour chaque critère.

Donc la normalisation des critères est faite selon les équations suivantes :

$$X_{normalisé} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{3.7}$$

Pour les critères à minimiser (makespan et coût).

$$X_{normalisé} = \frac{x_{max} - x}{x_{max} - x_{min}} \tag{3.8}$$

Pour le critère à maximiser (taux d'utilisation de ressources).

La fonction objectif pour l'optimisation multi-objective selon la méthode de la somme pondérée peut être exprimée de la manière suivante :

$$F = w1 * Makespan + w2 * Cout + w3 * Taux\ d'utilisation\ de\ ressources \tag{3.9}$$

Le vecteur de poids w = {w1, w2, w3} = {0.5, 0.25, 0.25} traduit l'importance ou l'exigence de l'utilisateur par rapport à chaque critère. Dans ce cas, les critères sont le makespan, le coût et le taux d'utilisation des ressources. Ce vecteur de poids indique que l'utilisateur accorde une plus grande importance au makespan (50%), tandis que le coût et le taux d'utilisation des ressources ont la même importance relative (25% chacun).

3.5 IHM développée

Étant donné que CloudSim s'exécute uniquement en mode console et ne dispose pas d'interface graphique, nous avons développé notre propre interface homme-machine (IHM) pour faciliter l'accès et la manipulation de la simulation.

Les étapes de configuration de la simulation sont présentées à travers les interfaces suivantes.

Interface principale : dès le démarrage de notre application, la première interface que vous verrez est l'interface principale. Cette interface, représentée dans la figure 3.2, affiche toutes les informations relatives à notre projet de fin d'études.

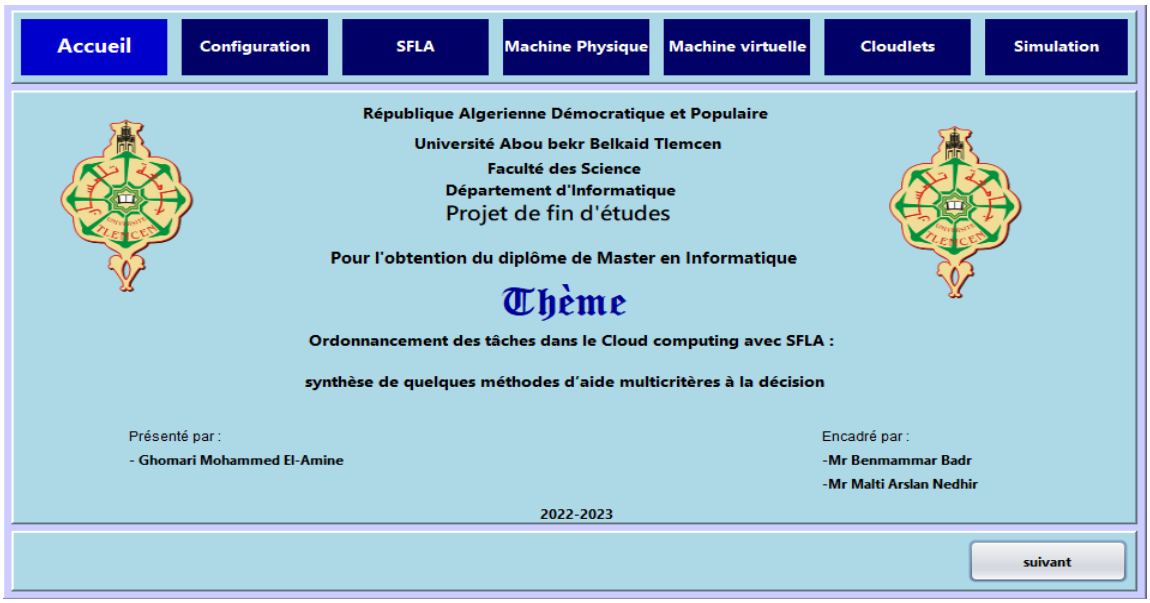

Figure 3.2 : interface principale.

Choix de configuration : au sein de l'interface représentée dans la figure 3.3, l'utilisateur est confronté à un choix entre deux options : la configuration par défaut et la configuration personnalisée.

Ces deux options donnent à l'utilisateur le contrôle sur la configuration de la simulation, lui permettant de choisir entre la configuration par défaut préétablie ou de personnaliser les paramètres selon ses propres critères et objectifs.

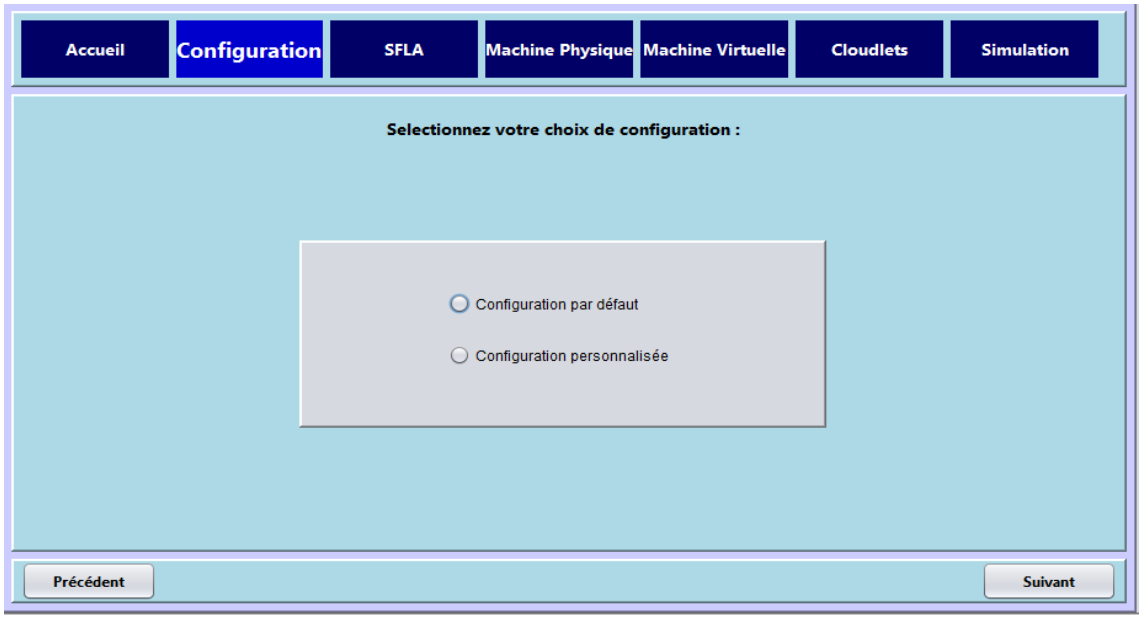

Figure 3.3 : choix de configuration.

Lancement des résultats de simulation : une fois que l'utilisateur a sélectionné la configuration par défaut, l'interface utilisateur illustrée dans la figure 3.4 apparaît. Cette interface permet de visualiser les résultats de l'évaluation en cliquant sur le bouton "Démarrer".

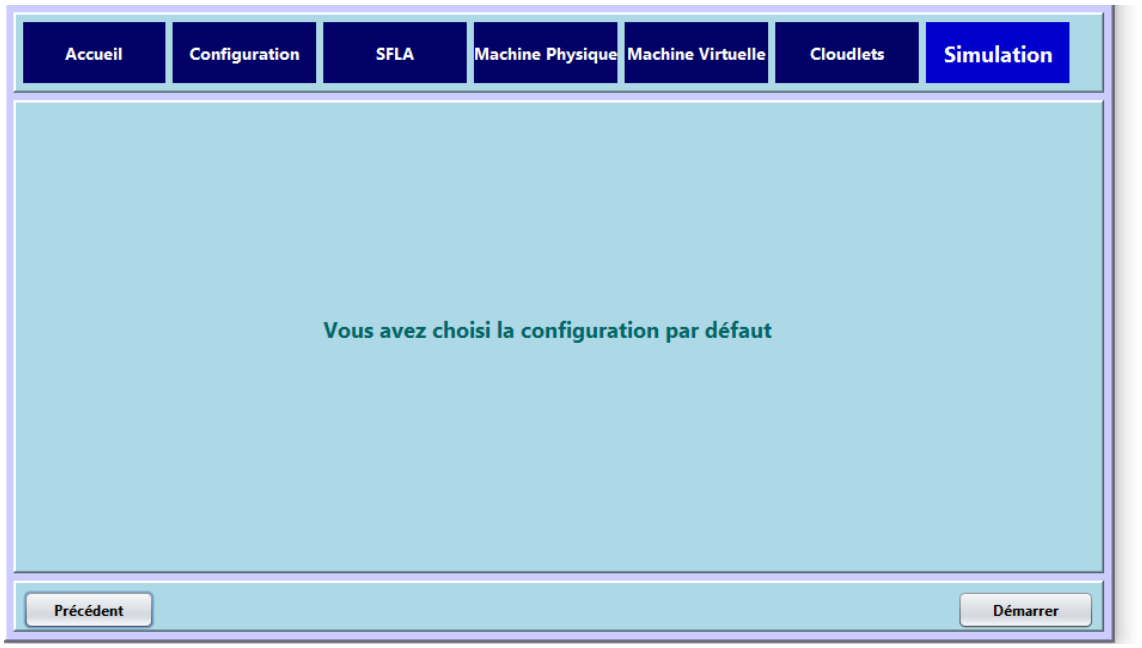

Figure 3.4 : lancement des résultats de simulation.

Configuration des paramètres de l'algorithme SFLA : après avoir choisi la configuration personnalisée dans l'interface précédente, l'interface utilisateur affichée dans la figure 3.5 se présente à l'utilisateur. Cette interface offre la possibilité de modification au caractéristiques de l'algorithm SFLA comme la taille de population, la taille du memeplex, le nombre d'itérations locale et globale.

Figure 3.5 : configuration des caractéristiques de l'algorithme SFLA.

Configuration des machines physiques : une fois que nous avons configuré les caractéristiques de l'algorithme, la prochaine étape consiste à configurer les machines physiques. Cette configuration comprend plusieurs éléments tels que le nombre de datacenters, le nombre d'hôtes, le nombre de processeurs (PE), la vitesse de chaque processeur (MIPS), la capacité de la RAM, la bande passante et la capacité de stockage.

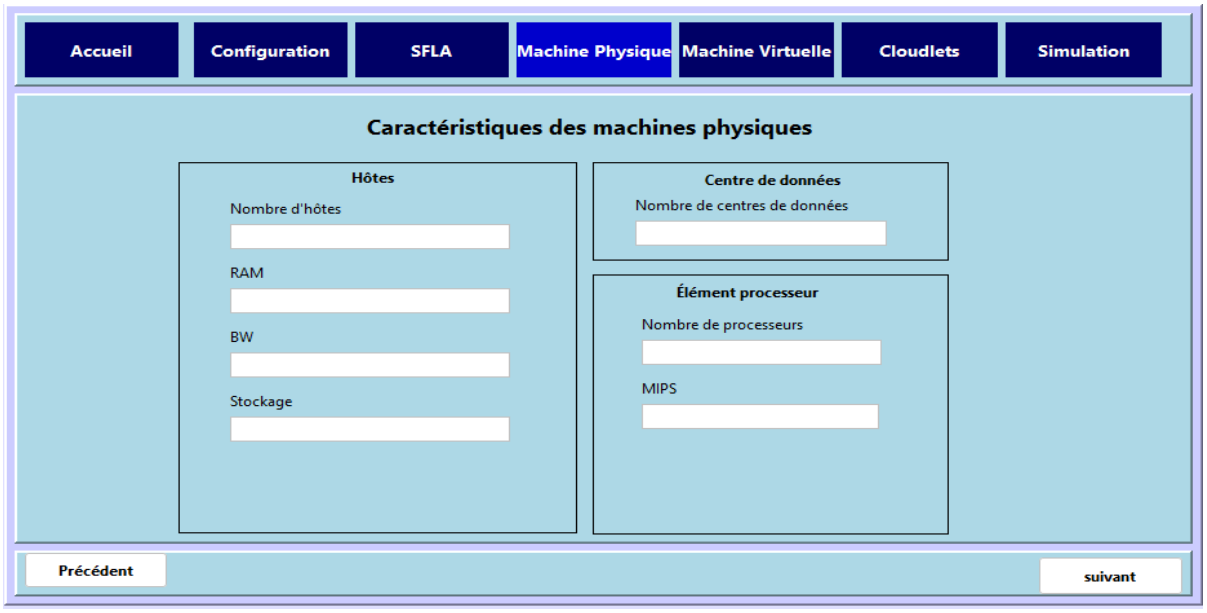

Figure 3.6 : configuration des machines physiques.

Configuration des machines virtuelles : une fois que nous avons terminé la configuration des différentes caractéristiques des machines physiques, la prochaine étape consiste à configurer les machines virtuelles (VMs).

Les machines virtuelles possèdent des caractéristiques telles que la vitesse de traitement (MIPS), la capacité de RAM, la bande passante et le stockage.

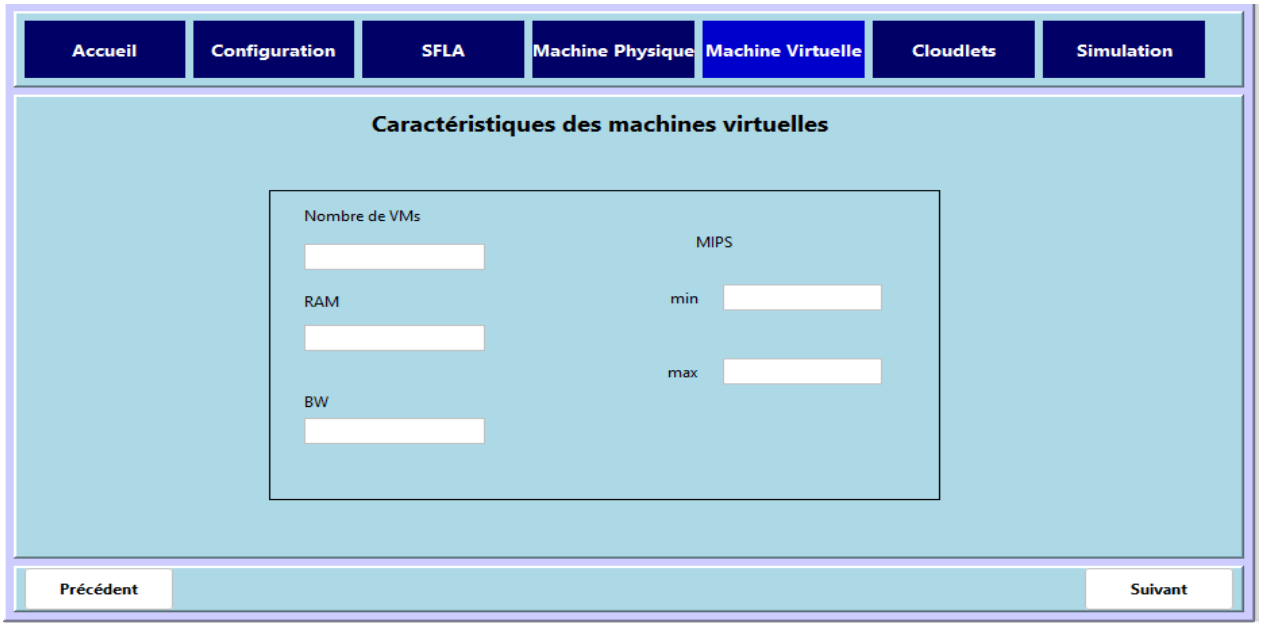

Figure 3.7 : configuration des machines virtuelles.

Configuration des Cloudlets : pour configurer les Cloudlets, il vous suffit d'effectuer une saisie simple pour définir leurs caractéristiques, telles que leur nombre et leur longueur.

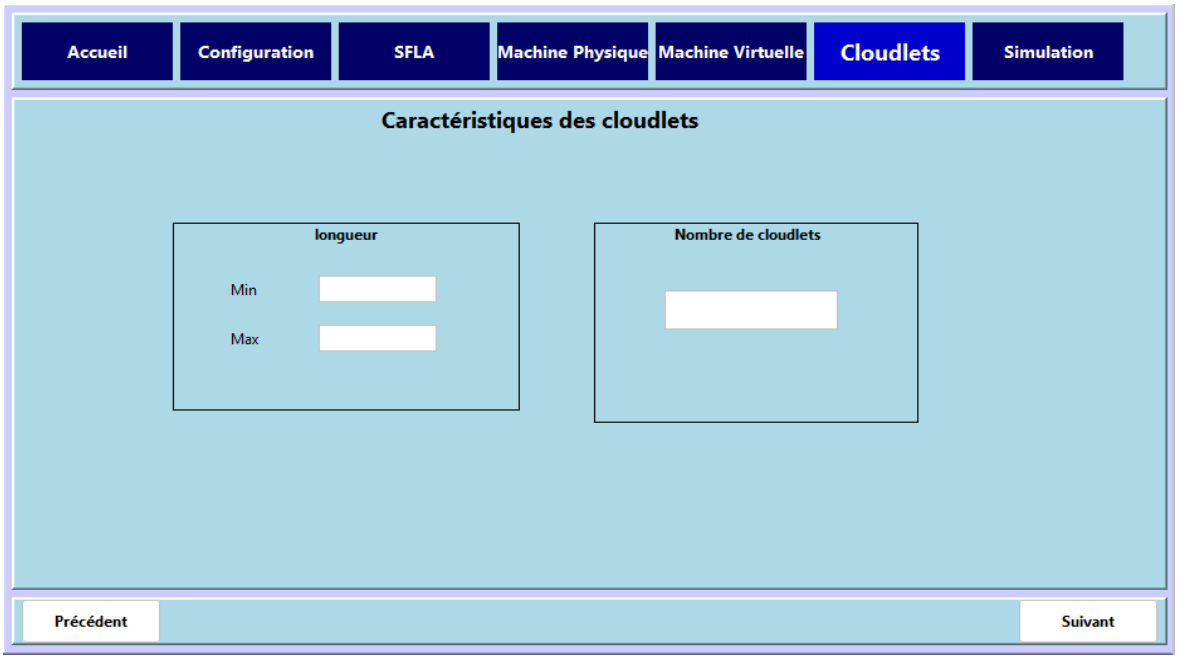

Figure 3.8 : configuration des Cloudlets.

Simulation : à l'aide de l'interface illustrée dans la figure 3.6, nous pouvons lancer l'exécution de la simulation. Cependant, avant cela, nous devons saisir le nombre de simulations à effectuer ainsi que le pourcentage attribué à chaque critère.

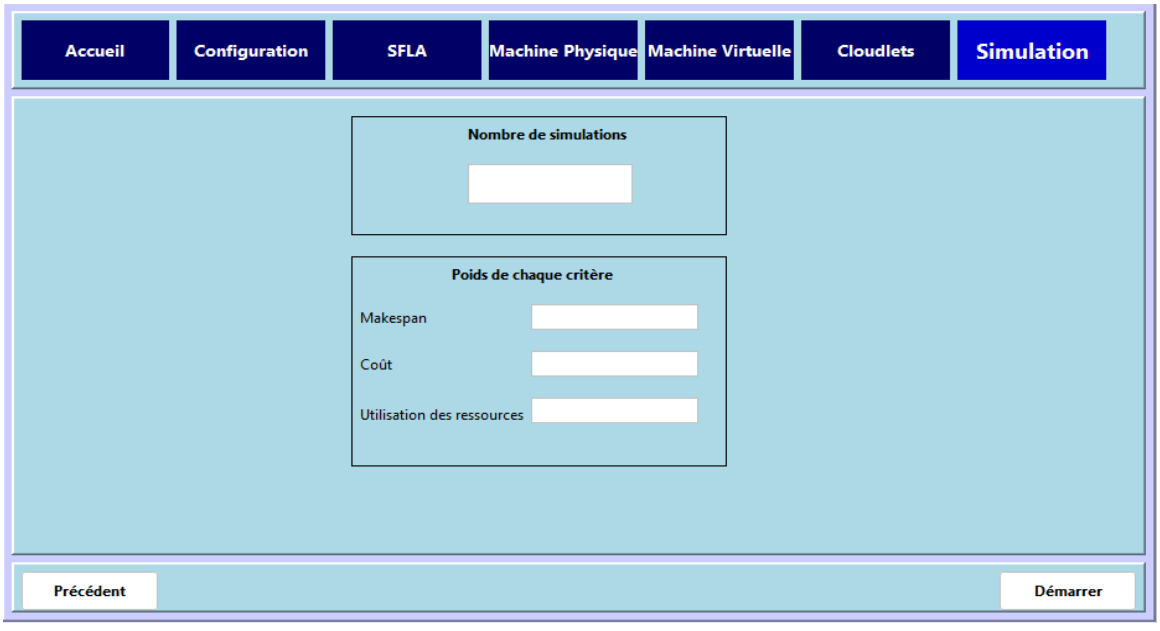

Figure 3.9 : interface de simulation.

3.6 Résultats obtenus et étude comparative

Cette section présente la description des différentes simulations et les résultats obtenus lors de l'exécution de l'algorithme SFLA tenant compte des méthodes multicritères utilisées dans ce PFE.

Les simulations ont été effectuées dans un environnement hétérogène, tel qu'indiqué dans le tableau 3.1, qui définit les valeurs associées à chaque paramètre.

Objet	Détails	Valeurs
Machines Virtuelles	Bonde passante	1000 Mb
	Nombre de processeurs	1
	Nombre de machines virtuelles	25
	RAM	1000
	Vitesse d'exécution (MIPS)	100-1000
	System d'exploitation	Linux
	Type de politique	Time shared
	Stockage	10000 MB
	VMM	Xen
Machines physiques	Bonde passante	3000 MB
	Stockage	1000000 MB
	Nombre d'hôtes	5
	RAM	8000 MB
	politique	Time shared
	processeurs	4 Dual core (4000 mips) 26 quad core (4000 mips)
Centres de données	nombre	2

Tableau 3.1 : les paramètres de simulation CloudSim[42].

Pour l les Cloudlets, nous avons choisi de varier la longueur selon le tableau 3.2.

Type de Cloudlet	Distribution
petite	35%
moyenne	40%
grande	5%
extra large	15%
énorme	5%

Tableau 3.2 : les paramètres de longueur des Cloudlets[42].

3.6.1 Comparaison de résultats en termes de makespan

La figure 3.10 représente la comparaison en termes de makespan entre les différentes méthodes analysées pour 25 Vms.

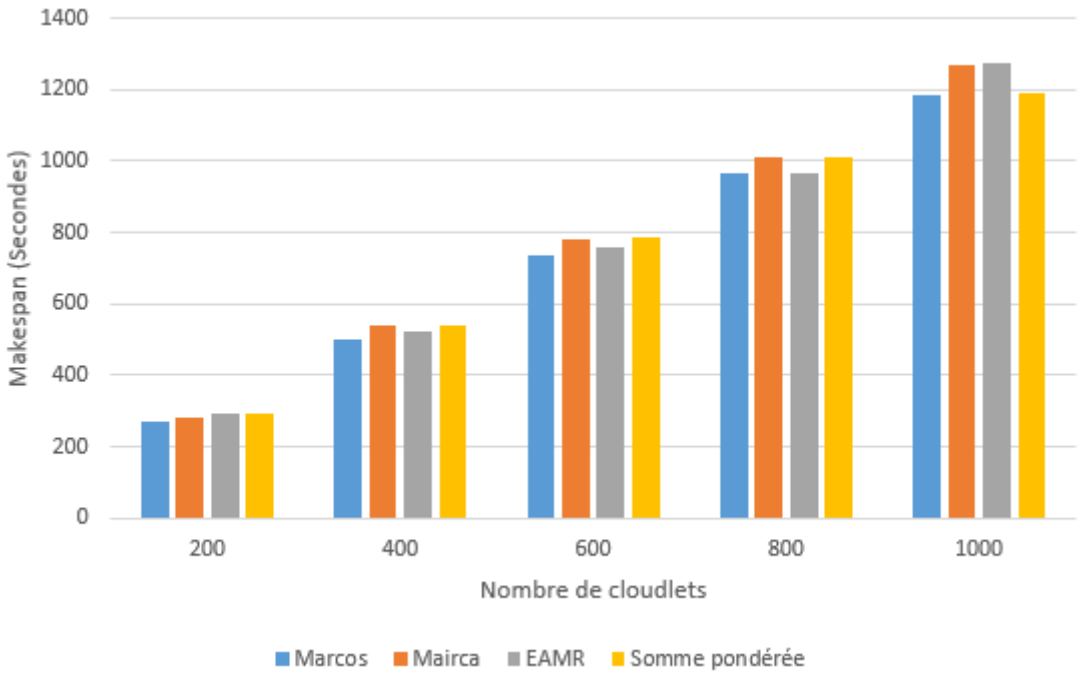

Figure 3.10 : comparaison en termes de makespan pour 25 Vms.

Le tableau 3.4 montre le détail des valeurs.

Nombre de cloudlets	200	400	600	800	1000
MARCOS	**273,216**	**500,444**	**733,56**	**967,66**	**1187,716**
MAIRCA	282,001	541,214	779,69	1009,696	1267,52
EAMR	291,955	524,457	761,262	**967,513**	1273,92
Somme pondérée	292,28	540,488	784,684	1012,754	1193,046

Tableau 3.3 : résultats de comparaison en termes de makespan pour 25 Vms.

Résultats :

En analysant les résultats présentés dans la figure 3.10 et le tableau 3.4, nous avons observé que la méthode MARCOS démontre une nette supériorité en termes de réduction du makespan, même en tenant compte de la variance du nombre de Cloudlets. D'autre part, nous avons constaté une variation des résultats de la méthode EAMR. Parfois, elle se rapproche des performances de la méthode MARCOS voire les dépasse, tandis que dans d'autres cas, elle obtient des résultats

inférieurs. Malgré cela, la méthode MARCOS reste supérieure à la méthode EAMR de manière générale.

En ce qui concerne les deux méthodes MAIRCA et la somme pondérée, elles ont obtenu des résultats inférieurs par rapport aux deux méthodes précédentes, mais elles présentent une convergence remarquable entre elles.

3.6.2 Comparaison de résultats en termes de coût

La figure 3.11 représente la comparaison en termes de coût entre les différentes méthodes analysées pour 25 Vms.

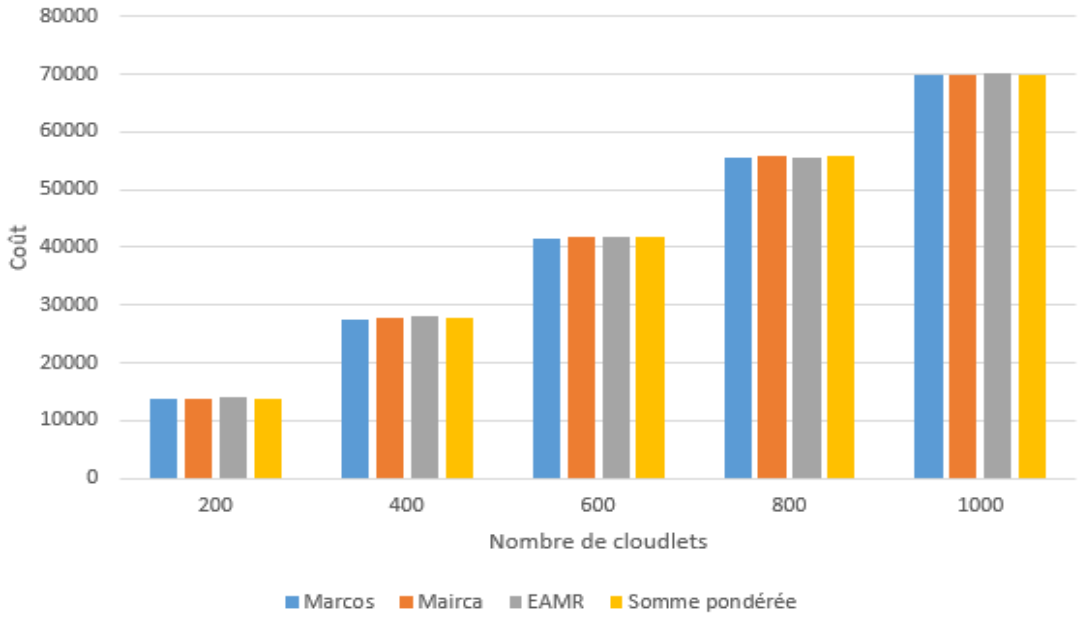

Figure 3.11 : comparaison en termes de coût pour 25Vms

Le tableau 3.5 montre le détail des valeurs.

Nombre de cloudlets	200	400	600	800	1000
MARCOS	**13765,911**	**27608,293**	**41616,682**	**55494,294**	**69745,367**
MAIRCA	13781,799	27903,221	41903	55787,351	69843,929
EAMR	14014,738	28153,4	41970,952	55662,374	70057,348
Somme pondérée	13824,655	27944,338	41852,448	55742,495	69892,721

Tableau 3.4 : résultats de comparaison en termes de coût pour 25 Vms.

Résultats :

En ce qui concerne la minimisation des coûts, nous avons constaté de que la méthode MARCOS obtient toujours les meilleurs résultats par rapport aux autres méthodes. De plus, nous avons remarqué que les résultats des méthodes MAIRCA et la somme pondérée sont inférieurs en terme de performance par rapport à MARCOS, mais elles se concurrencent entre elles. Enfin, la méthode EAMR présente les résultats les plus faibles.

3.6.3 Comparaison en termes de taux d'utilisation des ressources

La figure 3.12 représente la comparaison en termes de taux d'utilisation des ressources entre les différentes méthodes analysées pour 25 Vms.

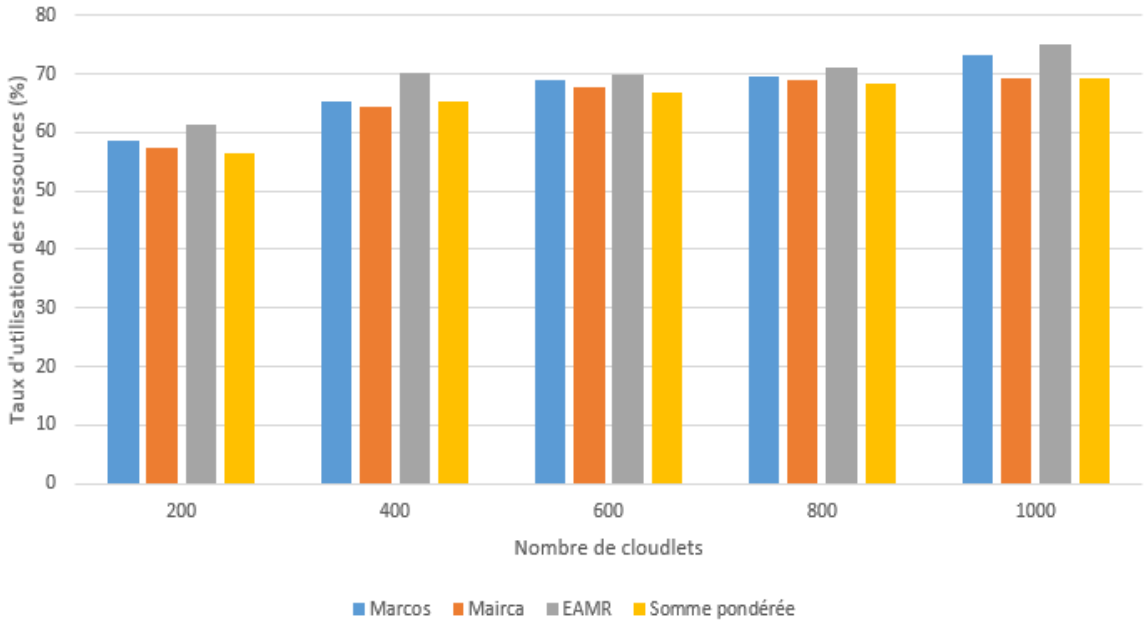

Figure 3.12 : comparaison de résultats en termes de taux d'utilisation des ressources pour 25Vms.

Le tableau 3.6 montre le détail des valeurs.

Nombre de cloudlets	200	400	600	800	1000
MARCOS	58,708	65,273	68,864	69,668	73,339
MAIRCA	57,345	64,459	67,614	69,067	69,316
EAMR	**61,322**	**70,045**	**69,853**	**71,007**	**74,997**
Somme pondérée	56,424	65,125	66,704	68,46	69,316

Tableau 3.5 : résultats de comparaison en termes de taux d'utilisation des ressources pour 25Vms.

Résultats :

En ce qui concerne le taux d'utilisation des ressources, la méthode EAMR a obtenu les résultats les plus élevés, suivie de près par la méthode MARCOS. Ensuite, les méthodes MAIRCA et la somme pondérée ont également montré une compétition entre elles, avec des résultats légèrement inférieurs mais proches.

Synthèse des résultats :

De manière générale, les résultats obtenus sont cohérents. Dans la plupart des cas, la méthode MARCOS présente de meilleures performances en termes de makespan et de coût. Cependant, en ce qui concerne le taux d'utilisation des ressources, la méthode EAMR se révèle meilleure.

En suivant les étapes de la méthode EAMR jusqu'à la dernière ou la fonction objectif de l'évaluation est définie. Le score d'évaluation est calculé en prenant le rapport entre l'écart entre les valeurs de la somme des critères à maximiser et les valeurs de la somme des critères à minimiser. Ce score est utilisé pour classer les solutions, où la solution avec le score le plus élevé est considérée comme la meilleure. Cette approche donne un avantage aux critères à maximiser indépendamment des poids spécifiques attribués à chaque critère. Cela explique les résultats obtenu pour le taux d'utilisation des ressources par la méthode EAMR ou elles étaient meilleures par rapport les autres méthodes.

3.7 Conclusion

Dans ce chapitre nous avons présenté l'implémentation réalisée dans le cadre de ce PFE ainsi que les résultats obtenus pour l'ordonnancement multi-objectif des tâches avec l'algorithme SFLA. Quatre méthodes d'aide à la décision multicritère qui sont MAIRCA, MARCOS, EAMR et la somme pondérée ont été évaluées.

Les résultats obtenus sont cohérents, la méthode MARCOS se distingue par son efficacité accrue en termes de makespan et de coût. Cependant, en ce qui concerne le taux d'utilisation des ressources, EAMR se révèle être une option plus avantageuse par rapport aux autres méthodes.

Conclusion générale

Dans le Cloud computing, l'ordonnancement des tâches est essentiel pour optimiser les performances et l'efficacité du système. Grâce à des mécanismes d'ordonnancement appropriés, les avantages du Cloud computing, tels que l'évolutivité, la flexibilité et la disponibilité des ressources, sont pleinement exploités. En répartissant efficacement les charges de travail entre les serveurs, il permet d'optimiser l'utilisation des ressources, de réduire les temps de traitement et d'améliorer la qualité de service pour les utilisateurs finaux. L'ordonnancement des tâches continue d'évoluer pour faire face aux défis croissants posés par les applications et les charges de travail complexes du Cloud computing, grâce à l'utilisation de divers algorithmes et techniques.

Dans le cadre de ce PFE, nous avons utilisé la métaheuristique SFLA pour l'évaluation multi-objectif à travers les trois métriques de QoS qui sont le makespan, le cout et l'utilisation des ressources en utilisant les méthodes d'aide à la décision multicritère qui sont : MARCOS, MAIRCA, EAMR et la somme pondérée.

Nous avons lancé plusieurs simulations en utilisant CloudSim pour comparer l'efficacité des méthodes MCDM dans le choix de la solution optimal et les résultats obtenus ont montré que le choix de la méthode d'aide à la décision a un impact sur l'optimisation de l'ordonnancement des taches dans le Cloud computing.

Il serait intéressant dans un travail future d'élargir le problème d'optimisation à d'autres algorithmes, d'autres critères tels que la fiabilité et le degré de déséquilibre et d'autres méthodes d'aide à la décision multicritères tels que TOPSIS.

Bibliographie

[1] what is Cloud computing?, disponible sur https://www.ibm.com/topics/Cloud-computing. Consulté le 26/03/2023.

[2] Jothy Rosenberg et Arthur Mateos, the Cloud at your service, chapter 1 what is Cloud computing? Disponible sur livebook.manning.com/book/the-Cloud-at-your-service/chapter-1/. Consulté le 26/03/2023.

[3] Peter Mell Timothy Grance, the NIST Special Publication 800-145, The NIST Definition of Cloud Computing. Disponible sur nvlpubs.nist.gov/nistpubs/legacy/sp/nistspecialpublication800-145.pdf. Consulté le 28/03/2023.

[4] SaaS vs. IaaS vs. PaaS: Differences, Pros, Cons and Examples.Disponible sur: https://www.techtarget.com/whatis/SaaS-IaaS-PaaS-Comparing-Cloud-Service-Models. Consulé le 28/03/2023.

[5] Yuri Musienko, IaaS vs PaaS vs SaaS: General Comparison.Disponible sur https://merehead.com/blog/iaas-vs-paas-vs-saas-general-comparison/. Consulté le 28/03/2023.

[6] Cloud Deployment Models. Disponible sur : https://www.geeksforgeeks.org/Cloud-deployment-models/.Consulté le28/03/2023.

[7] Amitava Nag, Cloud Computing: A Paradigm Shift in IT Infrastructure. Disponible sur https://www.researchgate.net/publication/270958592_Cloud_Computing_A_Paradigm_Shift_in_IT_Infrastructure .Consulté le 28/03/2023.

[8] Cloud Computing architecture .Disponible sur https://www.javatpoint.com/Cloud-computing-architecture. Consulté le 28/03/2023.

[9] what is virtualization? Et [5] Advantages of Virtualization in Cloud Computing. Disponible sur https://ecomnews.in/technology/virtualization-in-Cloud-computing/. Consulté le 31/03/2023.

[10] Types of Virtualization. Disponible sur https://www.geeksforgeeks.org/virtualization-Cloud-computing-types/. Consulté le 02/04/2023.

[11] Types of virtualization in Cloud computing .disponible sur https://www.interviewbit.com/blog/virtualization-in-Cloud-computing/. Consulté le 02/04/2023.

[12] BOUZARA Ayoub, Sécurité dans le Cloud Computing, mémoire de Master Informatique. Université Ibn Khaldoun Tiaret 2018.

[13] Amazon Web Services disponible sur https://fr.wikipedia.org/wiki/Amazon_Web_Services. Consulté le 05/04/2023.

[14] Microsoft azure disponible sur https://fr.wikipedia.org/wiki/Microsoft_Azure. Consulté le 05/04/2023.

[15] Microsoft azure vs aws vs Google Cloud disponible sur https://adex.ltd/microsoft-azure-vs-aws-vs-google-Cloud/. Consulté le 05/04/2023.

[16] Harfouchi Fatima, Contribution à l'optimisation par colonies d'abeilles artificielles. Thèse de Doctorat. Université M'hamed Bougara Boumerdès, 2019.

[17] Hettab Seddik, Latrache Abderrazake. Une Approche pour la Résolution et l'Optimisation d'un Problème de Tournées de Véhicule. Mémoire de Master Informatique. Centre Universitaire Abd Elhafid Boussouf, Mila 2021.

[18] Daoudi Mourad, Approches de résolution par les métaheuristiques de problèmes d'optimisation combinatoire Np-Difficiles. Thèse de doctorat Informatique USTHB 2012.

[19] S. Le Digabel. Cour Introduction aux métaheuristiques. Ecole Polytechnique de Montréal (2018) :16-17.

[20] NOUREDDINE Rachid, Métaheuristiques Sur Grilles de Calcul, Mémoire de Magister Informatique, Université des Sciences et de la Technologie d'Oran (2010) :19.

[21] Cour Optimisation Combinatoire. Disponible sur https://www.mcours.net/cours/pdf/hassbg/hassbgli902.pdf. Consulté le 27/04/2023.

[22] Cour Optimisation des Réseaux. CHAPITRE 6 : Méthodes de Résolution Exactes Heuristiques et Métaheuristiques. Université de m'sila 2023 disponible sur : https://elearning.univ-msila.dz/moodle/pluginfile.php/628992/mod_resource/content/1/Chapitre%206.pdf

[23] Mohammedi El hadi, Shuffled Frog Leaping Algorithm pour l'ordonnancement des tâches dans le Cloud computing, mémoire de Master Informatique, université de Tlemcen, 2022.

[24] Eusuff MM, Lansey KE. Optimization of water distribution network design using the shuffled frog leaping algorithm. Journal of Water Resources planning and management. 2003 May;129(3):210-225.

[25] Wang, al. An improved shuffled frog-leaping algorithm with extremal optimisation for continuous optimisation. Information Sciences. Volume 192, 1 June 2012, Pages 143-151.

[26] Alireza Rahimi-Vahed, Ali Hossein Mirzaei. A hybrid multi-objective shuffled frog-leaping algorithm for a mixed-model assembly line sequencing problem Computers & Industrial Engineering. Volume 53, Issue 4, November 2007, Pages 642-666.

[27] Taher Niknam, Mohammad rasoul Narimani, Masoud Jabbari, Ahmad Reza Malekpour" A modified shuffle frog leaping algorithm for multi-objective optimal power flow" Energy Volume 36, Issue 11, November 2011, Pages 6420-6432.

[28] Chunlei Liu, Zhenzhou Ji, Yingsen Hong," A Parallel Shuffled Frog Leaping Algorithm Based on Stem Regions Combinatorial Optimization for RNA Secondary Structure Prediction" Advances in Intelligent Systems Research March 2013.

[29] Benmammar Badr. "Quality of service optimization in orthogonal frequency division multiplexing-based cognitive radio systems based on shuffled frog leaping algorithm." Concurrency and Computation: Practice and Experience 34, no. 1 (2022): e6530.

[30] Stević, Ž., Pamučar, D., Puška, A., Chatterjee, P. (2020). Sustainable supplier selection in healthcare industries using a new MCDM method: Measurement alternatives and ranking according to Compromise solution (MARCOS), Computers & Industrial Engineering, Vol. 140, Article No. 106231.

[31] Trung, D.D, Thinh, H.X. A multi-criteria decision-making in turning process using the MAIRCA, EAMR, MARCOS and TOPSIS methods: A comparative study. Advances in Production Engineering & Management ISSN 1854-6250 Vol 16 , No 4 , December 2021 | pp 443–456.

[32] Pamucar, D.S., Pejcic Tarle, S., Parezanovic, T. (2018). New hybrid multi-criteria decision-making DEMATEL - MAIRCA model: Sustainable selection of a location for the development of multimodal logistics centre, Economic Research-Ekonomska Istraživanja Vol. 31, No. 1, 1641-1665

[33] Aksoy, E. (2021). An analysis on Turkey's merger and acquisition activities: MAIRCA method, Gümüşhane Üniversitesi Sosyal Bilimler Dergisi, Vol. 12, No. 1, 1-11.

[34] Keshavarz Ghorabaee, M., Zavadskas, E.K., Amiri, M., Antuchevience, J. (2016). Evaluation by an area-based method of ranking interval type-2 fuzzy sets (EAMRIT-2F) for multi-criteria group decision making. Transformations in Business & Economics, Vol. 15, No. 3, 76-95.

[35] Histoire de java .Disponible sur : https://web.maths.unsw.edu.au/~lafaye/CCM/java/javaintro.htm Consulté le 28/05/2023.

[36] Documentation netbeans. Disponible sur :http://doc.ubuntu-fr.org/netbeans. Consulté le28/05/2023.

[37] Documentation JfreeChart. Disponible sur : https://www.jfree.org/jfreechart/ Consulté le 28/05/2023.

[38] CloudSim Architecture. Disponible sur : https://www.geeksforgeeks.org/what-is-Cloudsim/ Consulté le 28/05/2023.

[39] Sonia Yassa. Allocation optimale multi-contraintes des workflows aux ressources d'un environnement cloud computing. Thèse de Doctorat. Université de Cergy-Pontoise 2014.

[40] Alsadie, Deafallah. "TSMGWO: Optimizing task schedule using multi-objectives grey wolf optimizer for cloud data centers." IEEE Access 9 (2021): 37707-37725.

[41] Xin-She Yang. Nature-Inspired Optimization Algorithms. Chapter 14 - Multi-Objective Optimization. 2014, Pages 197-211.

[42] Trung, D.D., Thinh, H.X., A multi-criteria decision-making in turning process using the MAIRCA, EAMR, MARCOS and TOPSIS methods: A comparative study, Advances in Production Engineering & Management ISSN 1854-6250 Volume 16 | Number 4 | December 2021 | pp 443–456.

Résumé

Le Cloud computing est la fourniture de différents services informatiques à la demande sur Internet. Dans ce contexte, une prise de décision multicritères pour l'ordonnancement des tâches est nécessaire pour déterminer la meilleure alternative parmi de nombreuses possibilités. Dans ce travail, nous avons évalué la métaheuristique SFLA en utilisant le simulateur CloudSim dans un contexte multi-objectif tenant compte de trois paramètres qui sont le makespan, le coût et l'utilisation des ressources. En plus de la somme pondérée, trois méthodes d'aide multicritères à la décision qui sont MARCOS, MAIRCA et EAMR ont été utilisées pour l'évaluation des individus. Les résultats obtenus sont très satisfaisants.

Mots clés : Cloud computing, SFLA, CloudSim, MARCOS, MAIRCA, EAMR.

Abstract

Cloud computing is the provision of various IT services on demand over the Internet. In this context, multi-criteria decision making for task scheduling is necessary to determine the best alternative among many possibilities. In this work, we evaluated the SFLA metaheuristics using the CloudSim simulator in a multi-objective context taking into account three parameters: makespan, cost and resource utilization. In addition to the weighted sum, three multi-criteria decision support methods, MARCOS, MAIRCA and EAMR, were used for the assessment of individuals. The results obtained are very satisfactory.

Keywords: Cloud computing, SFLA, CloudSim, MARCOS, MAIRCA, EAMR.

ملخص

الحوسبة السحابية هي توفير خدمات تكنولوجيا المعلومات المختلفة عند الطلب عبر الإنترنت. وفي هذا السياق، فإن اتخاذ قرارات متعددة المعايير لجدولة المهام أمر ضروري لتحديد أفضل بديل من بين إمكانيات كثيرة. في هذا العمل، قمنا بتقييم SFLA باستخدام محاكي CloudSim في سياق متعدد الأهداف مع الأخذ في الاعتبار ثلاثة معايير: وقت التنفيذ والتكلفة واستخدام الموارد. وبالإضافة إلى المتوسط المرجح، استُخدمت ثلاث طرائق متعددة المعايير لدعم اتخاذ القرارات، هي MARCOS، وMAIRCA، وEAMR، لتقييم الأفراد. النتائج التي تم الحصول عليها مرضية للغاية.

الكلمات الرئيسية: الحوسبة السحابية,SFLA, CloudSim, MARCOS, MAIRCA, EAMR .