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 (R.S.D)

Thème :

Ordonnancement des tâches dans le cloud computing : utilisation d'une méthode exacte

Réalisé par :

- SENOUSI Saliha

Présenté le 25 juin 2024 devant le jury composé de :

- **Mr. SETTOUTI Ahmed Khalid Yassine (Président)**
- **Mme. MEHIAOUI Asma (Examinatrice)**
- **Mr. BENMOUNA Youcef (Encadrant)**

Année universitaire : 2023-2024.

Remerciements

Tout d'abord **El-hamdoulillah**, c'est lui qui m'a donné l'effort, le temps et (el-taofik) pour réaliser ce mémoire et m'a ouvert les bonnes portes et les justes réflexes, sans sa miséricorde, ce travail n'aurait pas été accompli.

Je profite de cette occasion afin de transmettre mes sincères remerciements et mes profondes reconnaissances À :

- Mes formidables parents qui m'ont aidé à surmonter les obstacles par leurs encouragements et leurs prières.
- Mon encadrant **Mr. BENMOUNA Youcef** qui m'a donné l'opportunité de réaliser ce sujet sous sa direction, et ses conseils fascinant ainsi que son temps consacré tout au long du travail.
- Tous mes enseignants qui m'ont permis d'arriver là où nous sommes aujourd'hui.
- Aux membres de jury **Mr. SETTOUTI Ahmed Khalid Yassine et Mme.**

MEHIAOUI Asma qui m'ont honoré de leur présence en tant qu'examinateurs. Je tiens à remercier toutes les personnes qui ont participé de près ou de loin au bon déroulement de ce travail.

Dédicace

Je d´dédie cet humble travail :

A ma mère, avec tous mes sentiments de respect, d'amour et de reconnaissance pour tous les sacrifices déployés pour m´élever dignement et assurer mon éducation dans les meilleures conditions. `

A mon père qui était toujours là pour moi tout au long de mes études.

A ma sœur Houria, mes frères Mohamed et Abd el Hakim.

A mes chères amies Fatima, Mebarka, Maria, Sara ..... et enfin à tous ceux qui m'ont aidé de près ou de loin.

Saliha SENOUSI

Tables des matières :

Tables des figures :

Liste des tableaux :

Liste des abréviations :

IaaS:	Infrastructure as a Service
PaaS:	Platform as a Service
SaaS:	Software as a Service
AWS:	Amazon Web Services
VM:	Virtual machine
CS:	Cuckoo Search
B&B:	Branch and Bound
MS:	Makespan

Résumé:

L'optimisation des ressources et la réduction des temps d'exécution sont des impératifs majeurs dans l'ordonnancement des tâches dans le cloud computing. Pour relever ce défi complexe, la méthode Branch and Bound s'avère être une approche algorithmique très efficace. Elle découpe l'espace de recherche en sousensembles plus gérables, appelés branches, et itère à travers chaque branche pour déterminer la solution la plus adaptée. Dans le cadre du cloud computing, cela se traduit par la segmentation des tâches à exécuter en sousgroupes pouvant être répartis sur différents serveurs ou machines virtuelles. En analysant systématiquement chaque combinaison de tâches et de ressources, cette méthode permet d'identifier l'allocation optimale des ressources et de concevoir un planning d'exécution efficace. Cette approche contribue ainsi à minimiser les temps d'attente et à maximiser l'utilisation des ressources disponibles, améliorant ainsi les performances globales du système. Les prestataires de services cloud peuvent ainsi proposer des solutions d'ordonnancement efficaces qui répondent précisément aux besoins de leurs clients, tout en optimisant la rentabilité et l'efficacité opérationnelle de leurs infrastructures.

Mots clés: Optimization des resources, Branch and Bound, cloud computing,performance, infrastructure…

Abstract:

The scheduling of tasks in cloud computing is crucial for optimizing resources and minimizing execution times. The branch and bound method is an efficient algorithmic approach to solving this complex problem. It involves dividing the search space into smaller subsets, known as branches, and iteratively evaluating each branch to find the optimal solution. In the context of cloud computing, this means dividing the set of tasks to be executed into subsets that can be distributed across different servers or virtual machines. By systematically evaluating each combination of tasks and resources, the branch and bound method can identify the best resource allocation and the most efficient execution plan. This helps optimize system performance by minimizing waiting times and maximizing the use of available resources. By employing this approach, cloud service providers can offer effective scheduling solutions that meet their clients' needs while maximizing the operational efficiency of their infrastructures.

Keywords: cloud computing,optimizing resources,branch and bound method,tasks,virtual machines,performance,service providers,infrastructure

ملخص:

جدولة المهام في الحوسبة السحابية أمر أساسي لتحسين الموارد وتقليل أوقات التنفيذ. الطريقة الفرعية والمقيدة هي نهج خوارزمي فعآل لحل هذه المشكلة المعقدة. ينطوي ذلك على تقسيم مساحة البحث إلى مجموعات فرعية أصغر، تسمى فروعًا، وتقييم كل فرع تدريجياً للعثور على الحل الأمثل. في سياق الحوسبة السحابية، يعني ذلك تقسيم مجموعة المهام التي يجب تنفيذها إلى مجموعات فرعية يمكن توزيعها عبر خوادم أو آلات افتراضية مختلفة. من خلال تقييم كل تركيبة من المهام والموارد بشكل منهجي، يمكن للطريقة الفرعية والمقيدة تحديد أفضل تخصيص للموارد والخطة الأكثر فعالية للتنفيذ. يساعد هذا في تحسين أداء النظام عن طريق تقليل أوقات الانتظار وزيادة استخدام الموارد المتاحة. من خلال تبني هذا النهج، يمكن لمزودي خدمات السحابة تقديم حلول جدولة فعالة تلبي احتياجات عملائهم بشكل دقيق مع تحسين الكفاءة التشغيلية لبنيتهم التحتية

الكلمات الرئيسية:جدولة المهام، نهج خوارزمي فعآل، الحوسبة السحابية، المهام...الخ

Introduction générale :

Le Cloud Computing s'impose très rapidement comme étant un standard pour l'hébergement des applications et des services logiciels. La plupart des entreprises, individus et même des corps gouvernementaux se réfugient vers le Cloud en raison de la réduction des prix, la facilité de développement et le stockage illimité. Plusieurs applications dans de nombreux domaines contiennent généralement de nombreuses tâches. Ces tâches requièrent pour leurs exécutions sur le Cloud un ordonnancement, c'est l'affectation des tâches aux ressources disponibles sur la base des exigences et des caractéristiques des tâches. C'est un processus très important pour un fonctionnement efficace du Cloud.

L'ordonnancement de tâche dans un Cloud est considéré parmi les problèmes les plus difficiles a traiter. Ce problème est d'autant plus difficile lorsqu'il y a plusieurs facteurs à prendre en compte, donc c'est un problème d'optimisation combinatoire, où 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 notre travail est de développer un algorithme d'ordonnancement des tâches basé sur la méthode « Branch and Bound» pour l'optimisation le makespan.

Dans le premier chapitre, nous présenterons quelques notions de base sur le Cloud Computing. Nous présenterons quelques notions fondamentales tel que les définitions et les caractéristiques, l'évolution du cloud, les services, les modèles de déploiement …

Chapitre1 : Introduction sur le Cloud Computing

Dans le second chapitre, nous présenterons quelques notions fondamentales sur les méthodes de résolutions notamment leurs définitions, mode de fonctionnement, caractéristiques et une classification. Ensuite nous présenterons de manière détaillée la méthode « Branch and Bound » utilisée dans notre projet.

Dans le dernier chapitre, nous expliquerons le choix du simulateur utilisé et l'approche adoptée pour la réalisation de notre travail, nous présenterons ensuite les résultats trouvés avec des comparaisons.

Chapitre1 : Introduction sur le Cloud Computing

Chapitre1 :
CLOUD COMPUTING

Chapitre 1 : introduction sur le Cloud Computing

1.1 Introduction :

Le Cloud Computing est une force de transformation qui redéfinit la manière dont les entreprises conçoivent, configurent et exploitent leurs ressources informatiques.Ce chapitre décrit le monde dynamique du cloud computing, un paradigme innovant qui transcende les frontières traditionnelles de l'informatique d'entreprise. Du libre-service à la demande à la mise en commun agile des ressources, ce modèle informatique offre une large gamme de services sur Internet et restructure fondamentalement la manière dont les organisations interagissent avec leur infrastructure. Ce chapitre explore les fondamentaux du cloud computing, ses caractéristiques uniques, ses modèles de services et ses avantages qu'il apporte tout en considérant les défis associés à cette innovation technologique.

1.2 Définition du Cloud computing :

Le Cloud Computing est un modèle permettant un accès réseau omniprésent, pratique et à la demande à un pool partagé de ressources informatiques configurables [1]. Ces ressources peuvent être rapidement approvisionnées et mises à disposition avec un effort de gestion minimal ou une interaction limitée avec le fournisseur de services. En d'autres termes, le Cloud Computing est une approche qui vise à fournir des applications, de la puissance de calcul, des moyens de stockage, etc., sous forme de services mutualisés, dématérialisés, contractualisés, évolutifs et en libre-service. Cette définition souligne l'aspect essentiel du Cloud Computing, qui transcende les contraintes matérielles et logicielles traditionnelles, offrant une flexibilité et une accessibilité sans précédent aux ressources informatiques.

1.3 Historique du Cloud Computing :

La mise en réseau des ordinateurs pour accroître la puissance de calcul et l'espace de stockage disponible n'est pas une innovation récente. Dans les années 1950, les mainframes représentaient déjà une avancée vers le cloud Computing. À cette époque, les utilisateurs pouvaient accéder à l'ordinateur central à travers plusieurs terminaux au sein de l'organisation, que ce soit dans les entreprises ou les universités, pour exploiter ses capacités. Initialement, cela impliquait une utilisation partagée du temps,

obligeant les utilisateurs à réserver des créneaux temporels pendant lesquels ils étaient autorisés à exploiter les capacités de l'ordinateur central pour leurs calculs.

Au fil des décennies, la virtualisation a émergé, permettant la construction abstraite d'instances de calcul de manière purement virtuelle. Avec l'avènement d'Internet, ces environnements virtualisés sont devenus accessibles en ligne pour tous. À partir des années 1990, ces modèles ont été progressivement mis à disposition d'un public plus vaste sur le marché. C'est également à cette époque que le concept de Cloud Computing a gagné en popularité. Cependant, ce n'est qu'au cours du nouveau millénaire que les entreprises et les particuliers ont commencé à manifester un intérêt croissant pour cette technologie. Les premiers services de Cloud Computing étaient souvent des offres individuelles, telles que des espaces de stockage pour le partage de fichiers ou des applications telles que les feuilles de calcul et les documents collaboratifs de Google. Simultanément, Amazon a ouvert ses vastes fermes de serveurs à d'autres utilisateurs, donnant naissance à AWS, qui permet à d'autres entreprises d'utiliser l'infrastructure du géant du commerce électronique pour exécuter leurs propres logiciels.

Aujourd'hui, le Cloud Computing est devenu une composante intégrale de la vie quotidienne pour de nombreuses personnes. La plupart des smartphones, et plus largement, l'internet des objets, maintiennent une connexion constante avec le Cloud. Les utilisateurs peuvent prendre une photo avec l'appareil photo de leur smartphone, qui est automatiquement téléchargée dans le Cloud d'Apple ou de Google, offrant ainsi un accès facile depuis d'autres appareils.[3]

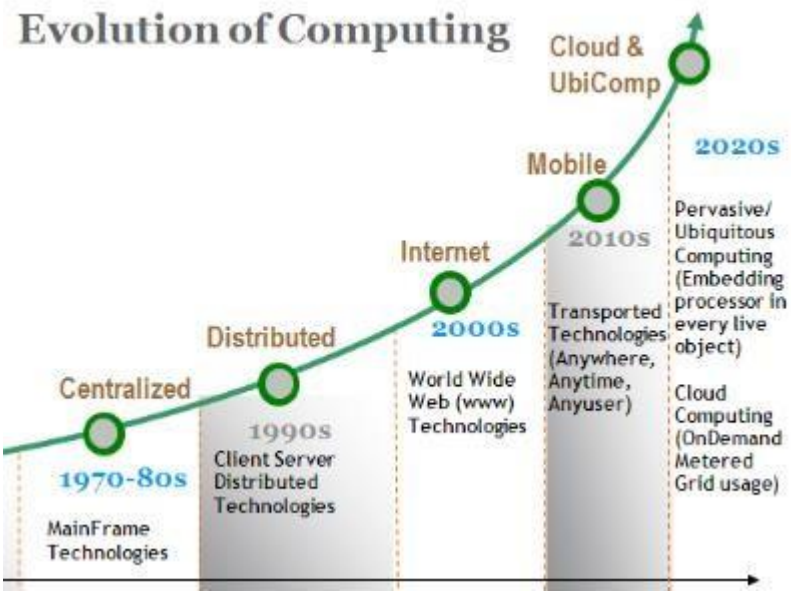

Figure2 : Évolution de l'information vers le Cloud Computing [4]

1.4 Caractéristiques du Cloud computing :

- **Multi location et mutualisation des ressources :**

- La capacité à fournir des services à plusieurs utilisateurs finaux via une seule instance de programme.
- Permet le partage des mêmes ressources informatiques entre plusieurs clients.
- Utilisation de mécanismes informatiques standards pour favoriser la compatibilité avec différentes plates-formes.
- Brise les barrières géographiques en permettant un accès sur diverses zones.

- **Élasticité et évolutivité rapides :**

- Possibilité de libérer de manière élastique les capacités du cloud en fonction de la demande des utilisateurs.
- Permet d'ajuster rapidement les services en fonction des besoins changeants.
- Offre des solutions de mutualisation de ressources abordables pour réduire les coûts informatiques.
- Favorise la mise en commun dynamique des ressources pour fournir des services à plusieurs consommateurs.

- **Service de mesure et de reporting :**

- Fournit des capacités de mesure pour surveiller et optimiser l'utilisation des ressources cloud.
- Apporte une transparence aux fournisseurs et aux consommateurs.

- **Automation :**

- Automatise la gestion et la modification des services cloud, minimisant l'interaction humaine.
- Facilite une expansion rapide des services grâce à des configurations automatisées.

- **Résilience :**

- Assure une disponibilité continue des serveurs et la récupération en cas d'interruption.
- Implémente des stratégies de gestion des catastrophes avec des nœuds cloud de sauvegarde.

- **Accès au grand réseau :**

- Permet aux utilisateurs d'accéder aux services cloud depuis n'importe où avec une connexion Internet.
- Favorise le téléchargement de données vers le cloud à partir de divers appareils.

- **Travailler depuis n'importe quel endroit :**

- Favorise le travail à distance, permettant aux utilisateurs de fonctionner et de se connecter depuis n'importe quel emplacement.

- **Structure de paiement confortable :**

- Offre une structure de paiement flexible basée sur les caractéristiques choisies par le client.

- Simplifie les options de paiement pour économiser du temps et des coûts.

- **L'excellence du service :**

- Propose une large gamme de services avec des accords de niveau de service partagés.
- Assure la disponibilité continue et la bande passante aux utilisateurs finaux.

- **Entretien facile :**

- Gestion de la maintenance de manière transparente sans implication du client.
- Mise à jour régulières permettant d'optimiser les capacités du Cloud avec des temps d'arrêt minimes.

- **Flexibilité :**

- Offre aux utilisateurs la flexibilité de l'hébergement de données dans le cloud dédié.
- Élimine la nécessité de changer fréquemment de fournisseur de services.

- **Économique et sécurité :**

- Permet des économies substantielles sur les dépenses informatiques avec des coûts modiques.
- Renforce la sécurité moyennant des frais nominaux.

- **Disponibilité :**

- Offre des services résilients disponibles 24h/7.
- Récupération rapide en cas de temps d'arrêt, assurant la sécurité Des informations stockées.

- **Libre-service a la demande :**

Permettant aux utilisateurs finaux d'accéder instantanément à des capacités informatiques. Il intègre une fonction de surveillance de la disponibilité des serveurs et offre un stockage réseau préconfiguré pour permettre aux utilisateurs de superviser leurs ressources informatiques. Fonctionnant sur un modèle en libre-service, le Cloud Computing habilite les utilisateurs finaux à prendre des décisions éclairées en matière d'utilisation des services, renforçant ainsi leur compréhension des capacités disponibles

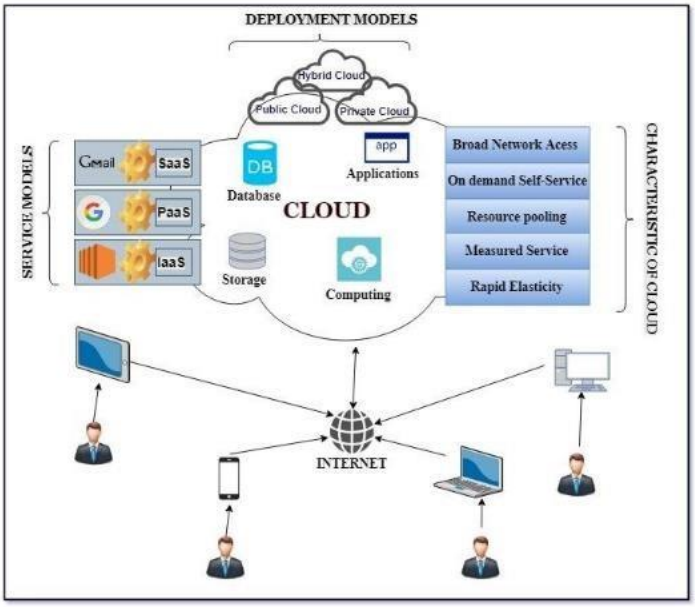

Figure 3 : L'environnement du Cloud Computing [5]

1.5 Technologies du Cloud Computing :

Le cloud Computing exploite différentes technologies telles que la virtualisation, l'architecture orientée services et les services web. Il est parfois confondu avec d'autres paradigmes informatiques tels que le Grid Computing, l'Utility Computing et l'Autonomic Computing, partageant chacun des similitudes avec le Cloud Computing.

Virtualisation :

La virtualisation permet le partage d'une instance physique d'une application ou d'une ressource entre plusieurs clients ou organisations.

Son objectif principal est de masquer les caractéristiques physiques des ressources informatiques pour permettre aux systèmes, applications ou utilisateurs finaux d'interagir avec ces ressources de manière standard.

La virtualisation constitue la base du Cloud Computing, offrant aux fournisseurs la possibilité de personnaliser la plate-forme pour répondre aux besoins des clients, notamment en exposant des applications en cours d'exécution dans des machines virtuelles.

Grid computing :

L'informatique en grille est une structure de l'informatique distribuée où des ressources informatiques de divers emplacements sont connectées pour atteindre un objectif commun.

Les systèmes de grille facilitent le partage des ressources à grande échelle via une informatique en cluster distribuée, divisant les tâches en morceaux distribués aux CPU.

Informatique utilitaire :

Basée sur le modèle de paiement à l'utilisation, l'informatique utilitaire fournit des ressources de calcul à la demande en tant que service facturé.

Les services informatiques tels que le Grid computing et le cloud computing reposent sur le concept d'informatique utilitaire, permettant une utilisation maximale des ressources et une minimisation des coûts d'exploitation grâce à l'approvisionnement à la demande et au paiement à l'usage.

Autonomic computing :

Vise à construire des systèmes informatiques capables de s'auto-administrer en s'adaptant à des changements internes et externes sans intervention humaine.

Bien que le cloud computing présente certaines caractéristiques autonomes, telles que l'approvisionnement automatique des ressources, son objectif principal est de réduire les coûts plutôt que de diminuer la complexité du système.

1.6 Services du Cloud Computing :

Les solutions de Cloud computing se regroupent en trois catégories principales :

1.6.1 Infrastructure as a Service (IaaS) :

Il s'agit de la catégorie fondamentale des services de cloud computing. Avec l'IaaS, vous louez une infrastructure informatique (serveurs, machines virtuelles, stockage, réseaux, systèmes d'exploitation) auprès d'un prestataire de services Cloud, avec une tarification basée sur l'utilisation.

1.6.2 Platform as a Service (PaaS):

Le terme Plateforme en tant que Service (PaaS, Platform-as-a-Service) désigne les services de Cloud Computing qui fournissent un environnement à la demande pour le développement, le test, le déploiement et la gestion d'applications logicielles. Le PaaS est conçu pour permettre aux développeurs de créer rapidement des applications web ou mobiles sans se soucier de la configuration ou de la gestion de l'infrastructure sous-jacente, telle que les serveurs, le stockage, le réseau et les bases de données nécessaires au développement.

1.6.3 Software as a Service (SaaS):

Le logiciel en tant que service (SaaS, Software-as-a-Service) est une méthode de distribution d'applications logicielles via Internet, à la demande et généralement sur abonnement. Avec le SaaS, les fournisseurs de services cloud hébergent et gèrent les applications logicielles ainsi que l'infrastructure sous-jacente, prenant en charge la maintenance telle que les mises à jour logicielles et l'application de correctifs de Sécurité. Les utilisateurs accèdent à l'application via Internet, généralement via un navigateur web sur leur téléphone, tablette ou ordinateur personnel.

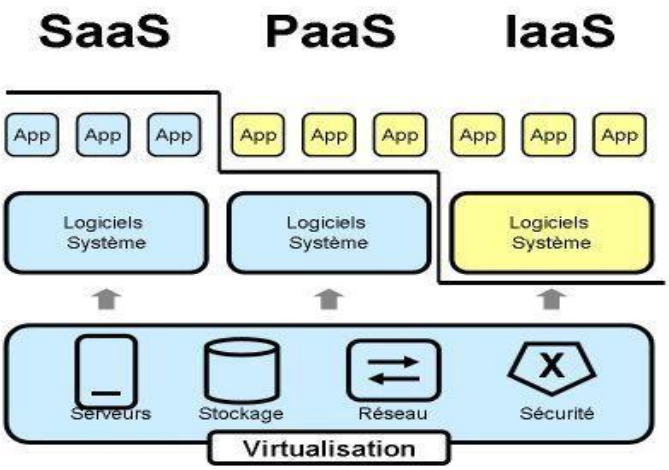

Figure 4 : Services du cloud computing.[6]

1.7 Modèles de déploiement :

Un modèle de déploiement Cloud représente une "configuration" spécifique des paramètres d'environnement cloud tels que la taille du stockage, l'accessibilité et la propriété. Il existe quatre modèles de déploiement de Cloud Computing :

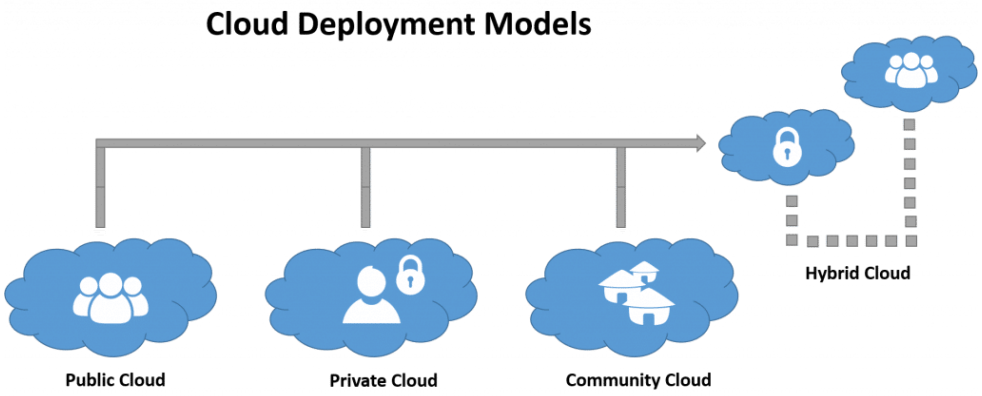

Figure 5 : Modèles de déploiements du cloud computing.[7]

1.7.1 Cloud publique :

Dans ce schéma de déploiement, le fournisseur de solutions cloud est externe, possédant sa propre infrastructure, et ses services sont ouverts à tous (sous réserve de paiement).

Cette utilisation des systèmes informatiques permet aux entreprises de se focaliser sur leurs processus métiers fondamentaux en déléguant la gestion de leurs systèmes informatiques à des fournisseurs distants.

Les services publics offrent divers avantages, notamment la mobilité grâce à un accès rapide et omniprésent aux ressources, ainsi que le partage de ressources telles que des machines de haute performance accessibles à partir de clients légers. Des acteurs majeurs de ce service incluent Google, Amazon et Salesforce.

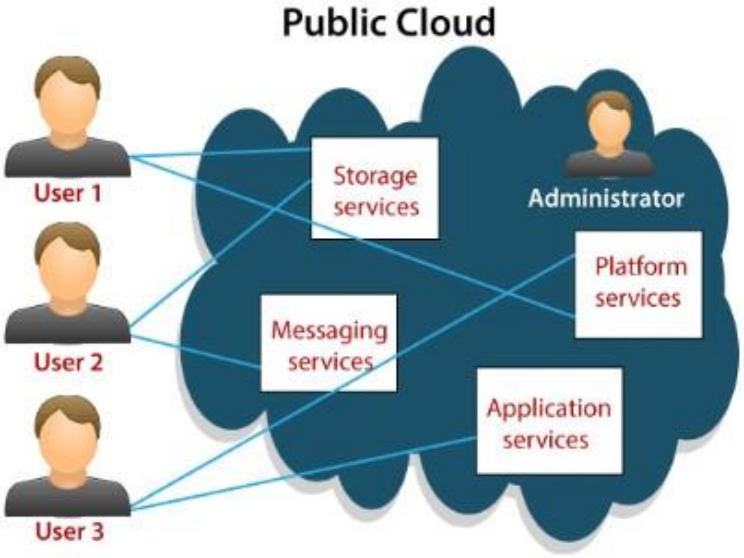

Figure6 : Schéma de l'architecture matérielle du Cloud publique [8]

1.7.2 Cloud privé :

Est un réseau informatique réservé exclusivement à une entreprise. Les données et les processus sont gérés en interne, évitant les restrictions de bande passante et les risques de sécurité associés au Cloud publique. L'accès est limité à un nombre restreint d'utilisateurs, ce qui renforce la sécurité et la résilience. Le Cloud privé peut être administré par l'organisation elle-même ou par un tiers, mais son utilisation est

exclusive à cette organisation. Il existe deux types de cloud privé : le Cloud privé interne et le Cloud privé externe.

- **Cloud privé interne** : ce type de cloud est géré et utilisé au sein de l'organisation, sur des infrastructures qui lui appartiennent. La figure suivante illustre un Cloud privé interne.

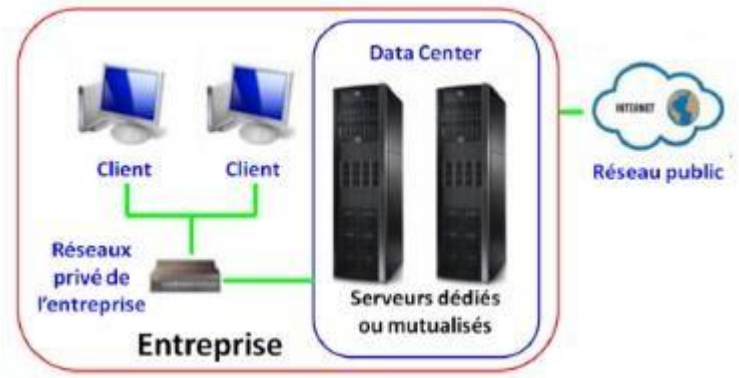

Figure 7 : Schéma de l'architecture matérielle du Cloud privé interne [9]

- **Cloud privé externe** : ce type de cloud est utilisé par l'organisation, mais hébergé chez un prestataire externe. La figure suivante détaille le schéma matériel d'un cloud privé externe.

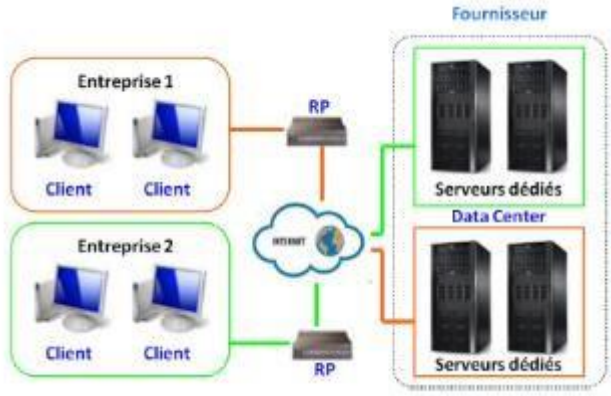

Figure 8 : Schéma de l'architecture matérielle du Cloud privé externe [9]

1.7.3 Cloud communautaire :

Dans ce modèle de Cloud, les ressources, les services et la propriété sont partagés au sein d'une communauté (par exemple, à l'échelle d'un État, d'une ville, d'une académie, d'un GIE, etc.) [10]. Dans un Cloud communautaire, l'infrastructure est déployée exclusivement pour un groupe d'entreprises ou d'organisations partageant

des intérêts communs. L'administration du système peut être assurée par l'une ou plusieurs des organisations qui partagent les ressources du Cloud.

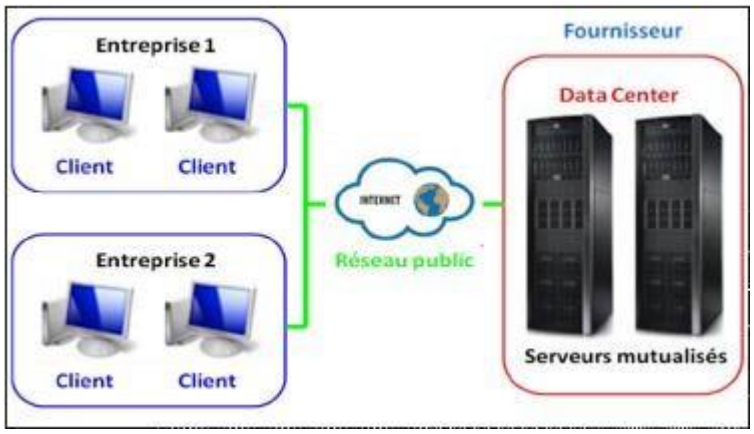

Figure 9 : Schéma de l'architecture matérielle du Cloud Communautaire [9]

1.7.4 Cloud hybride :

Ce modèle représente une combinaison de deux ou trois des modèles décrits ci-dessus. L'avenir devrait confirmer l'émergence de ce type de Cloud, avec une combinaison de Cloud privé et publique.

Parameters\Type	Public Cloud	Private Cloud	Hybrid Cloud	Community Cloud
Description	In public cloud, services are available for public users.	Private cloud is build up with existing private infrastructure. This type of cloud has some authentic users who can dynamically provision the resources.	Hybrid cloud is a heterogeneous distributed system, resulting from a private cloud, which incorporates diff erent types of services and resources from public clouds.	Different types of cloud are integrated together to meet a common or particular need for some organizations.
Scalability	Very High	Limited	Very High	Limited
Reliability	Moderate	Very High	Medium to High	Very High
Security	Totally on D ependds service provider	High class security	Secure	Secure
Performance	Low to medium	Good	Good	Very Good
Cost	Cheaper	High Cost	Costly	Costly

Examples	Amazon EC2, Google AppEngine	VMWare, Microsoft, KVM, Xen	IBM, HP, VMWar vCloud, Eucalyptus	SolaS Community Cloud, VMWare

Tableau 2: Comparaison entre Cloud Public, Privé, Hybride et communautaire [10]

Figure 10 : Schéma de l'architecture matérielle du Cloud hybride [8]

1.8 Acteurs du cloud computing :

Les acteurs du cloud computing sont des entités qui interviennent dans l'écosystème du cloud, jouant des rôles spécifiques dans la fourniture, la gestion et l'utilisation des services cloud. Voici une liste des principaux acteurs du cloud Computing :

1.8.1 Consommateur (Cloud Consumer) : désigne une personne ou une organisation utilisant un service fourni par un prestataire, le terme "utilisateur" est également employé pour décrire un consommateur.

1.8.2 Fournisseur (Cloud Provider) : correspond à une personne, une organisation ou une entité chargée de mettre un service à disposition des parties intéressées.

1.8.3 Courtier (Cloud Broker) : désigne une entité qui gère l'utilisation, la performance et l'approvisionnement des services, tout en négociant les relations entre les fournisseurs et les consommateurs.

1.8.4 Auditeur (cloud Auditor) : représente une entité capable de réaliser une évaluation indépendante des services cloud, portant notamment sur la performance et la sécurité du cloud.

1.8.5 Transporteur (Cloud Carrier) : décrit un intermédiaire fournissant la connectivité et le transport des services depuis les fournisseurs vers les consommateurs.

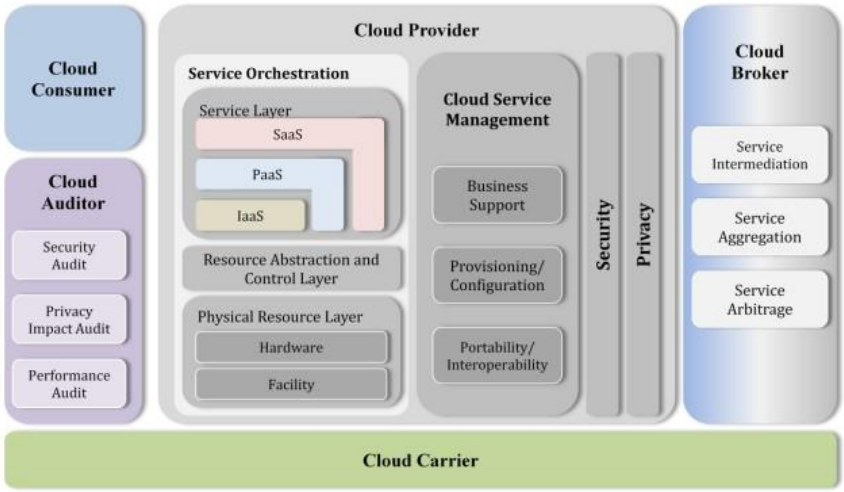

Figure11 : Acteurs du cloud computing [11]

1.9 Avantages du Cloud Computing :

Parmi les avantages les plus significatifs offerts par cette technologie, on peut citer :

- **L'accès facile à distance :** le Cloud facilite la collaboration mondiale et permet aux utilisateurs d'accéder aux services depuis n'importe quel endroit, favorisant la mobilité et la flexibilité.
- **Economisation des couts :** le Cloud Computing permet des économies substantielles en éliminant le besoin d'investir massivement dans une infrastructure physique.
- **La flexibilité des types de services :** les modèles de services cloud (IaaS, PaaS, SaaS) offrent une flexibilité considérable, permettant aux entreprises de choisir des solutions adaptées à leurs besoins spécifiques.
- **Souplesse pour l'entreprise :** gérer les problèmes informatiques devient une tâche simple sans nécessité d'engagements à long terme.
- **Lancement rapide :** grâce au cloud computing, il devient possible de mettre rapidement à l'épreuve le business plan, et ce, à moindre coût et en toute simplicité.

1.10 Inconvénients du cloud computing :

Bien que le Cloud Computing offre de nombreux avantages, il présente également des défis tels que :

- **Problème de la connectivité :** un accès Internet constant est nécessaire pour utiliser les services cloud, ce qui peut poser problème dans les zones à faible connectivité.
- **La sécurité des données :** la confiance dans la protection des données reste une préoccupation, surtout pour les données sensibles.
- **La perte de contrôle :** les entreprises peuvent perdre une partie du contrôle direct sur leurs ressources informatiques.
- **Impact potentiel sur les performances des applications :** l'utilisation d'un Cloud public ne garantit en aucun cas une amélioration des performances des applications.
- **Fiabilité des services Cloud :** le passage à un Cloud comporte des risques significatifs, notamment lorsque l'application hébergée revêt une importance stratégique ou contient des informations sensibles des clients.

1.11 Conclusion :

Le Cloud Computing a révolutionné la manière dont les organisations abordent l'informatique. Ce modèle offre des solutions flexibles et économiques, stimulant l'innovation et la croissance. Ce chapitre a introduit les concepts fondamentaux du Cloud Computing, jetant les bases pour explorer plus en détail les aspects techniques, les meilleures pratiques et les tendances émergentes de cette technologie passionnante.

Chapitre1 : Introduction sur le Cloud Computing

Chapitre2 :
Méthodes d’optimisations

Chapitre2 : méthodes d’optimisations

2.1 Introduction :

Les méthodes d'optimisation sont des techniques utilisées pour chercher la meilleure solution possible à un problème donné, soit en minimisant ou maximisant une fonction objective sous contraintes. Ces méthodes sont largement utilisées dans de nombreux domaines tels que l'ingénierie, les sciences économiques, les sciences informatiques, etc… L'objectif principal de ces méthodes est de trouver la valeur optimale d’une fonction objective en explorant un ensemble de solutions possibles de manière efficace. Les méthodes d'optimisation peuvent être classées en deux catégories : les méthodes d'optimisation déterministes et les méthodes d'optimisation stochastiques. Ces méthodes peuvent être appliquées à des problèmes d'optimisation continue ou discret.

Dans notre travail, nous nous intéressons aux méthodes exactes, qui garantissent la détermination de la solution optimale à un problème, nous mettons l'accent sur une méthode spécifique, à savoir la méthode Branch and Bound, reconnue pour son efficacité dans la résolution de problèmes complexes et difficiles.

2.2 Méthodes de résolution :

Les méthodes de résolutions peuvent être catégorisées en deux grandes classes : les méthodes exactes et les méthodes approchées [12] .

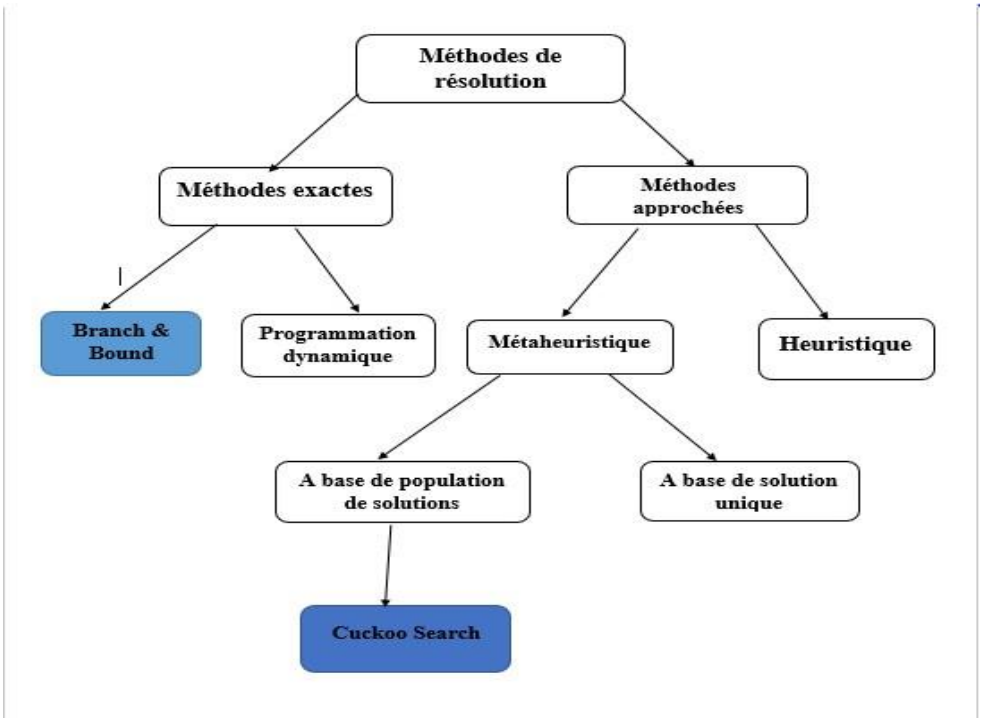

Figure12 : Classification des méthodes de résolution de problèmes d'optimisation

2.2.1 Méthodes approchées :

Les méthodes approchées sont des techniques utilisées en mathématiques, en statistiques et dans d'autres domaines pour obtenir des solutions approximatives à des problèmes complexes. Elles sont souvent utilisées lorsque les solutions exactes ne sont pas possibles ou trop coûteuses en termes de temps ou de ressources.

Les méthodes approchées, d'autre part, visent à fournir des solutions de haute qualité dans un temps raisonnable, sans garantie de trouver la solution optimale. Elles sont particulièrement adaptées aux problèmes complexes où la recherche exhaustive n'est pas réaliste.

Principale caractéristique des méthodes approchés :

Facilité : Les méthodes approchées sont généralement simples à comprendre et à appliquer.

Efficacité : Elles cherchent à fournir des solutions dans un laps de temps raisonnable, même pour des problèmes complexes.

Catégories des méthodes approchés :

Heuristique : en optimisation combinatoire, une heuristique est un algorithme approché qui permet de trouver au moins une solution réalisable rapidement, même si elle n'est pas nécessairement optimale. L'utilisation d'une heuristique est efficace pour obtenir une solution approximative à un problème et accélérer le processus de résolution exacte **[13]**. Les heuristiques sont généralement conçues pour des problèmes spécifiques, en exploitant leur structure propre sans garantir la qualité de la solution obtenue. On peut classer les heuristiques en deux catégories :

Les méthodes constructives : qui construisent des solutions à partir d'une solution initiale en ajoutant progressivement des éléments jusqu'à obtenir une solution complète,

Les méthodes de recherche locale : qui partent d'une solution complète initiale (peut-être moins intéressante) et cherchent à l'améliorer de manière itérative en explorant son voisinage.**[13]**

Méta-heuristique : métaheuristique sont des méthodes d'optimisation approximative qui peuvent impliquer l'aspect aléatoire dans leur processus de

recherche et qui permettent d'aborder des instances de problèmes difficiles en fournissant des solutions satisfaisantes (solution optimal ou proche de l'optimale) dans un délai raisonnable. Les métaheuristiques ne fournissent aucune garantie de trouver une solution optimale global .la réutilisation dans des nombreux applications montrent leur puissance et leur efficacité pour résoudre des problèmes importants et complexes. Les métaheuristiques sont généralement plus rapide en termes de calcul que la recherche exhaustive.

Catégories de métaheuristique : il existe deux grandes catégories : recherche locale et recherche globale.

Recherche Locale : parmi les métaheuristiques basées sur la recherche locale on trouve le recuit simulé qui s'inspire du processus de refroidissement des métaux pour explorer de façon approfondie l'ensemble des solutions possibles, et la recherche taboue qui emploie une liste taboue pour éviter de revisiter des solutions déjà examinées.

Recherche globale : parmi les métaheuristiques basées sur la recherche globale on trouve les algorithmes génétiques qui tirent leur inspiration de la sélection naturelle et utilisent des opérateurs génétiques tels que la mutation et la recombinaison pour parcourir l'espace des solutions, et les essaims particulaires qui reproduisent le comportement collectif des essaims d'oiseaux pour explorer de manière collaborative l'espace des solutions.

Algorithme de recherche du coucou (Cuckoo Search, CS) :

C'est une métaheuristique inspirée du comportement de ponte des coucous parasitaires. Initialement développé par Xin-She Yang et Suash Deb en 2009, il simule le processus naturel de reproduction et de remplacement des œufs des coucous pour résoudre des problèmes d'optimisation. CS commence par une population aléatoire de solutions candidates, évaluées selon une fonction

objective. Des opérations de recherche locale et de mouvement de ponte sont ensuite appliquées pour améliorer les solutions et maintenir la diversité de la population. Bien que CS soit simple à comprendre et à implémenter, son efficacité dépend de la sélection appropriée des paramètres et de la compréhension du problème spécifique.

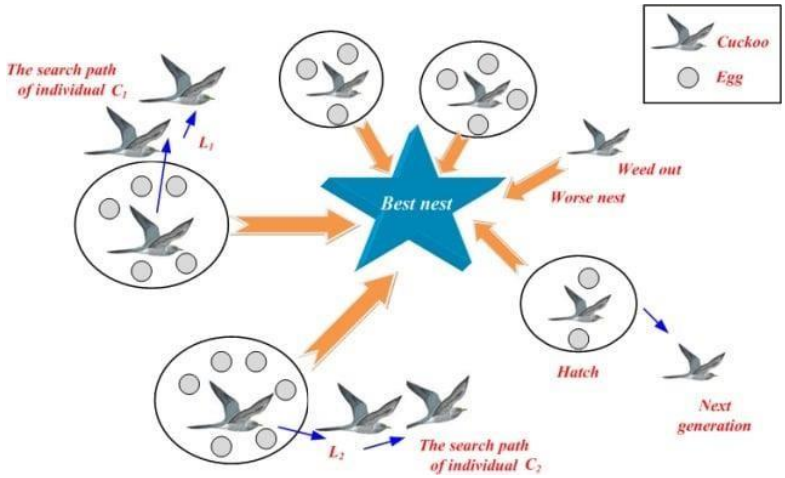

Figure13 : Environnement de Cuckoo Search [14]

Ci-dessous le pseudo code de l'algorithme Cuckoo Search :[15]

Algorithme : Cuckoo Search

Initialiser une population de n nids d'accueil *xi* (I = 1, 2, ..., n).

While *(t <NbrMaxGéneration)* **do**
Choisir une solution j au hasard et
générer une solution par vol de
Lévy Evaluation du fitness fi de la
solution **if** *(Fi >Fj)* **then**
Remplacer j par la nouvelle solution **end**

On classe les solutions par leur fitness et on trouve les meilleures solutions courantes. La fraction *Pa* des moins bonnes solutions sont abandonnés et une quantité équivalente de nouvelles solutions est générée par le vol de Lévy

Les meilleures solutions passent à l'itération suivante **end**

Classification des métaheuristique :

Les métaheuristiques peuvent être classifiées en deux catégories principales :

- **Métaheuristiques mono-solution (solution unique) :**

  Les métaheuristiques mono-solution se focalisent sur l'amélioration itérative d'une unique solution à la fois. Elles explorent l'espace de recherche en effectuant des déplacements à partir de cette solution initiale. Ces approches sont préférées lorsque l'espace de recherche est organisé de manière claire et que la qualité des solutions voisines peut être évaluée efficacement. Des exemples bien connus incluent le Recuit simulé et la Recherche tabou.

- **Métaheuristiques multi-solution (population de solutions) :**

  Les métaheuristiques multi-solution travaillent avec plusieurs solutions candidates simultanément, formant ce qu'on appelle généralement une population. Elles font appel à des processus de sélection, de reproduction et de mutation pour engendrer de nouvelles solutions à partir de cette population. Ces stratégies se révèlent efficaces lorsqu'on évolue dans un espace de recherche complexe où la qualité des solutions ne peut être déterminée de façon certaine. Des exemples courants comprennent les Algorithmes

  génétiques et l'Optimisation par essaim particulaire.

Avantages de métaheuristique :

On peut citer plusieurs avantages des Métaheuristiques :

- Les Métaheuristiques peut conduire a des solutions suffisamment bonnes pour les problèmes NP-difficiles c'est-à-dire des problèmes pour lesquels il n'existe aucun

algorithme exact connu pouvant les résoudre dans un délai raisonnable

- Contrairement à la plupart des méthodes classiques les métaheuristiques ne nécessitent aucune information de gradient
- La plupart des Métaheuristiques ont la capacité de récupérer les optima locaux en raison de leur stochastisité
- La plupart des Métaheuristiques peuvent gérer plusieurs objectifs avec quelques changements algorithmiques.
- Elles peuvent être appliqués à tous les problèmes d'optimisations.
- Elles peuvent être combinées avec d'autre techniques d'optimisations plus traditionnelle.
- Les Métaheuristiques sont très flexible et simples à comprendre à concevoir et à mettre en œuvre.

Inconvénients de métaheuristique : les métaheuristiques ont quelques inconvénients :

- Elles n'ont pas de fondement mathématique solide par rapport au techniques traditionnelles.

- Elles ne peuvent pas garantir l'optimalité.
- Leurs performances dépendent fortement de leurs paramètres.

2.2.2 Méthodes exactes :

Les méthodes exactes en optimisation représentent des stratégies algorithmiques assurant la détermination de la solution optimale pour un problème spécifique. Elles revêtent une importance particulière dans des secteurs où l'exactitude et l'optimalité des solutions sont essentielles, comme la planification, la conception de réseaux, la logistique, ainsi que d'autres domaines associés à l'ingénierie et à la recherche opérationnelle.

2.2.2.1 Recherche exhaustive :

La recherche exhaustive est l'une des méthodes exactes les plus simples mais également les plus coûteuses en termes de temps de calcul. Elle consiste à énumérer systématiquement toutes les solutions possibles d'un problème, évaluant chaque solution pour déterminer la meilleure. Bien que garantissant l'optimalité, cette méthode devient rapidement impraticable lorsque le nombre de solutions potentielles croît de manière exponentielle avec la taille du problème.

2.2.2.2 Programmation dynamique :

La programmation dynamique est une approche qui divise un problème en sous problèmes plus petits et résout chacun de manière itérative. Elle est particulièrement efficace pour les problèmes présentant une structure de chevauchement, où des solutions identiques ou partiellement identiques peuvent être réutilisées pour résoudre différents sous-problèmes. La programmation dynamique évite les recalculs inutiles en stockant les résultats intermédiaires, améliorant ainsi considérablement l'efficacité.

2.2.2.3 Programmation linéaire :

La programmation linéaire est une méthode mathématique utilisée pour résoudre des problèmes d'optimisation en cherchant à maximiser ou minimiser une fonction linéaire tout en respectant des contraintes linéaires. Pour trouver la solution optimale, on utilise des algorithmes comme le simplexe qui consiste à se déplacer le long des arêtes d'une forme géométrique définie par les contraintes. Cette approche efficace permet de résoudre des problèmes complexes en utilisant des équations linéaires, et les méthodes de résolution telles que le simplexe sont essentielles pour déterminer la meilleure solution possible.

2.2.2.4 Branch and Bound :

La méthode Branch and Bound est une technique systématique pour résoudre des problèmes d'optimisation combinatoire difficiles. Elle commence par diviser le problème initial en sous-problèmes plus gérables, formant ainsi un arbre de recherche. À chaque étape, des bornes inférieures et supérieures sont évaluées pour chaque sous problème, permettant d'élaguer les branches non prometteuses de l'arbre. Cette méthode est particulièrement efficace pour traiter des problèmes NP-difficiles, où l'espace de recherche est vaste et complexe.

Principe fondamentale de la méthode Branch and Bound (B&B) :

Décomposition du problème : initialement, la méthode B&B divise le problème initial en sous-problèmes, créant ainsi une structure arborescente de recherche. Chaque nœud de cet arbre représente un sous-problème qui doit être résolu.

Évaluation des bornes inférieures et supérieures : à chaque étape, la méthode analyse les bornes inférieures et supérieures pour chaque nœud de l'arbre. Les bornes inférieures permettent d'éliminer les sous-problèmes incapables d'accueillir une solution optimale, tandis que les bornes supérieures excluent ceux dont la solution actuelle est moins optimale que la meilleure solution connue.

Sélection Discernant : la méthode B&B sélectionne de manière discernant les sous problèmes les plus prometteurs en se basant sur les bornes inférieures et supérieures évaluées. Les nœuds non retenus sont élagués, réduisant ainsi la dimension de l'arbre de recherche.

Résolution Itérative des Sous-Problèmes : les sous-problèmes sélectionnés sont résolus de manière itérative en appliquant la même procédure, poursuivant la décomposition jusqu'à ce que les feuilles de l'arbre soient atteintes.

Mise à Jour de la Meilleure Solution : la méthode B&B maintient en permanence la meilleure solution identifiée à ce stade. Si une solution trouvée pour un sous-problème surpasse la meilleure solution actuellement connue, cette dernière est actualisée.

Application de la méthode :

Problème de sac à dos :

Le problème du Sac à Dos, une énigme classique dans le domaine de l'optimisation combinatoire, a fait l'objet de nombreuses investigations dans la littérature. Il implique la maximisation de la valeur totale des objets pouvant être inclus dans un sac à dos, sous la contrainte que la somme des poids des objets ne dépasse pas une capacité prédéfinie. L'efficacité de la Méthode Branch and Bound pour résoudre ce problème réside dans son approche systématique de l'exploration de l'espace des solutions.

L'application de la Méthode Branch and Bound est comme suit :

Décomposition du Problème :

- La première étape implique la décomposition du problème du sac à dos en sous-problèmes.
- Chaque sous-problème représente une décision binaire pour un objet spécifique : inclusion ou exclusion dans le sac.

Bornes inférieures et supérieures :

- À chaque nœud de l'arbre de recherche, la méthode évalue des bornes inférieures et supérieures.
- **Bornes inférieures :** calculées en tenant compte de la valeur des objets déjà inclus dans le sac et en extrapolant la valeur des objets restants.
- **Bornes supérieures :** évaluées en supposant que tous les objets restants peuvent être inclus dans le sac.

Élagage sélectif :

- Les nœuds de l'arbre qui ne présentent pas de promesses, c'est-à-dire ceux pour lesquels les bornes inférieures et supérieures ne suggèrent pas d'amélioration, sont élagués.

Résolution des sous-problèmes :

- Les sous-problèmes sélectionnés sont résolus de manière récursive, en appliquant la même procédure de décomposition.
- Cette étape se poursuit jusqu'à atteindre le parcours de l'arbre.

Actualisation de la meilleure solution :

- La méthode Branch and Bound maintient en permanence la meilleure solution connue.
- Si une solution pour un sous-problème dépasse la meilleure solution actuelle, cette dernière est mise à jour.

Problème du Voyageur de Commerce :

Le problème du Voyageur de Commerce (TSP - Traveling Salesman Problem) est un défi classique en optimisation combinatoire. Il pose la question suivante : quel est le chemin le plus court qu'un voyageur de commerce peut emprunter pour visiter chaque ville exactement une fois et revenir à son point de départ ? La méthode Branch and Bound s'avère être une approche puissante pour résoudre ce problème complexe. L'application de la méthode Branch and Bound du probleme TSP est comme suit :

Décomposition du problème :

- La première étape englobe la décomposition du problème initial en sous-problèmes.

- Chaque sous-problème représente une décision binaire pour une arête du graphe du TSP :

  inclusion ou exclusion dans le parcours.

Bornes inférieures et supérieures : à chaque nœud de l'arbre de recherche, la méthode évalue des bornes inférieures et supérieures.

- **Bornes Inférieures :** calculées avec une heuristique pour estimer la distance minimale possible du parcours en considérant les arêtes incluses.
- **Bornes Supérieures :** calculées en prenant en compte toutes les arêtes restantes et en supposant que le voyageur de commerce pourrait choisir la meilleure option.

Élagage sélectif :

- Les nœuds de l'arbre qui ne présentent pas de promesses, c'est-à-dire ceux pour lesquels les bornes inférieures et supérieures ne suggèrent pas d'amélioration par rapport à la meilleure solution actuelle, sont élagués.

Résolution des sous-problèmes :

- Les sous-problèmes sélectionnés sont résolus de manière récursive en appliquant la même procédure de décomposition.
- La recherche se poursuit jusqu'à ce que toutes les feuilles de l'arbre soient atteintes.

Actualisation de la meilleure solution :

- La méthode Branch and Bound maintient constamment la meilleure solution trouvée jusqu'à présent.
- Si une solution pour un sous-problème dépasse la meilleure solution actuelle, cette dernière est mise à jour.

Avantage de la méthode B&B:

Exploration Systématique : la méthode assure une exploration ordonnée et exhaustive de l'espace des solutions, assurant une couverture complète du domaine du problème.

Efficacité de l'Élagage : l'utilisation de bornes inférieures et supérieures permet un élagage judicieux des branches non prometteuses, simplifiant ainsi la complexité du problème et accélérant le processus de recherche.

Adaptabilité : la méthode est polyvalente, adaptée à une diversité de problèmes d'optimisation combinatoire. Sa flexibilité permet une application efficace à différentes instances de problèmes.

Optimalité : la méthode assure la convergence vers une solution optimale, fournissant ainsi une garantie cruciale d'atteindre le meilleur résultat possible pour les problèmes traités.

Inconvénients de la méthode B&B :

Complexité de Mise en Œuvre : la mise en œuvre de la méthode peut être exigeante, nécessitant une connaissance approfondie du problème spécifique ainsi qu'une modélisation précise.

Dépendance aux heuristiques : l'efficacité de la méthode repose souvent sur la qualité des heuristiques utilisées pour estimer les bornes inférieures, ce qui peut constituer un défi dans certaines situations.

Coût Computationnel : dans certains cas, le coût computationnel associé à la méthode peut être élevé, en particulier pour des problèmes de grande taille ou complexes, impactant les performances globales.

Variabilité de Performances : l'efficacité de la méthode peut présenter des variations en fonction de la nature spécifique du problème traité, de la qualité des bornes utilisées, et de la distribution des valeurs, introduisant ainsi une certaine incertitude dans ses performances.

2.2.3 Comparaison entre les méthodes exactes et les méthodes approchés :

La comparaison entre les méthodes exactes et les méthodes approchées est fondamentale en recherche opérationnelle et en intelligence artificielle, surtout lorsqu'il s'agit de résoudre des problèmes d'optimisation. Voici les principales différences entre ces deux types de méthodes :

Méthodes exactes	Méthodes approchées :
Elles garantissent de trouver la solution optimale d'un problème.	Elles ne garantissent pas la solution optimale, mais cherchent une solution de bonne qualité en un temps raisonnable.
Elles sont basées sur une exploration systématique de l'espace de recherche.	Elles sont souvent basées sur des heuristiques qui guident la recherche vers des régions prometteuses de l'espace de recherche.
Leur temps de calcul peut être très long, surtout pour les problèmes de grande taille.	Elles sont plus adaptées aux problèmes de grande taille ou aux problèmes où une solution exacte n'est pas nécessaire.
Des exemples incluent la méthode de Branch and Bound, la programmation linéaire, et la programmation dynamique.	Des exemples incluent le recuit simulé, la recherche taboue, et les algorithmes génétiques.

Tableau 1 : comparaison entre méthode exacte et méthode approchées

2.3Conclusion :

Ce chapitre a présenté les méthodes de résolution de problèmes en général, en mettant l'accent sur la méthode Branch and Bound. Nous avons exploré les fondements de cette méthode, ses applications, ainsi que ses avantages et

limitations. Dans le chapitre suivant, nous approfondirons davantage notre compréhension en examinant des exemples concrets et des cas d'application spécifiques de la méthode Branch and Bound.

Chapitre2 : Méthodes d’optimisations

Chapitre3 :
Implémentation

Chapitre3 : Implémentations et Simulations

3.1 Introduction :

Le problème d'ordonnancement des tâches dans le Cloud computing représente un défi significatif dans la gestion efficace des ressources informatiques. Dans cet environnement dynamique et hautement distribué, où de multiples tâches doivent être exécutées simultanément sur un ensemble de ressources, l'optimisation du temps d'exécution total des tâches est cruciale. Pour résoudre ce problème, différentes approches sont utilisées, parmi lesquelles les méthodes exactes et les métaheuristiques occupent une place importante.

Ce chapitre, présente notre travail en se basant sur la méthode "Branch and Bound"

Dans ce chapitre, nous allons présenter le langage java, l'environnement de développement NetBeans.

Ce chapitre expérimental se concentre sur une analyse approfondie comparant deux approches clés pour résoudre le défi de l'ordonnancement des tâches dans le domaine du Cloud computing : la méthode exacte **Branch and Bound** et la métaheuristique **Cuckoo Search**. Notre objectif principal est d'évaluer et de comparer de manière exhaustive les performances de ces deux méthodes dans une variété de situations expérimentales en utilisant le critère : « makespan » et « le temps d'exécution ».

Par la suite, nous expliquerons notre approche et nous citerons aussi les différentes étapes et les configurations nécessaires avant le lancement de la simulation.

Nous conclurons le chapitre par une analyse des résultats.

3.2 Environnement de développement et de simulation :

Ce projet a été implémenté et simulé dans un environnement qui possède les Caractéristiques suivantes :

- Une machine avec un processeur Intel(R) core i3, une vitesse de 2.40GHzavec une capacité de mémoire de 4GB. Sous le système d'exploitation windows10 de 64bits.
- Le simulateur CloudSim version 2.1.1 développées avec langage de programmation java.
- L'IDE NetBeans

3.2.1Java :

Java est un langage de programmation et une plate-forme de calcul lancé par Sun Microsystems en 1995. Depuis ses débuts modestes, Java a beaucoup évolué. A l'heure actuelle, une grande partie du monde numérique dépend de Java : de nombreux services et applications reposent sur cette plate-forme fiable. De même, de nouveaux produits et services numériques innovants et tournés vers l'avenir dépendent aussi de Java [16]. L'une de ses caractéristiques les plus marquantes est sa portabilité, permettant aux programmes Java de s'exécuter sur différentes plates-formes sans nécessiter de modifications majeures. De plus, Java met l'accent sur la sécurité grâce à des fonctionnalités telles que la vérification de byte code et la gestion automatique de la mémoire, ce qui en fait un choix populaire pour le développement d'applications critiques. Avec sa large bibliothèque standard et sa prise en charge native du multithreading, Java offre aux développeurs un environnement de développement robuste et efficace pour répondre à divers besoins logiciels.

Figure14 : Logo de java [17]

3.2.2NetBeans :

NetBeans IDE est un environnement de développement intégré (IDE) open source qui offre une gamme complète d'outils pour le développement d'applications Java, ainsi que pour d'autres langages de programmation comme JavaScript, HTML5, PHP, et C/C++. Lancé par Sun Microsystems et maintenant soutenu par la Fondation Apache, NetBeans propose un éditeur de code avancé, des fonctionnalités de débogage intégrées et des outils de gestion de projets. Sa fonctionnalité la plus appréciée est son concepteur graphique d'interfaces utilisateur (GUI), qui permet aux développeurs de créer facilement des interfaces attrayantes. Avec son intégration transparente aux serveurs d'applications et aux bases de données, ainsi que son support pour les Framework de développement, NetBeans simplifie le processus de développement et de déploiement des applications. Grâce à sa communauté active de développeurs, NetBeans continue de s'améliorer et de proposer de nouvelles fonctionnalités pour répondre aux besoins des développeurs de tous niveaux.

Figure15 : Logo d'apacheNetBeans [18]

3.3Approche proposée :

3.3.1 Objectif du travail :

L'objective de notre projet, est de proposer un mécanisme d'ordonnancement des tâches dans le Cloud Computing. Nous avons proposé une approche qui consiste à gérer les

ressources disponibles au sein d'un système Cloud Computing de manière optimale, et cela dans le but d'améliorer les performances du système. Cette approche est basée sur la méthode « Branch and Bound », qui prend en considération les deux critères suivants : temps d'exécution et le makespan. Nous avons aussi implémenté la métaheuristique coucou pour justifier nos résultats.

3.3.2Description de l'approche :

Cette section décrit l'architecture de la solution proposée. Le but de cette approche est de trouver une meilleure affectation des tâches aux machines virtuelles en matière de minimiser le makespan de l'algorithme. La démarche principale de notre proposition est focalisée sur deux phases : phase d'initialisation et phase d'ordonnancement

Dans notre travail nous avons choisi la metaheuristique cuckoo-search pour comparer avec notre approche. Voici le pseudo code de la metaheuristique utilisé :

```
Début
    Appeler initializeRandomData() pour initialiser les données aléatoires

    Initialiser les nids cuckoo aléatoires en appelant initializeRandomNids()

    Pour chaque itération de 1 à NB_ITERATIONS faire
      Pour chaque nid cuckoo dans les nids cuckoo faire
        Pour chaque tâche dans les tâches faire
          Générer aléatoirement une machine pour la tâche
          Mettre à jour les bits correspondant à la tâche dans le nid avec la machine générée
        Fin Pour
      Fin Pour

      Évaluer les solutions des nids cuckoo et stocker les makespans dans un tableau

      Trier les makespans par ordre croissant

      Remplacer une fraction LB des pires nids cuckoo par de nouveaux nids aléatoires

    Fin Pour

    Sélectionner la meilleure solution parmi les nids cuckoo restants

    Retourner la meilleure solution et son makespan
  Fin
```

La taille de la population doit être fixée avant le lancement de l'exécution de cet algorithme, pour cela nous avons varier la taille population, dont le but de choisir la meilleure (voir Tableau 3)

Choix de la population :

population	makespan
10	20
20	19
30	20
40	20
50	20
60	21

Tableau 3 : tableau de choix de population

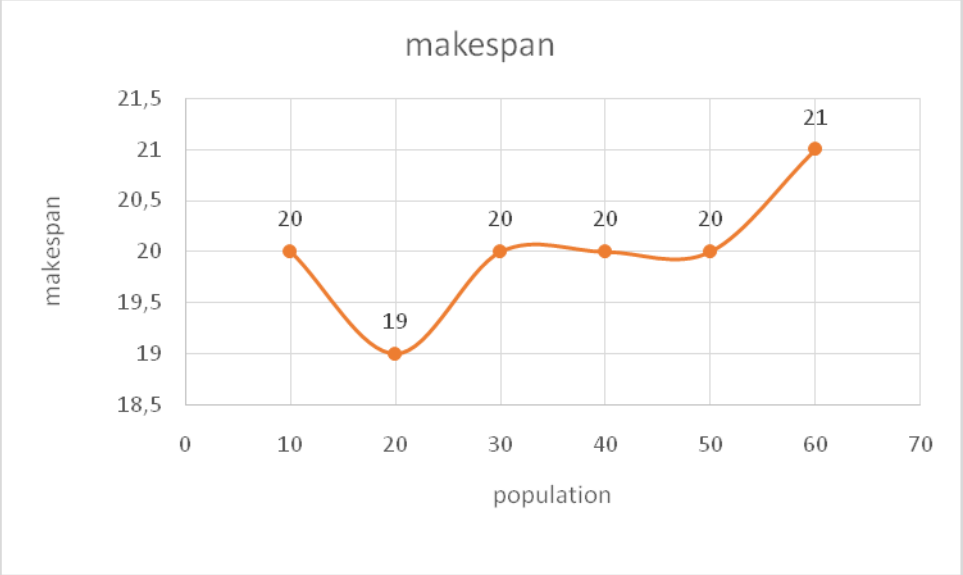

Figure18 : Graphe de choix de population

Selon le graphe de choix de population qui signifie la variation de makespan en fonction de changement de population on remarque que l'algorithme de recherche Cuckoo montre une stabilité et une capacité d'optimisation jusqu'à une certaine taille de population. La population de 20 nids semble fournir un résultat similaire et optimal (makespan de 19).

Chapitre3 : Implémentation

Apres avoir fixer les paramètres de la métaheuristique Cuckoo qui sera utilisé pour justifier notre proposition, nous passons à la description de la méthode Branch and Bound, en commençant par son pseudo code :

Pseudo code de la fonction B&B:

```
Debut
ProcedurebranchAndBound(espace, affectation, indexTache, chargesVMs)
siindexTache == longueur de espace alors max = maximum de
chargesVMs si max <meilleurMakespan alors attribuer max à
meilleurMakespan copier affectation à meilleureAffectation retourner
fin si

pour i de 0 à longueur de espace[0] - 1 faire
copierchargesVMs à nouvelleChargeVMs ajouter
espace[indexTache][i] à nouvelleChargeVMs[i]

si nouvelle Charge VMs[i] <meilleur Makespan alors copier affectation à nouvelleAffectation
affecter index Tache à i dans nouvelleAffectation appeler branch And Bound(espace, nouvelle
Affectation, index Tache + 1, nouvelle Charge VMs) fin pour fin procedure
Fin
```

Le multithreading est aussi utilisé pour améliorer le temps de calculs de la méthode B&B

Pseudo code de la fonction branch and bound en utilisant les threads:

```
Début
fonctionexecuterParallele(nombreDeThreads, espace) initialisermeilleurMakespan
à INFINITY initialisermeilleureAffectation à null

créer tableau threads de taille nombreDeThreads

nombreDeVMs = longueur de la première ligne de espace
```

```
nombreDeTâchesInitiales = min(nombreDeThreads, longueur de espace)

pour i de 0 à nombreDeThreads - 1 faire initialiserinitialTaskIndex à i %
nombreDeTâchesInitiales initialiservmStart à (i / nombreDeTâchesInitiales) %
nombreDeVMs threads[i] = créer nouveau BranchAndBoundThread(espace,
initialTaskIndex, vmStart) démarrer threads[i]

pour i de 0 à nombreDeThreads - 1 faire essayer attendre que
threads[i] se termine (join) attraper (InterruptedException e)
afficher erreur pour i de 0 à nombreDeThreads - 1 faire si
threads[i].getMeilleurMakespan() <meilleurMakespanalors
meilleurMakespan = threads[i].getMeilleurMakespan()
meilleureAffectation = threads[i].getMeilleureAffectation()

afficher "Meilleur Makespan: " + meilleurMakespan
afficher "Meilleure Affectation: " + meilleureAffectation
finprocedure
Fin
```

La fonction principale (l'appel des fonctions) :

```
Debut
Procedurerun() initialiserchargesVMs à tableau de zéros de taille
espace[0].length initialiser affectation à tableau de -1 de taille
espace.length affecterinitialTaskIndex à vmStart dans affectation
ajouter espace[initialTaskIndex][vmStart] à chargesVMs[vmStart]

appelerbranchAndBound(espace, affectation, initialTaskIndex + 1, chargesVMs)
finprocedure
Fin
```

Fixer le nombre de thread :

Le tableau suivant montre le temps d'exécution (en millisecondes) pour différents nombres de threads (nbthread) :

Nb thread	Temp d'exécution
1	66287
2	56161
4	55988
6	55585
8	50508

Tableau 4 : la variation de temp d'exécution en fonction de nombre de threads

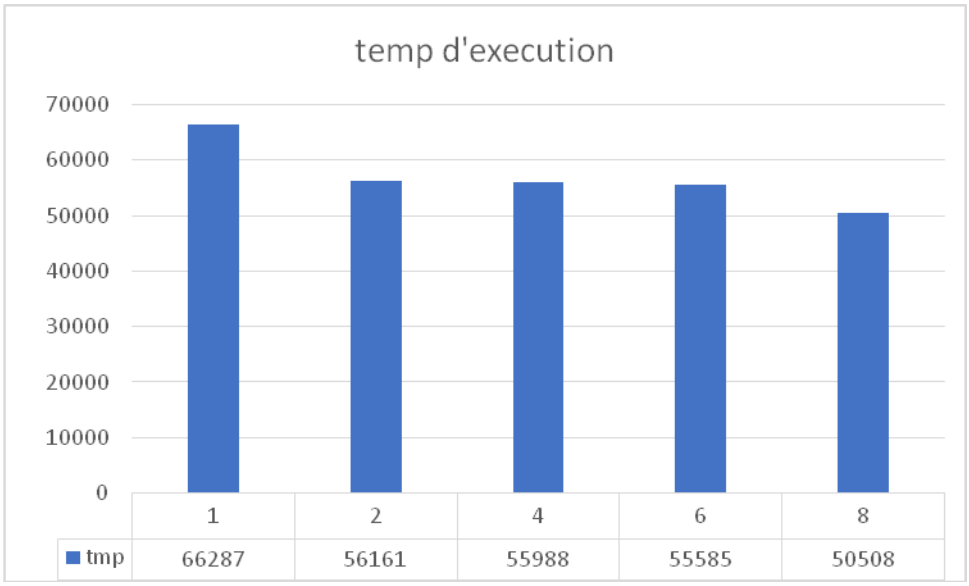

Figure 19: Graphe de temps d'exécution en fonction de nombre de threads Analyse des résultats :

 Un seul thread (1 thread) : l'algorithme prend le plus de temps (66287 ms) pour s'exécuter, ce qui est attendu car il n'y a pas de parallélisme.

Deux threads (2 threads) : l'utilisation de 2 threads réduit considérablement le temps d'exécution à 56161 ms, ce qui montre un bon usage du parallélisme.

Quatre threads (4 threads) : avec 4 threads, le temps d'exécution diminue légèrement à 55988 ms. La différence par rapport à 2 threads est minime, ce qui suggère une saturation des capacités de votre CPU (puisqu'il a 4 threads matériels).

Six et Huit threads (6 et 8 threads) : l'augmentation du nombre de threads au-delà de 4 (6 et 8 threads) continue de réduire le temps d'exécution, mais les gains deviennent de plus en plus marginaux. L'utilisation de 8 threads donne le temps d'exécution le plus court (50508 ms), mais l'amélioration par rapport à 4 threads n'est pas proportionnelle à l'augmentation du nombre de threads.

Choix optimal de nombre de threads : étant donné que notre PC dispose de 2 cœurs et 4 threads, et à cause de la faiblesse de la machine, 4 threads n'améliorent pas les résultats. Les résultats montrent que l'utilisation de 2 threads réduit le temps d'exécution à 56161ms.Ainsi, pour un équilibre optimal entre performance et efficacité, en opte pour 2 threads.

3.3.3 Comparaison entre les deux approches :

Le tableau ci-dessous résume la comparaison entre les deux approches B&B et Cuckoo Search :

Selon makespan :

Nombre de taches	10	15	20
Nombre de vms	2	3	4
Makespan Cuckoo search	17	26	20
Makespan Branch and bound	17	17	17

Tableau 5 : Tableau de comparaison entre B&B et Cuckoo Search en fonction de makespan

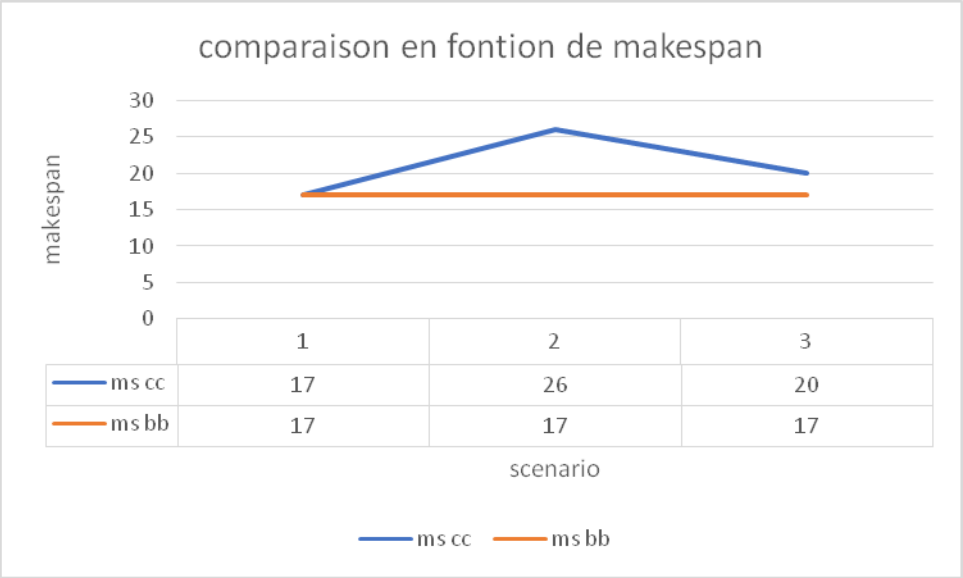

Figure 20 : Graphe de comparaison entre B&B et Cuckoo Search en fonction de makespan

Analyse des résultats (stabilité et efficacité) **:**

Stabilité **:**

- La méthode Branch and Bound montre une stabilité remarquable avec un makespan constant de 17, quel que soit le nombre de tâches ou de VMs. Cela suggère que cette méthode est robuste et capable de fournir des solutions optimales.

- En revanche, la méthode Cuckoo Search présente une variabilité dans ses résultats. Elle est parfois aussi efficace que Branch and Bound (pour 10 tâches), mais dans d'autres cas, elle est nettement moins performante (par exemple, 26 pour 15 tâches).

Efficacité :

- Branch and Bound est plus efficace globalement, car il maintient un makespan optimal de 17 dans tous les cas.
- Le Cuckoo Search est moins prévisible. Pour 15 tâches et 3 VMs, il est significativement moins efficace avec un makespan de 26.

Nombre de taches	10	15	20
Nombre de vms	2	3	4
Temp d'execution Cuckoo search (ms)	1	2	14
Temp d'execution Branch and bound (ms)	1	21	54538

Tableau6 : Comparaison entre B&B en fonction de temp d'exécution

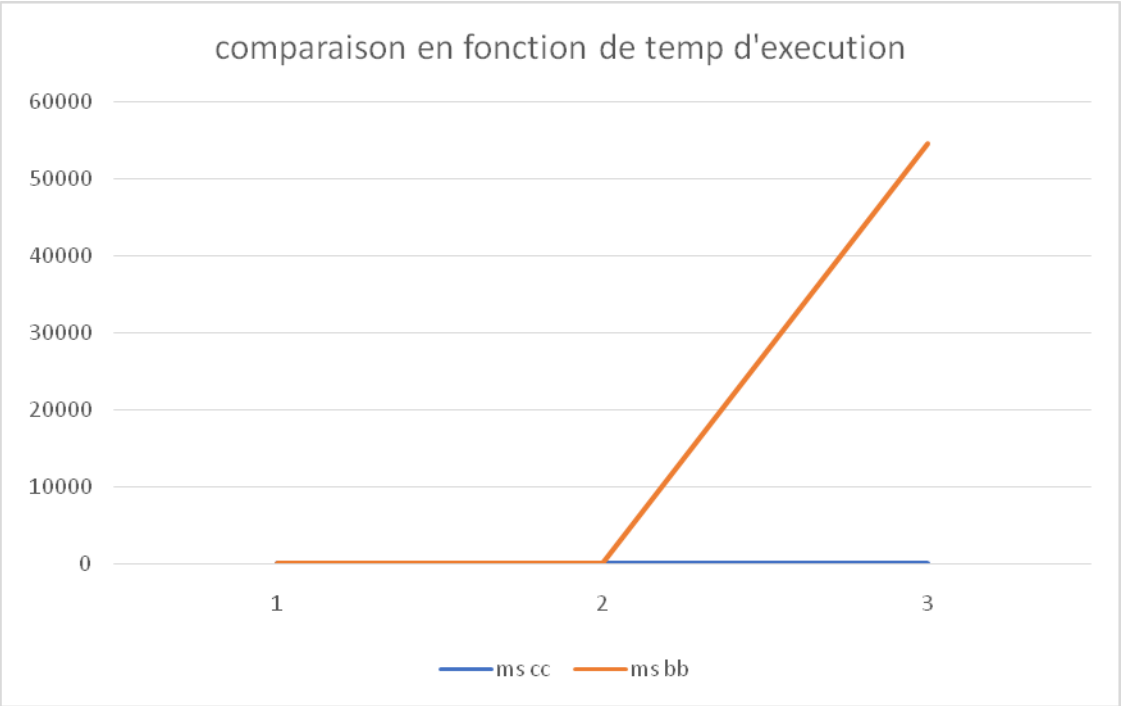

Figure21: Graphe de comparaison entre B&B en fonction de temp d'exécution

Analyse des résultats (Temps d'exécution):

- **Temps d'exécution Cuckoo Search :**

- Pour 10 tâches avec 2 VMs, le temps d'exécution est de 1 milliseconde.
- Pour 15 tâches avec 3 VMs, le temps d'exécution augmente légèrement à 2 millisecondes.
- Pour 20 tâches avec 4 VMs, le temps d'exécution augmente de manière plus significative à 14 millisecondes.
- Le Cuckoo Search montre une augmentation du temps d'exécution en fonction du nombre de tâches et de VMs, mais reste relativement rapide même pour le plus grand nombre de tâches.

Temps d'exécution Branch and Bound :

- Pour 10 tâches avec 2 VMs, le temps d'exécution est de 1 milliseconde, similaire à celui du Cuckoo Search.
- Pour 15 tâches avec 3 VMs, le temps d'exécution augmente considérablement à 21 millisecondes.
- Pour 20 tâches avec 4 VMs, le temps d'exécution augmente de manière exponentielle à 54,538 millisecondes.
- Le Branch and Bound montre une augmentation drastique du temps d'exécution avec l'augmentation du nombre de tâches et de VMs, indiquant une complexité computationnelle élevée pour cette méthode.

3.3.4 Implémentation :

Pour 20 taches et 4 vm :

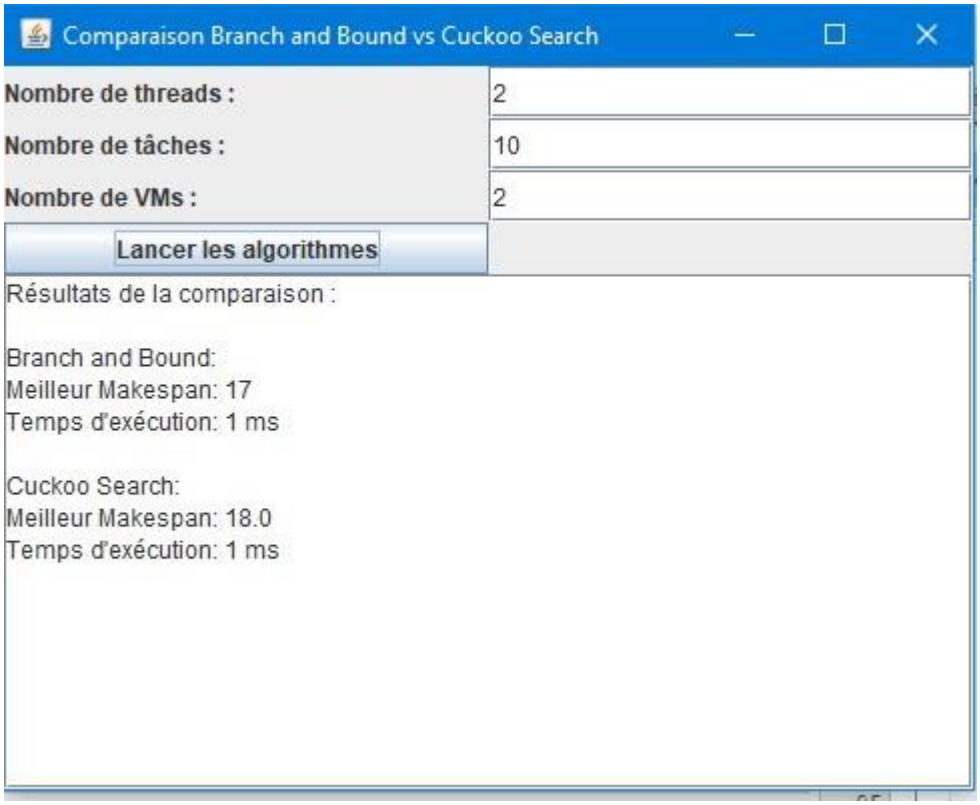

Figure 22: Comparaison entre les approches en fonction de makespan et temp d'exécution(10,2)

Configuration :

- Nombre de threads : 2
- Nombre de tâches : 10
- Nombre de VMs : 2

Résultats :

- **Branch and Bound :**
  - **Meilleur Makespan : 17**
    - Le temps total pour exécuter toutes les tâches en utilisant l'algorithme Branch and Bound est de 17 unités de temps.
  - **Temps d'exécution : 1 ms**
    - L'algorithme a pris 1 milliseconde pour calculer le makespan optimal.
- **Cuckoo Search :**
  - **Meilleur Makespan : 18.0**
    - Le temps total pour exécuter toutes les tâches en utilisant l'algorithme Cuckoo Search est de 18 unités de temps.
  - **Temps d'exécution : 1 ms**
    - L'algorithme a pris 1 milliseconde pour calculer le makespan optimal.

Pour 15 taches et 3vms :

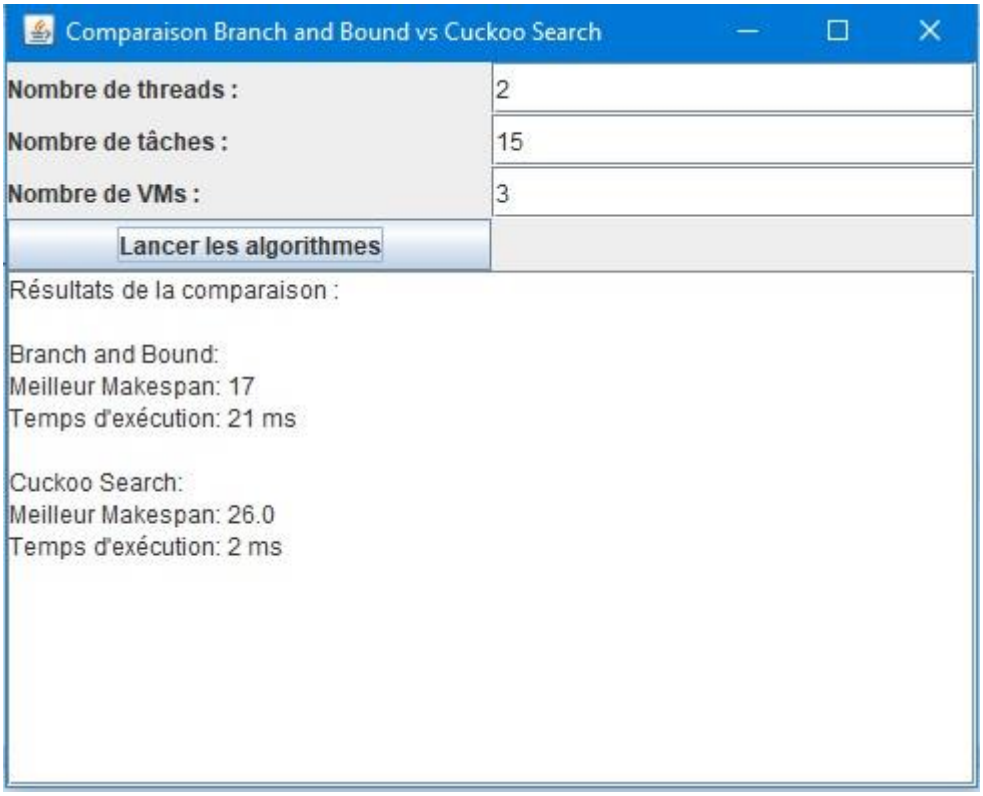

Figure 23: Comparaison entre les approches en fonction de makespan et temp d'exécution(15,3)

Configuration :

- Nombre de threads : 2
- Nombre de tâches : 15
- Nombre de VMs : 3

Résultats :

- **Branch and Bound :**
  - **Meilleur Makespan : 17**
    - Le temps total pour exécuter toutes les tâches en utilisant l'algorithme Branch and Bound est de 17 unités de temps.
  - **Temps d'exécution : 21 ms**
    - L'algorithme a pris 21 millisecondes pour calculer le makespan optimal.
- **Cuckoo Search :**
  - **Meilleur Makespan : 26.0**
    - Le temps total pour exécuter toutes les tâches en utilisant l'algorithme Cuckoo Search est de 26 unités de temps.
  - **Temps d'exécution : 2 ms**
    - L'algorithme a pris 2 millisecondes pour calculer le makespan optimal

Pour 20 taches et 4 vm :

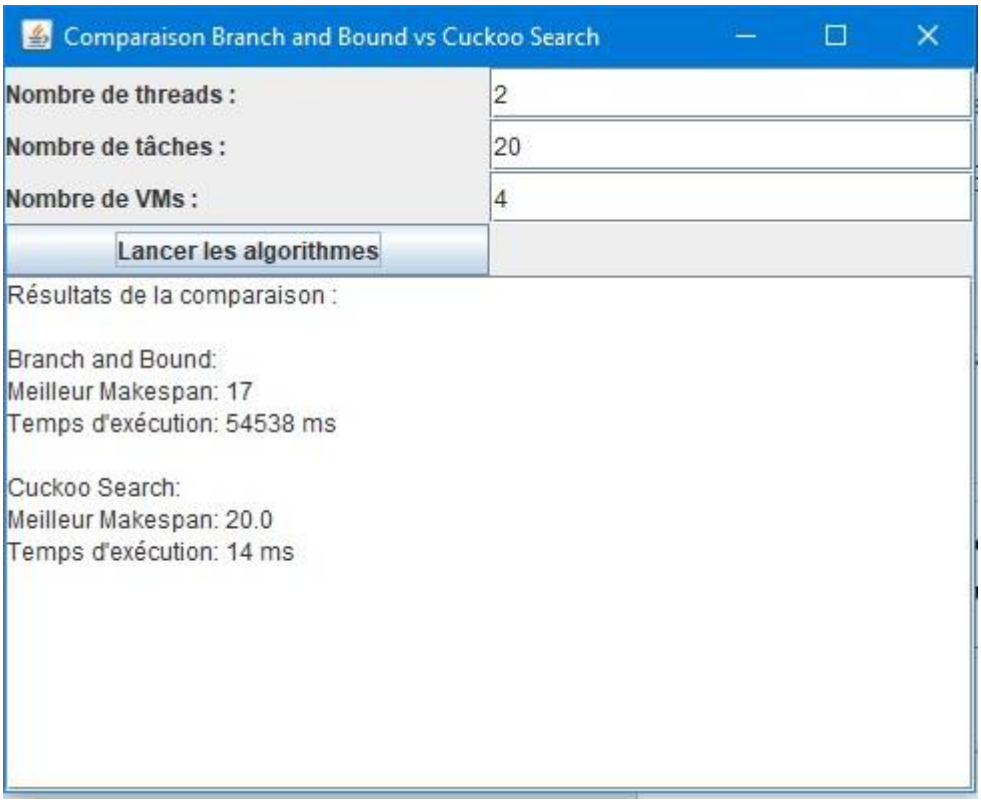

Figure 24: Comparaison entre les approches en fonction de makespan et temp d'exécution(20,4)

Configuration :

- Nombre de threads : 2
- Nombre de tâches : 20
- Nombre de VMs : 4

Résultats :

- **Branch and Bound :**
  - **Meilleur Makespan : 17**
    - Le temps total pour exécuter toutes les tâches en utilisant l'algorithme Branch and Bound est de 17 unités de temps.
  - **Temps d'exécution : 54538 ms**
    - L'algorithme a pris 54538 millisecondes (soit environ 54.5 secondes) pour calculer le makespan optimal.
- **Cuckoo Search :**
  - **Meilleur Makespan : 20.0**
    - Le temps total pour exécuter toutes les tâches en utilisant l'algorithme Cuckoo Search est de 20 unités de temps.
  - **Temps d'exécution : 14 ms**
    - L'algorithme a pris 14 millisecondes pour calculer le makespan optimal.

Discussion :

Branch and Bound est efficace pour obtenir un meilleur makespan, mais son temps d'exécution augmente considérablement avec la complexité du problème, particulièrement pour un grand nombre de tâches et de VMs.

Cuckoo Search est moins optimal en termes de makespan, mais il offre des temps d'exécution beaucoup plus constants et raisonnables, même pour des configurations plus complexes.

3.4 Conclusion :

Pour des problèmes de petite taille et des contraintes de temps faibles, Branch and Bound peut être préféré pour obtenir le makespan optimal. Pour des problèmes de grande taille ou lorsque le temps d'exécution est crucial, Cuckoo Search est une meilleure option.

Chapitre3 : Implémentation

Conclusion générale :

Notre étude a exploré diverses méthodes d'ordonnancement des tâches dans le Cloud computing, en mettant l'accent sur la comparaison entre les méthodes exactes et les métaheuristiques. Le principal objectif était d'optimiser le temps d'exécution total (makespan) des tâches réparties sur plusieurs machines virtuelles (VMs). La méthode Branch and Bound, bien que capable de garantir une solution optimale, présente une complexité computationnelle élevée, ce qui rend son utilisation peu pratique pour les grandes configurations de tâches et de VMs. À l'inverse, l'algorithme Cuckoo Search, une métaheuristique, offre des temps d'exécution plus constants et raisonnables, en dépit de l'absence de garantie d'optimalité. Ainsi, pour des problèmes de petite taille où le temps n'est pas un facteur limitant, la méthode Branch and Bound peut être privilégiée. Cependant, pour des problèmes de grande envergure ou lorsque le temps d'exécution est crucial, l'algorithme Cuckoo Search est une alternative plus viable. L'optimisation de l'ordonnancement des tâches dans le Cloud computing reste un domaine de recherche prometteur, avec des perspectives d'améliorations par l'intégration de l'apprentissage automatique pour adapter dynamiquement les paramètres des algorithmes et le développement de nouvelles approches hybrides. En somme, le choix de la méthode d'optimisation dépend des contraintes spécifiques du problème, avec les méthodes exactes convenant aux besoins de précision maximale et les métaheuristiques aux exigences de rapidité et d'efficacité.

Bibliographie

[1]Zhang, Q., Cheng, L., &Boutaba, R. (2010). "Cloud computing: state-of-the-art and research challenges." Journalof Internet Services and Applications, 1(1), 7-18[1]

[2]Les bonnes pratiques du cloud computing (Consulté le 27/04/2024). "Qu'est-ce que le Cloud Computing." Disponible sur FrameIP [2]**https://www.frameip.com/cloudcomputing-bonnes-pratiques/**

[3]IONOS. (Consulté le 13/02/2024). "Cloud Computing : définition, explication, histoire." Disponible sur IONOS [3].https://www.ionos.fr/digitalguide/serveur/know-how/cloudcomputing/consulter 13 Feb.2024

[4]Tim, M. (2009, Octobre). "Security and Privacy." Publié par: O'Reilly Media, édition 1, USA.

[5]A Systematic Review on Cloud Computing. (Consulté le 21/05/2024). Disponible sur ResearchGate .https://www.researchgate.net/publication/265232168_A_systematic_review_on_cloud_computing

[6]Figer. (Consulté le 27/04/2024). "Les services du cloud computing." Disponible sur Figer .

[7]Mecheri, R. B. A. M. "Réplication des données dans le Cloud Computing." Dans I.5.3. Cloud Communautaire (p. 18). Disponible sur Université Ouargla .

[8]Srinivas, K. S. R. D. Q. J. "CLOUD COMPUTING BASICS." International Journal of Advanced Research in Computer and Communication Engineering, 1(5). .

[9]Gueltoum, M. B. (Consulté le 21/05/2024). "Sécurité des applications métiers au niveau du Cloud Computing : Contrôle d'accès au niveau des APIs du Cloud Computing.

[10]Nag, A. (Consulté le 21/05/2024). "Cloud Computing: A Paradigm Shift in IT

Bibliographie

Infrastructure.” Disponible sur ResearchGatehttps://www.researchgate.net/publication/270958592_Cloud_Computing_A_Paradigm_Shift_in_IT_Infrastructure

[11]Le Cloud. (Consulté le 27/04/2024). “Les architectures générales du cloud computing.” Disponible sur **https://lecloud.info/cloud-computing/**

[12]Université M’sila.. “Méthodes de Résolution Exactes Heuristiques et Métaheuristiques.” Classification des méthodes de résolutions chapitre6 (p. 4).

[13]Douiri, S. M., Elbernoussi, S., &Lakhbab, H. “Cours des Méthodes de Résolution Exactes Heuristiques et Métaheuristiques.” Dans le 4ème titre Heuristiques (p. 13).

[14]Baeldung. (Consulté le 21/05/2024). “Algorithme de recherche Cuckoo.” Disponible sur https://www.baeldung.com/cs/cuckoo-search .

[15]Kharabi, S. A. R., &Benallal, M. I. “Ordonnancement Multi-critères dans le Cloud Computing.” Mémoire sous le titre “Description de l’algorithme” (p. 32)

[16] Qu'est-ce que la technologie Java et pourquoi en ai-je besoin ?(consulterle 14/05/2024**)**

Disponible sur https://www.java.com/fr/download/help/whatis_java.html

[17]java logo png consulté **le 18/05/2024**Disponible sur https://logos-marques.com/java-logo/

[18] Netbeans logo **consulté le 18/05/2024**Disponible sur https://eberloadfree.blogspot.com/2017/04/netbeans-82-full-mega.html

[19]CloudSim Structure and Entities.consulter le 21/05/2024 .Disponible surhttps://medium.com/ingkwan/getting-started-with-cloudsim-631e7f6b85d6

Bibliographie