الجمهورية الجزائرية الديمقراطية الشعبية

وزارة التعليم العالي والبحث العلمي

جامعة سعيدة د. مولاي الطاهر

كلية التكنولوجيا

قسم: الإعلام الآلي

Mémoire de Master

Spécialité : Réseaux Informatiques et Systèmes Réparties

Thème

Gestion de placement des données basée sur algorithme BEA pour les workflows scientifiques dans le Cloud Computing

Présenté par :

BELHADJI Mohammed Seddik

BENHAMIDI Abdelkader

Dirigé par :

KOUIDRI Siham

Promotion 2021 - 2022

Dédicace Mohammed seddik

Je dédie ce travail fruit, de plusieurs années de réflexions

A ma très chére mère et mon joyau père qui m'ont aidé par

Leurs invocations ainsi, que leurs bénédictions et leur soutien Durant toute ma vie.

J'attire un grand remercie à Mme **KOUIDRI Siham** de leur sacrifice avec nous durant la réalisation de ce mémoire. Ainsi que mon binôme Benhamidi abdelkader Merci d'être toujours là pour moi.

A tous l'ensemble des étudiants du département

D'informatique et spécialement la promotions

Master RISR 2021/2022

Dédicace benhamidi Abdelkader

Je dédie ce travail à :

A mes chers parents, pour tous leurs sacrifices, leur amour, leur tendresse, leur soutien et leurs prières tout au long de mes études,

A mes chers frères pour leurs appui et leur encouragement,

A ma chère sœur pour leur encouragements permanents, et leur soutien moral,

A toute ma famille pour leur soutien tout au long de mon parcours universitaire,

Que ce travail soit l'accomplissement de vos vœux tant allégués, et le fruit de votre soutien infaillible,

A tous ceux qui m'ont aidé de près ou de loin A la famille du lycée Mujahid de Lamari Mohammed

J'attire un grand remercie à Mme **KOUIDRI Siham** de leur sacrifice avec nous durant la réalisation de ce mémoire. Ainsi que mon binôme Belhadji Mohammed Seddik Merci d'être toujours là pour moi.

A tous l'ensemble des étudiants du département

d'informatique et spécialement la promotions Master

Remerciments

Nous remercions Allah de nous avoir donner le courage et la volonté ainsi que la conscience et la patience d'avoir pu terminer notre mémoire de Master.

Nous tenons à exprimer nos vifs remerciements à notre encadreure Mme **KOUIDRI Siham** pour nous avoir donner l'opportunité de réaliser ce sujet sous sa direction, la confiance faite ainsi que ses conseils fructueux, et son temps consacré tout au long du travail.

Nous tenons à remercier l'ensemble des membres de jury qui on acceptés d'examiner notre mémoire.

Un merci particulier à tous ce qui nous ont soutenu de prés ou de loin par leurs soutiens et encouragements.

Résumé

Les applications scientifiques bénéficient du paradigme de l'informatique en cloud, qui offre un accès aux ressources virtuelles à la demande et à la carte. Ces applications utilisent de grandes quantités de données qui doivent être stockées dans des centres de données distribués. Pour stocker efficacement ces données, un gestionnaire de données doit sélectionner intelligemment les machines virtuells dans lesquelles ces données résideront. Ce n'est cependant pas le cas pour les données qui doivent avoir un emplacement fixe. Lorsqu'une tâche nécessite plusieurs ensembles de données situés dans différents centres de données, le déplacement de gros volumes de données devient un défi. L'objectif de ce thème est d'implémenter une stratégies de placement de données basé sur L'algorithme Bond Energy Algorithme (BEA),ce dernier a fait ces preuves sur les base de données réparties .Nos simulations montrent l'efficacité de cette stratégie et surtout sur les applications intensives en terme de données.Les simulations ont été réalisé à l'aide du simulateur Cloudsim

Mots clés : cloud computing, Placement de Données, algorithme BEA ,Regroupement ,planification du flux de travail, CloudSim,planification des tâches

Abstract

Scientific applications benefit from the cloud computing paradigm, which provides on-demand and pay-as-you-go access to virtual resources. These applications use large amounts of data that must be stored in distributed data centers. To efficiently store this data, a data manager must intelligently select the virtual machines in which this data will reside. However, this is not the case for data which must have a fixed location. When a task requires multiple sets of data located in different data centers, moving large volumes of data becomes a challenge. The objective of this theme is to implement a data placement strategy based on the Bond Energy Algorithm (BEA), the latter has proven itself on distributed databases. Our simulations show the effectiveness of this method. strategy and especially on intensive applications in terms of data. The simulations were carried out using the Cloudsim simulator

Keywords : Cloud computing, scheduling tasks, workflows,Clustering, Data Placement ,BEA algorithm, workflow scheduling

تستفيد التطبيقات العلمية من نموذج الحوسبة السحابية ، الذي يوفر إمكانية الوصول عند الطلب والدفع أولاً بأول إلى الموارد الافتراضية. تستخدم هذه التطبيقات كميات كبيرة من البيانات التي يجب تخزينها في مراكز البيانات الموزعة. لتخزين هذه البيانات بكفاءة ، يجب على مدير البيانات أن يختار بذكاء الأجهزة الافتراضية التي ستوجد فيها هذه البيانات. ومع ذلك ، ليس هذا هو الحال بالنسبة للبيانات التي يجب أن يكون لها موقع ثابت. عندما تتطلب مهمة ما مجموعات متعددة من البيانات الموجودة في مراكز بيانات مختلفة ، فإن نقل كميات كبيرة من البيانات يصبح تحديًا. الهدف من هذا الموضوع هو وقد أثبت هذا ، (Bond Energy (BEA تنفيذ إستراتيجية وضع البيانات على أساس خوارزمية الأخير وجوده على قواعد البيانات الموزعة. تظهر عمليات المحاكاة لدينا فعالية هذه الاستراتيجية ، .وخاصة على التطبيقات المكثفة من حيث البيانات

تم إجراء عمليات المحاكاة باستخدام محاكي CloudSim.

كلمات مفتاحية : الحوسبة السحابية ، وضع البيانات ، خوارزمية **BEA** ، التجميع ، تخطيط سير العمل، تخطيط المهام

Table des matières

Introduction générale

Le cloud computing est en plein essor et devient rapidement l'un des concepts les plus populaires de l'industrie informatique. Il s'agit d'une infrastructure informatique et de stockage administrée par des serveurs auxquels les utilisateurs accèdent via une connexion Internet sécurisée. Les objets liés servent de points d'entrée pour exécuter des programmes et accéder aux données stockées sur les serveurs. Le Cloud se distingue notamment par son adaptabilité, qui permet aux fournisseurs d'ajuster automatiquement l'espace de stockage et la puissance de traitement pour répondre aux besoins des consommateurs.

Le cloud computing permet d'allouer des ressources informatiques. Lorsque ces ressources ne suffisent pas à répondre à la demande, des techniques de planification sont nécessaires. L'une des difficultés les plus difficiles est la planification des tâches dans un environnement diversifié comme le Cloud. Lorsqu'il y a plusieurs facteurs à examiner pour l'optimisation, le problème devient considérablement plus compliqué.

Avec la popularité croissante du Cloud Computing, la théorie de la planification des tâches attire de plus en plus l'attention. La planification des tâches, en général, est le processus d'attribution des tâches aux ressources disponibles en fonction des caractéristiques et des conditions des tâches. Parce que de nombreux critères métiers doivent être pris en compte pour une commande correcte, c'est un facteur crucial dans le fonctionnement fonctionnel du Cloud. Les ressources disponibles doivent être utilisées efficacement sans affecter les caractéristiques du service cloud.

Les nouvelles solutions proposées doivent répondre aux problèmes que présentent les attributs du réseau et les besoins des utilisateurs. Les nouvelles techniques peuvent combiner certains principes de planification traditionnels avec des stratégies de réseau pour créer une solution de planification des tâches meilleure et plus efficace.

Le recours aux infrastructures de type cloud se justifie par deux motivations essentiells :

- La nécessité de disposer de puissance de calcul élevée pour répondre aux besoins des applications qui nécessitent un calcul de haute performance
- l'utilisation d'ensembles des données très larges.

Problématique :
Dans ce mémoire, nous nous intéressons particulièrement aux Cloud Computing, qui permet de connecter des centaines de machines et de ressources de stockage distribuées à travers le monde. Ce type d'infrastructure est utilisé par les applications intensives qui génèrent des quantités importantes de données. Ces applications peuvent avoir besoin de données produites par d'autres applications dans un endroit géographiquement distant. Comme les données représentent la ressource la plus importante, une gestion et un accès efficace à ce volume de données présentent des défis majeurs.

Les applications scientifiques peuvent être très complexes. Une tâche peut nécessiter l'exécution de nombreuses données. En outre, un même ensemble de données peut également être requis par plusieurs tâches, si plusieurs ensembles de données sont toujours utilisées par de nombreuses tâches, on dit que cet ensemble de données dépendent les uns des autres.

Objectif :

L'objectif de notre travail et de proposer une stratégie d'ordonnancement de tache basée sur le placement de données pour un type particulier des workflows scientifiques, Notre stratégie de gestion de données comporte cinq phases :

la première phase consiste à la construction des ensembles de données à partir des graphes qui représentent les workflow scientifiques, la deuxième phase consiste à la création de la matrice de dépendance (DM), ensuite nous utilisons l'algorithme BEA(Bond Energy Algorithm) [29] pour clusteriser les données afin de minimiser le déplacement de données suivi d'une phase de partitionnement et la dernière phase représente l'exécution des taches.

Organisation du mémoire

Le présent mémoire est structuré autour de quatre principaux chapitres qui se résument comme suit :

Chapitre 1 : Dans le premier chapitre, décrit les caractéristiques de Cloud Computing.

Chapitre 2 : Le second chapitre présentera quelques techniques l'ordonnancement des tâches d'un Worflow scientifique .

Chapitre 3 : Le troisième chapitre sera réservé à la description détaillée de notre stratégie proposée .

Chapitre 4 : Ce dernier chapitre présentera les étapes de l'implémentation de l'approche proposée.

Enfin, Une synthèse et un ensemble de perspectives viendront pour cloturer notre travail.

Chapitre 1

Contexte sur le Cloud Computing

Sommaire

1.1. Introduction

L'informatique dans le nuage est plus connue sous sa forme anglo-saxonne :" Cloud Computing ", mais il existe de nombreux synonymes francophones tels que : " informatique dans les nuages ", " infonuagique" (Québec) ou encore "informatique dématérialisée ". C'est un domaine qui regroupe les technologies de distribution, a la demande et via Internet, de services informatiques logiciels et matériels. L'idée principale de ces technologies est de distribuer des ressources informatiques comme un service d'utilite publique, conformement a ce qui avait été imaginé par les pionniers de l'informatique moderne, il y a plus de 40 ans [1]. Ce principe de distribution publique de ressources informatiques anime également la communauté de la grille informatique, si bien qu'il est parfois diffcile de distinguer la frontière entre "Grille " et " Informatique dans le nuage ". Cette difficulté est d'autant plus réelle que l'informatique dans les nuages, est un concept jeune, dont les premières implantations datent de 2006, et dont le développement s'est accélèré durant ces dernières années[2].
Dans ce chapitre, nous allons présenter globalement l'historique du Cloud Computing et l'origine de ce terme, suivi d'une définition explicite de ce dernier qui sera basée sur une analyse des définitions proposées par le monde académique. Nous décrivons aussi la virtualisation qui est une partie essentielle dans " l'informatique en nuages ", sans oublier les services de Cloud, les types de Cloud et ses acteurs ainsi que les avantages, les inconvénients, les objectifs principaux et les domaines 'utilisation du Cloud Computing.[2]

1.2. Les concepts du Cloud Computing

Il y a une certaine confusion dans même l'esprit d'analyse des praticiens expérimentés sur ce que constitue le Cloud Computing et ce qui est le partage en temps ou tout simplement une grande collection de serveurs distants. Cette confusion est aggravée par un grand nombre de fournisseurs de services qui prétendent donner le meilleur et le moins cher pour le calcul dans le nuage sans élucider comment cela est différent de la génération de l'informatique [3, 4].

Puisque nous croyons que le Cloud Computing est plus qu'un mot a la mode, nous reproduisons ici la définition du Cloud Computing par le NIST réputé [5]. Selon l'Institut national des normes et de la technologie, Cloud Computing est un modéle pour permettre un accés pratique à la demande du réseau à un ensemble partagé de ressources informatiques configurables (par exemple, les réseaux, les serveurs, le stockage, les applications et les services) qui peuvent être provisionnés rapidement et libérés avec un effort de gestion minimale ou par l'interaction de fournisseur de services . Ce modèle favorise l'accessibilité et est composé de cinq caractéristiques essentielles[6]

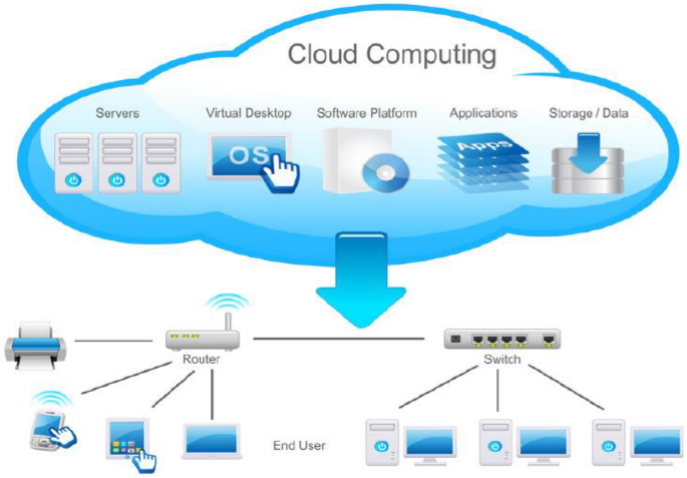

FIGURE 1.1 – *L'environnement de Cloud Computing [6]*

1. La demande libre des services
2. Un accés en discusion via le réseau
3. La mise en commun des ressources
4. L'élasticité rapide
5. Un service mesuré

Trois modéles de services (SaaS, PaaS et IaaS) , et quatre modéles de déploiement (privé, public, communautaire et hybride). Les technologies clés comprennent :

1. Des réseaux rapides
2. Des ordinateurs bon marché
3. La virtualisation pour du matériel de base

Les principaux obstacles a la plus large adoption du Cloud sont : La sécurité, l'interopérabilité et la portabilité. Nous résumons en termes simples et courts, le Cloud Computing est une grande puissance évolutive et personnalisée de calcul disponible par loyer/par heure et accessible à distance. Il peut aider à faire plus de calcul à une fraction de cout.

1.2.1. Architecture d'un système Cloud

Un système de nuage, c'est-à-dire un système qui adopte le paradigme de Cloud Computing, peut être caractérisé par son architecture et les services qu'il offre. L'architecture d'un système de Cloud Computing est généralement structurée en un ensemble de couches.Une architecture typique d'un système de nuages est illustrée à la figure 1.2 .(à partir de Zhang et al 2010)[7]. Au niveau le plus bas de la hiérarchie se trouve la couche matérielle, qui est responsable de la gestion des ressources physiques du système Cloud, telles que les serveurs, le stockage, les périphériques réseau, les systèmes d'alimentation et de refroidissement. Au sommet de la couche matérielle, réside la couche infrastructure, qui fournit un pool de ressources informatiques et de stockage en partitionnant les ressources physiques de la couche matérielle au moyen de technologies de virtualisation. Construite au-dessus de la couche d'infrastructure, La couche plateforme est constituée de systèmes d'exploitation et de cadres d'application. Enfin, au niveau le plus élevé de la hiérarchie se trouve la couche application, constituée d'applications Cloud.

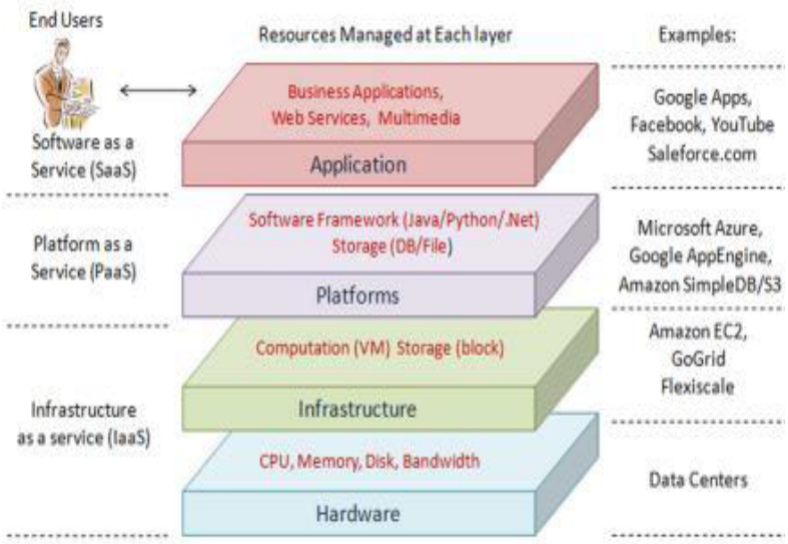

FIGURE 1.2 – *L'Architecture de Cloud Computing[7]*

1.2.2. LA VIRTUALISATION :

La virtualisation consiste à faire fonctionner un ou plusieurs systèmes d'exploitation sur un ou plusieurs ordinateurs. Cela peut sembler étrange d'installer deux systèmes d'exploitation sur une machine conçue pour en accueillir qu'un, mais comme nous le verrons par la suite, cette technique a de nombreux avantages. Il est courant pour des entreprises de posséder de nombreux serveurs, tels que les serveurs de mail, de nom de domaine, de stockage pour ne citer que ceux-ci. Dans un contexte économique où il est important de rentabiliser tous les investissements, acheter plusieurs machines physiques pour héberger plusieurs serveurs n'est pas judicieux. De plus, une machine fonctionnant à 15 pour cent ne consomme pas plus d'énergie qu'une machine fonctionnant à 90 pour cent. Ainsi, regrouper ces serveurs sur une même machine peut donc s'avérer rentable si leurs pointes de charge ne coïncident pas systématiquement. Enfin, la virtualisation des serveurs permet un bien plus grande modularité dans la répartition des charges et la reconfiguration des serveurs en cas d'évolution ou de défaillance momentanée. Les intérêts de la virtualisation sont multiples. On peut citer :[8]

- L'utilisation optimale des ressources d'un parc de machines (répartition des machines virtuelles sur les machines physiques en fonction des charges respectives)

- L'économie sur le matériel (consommation électrique, entretien physique, surveillance)
- L'installation, tests, développements sans endommager le système hôte

1.2.2.1. La paravirtualisation (virtualisation type 1)

La paravirtualisation est une technique de virtualisation qui présente à la machine invitée une interface logicielle similaire mais non identique au matériel réel. Ainsi, elle permet aux systèmes d'exploitation invités d'interagir directement avec le système d'exploitation hôte et donc ils seront conscients de la virtualisation.[9]

1.2.2.2. La virtualisation complète (virtualisation type 2)

La virtualisation complète (en anglais full-virtualization) est une technique de virtualisation qui permet de créer un environnement virtuel complet. En utilisant cette technique, le système d'exploitation invité n'interagit pas directement avec le système d'exploitation hôte et donc il croit s'exécuter sur une véritable machine physique.[8]

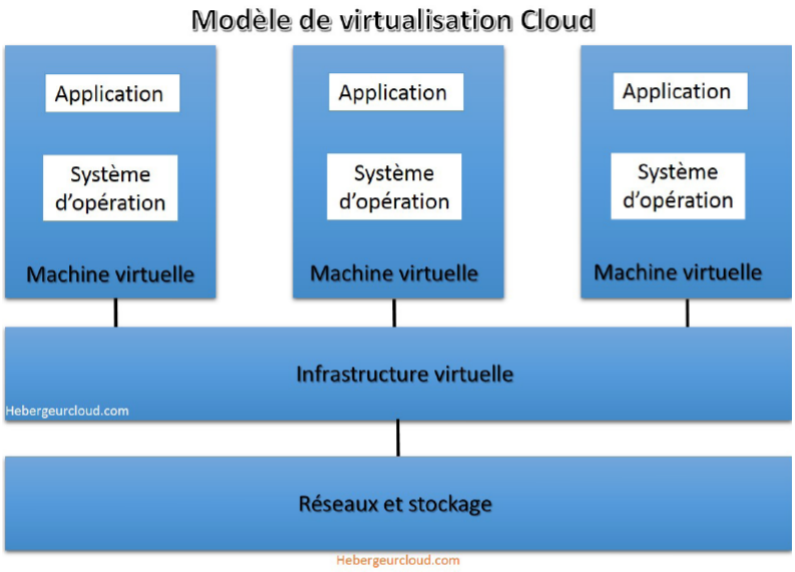

Figure 1.3 – *La virtualisation dans les environnements de Cloud [8]*

1.2.3. La grille informatique :

Grid Computing est un paradigme de calcul distribué qui coordonne en réseau les ressources pour atteindre un objectif commun de calcul. Le développement de la grille informatique a été tirée par les applications scientifiques qui nécessite hàbituellement un calcul intensif, mais les applications nécessitant le transfert et la manipulation d'une quantité massive de données a également été en mesure de tirer parti des grilles. Le Cloud Computing semble être similaire à la grille informatique dans la façon dont il a également employé les ressources distribuées pour atteindre les objectifs au niveau de l'application. Cependant, le Cloud Computing prend un peu plus loin en mettant à profit les technologies de virtualisation pour atteindre le partage à la demande des ressources et le provisionnement dynamique des ressources.[2]

1.2.4. L'informatique utilitaire (Utility Computing)

L'informatique utilitaire représente le modèle d'affaires des ressources d'emballage en tant que services comptés similaires à ceux fournis par les entreprises traditionnelles d'utilité publique. En particulier, il permet aux ressources d'approvisionnement sur les clients à la demande et à la charge basé sur l'utilisation plutôt que sur un taux forfaitaire. Le principal avantage de l'informatique utilitaire est l'économie. Le Cloud Computing peut être perçu comme une réalisation de l'informatique utilitaire. Avec un approvisionnement à la demande des ressources et de la tarification fondée sur l'utilité, les clients sont en mesure de recevoir davantage de ressources pour gérer les pics inattendus et ne payer que pour les ressources dont ils avaient besoin ; Pendant ce temps, les fournisseurs de services peuvent maximiser l'utilisation des ressources et minimiser leurs couts d'exploitation.[2]

1.3. Les technologies connexes liées au Cloud Computing

Le Cloud Computing a évolué sur des décennies de recherche dans différentes technologies, dont il a hérité des caractéristiques et des fonctionnalites telles que les environnements virtualisés, le Computing autonome, la grille informatique, et le calcul distribué. La Figure 1.4 illustre l'évolution vers le Cloud Computing dans l'hébergement des applications logicielles . En fait, le Cloud Computing est souvent comparé aux technologies connexes, dont chacun partage certains aspects avec le Cloud Computing.[9]

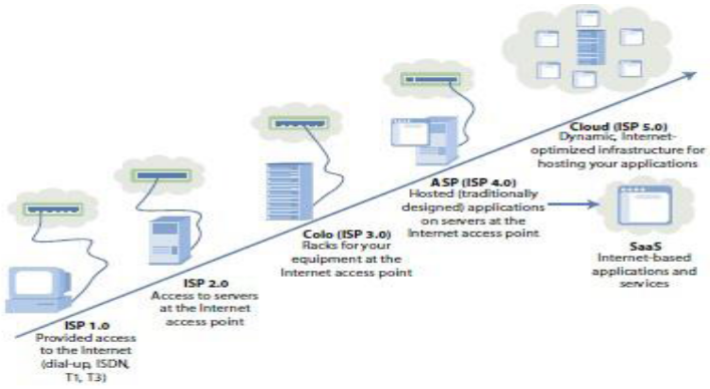

FIGURE 1.4 – *L'évolution vers le Cloud Computing dans l'hébergement d'applications logicielles .[9]*

1.4. LES PRINCIPALES CARACTÉRISTIQUES DES CLOUDS

Le modéle Cloud Computing se différencie par les cinq caractéristiques essentielles suivantes [14] :

- **Un ensemble de ressources en réseau :** accessibles de partout : les ressources sont accessibles via un réseau (Internet ou réseau privé), à partir d'un ou plusieurs sites clients.
- **La mutualisation de ressources :** le fournisseur mutualise les ressources et services qu'il propose à ses clients, ressources qui peuvent se trouver dans plusieurs centres de données réparties à travers le monde (d'où le terme de "nuages") et dont la fourniture est indépendante de la localisation : l'usager ne connaît pas (et n'a a priori pas besoin de connaître) leur situation géographique
- **Un développement plus rapide des produits :** Réduisons le temps de recherche pour les développeurs sur le paramétrage des applications.
- **Un libre-service à la demande :** l'usager peut réserver ou libérer unilatéralement les ressources en fonction de ses besoins sans interaction avec le fournisseur.
- **Un accès rapide et souple à ces ressources :** les ressources peuvent être réservées rapidement pour répondre à des besoins qui évoluent et être libérées tout aussi rapidement lorsque le besoin disparaît.

- **Enfin, une facturation à l'usage :** l'utilisation des ressources et des services associés est contrôlée et mesurée et l'utilisateur facturé en fonction de l'usage qu'il en fait.

1.5. Modéles de déploiement :

Il existe 4 modéles de déploiement du Cloud Computing : - Le Cloud privé, qui peut se déployer sous forme interne ou externe - Le Cloud communautaire - Le Cloud public - Le Cloud hybride

1.5.1. Cloud privé

1.5.1.1. Privé interne :

L'architecture est hébergée par l'entreprise. Ce Cloud privé interne est à l'usage de plusieurs consommateurs appartenant à cette seule entreprise qui est propriétaire de l'infrastructure. Elle peut également être partagée ou mutualisée de façon privée avec les filiales.[14]

1.5.1.2. Privé externe :

Le Cloud privé externe suit la même logique que le Cloud privé interne. La différence est que l'architecture est hébergée chez un prestataire. Elle est entièrement dédiée à l'entreprise et accessible via des réseaux sécurisés ou VPN. Dans la pratique, ces ressources virtualisées «privées » sont directement administrées par l'entreprise par le biais de son service IT (Internet Technology) interne.[14]

1.5.2. Cloud public :

Un Cloud public est basé sur un modèle standard de Cloud Computing, dans lequel un prestataire de services met les ressources, tels que les applications, ou le stockage, à la disposition du grand public via internet. Le Cloud public peut être gratuit ou fonctionner selon paiement à la consommation. L'avantage de ce genre d'architecture est d'être facile à mettre en place, pour des coûts relativement raisonnables. La charge du matériel, des applicatifs, de la bande passante est couverte par le fournisseur. De cette manière ce modèle permet de proposer une souplesse et une évolutivité accrue afin de répondre rapidement au besoin. Il n'y a pas de gaspillage de ressources car le client ne paye que ce qu'il consomme. [14]

1.5.3. Cloud hybride :

Comme nous venons de le voir avec le Cloud public, certains problèmes peuvent se poser, liés à la sécurité de l'information. Il est alors possible de « mélanger » les deux approches du Cloud, privé et public, en débouchant sur une plateforme hybride. Ce terme n'évoque pas vraiment un Cloud en tant que tel, puisqu'il s'agit de faire cohabiter et communiquer un Cloud privé et un Cloud public. Dans le public, on déportera les éléments non sensibles et dans le privé, on gardera les données, applications sensibles liées au business de l'entreprise. Derrière ce terme de « communiquer », beaucoup de problèmes techniques peuvent être rencontrés, comme l'interopérabilité des plateformes. Ce type de Cloud est souvent utilisé dans un but de montée en charge ponctuelle comme le permet le Cloud public, que nous avons vu précédemment. La seule différence est que dans ce cas, il serait lié à un Cloud privé ou interne afin de faire communiquer les deux infrastructures.[14]
La figure 1.5 suivante illustre et résume parfaitement le concept de Cloud hybride :

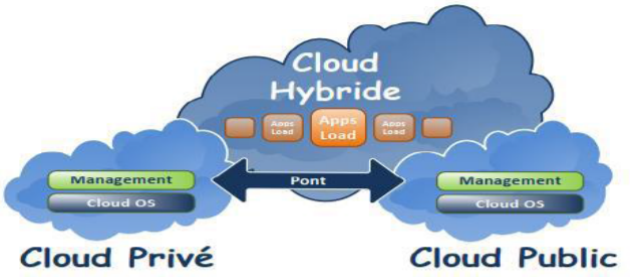

Figure 1.5 – *Cloud hybride [14]*

1.5.4. Cloud communautaire :

L'architecture est dédiée à une communauté professionnelle spécifique incluant partenaires, sous-traitants, etc., pour travailler de manière collaborative sur un même projet. Il permet à plusieurs entreprises ou organisations de partager des ressources en mode Cloud, qui sont alors exclusivement dédiées à ces organisations. Le Cloud communautaire peut être géré par les organisations membres ou par un prestataire externe.[14]

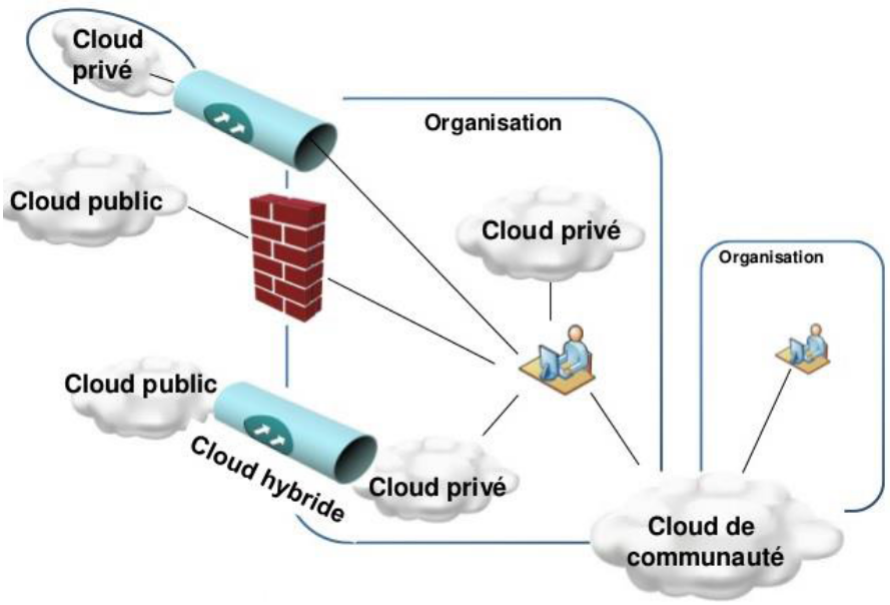

FIGURE 1.6 – *Les modèles de déploiement dans le Cloud Computing[14]*

1.6. MODÈLES DE SERVICE

Pour continuer avec la définition du NIST, celle-ci distingue trois niveaux de services pour le Cloud Computing [14] :

- Software as a Service (SaaS).
- Platform as a Service (PaaS).
- Infrastructure as a Service (IaaS).

Ces trois éléments constituent la pyramide du Cloud Computing.

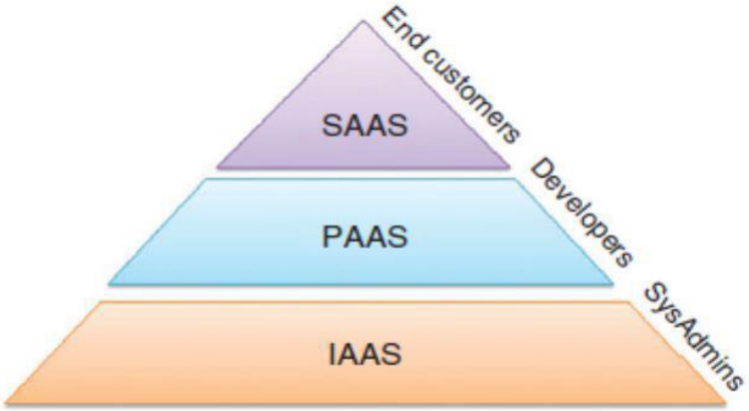

FIGURE 1.7 – *Pyramide du Cloud Computing[14]*

Il est indispensable de bien différencier ces 3 couches pour lesquelles l'entreprise et le fournisseur ont un domaine de responsabilité qui diffère. La figure 1.8 suivante nous présente ses différents domaines en fonction du niveau de service.

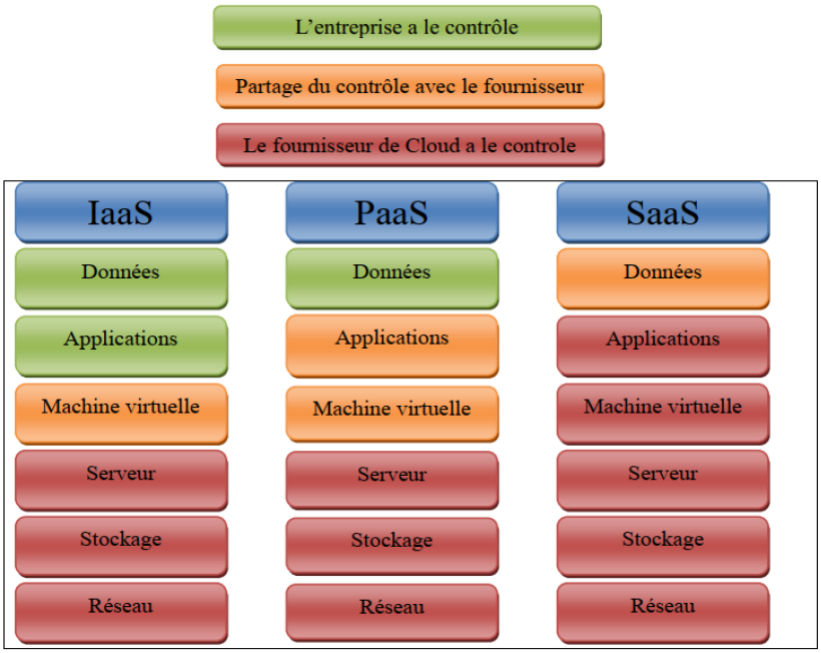

FIGURE 1.8 – *Répartitions des responsabilités[14]*

1.6.1. Software as a Service (SaaS)

Ce modèle de service est caractérisée par l'utilisation d'une application partagée qui fonctionne sur une infrastructure Cloud. L'utilisateur accède à l'application par le réseau au travers de divers types de terminaux (via un navigateur web). L'administrateur de l'application ne gère pas et ne contrôle pas l'infrastructure sous-jacente (réseaux, serveurs, applications, stockage). Il ne contrôle pas les fonctions de l'application à l'exception d'un paramétrage de quelques fonctions utilisateurs limitées.[14]

1.6.2. Platform as a Service (PaaS) :

L'utilisateur a la possibilité de créer et de déployer sur une infrastructure Cloud PaaS ses propres applications en utilisant les langages et les outils du fournisseur. L'utilisateur ne gère pas ou ne contrôle pas l'infrastructure Cloud sous jacente (réseaux, serveurs, stockage) mais l'utilisateur contrôle l'application déployée et sa configuration.[14]

1.6.3. Infrastructure as a Service (IaaS) :

L'utilisateur loue des moyens de calcul et de stockage, des capacités réseau et d'autres ressources indispensables (partage de charge, pare-feu, cache). L'utilisateur a la possibilité de déployer n'importe quel type de logiciel incluant les systèmes d'exploitation. L'utilisateur ne gère pas ou ne contrôle pas l'infrastructure Cloud sous jacente mais il a le contrôle sur les systèmes d'exploitation, le stockage et les applications. Il peut aussi choisir les caractéristiques principales des équipements réseau comme le partage de charge, les pare-feu, etc.
Fournisseurs actuels : Amazon AWS , Google Compute Engine , Microsoft Azure , IBM SmartCloud Enterprise , Rackspace Cloud , HP Enterprise Converged Infrastructure .[14]

1.6.4. Avantages et Inconvénients des services :

Le tableau 1.1 illustre les services de Cloud Computing qui ont été décrites dans la section précédente tout en montrant les avantages et les inconvénients de chaque service .

	Avantage	inconvenient
Saas	-pas d'installation -plus de licence -migration	-logiciel limité -sécurité -dépendance des prestataires
PaaS	-pas d'infrastructure nécessaire -pas d'installation -environnement hétérogène	-limitation des langages -pas de personnalisation dans la configuration des machines virtuelles
IaaS	-administration -personnalisation -flexibilité d'utilisation	-sécurité -besoin d'un administrateur système

TABLE 1.1 – *Les avantages et les inconvénients des différents services.[12]*

1.7. Les avantages et Les inconvénients du Cloud Computing

1.7.1. Les avantages :

- **Reprise après sinistre :** la perte de données est une préoccupation majeure pour toutes les organisations, ainsi que la sécurité des données. Le stockage de notre données dans le Cloud garantit leur disponibilité permanente, même si notre équipement, tel qu'un ordinateur portable est endommagé. Les services en nuage permettent une récupération rapide des données pour tous les types de scénarios d'urgence, des catastrophes naturelles aux pannes de courant.[13]

- **Contrôle :** avoir le contrôle des données sensibles est vital pour toute entreprise. On ne sait jamais ce qui peut arriver si un document tombe entre de mauvaises mains, même s'il ne s'agit que des mains d'un employé non formé. Le Cloud permet une visibilité et un contrôle complets sur nos données. On peut facilement décider quels utilisateurs ont quel niveau d'accès à quelles données. Cela nous donne le contrôle, mais cela simplifie également le travail, car le personnel saura facilement quels documents lui sont affectés. Cela augmentera et facilitera également la collaboration. Etant donné qu'une version du document peut être travaillée par différentes personnes, il n'est pas nécessaire d'avoir des copies du même document en circulation.[13]

- **Simplicité :** l'entreprise cliente n'a plus besoin de développements coûteux et déplace la responsabilité du fonctionnement et de maintenance sur le fournisseur.[13]
- **Liberté de changer :** le Cloud étant généralement facturé à la demande ou par abonnement mensuel, il est très facile pour une entreprise d'arrêter le service si elle n'en a plus besoin ou si elle souhaite aller chez un concurrent.[13]

1.7.2. Les inconvénients :

- **Confidentialité et sécurité des données :** les données sont hébergées en dehors de l'entreprise. Cela peut donc poser un risque potentiel pour l'entreprise de voir ses données mal utilisées ou volées. Il s'agit donc de s'assurer que le fournisseur dispose d'une sécurité suffisante et qu'il propose une politique de confidentialité concernant les données de l'utilisateur par les SLA (Service Level Agreement) qui est un document ou contrat qui définit la qualité de service requise entre un prestataire et un client. [13]
- **Dépendance à Internet :** en absence de connexion Internet, on n'a plus accès aux services.[13]
- **La bande passante peut faire exploser votre budget :** La bande passante qui serait nécessaire pour mettre cela dans le Cloud est gigantesque, et les coûts seraient tellement importants qu'il est plus avantageux d'acheter le stockage nous-mêmes plutôt que de payer quelqu'un d'autre pour s'en charger.[13]
- **Les performances des applications peuvent être amoindries :** Un Cloud public n'améliorera définitivement pas les performances des applications.[13]
- **La fiabilité du Cloud :** Un grand risque lorsqu'on met une application qui donne des avantages compétitifs ou qui contient des informations clients dans le Cloud.[13]
- **Taille de l'entreprise :** Si votre entreprise est grande alors vos ressources sont grandes, ce qui inclut une grande consommation du Cloud. Vous trouverez peut- être plus d'intérêt à mettre au point votre propre Cloud plutôt que d'en utiliser un externalisé. Les gains sont bien plus importants quand on passe d'une petite consommation de ressources à une consommation plus importante.[13]

1.8. Sécurité dans le Cloud :

La sécurité est un des critères les plus importants lors de la recherche d'un prestataire pour stocker des données dans le Cloud. En effet, c'est vrai pour des données personnelles, mais ça l'est encore plus pour les données d'une entreprise, souvent sensibles et confidentielles. Plusieurs points doivent être abordés afin de s'assurer que toutes les mesures de sécurité sont mises en place :

- **La localisation des data centres :** certains hébergeurs possèdent plusieurs centres de données. Il faut donc que l'entreprise cliente soit en mesure de savoir où se localisent géographiquement ses données. [14]
- **Des garanties d'audits et de contrats :** le prestataire choisi doit pouvoir démontrer via différents audits que toutes les règles de sécurité sont optimales et à jour. [14]
- **Une disponibilité des données constantes :** la sécurité des données, c'est aussi s'assurer que ces dernières seront disponibles à tout moment et ne seront pas perdues même en cas de soucis techniques. Un bon prestataire doit proposer à l'entreprise un Plan de Reprise d'Activité (PRA) après sinistre, qui permettra de dupliquer les données et l'infrastructure Cloud afin que celles-ci restent disponibles même si les serveurs initiaux rencontrent un problème.[14]
- **La sécurisation des données en mouvement :** la sécurité des données passe donc par la qualité du relationnel et la transparence administrative avec son prestataire, mais aussi par la qualité purement technique de son offre.Les données sont vulnérables lorsqu'elles sont en mouvement. C'est pourquoi les flux sont hautement sécurisés par les hébergeurs de confiance. [14]

1.9. Conclusion

Dans ce chapitre nous avons introduit les principales caractéristiques des systèmes à large échelle et en particulier les Cloud Computing dans un système distribué mais l'apparition de nouveaux problèmes on outre .ordonnancer les diverses tâches et de placer les données de façon á ce que l'exécution se fasse dans des conditions optimales Le chapitre suivant décrira la technique de l'ordonnancement de tâches et son rôle.

Chapitre 2

Stratégies d'ordonnancement de workflow scientifique dans le cloud computing

Sommaire

2.1. Introduction

Les workflows ont été adoptés comme mécanisme attrayant de représentation des applications scientifiques intensives en données. Les workflows sont utilisés dans plusieurs domaines pour représenter des applications, où les tâches sont liées par des dépendances de données et la représentation de ces applications se fait sous forme de DAG. Les workflows scientifiques sont des applications de calcul à haute performance, nécessitent beaucoup de calculs et de données ainsi q'un environnement HPC pour leur exécution. Le cloud prend en charge les workflows scientifiques tout en offrant des ressources indispensables à leur exécution sous forme d'instance de calcul, de stockage et de réseau. Le processus d'affectation d'une tâche à une instance pour son exécution est appelé ordonnancement de workflow (scheduling workflow).

2.2. Ordonnancement : Concepts et définitions :

Le probléme d'ordonnancement consiste a organiser dans le temps la réalisation de tâches, compte tenu de contraintes temporelles (contraintes de délai, contraintes d'enchainement, ...) et de contraintes portant sur l'utilisation et la disponibilité des ressources requises

2.2.1. Ordonnancement :

Un problème d'ordonnancement consiste a ordonner dans le temps un ensemble de tâches contribuant a la réalisation d'un meme projet. L'objectif est de minimiser la durée de réalisation du projet compte tenu des contraintes d'antériorité reliant les diférentes tâches. De plus, on détermine les calendriers de réalisation de chacune de ces tâches ainsi que les marges de manoeuvre associées..[15, 16]

2.2.2. Les tâches :

Une tâche ou un job est une entité élémentaire localisée dans le temps, par une date de début et une date de fin, et dont la réalisation nécessite une durée préalablement définie. Elle est constituée d'un ensemble d'opérations qui

requiert, pour son exécution, certaines ressources et qu'il est necessaire de programmer de façon a optimiser un certain objectif.[2]

2.2.3. Les ressources :

La ressource est un moyen technique ou humain destiné a etre utilisé pour la réalisation d'une tâche et disponible en quantité limitée, sa capacité. Plusieurs types de ressources sont à distinguer. Une ressource est renouvelable si aprés avoir été allouée a une ou plusieurs tâches, elle est a nouveau disponible en mème quantité (les hommes, les machines, l'équipement en général) ; la quantité de ressource utilisable a chaque instant est limitée.

Dans le cas contraire, elle est consommable (matières premiéres, budget) ; la consommation globale (ou cumul) au cours du temps est limitée. Une ressource est doublement contrainte lorsque son utilisation instantanée et sa consommation globale sont toutes deux limitées (l'argent en est un bon exemple). Qu'elle soit renouvelable ou consommable, la disponibilité d'une ressource peut varier au cours du temps. Sa courbe de disponibilité est en général connue a priori, sauf dans les cas ou elle dépend du placement de certaines tâches génératrices. On distingue par ailleurs principalement dans le cas de ressources renouvelables, les ressources disjonctives qui ne peuvent exécuter qu'une tâche a la fois (machine-outil, robot manipulateur) et les ressources cumulatives qui peuvent etre utilisées par plusieurs tâches simultanément mais en nombre limité (équipe d'ouvriers, poste de travail).[2]

2.3. WORKFLOW SCIENTIFIQUES :

La notion de workflow ("flux de travail" en français) est apparue pour la toute première fois dans l'industrie de l'imagerie électronique et de la gestion assistée par ordinateur , afin d'automatiser les processus de travail au sein des organisations. De plus, dans les infrastructures actuelles distribuées, gérant des ressources hétérogènes, telles que le cloud computing, bénéficier d'un environnement autorisant la définition et l'exécution des chaînes de traitement constitue une des fonctionnalités essentielles recherchée par les scientifiques et par le grand public.
Deux grandes catégories d'usages utilisent la notion de workflow :
les protocoles expérimentaux, dans des domaines tels que la biologie, l'astronomie, la physique, la neuroscience, la chimie, etc. (appelées workflows scientifiques) et les chaînes de traitement pratiquées dans des domaines commerciaux, financiers, pharmaceutiques (appelés processus métiers). Elles donnent lieu à plusieurs pistes de recherche diverses, mais connexes. Dans le cadre de notre travail ,

nous traitons plus particulièrement les workflows scientifiques.[17]

2.3.1. Un workflow scientifique :

est la description d'un processus pour atteindre un objectif scientifique, généralement constitué d'un ensemble de tâches et des dépendances de données entre celles-ci.En règle générale, les tâches de workflows scientifiques sont des étapes de calcul pour des simulations scientifiques ou des étapes d'analyse de données. Les éléments ou étapes courants des workflows scientifiques sont : l'acquisition, l'intégration, la réduction, la visualisation et la publication (par exemple, dans une base de données partagée) de données scientifiques. Les tâches d'un workflow scientifique sont organisées (au moment de la conception) etorchestrées (au moment de l'exécution) en fonction du flux de données et éventuellement d'autres dépendances spécifiées par le concepteur de workflow.[18]

Les worflows peuvent être conçus visuellement, par exemple, à l'aide de diagrammes de blocs, ou textuellement à l'aide d'un langage spécifique au domaine.

2.3.2. Caractéristiques du workflow scientifique :

Les applications parallèles à forte intensité de données (notamment les workflows scientifiques) sont couramment utilisées dans la majorité des disciplines exploitant souvent des ressources de données riches et variées ainsi que des plateformes informatiques parallèles et distribuées. Les workflows fournissent un moyen systématique de décrire les méthodes nécessaires pour représenter une application parallèle. Ils assurent l'interface entre les spécialistes du domaine et les infrastructures informatiques. Les systèmes de gestion de workflow (WMS) effectuent les analyses complexes sur une variété de ressources distribuées. Avec l'augmentation spectaculaire des volumes de données et de la diversité dans chaque domaine, les workflows jouent un rôle de plus en plus important, permettant aux chercheurs de formuler des méthodes de traitement et d'analyse afin d'extraire des informations à partir de sources de données multiples et d'exploiter un très large éventail de données et de plateformes de calcul. Pour

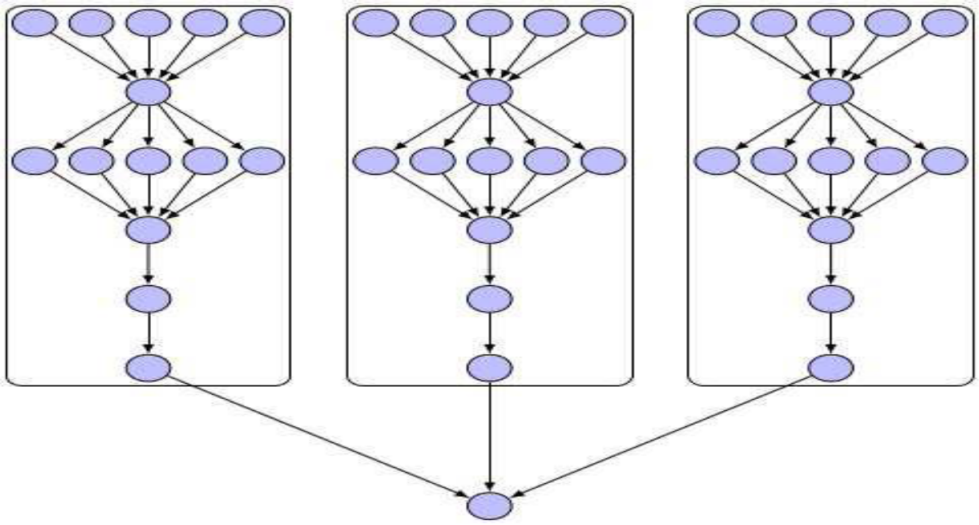

FIGURE 2.1 – *Exemple simple de workflow [18]*

mener à bien une étude, les workflows sont très souvent modélisés sous forme de DAG comme sus-mentionné.

Un workflow, comme l'illustre la Figure 2.1 , est un ensemble de noeuds et d'arcs. Les noeuds représentent les tâches du workflow et les arcs entre les différentes tâches représentent soit une dépendance de données, c'est-à-dire un transfert de fichier, soit une dépendance de flux entre deux tâches. Chacune des tâches composant le workflow a une durée estimée, nécessite un ensemble de fichier d'entrée pour démarrer son exécution et produit un ensemble de fichier de sortie à la fin. Lorsqu'un fichier de sortie produit par une tâche est consommé en entrée par une autre, cela crée une dépendance de données entre les deux tâches. La dépendance entre les deux est représentée par un arc orienté. Les fichiers d'entrée qui ne sont générés par aucune des tâches du workflow sont appelés fichiers d'entrée du workflow. Inversement, les fichiers de sortie qui ne sont pas consommés par une tâche sont appelés les fichiers de sortie du workflow.[18]

2.4. ORDONANCEMENT DES TÂCHES DANS LE CLOUD COMPUTING :

La performance des applications est reliée à la bonne gestion de l'utilisation des ressources, donc il est nécessaire de proposer des algorithmes et des outils

efficaces pour gérer une utilisation efficace des ressources. Plusieurs algorithmes d'ordonnancements ont été proposés dans la plate- forme de type cloud. Le travail de Bessai.K dans [19] a été proposé une classification, des algorithmes existants, en trois catégories selon les types de tâches constituant l'application : tâches indépendantes, tâches dépendantes (DAG) et les DAGs s'exécutant en parallèle.

2.4.1. L'ordonnancement de tâches indépendantes :

La plupart des études ont été proposé dans leurs études des tâches indépendantes par rapport aux deux autres types d'applications, cependant très rare des travaux utilisent des plates-formes hétérogènes. Les heuristiques d'ordonnancement dynamique des tâches indépendantes et séquentielles proposés dans [16] sont classées en deux classes [27]

- **Des heuristiques d'ordonnancement en ligne :** Les tâches qui arrivent dans le système sont ordonnancées et affectées à des ressources dès leurs arrivées.[20]
- **Des heuristiques d'ordonnancement par lot (batch sheduling) :** Les tâches ne sont pas exécutées à leurs arrivées mais regroupées dans des ensembles distincts et les dates de leurs ordonnancements sont déterminées ultérieurement. Les tâches parallèles sont aussi étudiées par des nombreuses heuristiques dyna- miques, L'objectif de ces heuristiques est essentiellement l'augmentation du taux d'utilisation des ressources et la minimisation du temps d'attente des tâches.[19]

2.4.2. Ordonnancement de tâches dépendantes :

Les applications composées de tâches dépendantes visent à optimiser leurs temps d'exécution en se focalisant sur des applications définies par des tâches liées par des contraintes de précédence (DAGs). L'ordonnancement de graphes de tâches composées détâches séquentielles a été largement étudié et plusieurs algorithmes heuristiques ont été proposés. Ces algo- rithmes peuvent être classés en trois catégories [19]

2.4.3. Algorithmes d'ordonancement :

Dans cette section, nous présentons certains algorithmes d'ordonnancement dans l'environnement Cloud, chacun d'eux est brièvement présenté.

2.4.3.1. Algorithme d'ordonnancement Min-Min :

Dans cet algorithme, nous attribuons la tâche qui a un temps d'exécution minimum à une ressource qui fournit le temps d'achèvement (completion time) minimum. Lorsque la ressource termine une tâche, son heure de disponibilité est mise à jour. Ce processus s'exécute séquentiellement jusqu'à ce que toutes les tâches souhaitées soient planifiées. L'algorithme d'ordonnancement Min-Min sélectionne d'abord la plus petite tâche et la planifie, puis passe aux autres tâches. Cet algorithme fonctionne en tenant compte de la durée d'exécution, le temps d'achèvement et de la disponibilité de la ressource. L'inconvénient majeur de cet algorithme est la charge déséquilibrée sur les ressources. Étant donné que cet algorithme est prioritaire sur le temps d'achèvement minimum d'une tâche, il planifie donc initialement les tâches qui ont des temps d'achèvement minimum. De cette façon, l'algorithme Min-Min est au profit des petites tâches et son principal avantage est son faible temps de réponse.[21]

2.4.3.2. Algorithme d'ordonnancement Max-Min :

Cet algorithme est similaire à l'algorithme Min-Min sauf que la tâche, qui a une durée d'exécution maximale, est affectée à la ressource, qui fournit le temps d'achèvement minimum. En d'autres termes, l'algorithme d'ordonnancement Max-Min planifie d'abord les tâches les plus importantes, puis passe aux autres tâches. Ce processus s'exécute séquentiellement jusqu'à ce que toutes les tâches souhaitées soient planifiées. Cet algorithme est utilisé lorsqu'il y a plus de tâches courtes que de grandes tâches. Le principal avantage de cet algorithme est sa simplicité de mise en oeuvre.[22]

2.4.3.3. L'algorithme d'ordonnancement du temps d'achèvement minimum (MCT) :

Cet algorithme attribue une tâche à une ressource qui a le temps d'achèvement le plus bas. Tout d'abord, la tâche est entrée dans la liste des tâches non planifiées, puis la recherche est effectuée pour trouver la ressource avec le temps d'achèvement le moins élevé, et la tâche est affectée à cette ressource. Cet algorithme améliore considérablement le temps d'exécutions totales des tâches. Cependant, la maintenance des informations sur les ressources et les tâches, qui sont calculée par le planificateur de tâches, a un coût élevé. Cet algorithme vise à ignorer la surcharge des communications dans l'ordonnancement et à réduire au maximum le temps moyen d'exécution des tâches sur les ressources.[24]

2.4.3.4. L'algorithme du tourniquet/ à tour de rôle (Round Robin) :

Il définit une boucle comme une file d'attente et indique également un quantum à temps constant. Chaque tâche ne peut être effectuée qu'avec ce quantum et à son tour. Si une tâche dans un quantum n'est pas terminée, elle retournera dans la file d'attente et attendra son prochain tour. Cet algorithme se concentre principalement sur l'ordonnancement des processus de manière équitable. Le principal avantage de cet algorithme est que chaque tâche est exécutée à son tour et il n'est pas nécessaire d'attendre que les tâches précédentes soient terminées. Cependant, si la file d'attente est pleine ou si la charge de travail est trop lourde, il faut beaucoup de temps pour terminer tout le travail. De plus, la sélection d'un quantum de temps approprié est difficile à planifier dans cet algorithme.[21]

2.4.3.5. Algorithme FIFO (First In First Out) :

dans cet algorithme, les tâches sont reçues et mises en file d'attente jusqu'à ce que la ressource soit disponible ; une fois que c'est le cas, les tâches leur sont mappées en fonction de leur heure d'arrivée. Aucun autre critère n'est envisagé pour cette technique.[25]

2.4.3.6. Algorithme le Plus court d'abord (Shortest Job First (SJF)) :

Il ressemble au FCFS algorithme, mais dans SJF on choisit la tâche avec le minimum temps d'exécution au lieu d'exécuter dans l'ordre d'arrivé (le cas de FCFS). L'inconvénient de cette méthode est de déterminer le temps d'exécution d'une tâches avant l'exécuter. [26]

2.5. Conclusion

La théorie d'ordonnancement de tâches dans les systèmes de Cloud computing suscite une attention croissante avec l'augmentation de la popularité de Cloud. En général, l'ordonnancement de tâches est le processus d'affectation des tâches aux ressources disponibles sur la base des caractéristiques et des conditions des tâches. C'est un aspect important dans le fonctionnement efficace du Cloud, car de divers paramètres de tâches doivent etre pris en considération pour un ordonnancement approprié. Les ressources disponibles devraient etre utilisees efficacement sans affecter les paramètres de service du Cloud.

Chapitre 3

Desciption et Modelisation de l'approche proposée

Sommaire

3.1. Introduction

La plate-forme cloud computing est caractérisée par l'utilisation d'un grand pool de ressources informatiques accessibles à la demande à travers Internet. Il offre une variété de ressources, tels que le réseau, le stockage, aux utilisateurs adoptées par IaaS, PaaS, SaaS ... etc. Dans les Workflows scientifiques intensifs en données ou calcul, les tâches nécessitent l'exécution de plusieurs jeux de données. Cependant, lorsque ces tâches sont exécutées dans différents centres de données, le transfert de données deviendrait inévitable. Pour résoudre ce problème, nous proposons d'implémenter l'algorithme BEA dans l'objectif de trouver une meilleure affectation des tâches et des données à l'ensemble des instances des machines virtuelles (VMs) pour une meilleure exécution des applications distribuées.[28]

3.2. Representation du workflow scientific

Un worfklow scientifique est défini comme un ensemble de tâches d'une (unique) application parallèle décrites par un DAG. Dans la suite nous utiliseron simplement le mot workflow pour désigner un workflow scientifique. Exécuter une application parallèle peut prendre énormément de temps si le nombre de ressources est insuffisant. C'est pourquoi quand la taille des données devient très importante ou quand les calculs deviennent extrêmement intensifs, il est nécessaire d'utiliser des plates-formes distribuées comme le cloud computing, afin d'obtenir des gains de temps. Le cycle de vie des workflows est divisé en trois parties :

- la composition du DAG, c'est-à-dire l'écriture du programme sous forme de DAG ;
- le placement (l'ordonnancement des tâches sur les ressources) ;
- l'exécution ;

Ce qui nous intéresse dans ce travail est l'étape du placement : l'ordonnancement, qui est un point crucial pour obtenir de bonnes performances et gagner du temps. Bien ordonnancer les tâches d'un workflow sur les ressources du cloud computing, c'est optimiser l'affectation des différentes tâches sur les différentes

ressources disponibles. Un workflow scientifique est représenté par un graphe orienté et sans cycle noté G =(T,E) Avec : T =t1, t2 , t3,. . .. . ., tn est l'ensemble fini des tâches qui le compose et qui sont exécutées de façon indépendantes.
E = est l'ensemble de ses arcs représentant les contraintes de données entre les tâches T. La Figure suivante représente un exemple d'une couche de workflow scientifique constituée de cinq tâches et cinq données notées respectivement t1 , t2 ,...,t5 et d1,d2,..,d5.

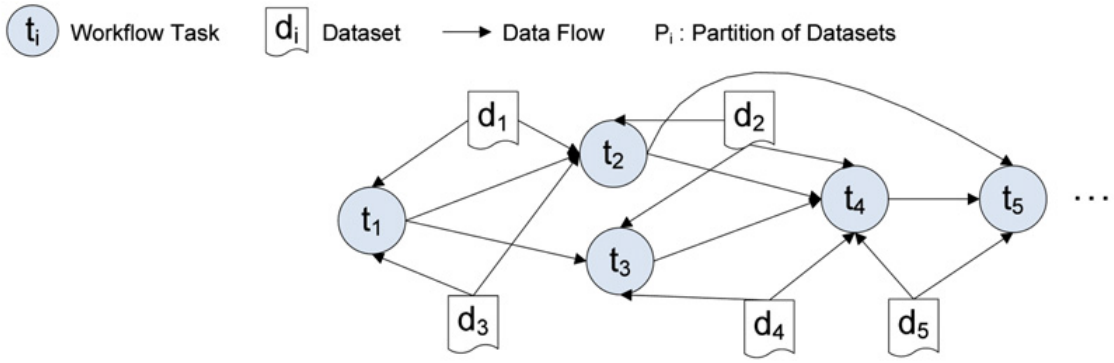

Figure 3.1 – *Exemple de workflow*

3.3. Description de l'approche proposée :

Dans ce travail , nous proposons une stratégie de regroupement basée sur une matrice pour le placement de données dans des systèmes de workflows scientifiques dans le cloud.
Les workflows scientifiques peuvent être très complexes, une tâche peut nécessiter de nombreux ensembles de données pour son exécution ; en outre, un ensemble de données peut également être requis par de nombreuses tâches. Si certains jeux de données sont toujours utilisés ensemble par de nombreuses tâches, on dit que ces jeux de données sont dépendants les uns des autres.
Dans notre stratégie, nous essayons de conserver ces ensembles de données dans une machine virtuelle, de sorte que lorsque des tâches ont été planifiées pour cette machine virtuelle, la plupart, ou la totalité, des données dont ils ont besoin soient stockées localement.
Notre stratégie de placement de données comporte cinq phases,la première phase pour la construction de ensembles de données à partir d'un graphe de workflow, la deuxiéme phase est la construction de matrice de dépendance(DM)

à partir de l'ensembles de données, la troisième phase est la construction d'une matrice clustérise(CM) à partir de matrice de dépendance(DM), la quatrième phase est le partitionnement de matrice CM aux ensembles de sous-matrice (CMt), et les distribuées aux différentes virtuelle machines, et la dernière phase est l'exécution des tâches .
Dans l'algorithme d'étape de construction, nous construisons une matrice de dépendance pour toutes les données d'application, qui représente les dépendances entre tous les ensembles de données. Ensuite, nous utilisons l'algorithme BEA [29] pour regrouper la matrice et la partitionner de sorte que les ensembles de données de chaque partition dépendent fortement les uns des autres.
Nous distribuons les partitions dans k machine virtuelle, où les partitions ont des ensembles de données à emplacement fixe sont également placés dans les machine virtuelle appropriés.
Notre stratégie tente de minimiser le mouvement total des données lors de l'exécution des workflows. De plus, avec la pré-allocation des données à d'autres machines virtuelles, notre stratégie peut empêcher la collecte de données dans une machine virtuelle et réduire le temps passé à attendre les données en garantissant que les données pertinentes sont stockées localement.
En effet, les applications scientifiques comportent un grand nombre de tâches et nécessitent beaucoup de temps pour leur exécution.
Dans cette section, nous discuterons en détail de notre stratégie de placement de données.

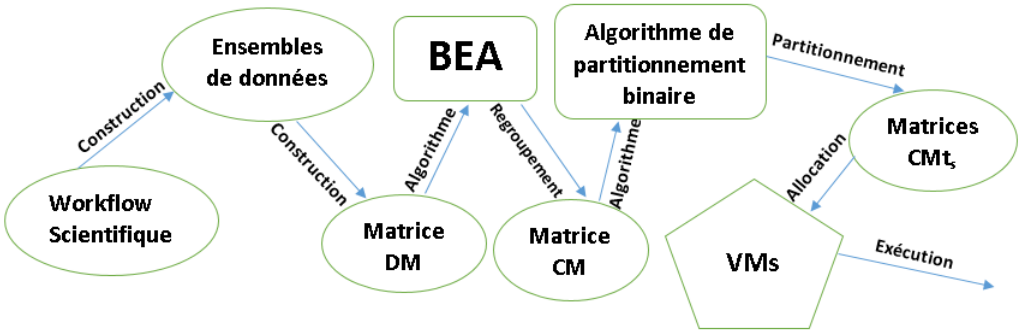

FIGURE 3.2 – *Architecture de notre stratégie*

3.3.1. PHASE01 :CONSTRUCTION DE L'ENSEMBLE DE DONNÉES

Dans cette étape, nous déterminons les ensembles des données et les tâches qui utilisent ces données . soit l'exemple suivant qui représente un modèle du

workflow :

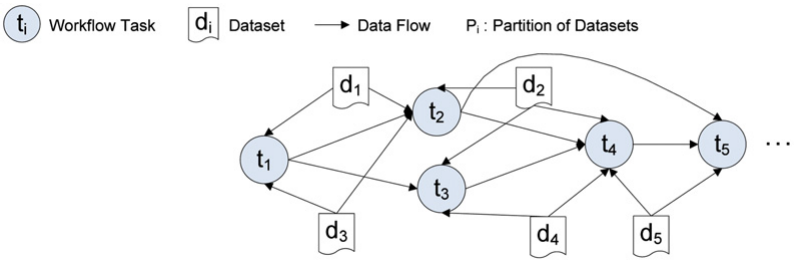

Figure 3.3 – *Exemple d'un instance de workflow scientifique*

Depuit l'exemple si desous nous constructions les ensembles de données comme suit :

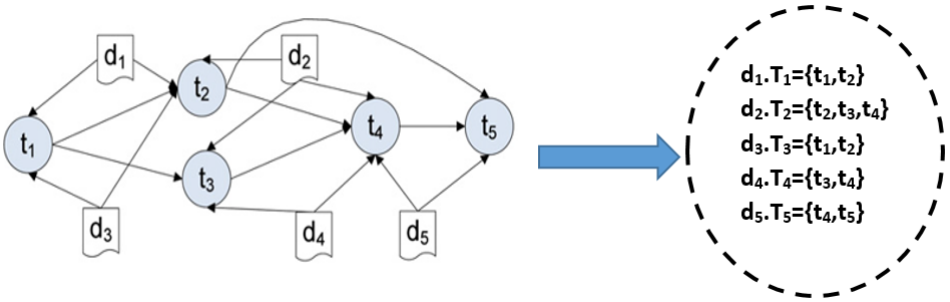

Figure 3.4 – *Construction des ensembles de données*

plus de précisions : d1.T1={t1,t2} : signifie que la donnée d1 est utilisée par les deux tâches t1 et t2 .
d2.T2={t2,t3,t4} : signifie que la donnée d2 est utilisée par les trois tâches t2 , t3 et t4 ... etc.
Dans cette phase, nous avons créé l'ensemble de données basé sur le graphe de dépendance taches avec les données. Nous utiliserons ces ensembles des données dans la prochaine phase pour construire la matrice de dépendance DM.

3.3.2. Phase02 : Construction de la matrice de dépendance (DM) :

Au cours de cette phase, nous utilisons un modèle matriciel pour représenter les données existantes.

Tout d'abord, nous calculons les dépendances de données de tous les ensembles de données tout en construisant une matrice de dépendance DM (Dependency Matrix), ou $DM_{ij} = dépendance_{ij}$ .est la valeur de dépendance entre les ensembles de données d_i et d_j. Il est calculé en comptant les tâches en commun entre les ensembles de tâches de di et dj, qui sont notées Ti et Tj.
En particulier, pour les éléments de la diagonale de DM, chaque valeur signifie le nombre de tâches qui utiliseront cet ensemble de données.
DM est une matrice symétrique n*n où n est le nombre total d'ensembles de données existants. Si nous prenons l'exemple de flux de travail simple de la figure 3.5 (avec seulement 5 jeux de données, à savoir d1 à d5, dans le système initialement), la matrice de dépendance DM est illustrée à la même figure

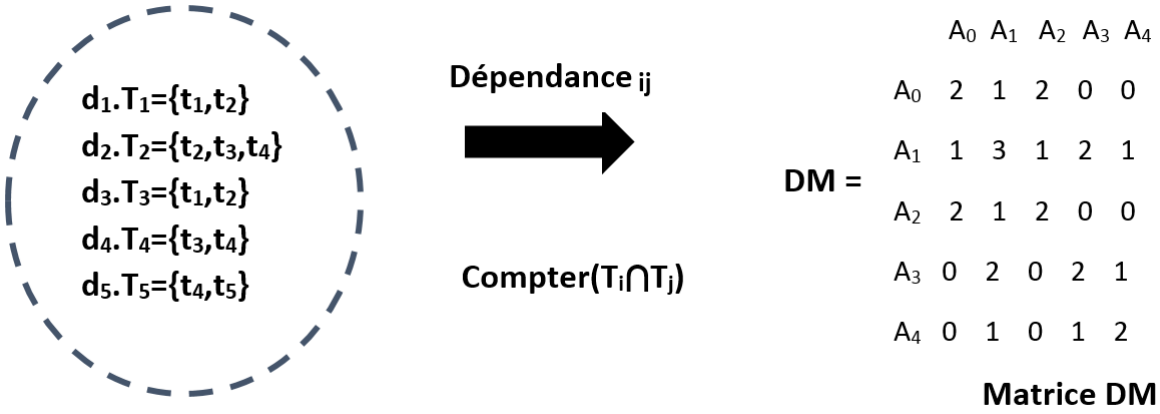

Figure 3.5 – *Construction de la matrice de dépendance DM*

Plus de précision :
La donnée d1 est utilisé par deux tâches(t1,t3) alors ($d_1 \cap d_1$=2).
Les données d1 et d2 ont une tâche commune (t2) alors ($d_1 \cap d_2$=1).
Les données d1 et d5 ont aucune tâche commune alors ($d_1 \cap d_5$=0)....etc.
Dans cette phase, nous avons construit la matrice de dépendance DM à partir de l'ensemble de données que nous avons créé à l'étape précédente. Dans l'étape suivante, nous utiliserons la matrice DM pour créer une autre matrice, clustérisé

appelé la matrice CM .

3.3.3. Phase03 : Construction de la matrice clustérisé (CM)

Après la construction de la matrice de dépendance DM, il faux regrouper les données en utilisons l'algorithme de BEA (Bond Energy Algorithm) pour transformer la matrice de dépendance DM. Le BEA a été proposé en 1972 [29] et a été largement utilisé dans les systèmes de bases de données distribuées pour la partition verticale de grandes tables . C'est un algorithme de permutation qui peut regrouper des éléments similaires dans la matrice en permutant les lignes et les colonnes. et Il est jugé approprié pour les raisons suivantes(Hoffer and Severance, 1975) :

- Il est conçu spécifiquement pour déterminer des groupes d'éléments similaires par opposition, par exemple, à un ordre linéaire des éléments (c'est-à-dire qu'il regroupe les attributs avec des valeurs d'affinité plus grandes et ceux avec des valeurs plus petites)
- Les regroupements finaux sont insensibles à l'ordre dans lequel les éléments sont présentés à l'algorithme.
- Le temps de calcul de l'algorithme est raisonnable : O(n2), où n est le nombre d'attributs.
- Les interrelations secondaires entre les groupes d'attributs groupés sont identifiables.

L'algorithme d'énergie de liaison(BEA) prend en entrée la matrice de dépendance DM , permute ses lignes et ses colonnes et génère une matrice groupée (CM). Nous définissons une mesure globale (GM) de la matrice de dépendance :

$$GM = \sum_{i=1}^{n} \sum_{j=1}^{n} DM_{ij}(DM_{i,j-1} + DM_{i,j+1})$$

La permutation est faite de manière à maximiser cette mesure.
- La génération de la matrice groupée (CM) se fait en trois étapes [30] **:**

- Initialisation : Placez et fixez arbitrairement une des colonnes de DM dans CM. La colonne1 a été choisie dans l'algorithme.
- Itération : Choisissez chacune des n-i colonnes restantes (où i est le nombre de colonnes déjà placées dans CM) et essayez de les placer dans les positions i+1 restantes dans la matrice CM. Choisissez l'emplacement qui

contribue le plus à la mesure d'affinité globale décrite ci-dessus. Continuez cette étape jusqu'à ce qu'il ne reste plus de colonnes à placer.

- Ordre des lignes : Une fois l'ordre des colonnes déterminé, le placement des lignes doit également être modifié afin que leurs positions relatives correspondent aux positions relatives des colonnes.

Algorithme de BEA(Bond Energy Algorithm) : Les détails de l'algorithme d'énergie de liaison(BEA)sont donnés dans l'algorithme suivante :

Input: AA: attribute affinity matrix
Output: CA: clustered affinity matrix
begin
  {initialize; remember that AA is an $n \times n$ matrix}
  $CA(\bullet,1) \leftarrow AA(\bullet,1)$;
  $CA(\bullet,2) \leftarrow AA(\bullet,2)$;
  $index \leftarrow 3$;
  while $index \leq n$ **do** {choose the "best" location for attribute AA_{index}}
    for *i from 1 to* $index-1$ *by 1* **do** calculate $cont(A_{i-1}, A_{index}, A_i)$;
    calculate $cont(A_{index-1}, A_{index}, A_{index+1})$; {boundary condition}
    $loc \leftarrow$ placement given by maximum *cont* value ;
    for *j from index to loc by* -1 **do**
      $CA(\bullet, j) \leftarrow CA(\bullet, j-1)$ {shuffle the two matrices}
    $CA(\bullet, loc) \leftarrow AA(\bullet, index)$;
    $index \leftarrow index + 1$
  order the rows according to the relative ordering of columns
end

FIGURE 3.6 – *Bond Energy Algorithm[30]*

Nous définissons la mesure globale (GM) de la matrice de dépendance[30] :

$$GM = \sum_{i=1}^{n} \sum_{j=1}^{n} DM_{ij}(DM_{i,j-1} + DM_{i,j+1})$$

On définis le lien (bond) entre deux attributs Ax et Ay comme[30] :

$$bond(Ai, Aj) = \sum_{z=0}^{n-1} DM(Az, Ai)DM(Az, Aj)$$

et on définis la contribution nette à la mesure d'affinité globale de placer l'attribut Ak entre Ai et Aj comme [30] :

$$cont(Ai, A_k, Aj) = 2bond(Ai, A_k) + 2bond(A_k, Aj) - 2bond(Ai, Aj)$$

Description de l'algorithme BEA :

- BEA a une matrice AA comme entrée et une matrice clusterisé CM comme sortie
- CM prend les deux premier colonne du AA .
- Pour le reste des colonnes, nous calculons la contribution de chaque attribut dans l'ordre approprié, et classons l'attribut dans l'ordre qui a la plus grande contribution.
- S'il n'y a plus de colonne, on ordone les lignes selon l'ordre des colonnes.

Dans notre travail, on prend la matrice de dépendance (DM) donnée à la figure 3.5 comme entrée et génère une matrice de dépendance groupée (CM). Dans CM les éléments avec des valeurs similaires sont regroupés (c'est-à-dire les grandes valeurs avec d'autres grandes valeurs et les petites valeurs avec d'autres petites valeurs : au début, la matrice CM contient deux colonnes A0 et A1 :

	A_0	A_1	A_2	A_3	A_4
A_0	2	1	2	0	0
A_1	1	3	1	2	1
A_2	2	1	2	0	0
A_3	0	2	0	2	1
A_4	0	1	0	1	2

Matrice DM

	A_0	A_1
A_0	2	1
A_1	1	3
A_2	2	1
A_3	0	2
A_4	0	1

Matrice CM

FIGURE 3.7 – *Etape01 :Placement de 02 colone dans CM*

maintenant on étude le placement de colonne A2 il ya trois ordres possibles :
Ordre1 (-1,2,0) : A2 ordone avant A0.
Ordre2 (0,2,1) : A2 ordone entre A0 et A1.
Ordre3 (1,2,3) : A2 ordone entre A1 et A3(A2reste à ça place)
On calcule la contibution de chaque ordre :

Ordre1 (-1,2,0) : cont(A_{-1},A_2,A_0)=2bond(A_{-1},A2) + 2bond(A_2,A_0) – 2bond(A_{-1},A_0)

bond(A_{-1},A_2)= $\sum_{z=0}^{n-1} DM(Az, A_{-1})DM(Az, A2)$=($A_0$,$A_{-1}$)*($A_0$,$A_2$)+($A_1$,$A_{-1}$)*($A_1$,$A_2$)+ ($A_2$, A_{-1})*(A_2, A2) + (A_3, A_{-1})*(A_3, A2) + (A_4, A_{-1})*(A_4, A_2) + 0*2+0*1+0*2+0*0+0*0= 0

Bond (A2, A0) = $\sum_{z=0}^{n-1} DM(Az, A_2)DM(Az, A0)$ = 2*2+1*1+2*2+0*0+0*0= 9

bond (A-1, A0)= $\sum_{z=1}^{n} DM(Az, A_{-1})DM(Az, A2)$ = 0*2+0*1+0*2+0*0+0*0 = 0

alors : cont (A_{-1}, A_2, A_0)= 2*0 + 2*9 + 2*0 = 18.

Ordre2 (0,2,1) : cont(A_0,A_2,A_1) = 2bond(A_0,A_2) + 2bond(A_2,A_1) – 2bond(A_0,A_1).

Bond (A_0, A_2) = 9 (déjà calculé)

Bond (A_2, A_1) = 2*1+1*3+2*1+0*2+0*1= 7

Bond (A_0, A_1) = 2*1+1*3+2*1+0*2+0*1 = 7

Alors: cont (A_0, A_2, A_1) = 2*9 + 2*7 – 2*7 = 18

Ordre3 (1,2,3) : cont(A_1,A_2,A_3) = 2bond(A_1,A_2) + 2bond(A_2,A_3) – 2bond(A_1,A_3)

Bond (A_1, A_2) = 7 (déjà calculé)

Bond (A_2, A_3) = 2*0 + 1*2 + 2*0 + 0*2 +0*1 = 2

Bond (A_1, A_3) = 1*0 + 3*2 + 1*0 + 2*2 + 1*1 = 11

Alors: cont (A_1, A_2, A_3) = 2*7 + 2*2 – 2*11 = -4

après le calcule des contributions de chaque ordre on trouve deux grandes contributions (18) alors on choisi un ordre depuit les deux ordres 1 ou 2 , que soit l'ordre1(-1,2,0) . Alors la colonne A2 est placée à la premier du matrice CM avant la colonne A0 comme suit :

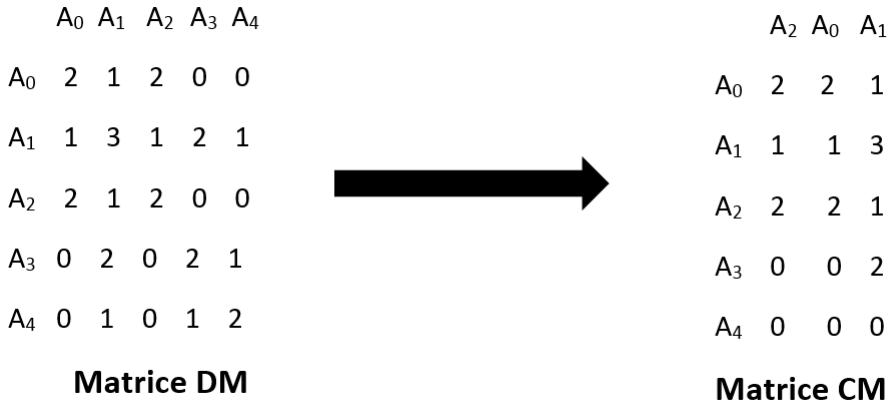

FIGURE 3.8 – *Etape02 :placement de colonne A2 dans la matrice CM*

On applique le même processus pour toutes les colonnes restantes. Finalement, on obtient la matrice CM suivante :

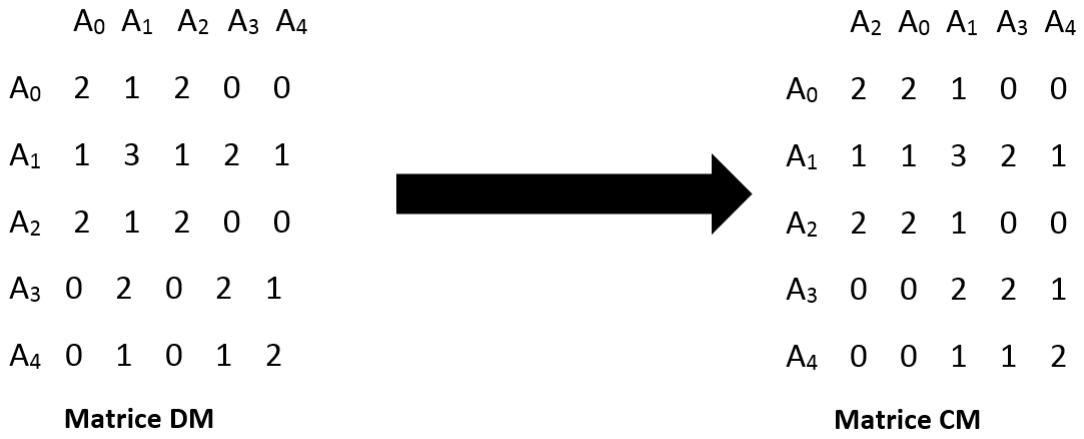

FIGURE 3.9 – *Resultat de BEA : matrice CM finale*

Dans L'étape suivante on ordonne les lignes selon l'ordre des colonnes, où nous obtenons la matrice CM clustérisé comme montre la figure 3.10

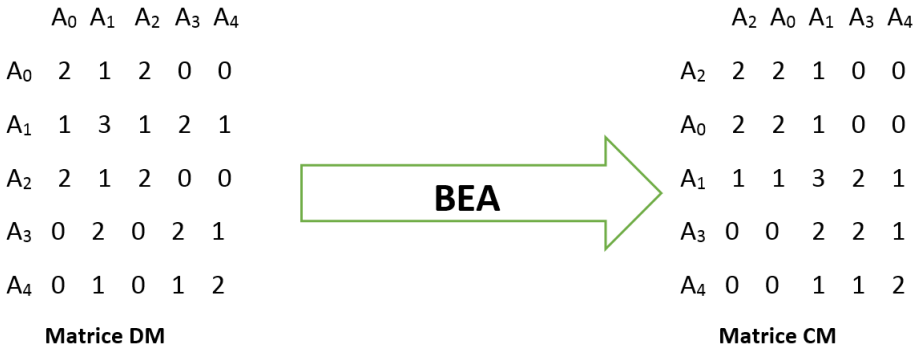

Figure 3.10 – *Matrice CM aprés l'application de l'algorithme BEA*

Dans cette phase , nous avons construit la matrice clustérisé CM à partir de la création des ensembles des données pour le workflow , puis nous avons créé la matrice de dépendance DM sur laquelle nous avons appliqué l'algorithme de regrouppement BEA, jusqu'à obtenir la matrice clustérisé CM que nous la utiliserons dans l'étape suivante.

3.3.4. Phase04 : partitionnement de CM et distribution des ensembles des CMt .

Au cours de cette partie, deux opérations importantes seront effectuées. Ces dernières sont le partitionnement et la distribution des datasets.
L'ensemble des machines virtuelles est noté VM dans lequel chaque VMj possède une capacité de stockage notée CSj .Un algorithme de partitionnement binaire 3.11sera applique sur la matrice CM dans le but d'obtenir le meilleur partitionnement binaire possible. Dans un premier temps, on partitionne CM en deux parties (d1,d2... dp) et (dp+1,dp+2... dn) , ce qui maximise la mesure suivante :

$$PM = \sum_{i=1}^{p}\sum_{j=1}^{p} CM_{ij} * \sum_{i=p+1}^{n}\sum_{j=p+1}^{n} CM_{ij} - (\sum_{i=1}^{p}\sum_{j=p+1}^{n} CM_{ij})^2$$

Ce mesure, PM, signifie que les ensembles de données de chaque partition ont des dépendances plus élevées les unes avec les autres et des dépendances plus faibles avec les ensembles de données des autres partitions. Sur la base de cette mesure, nous pouvons simplement calculer tous les PM pour p=1,2... n-1, et choisissez p tel qu'il ait la valeur PM maximale comme point de partition. Les détails de l'algorithme de partionnement binaire sont donnés dans la figure suivante :

Algorithme de partitionnement binaire :

Algorithme 1 : ALgorithme de paritionnement binaire

```
1 Input ; CM ; Matrice de dépendance clusterisee;
2 Output ; CM_T et CM_B ; Deux Matrice clusterisee representant CM ;
3 Description :;
4 for p = 1 ;p ≤ n − 1 ;p + + do
5     Calculer PM;
6     while PM_s obtenues do
7         Choisir p sa valeur PM = Max;
8         Prendre p point de coupure et Partitionner CM en CM_T et CM_B;
9         retour CMP ;
```

Figure 3.11 – *Algorithme de partitionnement binaire*

CM : matrice clustérisé.
CMt,CMb : deux sous matrices de la matrice CM .
P : le point de coupure.
n : la taille de la matrice entrée CM.
PM : mesure globale.
CMP : la variable qui contient les partitions trouvées après la fin de l'étape de partitionnement.

Après une partition, le CM forme deux nouvelles matrices clustérisées, nous notons celle du haut CMT , qui contient les dépendances des jeux de données DT= (d1,d2... dp) et celle du bas CMB, qui contient les dépendances des jeux de données DB =(dp+1,dp+2... dn). Chaque matrice groupée représente une partition d'ensembles de données et nous notons la taille totale des ensembles de données qu'elle contient comme : $dsT = \sum_{i=1}^{p} si$ et $dsB = \sum_{i=p+1}^{n} si$.
exemple de partitionnement : Nous prenons la matrice précédente CM de taille n=5 comme exemple pour illustrer le processus de partitionnement binaire : La matrice CM à prend comme entrée, et cette algorithme génère deux nouvelle matrices CMt et CMb , et on répète l'opération avec la matrice CMb jusq'au p<=n .

	A_0	A_1	A_2	A_3	A_4
A_0	2	2	1	0	0
A_1	2	2	1	0	0
A_2	1	1	3	2	1
A_3	0	0	2	2	1
A_4	0	0	1	1	2

Matrice CM

FIGURE 3.12 – *Matrice CM*

p=1 : on calcule le mesure PM :

$$\mathbf{PM}=\sum_{i=1}^{p}\ \sum_{j=1}^{p}\ \mathbf{AA_{ij}} * \sum_{i=p+1}^{n}\ \sum_{j=p+1}^{n}\ \mathbf{AA_{ij}} - \left(\sum_{i=1}^{p}\ \sum_{j=p+1}^{n}\ \mathbf{AA_{ij}}\right)^2$$

$$PM_1 = \sum_{i=1}^{1}\ \sum_{j=1}^{1}\ \mathbf{AA_{ij}} * \sum_{i=1+1}^{5}\ \sum_{j=1+1}^{5}\ \mathbf{AA_{ij}} - \left(\sum_{i=1}^{1}\ \sum_{j=1+1}^{5}\ \mathbf{AA_{ij}}\right) \mathbf{2}$$

$$= \mathbf{AA_{11}} * \sum_{i=2}^{5}\ \sum_{j=2}^{5}\ \mathbf{AA_{ij}} - \left(\sum_{j=2}^{5}\ \mathbf{AA_{1j}}\right) \mathbf{2}$$

$$= 2 * (AA_{22}+AA_{23}+AA_{24}+AA_{25} +AA_{32}+AA_{33}+AA_{34}+AA_{35}+AA_{42}+AA_{43}+AA_{44}+AA_{45} + AA_{52}+AA_{53}+AA_{54}+AA_{55}) - (AA_{12}+AA_{13}+AA_{14}+AA_{15})^2$$

$$= 29$$

On calcule le reste de même façon : p=2 : PM2 = 116 . p=3 : PM3 = 81 . p=4 : PM4 = 38 On a le max PM = 116, pour p=2 , alors on prend p=2 comme point de coupure . c'est t-à dire on génère deux matrice CMt de taille n=2 et matrice CMb de taille n=3 comme suit :

On répète l'opération avec la matrice CMb de taille n=n-p jusq'à la fin du partitionnement. À la fin de partitionnement, on obtient les matrices CMts suivantes :

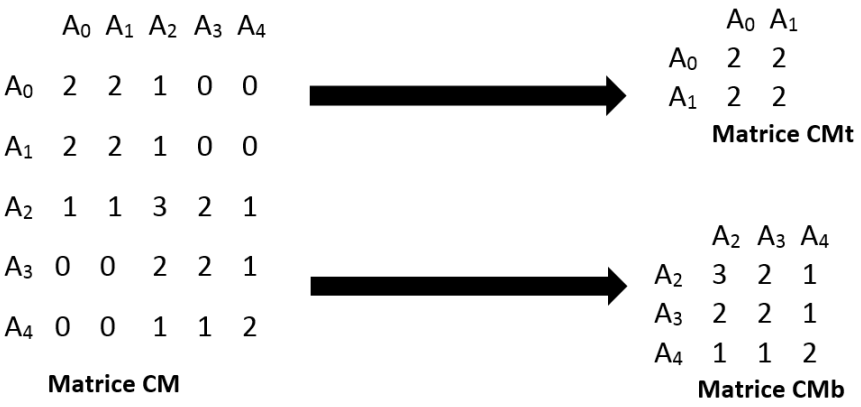

FIGURE 3.13 – *Matrice CMt et Cmb*

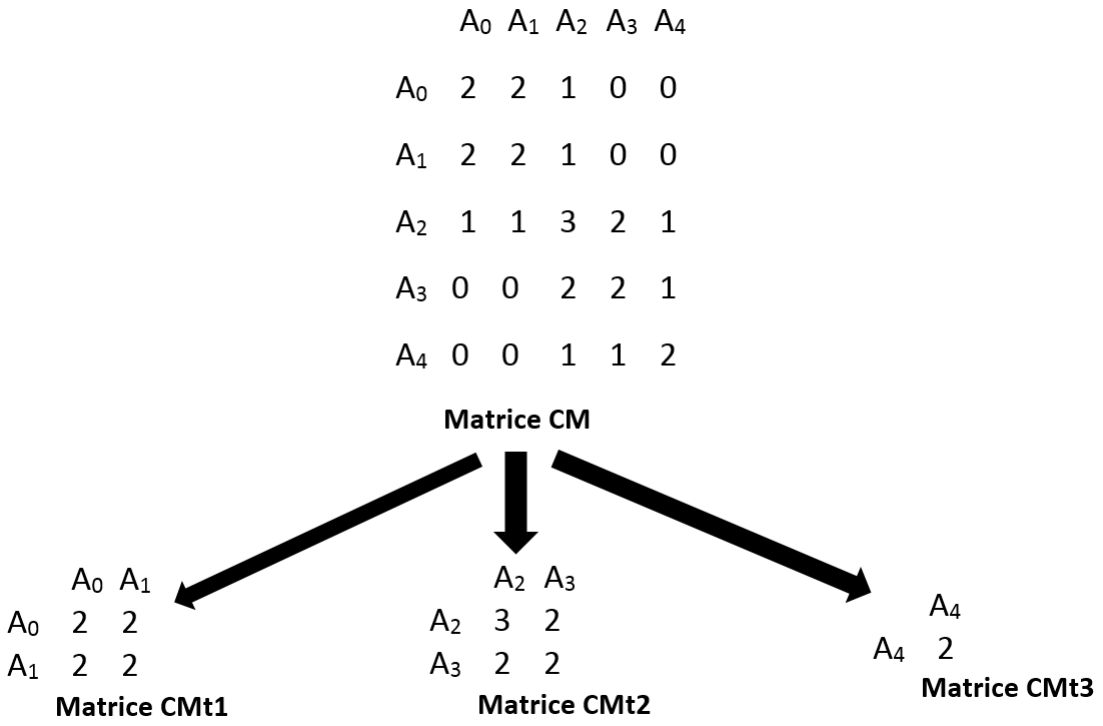

FIGURE 3.14 – *Ensembles de matrice CMt*

Distribution des ensembles de CMts : Dans cette étape, nous allons distribuer les ensembles de données CMt aux machines virtuelles, afin que chaque groupe

soit affecté à une machine virtuelle, en tenant compte de sa capacité de stockage. Après cela, on détermine la place d'exécution de chaque tâche, en tenant compte du nombre et du volume de données utilisées par chaque tâche, qui sont stockées dans les machines virtuelle. Où la tâche est exécutée dans la machine virtuelle qui possède le plus grand nombre de données parmi les données utilisées par cette tâche.

3.3.5. Phase05 : Exécution des tâches :

Après la distribution des datasets CMt aux virtuelles machines,nous allons exécuter les étapes décrites ci-dessous par un organigramme :

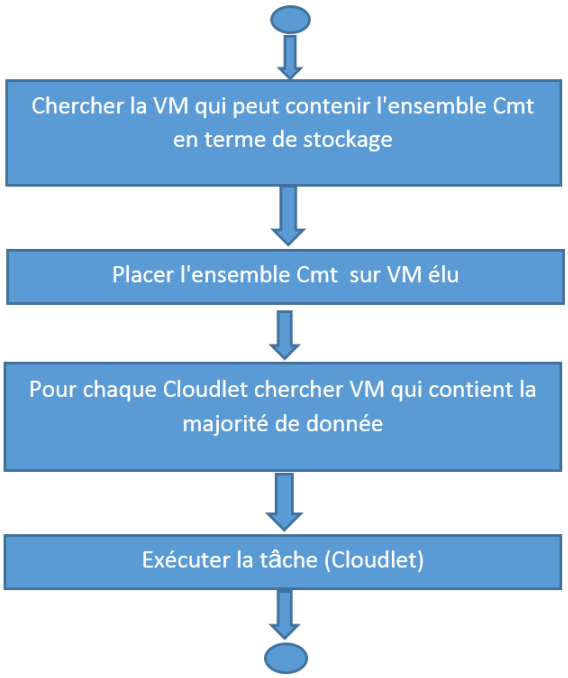

Figure 3.15 – *Etapes de L'exécution des tâches*

3.4. Conclusion

Au cours de ce chapitre, nous avons décrit les méthodes qu'on s'est basé dans notre stratégie , un ensemble de pseudocodes ont été montré pour bien illustré notre approche, Dans le dernier chapitre, on va présenter les résultats obtenus de plusieurs expérimentations et simulations.

Chapitre 4

Implémentation et Résultats

Sommaire

4.1. Introduction

Afin de concrétiser, de valider et évaluer notre approche d'ordonnancement, nous avons conçu un simulateur reflétant le fonctionnement de notre proposition, nous avons exécuté une série d'expérimentations dont les résultats et les interprétations font l'objet du présent chapitre.
Nous commencerons d'abord par décrire l'environnement dans lequel nous avons réalisé notre simulateur puis nous discuterons et analyserons les résultats que nous avons obtenu.

4.2. Langage de programmation Java

Le langage Java est un langage de programmation informatique orienté objet. Java a la particularit e principale d'être portable, c'est à dire, que les logiciels écrits avec ce dernier sont très facilement reutilisable sur plusieurs systemes d'exploitation tels que UNIX, Microsoft Windows, Mac OS ou Linux avec peu ou pas de modifications. C'est la plate-forme qui garantit la portabilité des applications développées en Java. Le langage reprend en grande partie la syntaxe du langage C++, trés utilisé par les informaticiens. Néanmoins, Java a été épurée des concepts du C++ et à la fois les plus déroutants, tels que l'héritage multiple remplacé par l'implémentation des interfaces. Les concepteurs ont privilégié l'approche orientée objet de sorte qu'én Java, tout est objet à l'exception des types primitifs (nombres entiers, nombres à virgule flottante).[31]

4.2.1. Avantages du java :

L'un des avantages évidents de ce langage est une bibliothèque d'exécution qui se veut indépendante de la plateforme : en théorie, il vous est possible d'utiliser le même code pour Windows 95/98/NT, Solaris UNIX, Macintosh, etc. Cette propriété est indispensable pour une programmation sur Internet, cependant, par rapport à la disponibilité sur Windows et Solaris, les implémentations sur d'autres plates-formes ont toujours un léger décalage. Un autre avantage de ce langage de programmation réside dans le fait que la syntaxe de Java est analogue à celle de C++ ce qui le rend économique et professionnel. Le fait de créer une autre version d'un langage C++ n'est cependant pas suffisant. Le point cl è est le suivant : il est beaucoup plus facile d'obtenir du code sans erreur à l'aide de Java qu'avec C++. Pourquoi ? Les concepteurs de Java ont beaucoup réfléchi à la raison pour laquelle le code C++ contenait autant d'erreurs. Cette réflexion les a amen è à ajouter dans Java des fonctions destinées à éliminer la possibilité de créer du code contenant les types d'erreurs les plus courants (selon certaines estimations, le code C++ contient au moins une erreur toutes les cinquante lignes)[31]

4.3. Environnement de devloppement

4.3.1. Environnements matériel :

L'approches proposée dans ce travail ont été implémentées et testées dans un environnement possédant les caractéristiques suivantes : Un processeur Intel(R) Core(TM) i3-4005 UCPU@1.7GHz, doté d'une capacité de mémoire de 6GB

4.3.2. Environnements logiciel :

Eclipse est un environnement de développement intégré (Integrated Development Environment) dont le but est de fournir une plate-forme modulaire pour permettre de réaliser des développements informatiques. Eclipse utilise énormément le concept de modules nommés "Plug-Ins" dans son architecture. D'ailleurs, hormis le noyau de la plate-forme nommé "Runtime", tout le reste de la plate-forme est développé sous la forme de Plug-Ins. Ce concept permet de fournir un mécanisme pour l'extension de la plate-forme et ainsi fournir La possibilité à des tiers de développer des fonctionnalités qui ne sont pas fournies en standard par Eclipse.[32]

4.4. Simulateur CloudSim

4.4.1. Dèfinition :

CloudSim est une nouvelle structure de simulation généralisée et extensible qui permet la modélisation des environnements hétèros, la simulation et l'expérimentation de Cloud émergent des infrastructures de calcul et des services d'application. CloudSim couvre la plupart des fonctionnalités ayant lieu dans un centre de traitement des données en détail qui contienne la simulation de la définition de matériel de centre de traitement des données (Datacenter) en termes de machines physiques composées de processeurs, de dispositifs de stockage, de mémoire et de largeur de bande interne, ainsi que la création et la destruction des machine virtuelle et le plus intéressant la simulation de l'exécution des programme de l'utilisateur sous forme de Cloudlet est cela en définissons leur charge et leur nombre.[33]

4.4.2. Caractéristiques du CloudSim :

CloudSim appuie la recherche et le développement dans le domaine émergent du Cloud Computing et les fonctionnalitès suivantes :

- Support pour la modélisation et la simulation à grande échelle d'infrastructure de Cloud Computing, y compris des centres de données sur un seul nœud physique.
- Une plateforme indépendante pour la modélisation des Datacenters, des Brokers, de l'ordonnancement et des politiques d'allocation des ressources.

Parmi les principales caractéristiques de CloudSim, nous pouvons citer :

- La disponibilité de moteur de virtualisation, ce qui facilite la création et la gestion de services virtualisés multiples, indépendants et hébergés sur un nœud du Datacenter.
- La flexibilité pour commuter entre l'allocation en espace partagé et en temps partagé des cœurs de traitement aux services virtualités.[33]

4.5. Implementation :

Dans cette partie, nous allons nous intéresser à la démonstration de notre application à travers un exemple en faisant référence à quelques interfaces graphiques. le diagramme de cas d'utilisation suivant décrit la méthode d'utilisation de notre application.

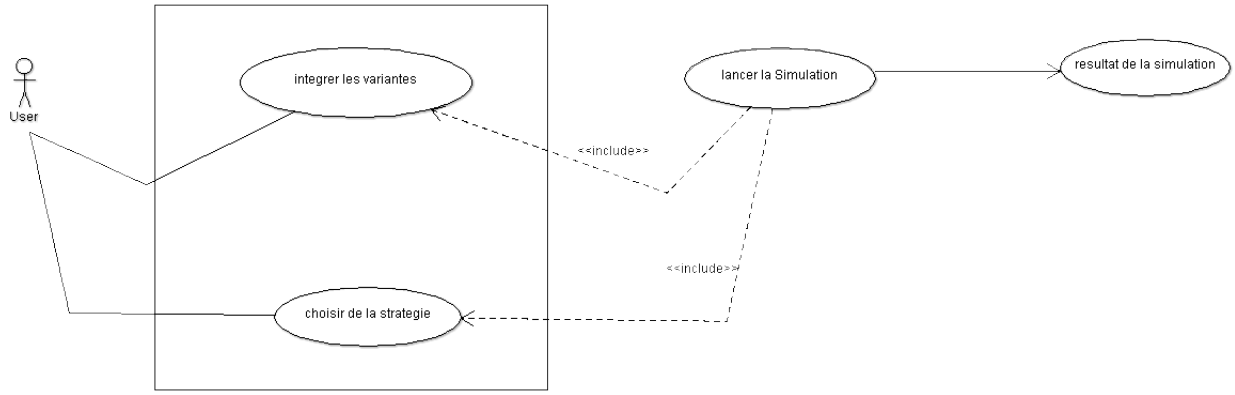

Figure 4.1 – *Diagramme de cas d'utilisation*

4.5.1. Interface principale

La version de CloudSim n'a pas d'interface graphique, son exécution se fait sur la console donc nous avons créé une interface qui facilite l'accès au simulateur. L'interface doit faire appel à CloudSim ainsi qu'aux différentes approches qui se trouvent dans différents packages.
La Figure 4.2 représente la première interface de notre simulateur qui apparaît à l'utilisateur.

Figure 4.2 – *Interface principale*

4.5.2. Configuration de simulation

Le premier pas de la simulation est la configuration des composants du Cloud. Cette configuration est réalisée en quatre étapes :

4.5.2.1. Configuration des Data Centers

Cette étape consiste à faire entrer les différents caractéristiques du Data Center comme : le coût de traitement, le coût de la mémoire, le coût de stockage et le coût de la bande passante .

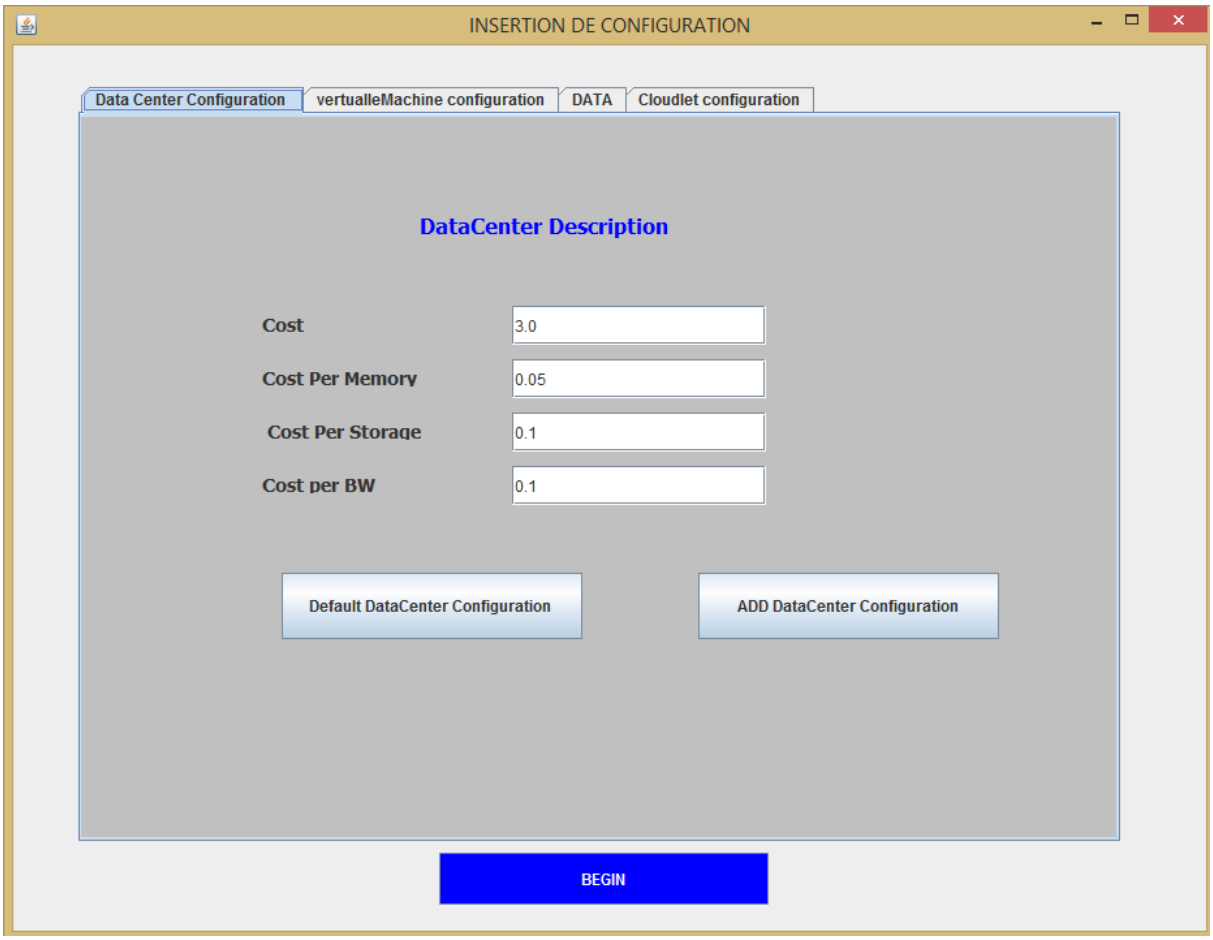

Figure 4.3 – *Configuration du Data Center*

4.5.2.2. Configuration des machines virtuelles

La création des machines virtuelles(VM) hétérogènes se fait en cliquant sur le bouton Add VM pour des parametres specifique et Bouton default parametre pour des parametre aleratoire . Pour cela, il faut préciser le nombre de processeur, MIPS, RAM, capacité de stockage, la bande passante.

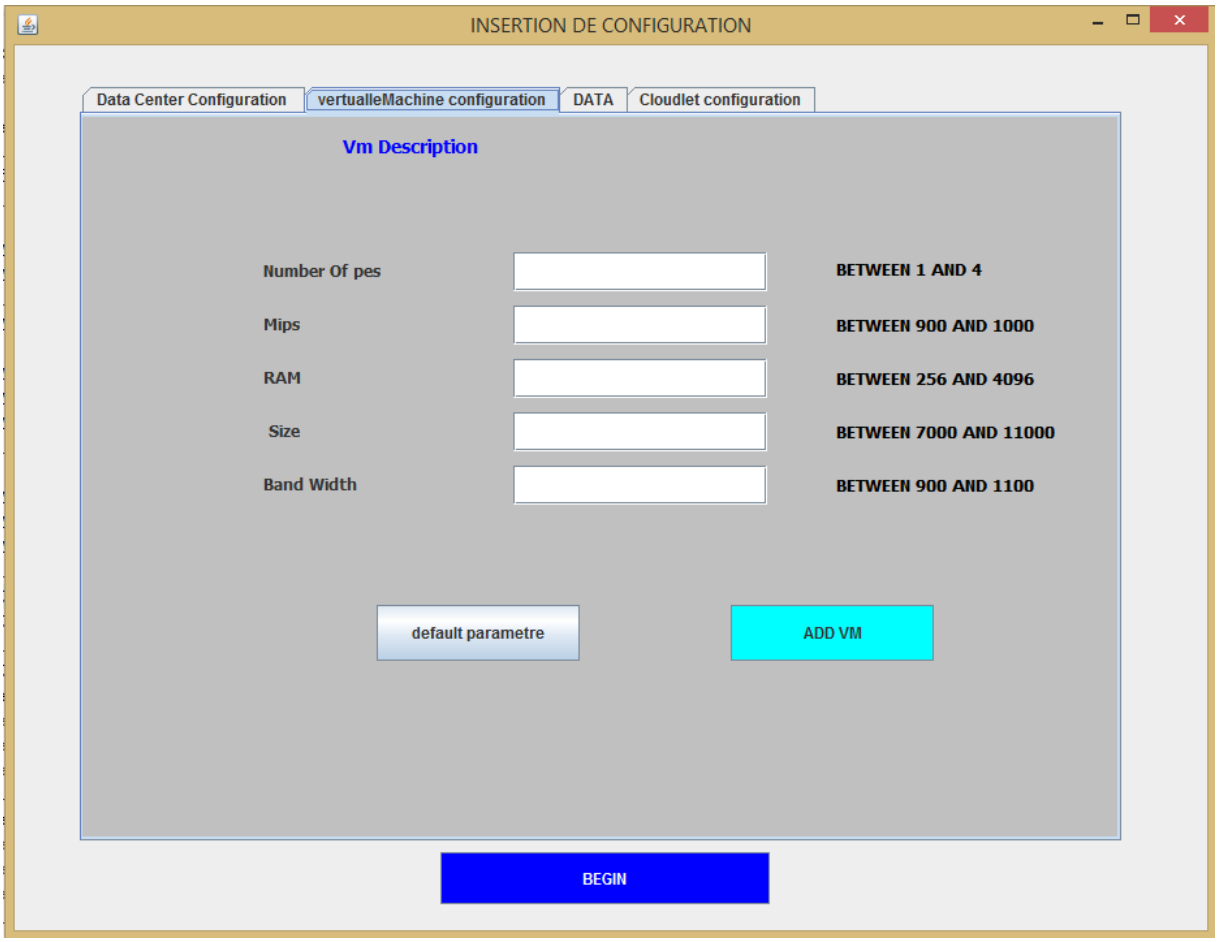

Figure 4.4 – *Configuration du machines virtuelles*

4.5.2.3. Préparation des données

La Figure présente l'étape de création des données. La création se fait par le click sur le boutton "Add Data" en précisant la taille des données, et Bouton default parametre pour les taille des données alératoire .

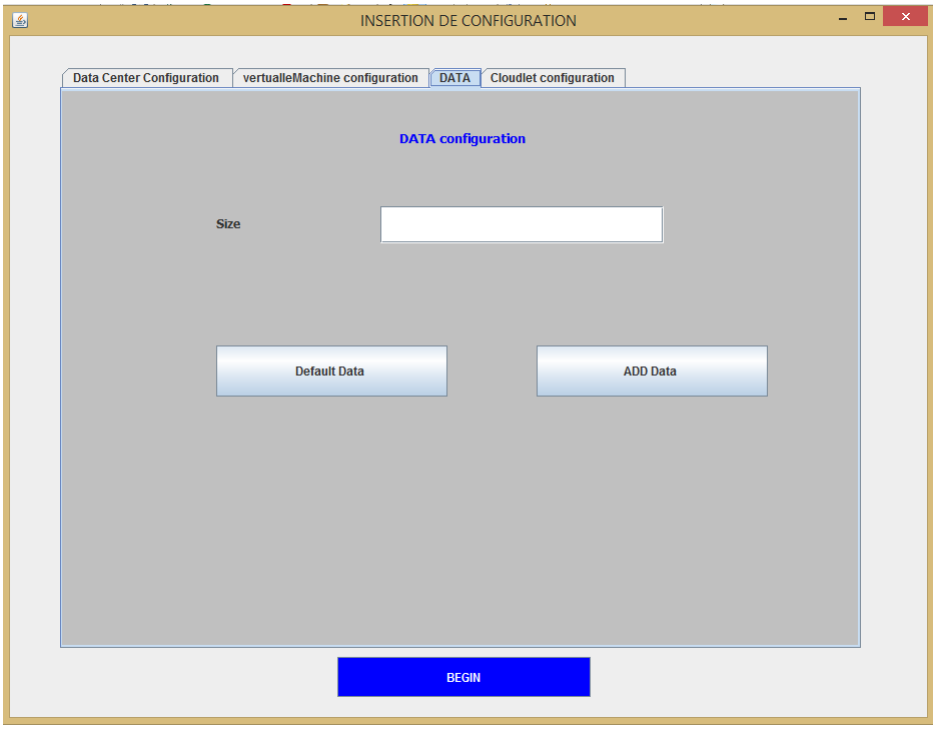

Figure 4.5 – *Préparation de Data*

4.5.2.4. Configuration des Cloudlets

L'onglet Cloudlet affiche l'interface pour configurer les Cloudlets : File Size ,Length,Output Size, La création se fait par le click sur le boutton Add Cloudlet en précisant les configuration des clouldlet, et Bouton default parametre pour configuration alératoire

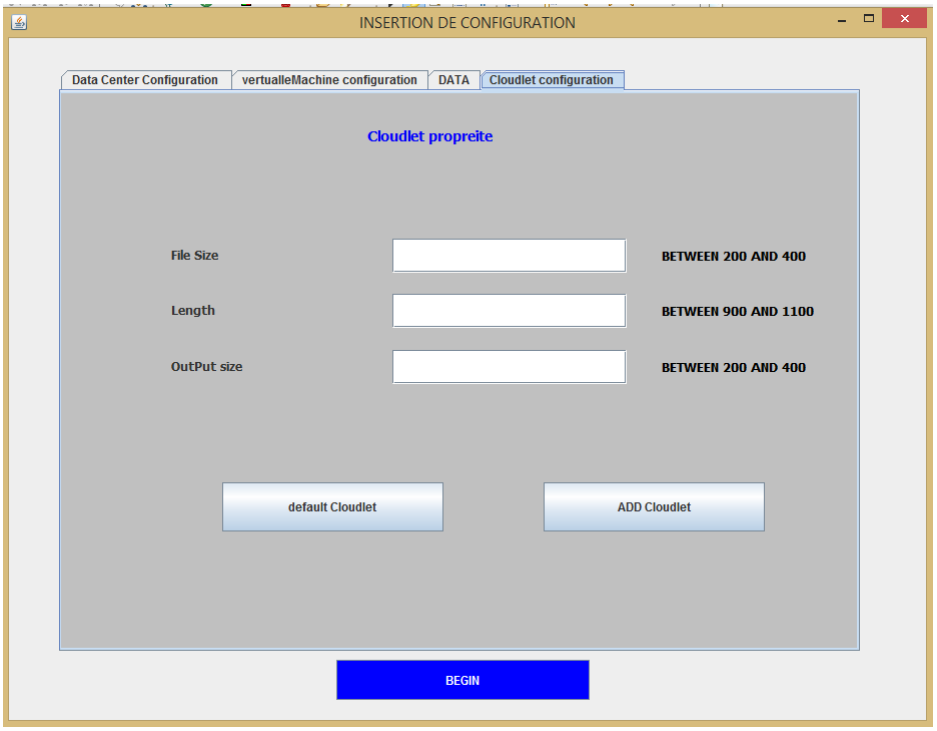

Figure 4.6 – *Configuration des Cloudlets*

4.5.2.5. Les Cloudlets , Les Donnes ET les VMS

L'onglet "second config " est l'interface pour configurer les Cloudlets : Nombre de Cloudlets,les VM : Nombre de vms et les données : nombre de données ,les centres des données : nombre de Data Center La Figure 4.7 représente l'étape de création des données,des cloudlets et virtuelle machines. La création se fait par le click sur le boutton "Insert Values" et entrer a l'étape de simulation .

Figure 4.7 – *Second configuration*

Pour lancer la simulation on clique sur "Start Simulation" pour voir les résultats de la simulation.

4.5.3. Simulation

Après avoir entré toutes les données et tous les paramètres, le processus de simulation a été lancé et les résultats suivants ont été obtenus.

4.5.3.1. Déploiement des listes des données pour chaque tâches

Dans cette fenêtre, l'ensemble des tâches et les données qu'elles contiennent s'affichent en appuyant sur le bouton "aficher des ensembles des donnes".

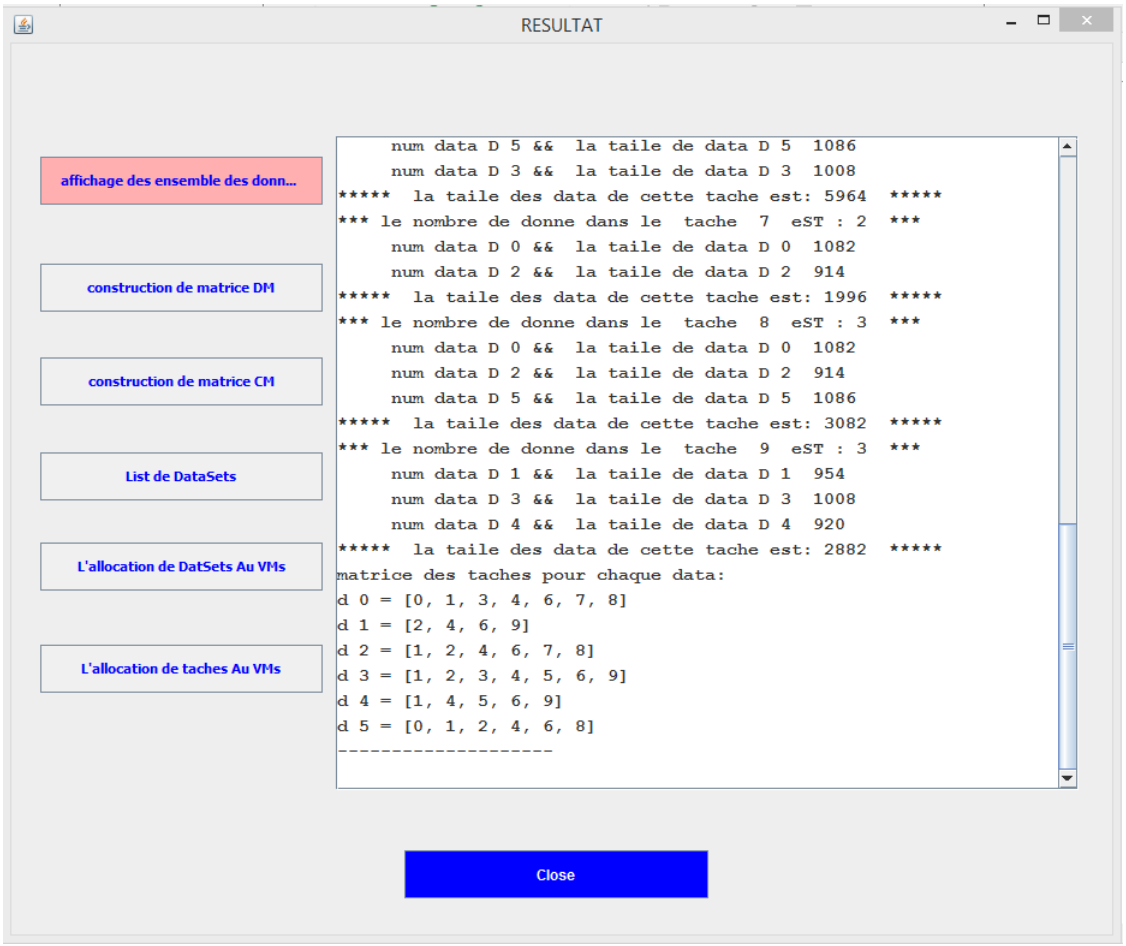

Figure 4.8 – *Affichage des ensembles des données*

4.5.3.2. Déploiement et clusterisation de la matrice de depéndance

A l'issue de l'etape de saisie, toutes les configurations concernant le Cloud et le workflow, sont effectuees. Ainsi, les calculs concernant la strategie peuvent commencer. Ces derniers commencent par l'elaboration de la matrice de dépendance. La Figure 4.9 suivante montre la création de la matrice de dépendance :

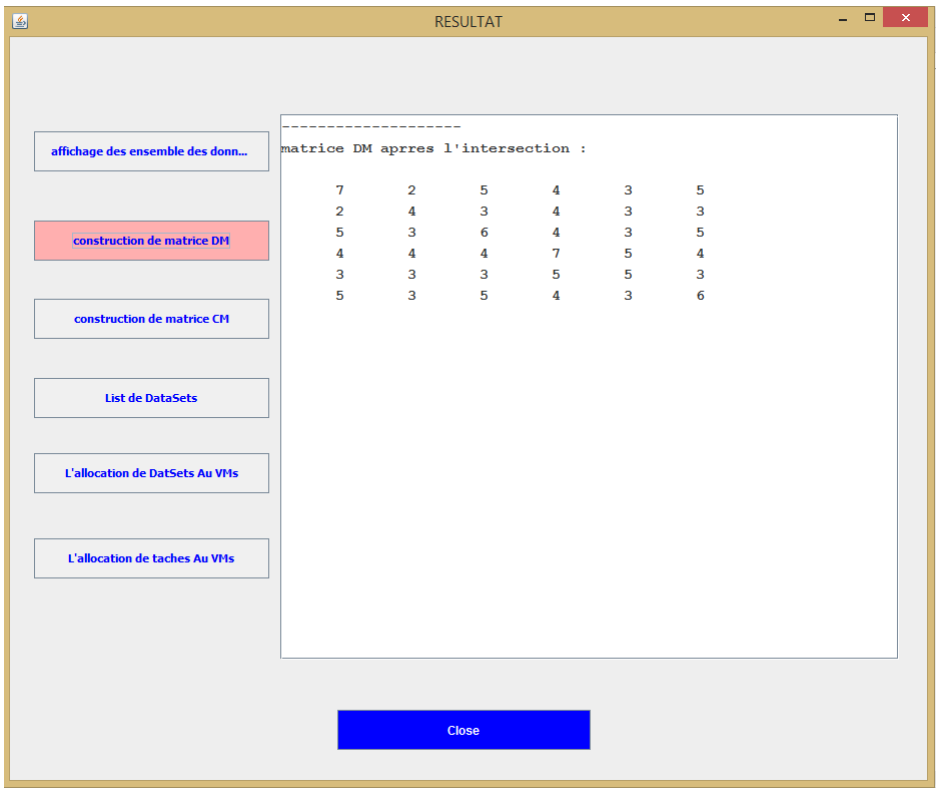

Figure 4.9 – *Déploiement de la matrice de dépendance DM*

4.5.3.3. Clusterisation de la matrice de CM

Après l'étape de construction de la matrice de DM, on passe à l'étape de clusterisation de la matrice CM avec l'algorithme BEA en appuyant sur le bouton "construction de la matrice CM". La Figure 4.10 montre le résultat obtenu :

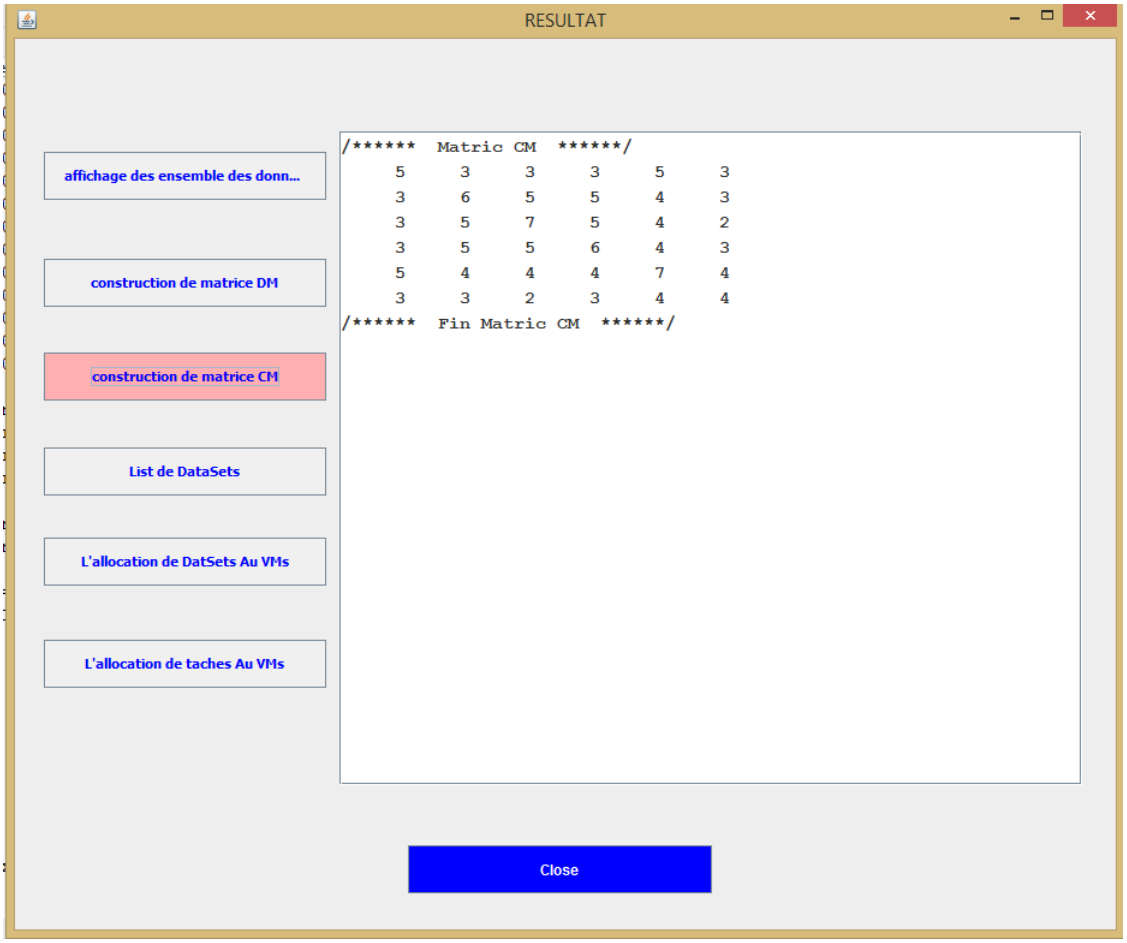

Figure 4.10 – *MATRICE CM*

4.5.3.4. Partitionnement et distribution des Cmt

Le quatrième phase de la stratégie est le partitionnement et la distribution des données sur les virtuelle machines. l'ensemble des Cmt s'affichent en appuyant sur le bouton "List de Datasets".

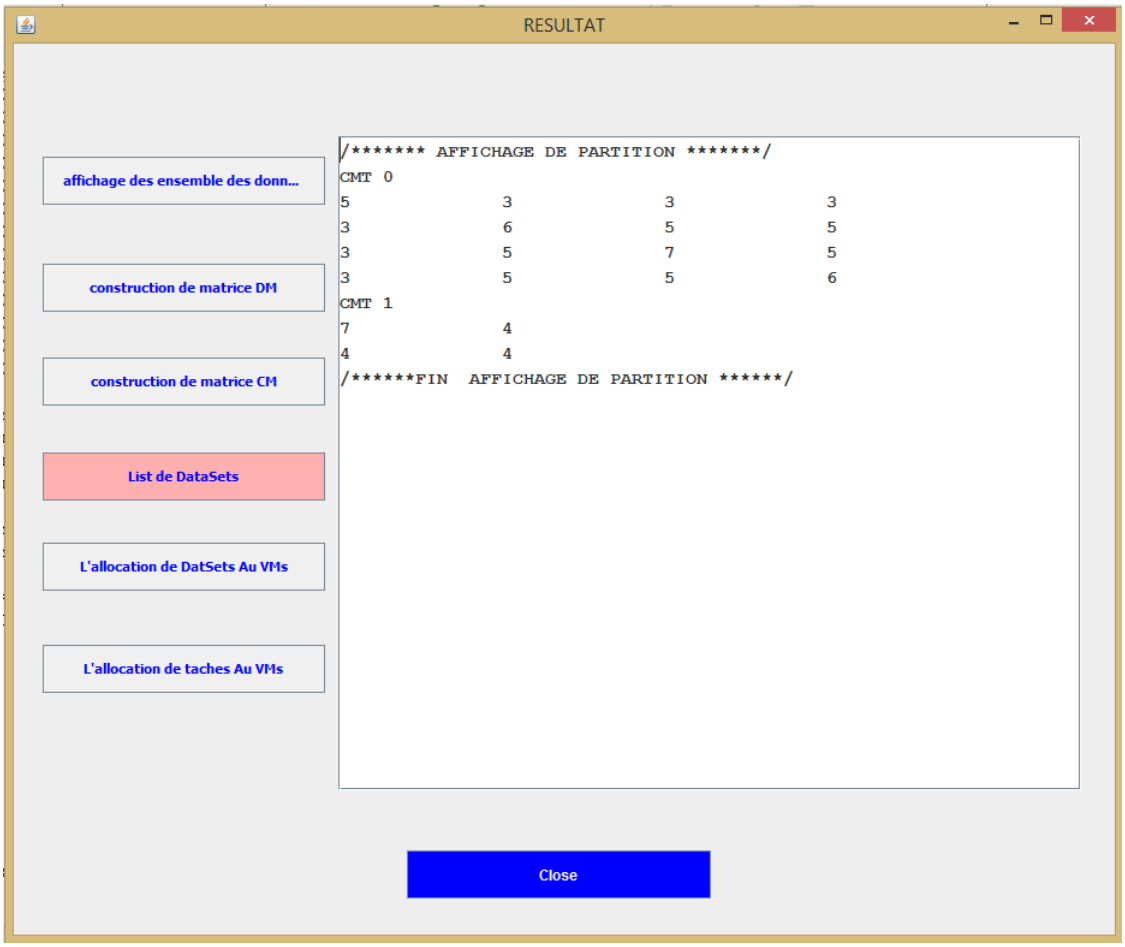

Figure 4.11 – *Partitionnement et distribution des CMts au VM*

4.5.3.5. L'allocation :

A cette phase , les données et les cloudlets sont affectées aux machines virtuelles selon les valeurs des matrices Cmt :

1. **Allocation des données Au VMs :** Dans cette étape les données de CMts sont affèctées aux différentes VMs en prenant de compte du stockage du chaque VM.

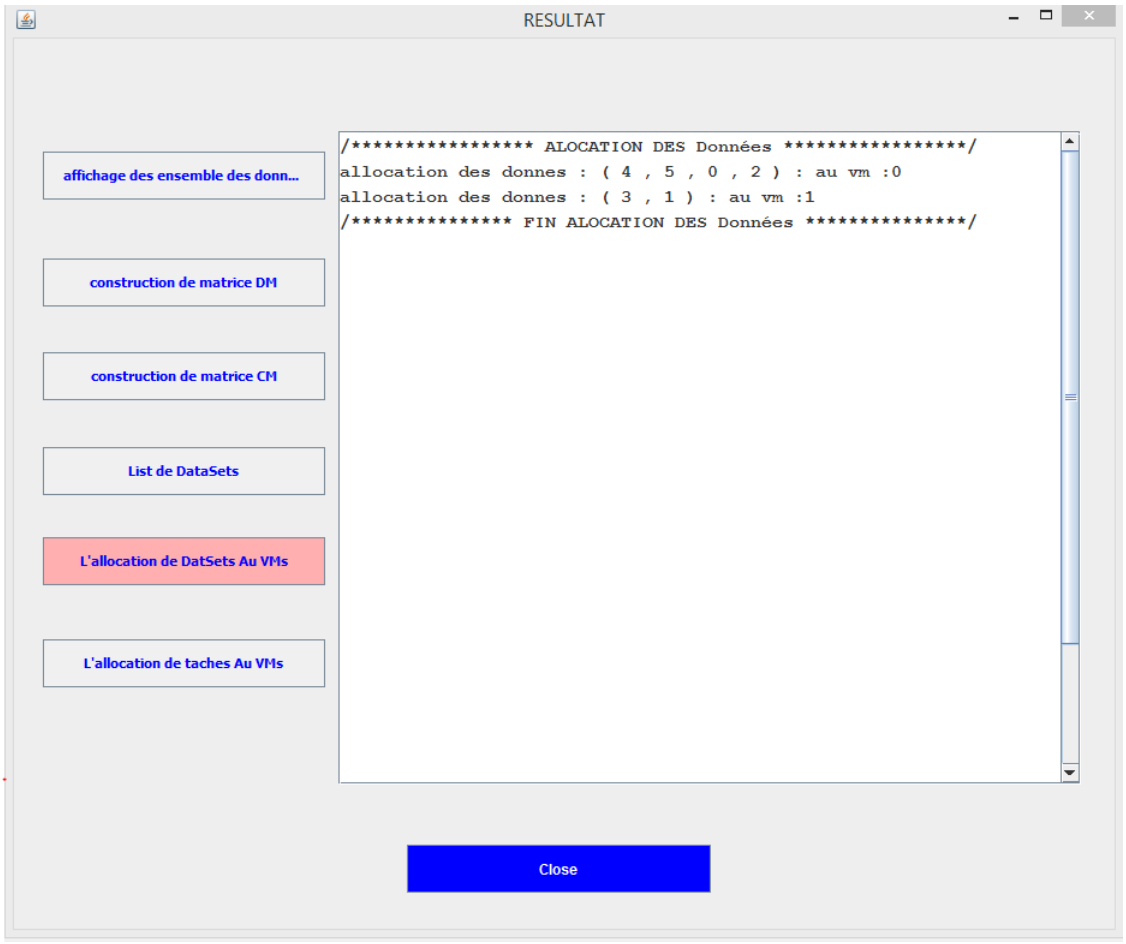

Figure 4.12 – *Allocation des données Au VMs*

2. **Allocation des cloudlet Au VMs :** Dans cette étapes chaque cloudlet est affecté au VM qui a la majorité de données.

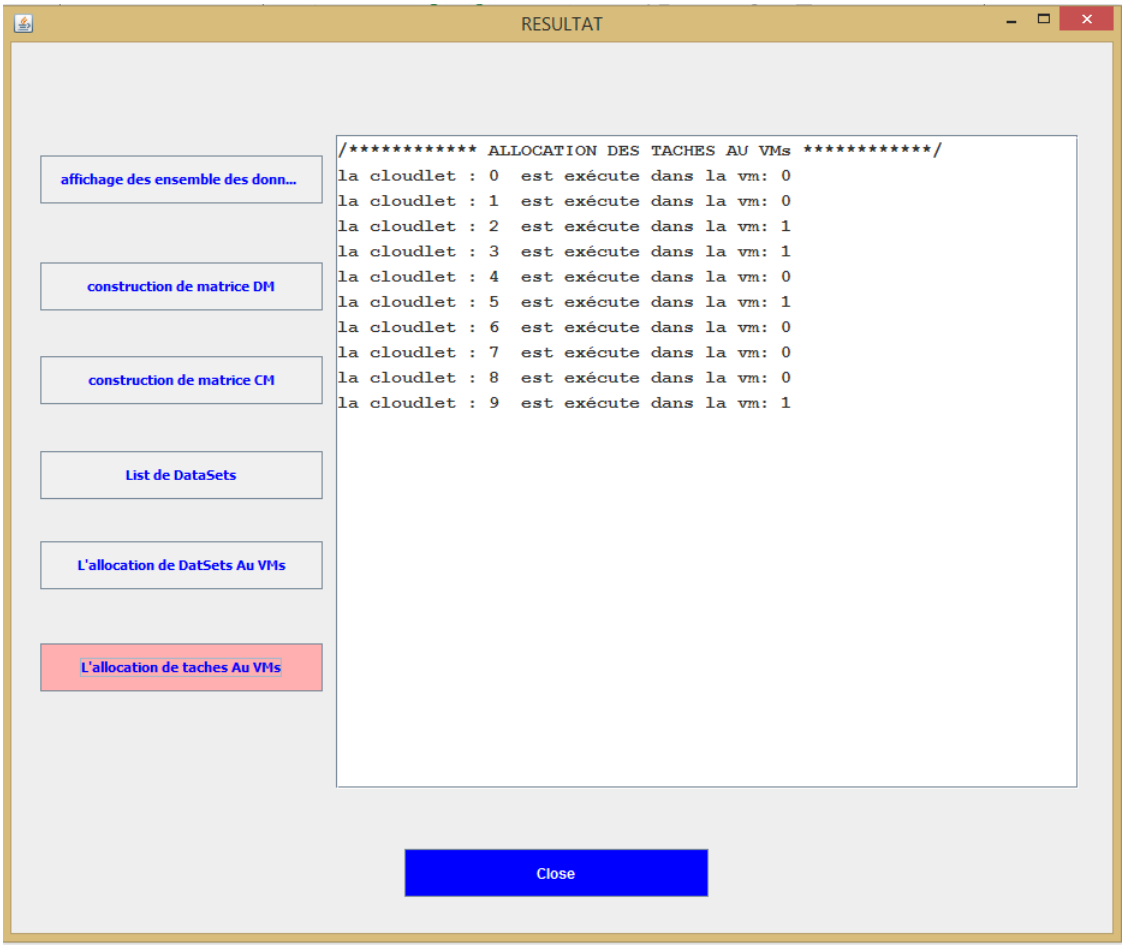

FIGURE 4.13 – *Allocation des données Au VMs*

4.5.4. RÉSULTATS EXPÉRIMENTAUX :

4.5.4.1. SIMULATION 1 :

Dans cette simulation, nous avons créé deux Data Center contenant 4 Host hétérogène, chaque Host posséde un seul processeur avec une vitesse variante en MIPS entre(20000, 200000), bande passante entre (10000,200000), la taille de la donnée est générée aléatoirement entre (900MB - 1100 MB) . Cette simulation consiste à varier le nombre de Cloudlet et voir l'impact sur le nombre des données transférés.

stratégies.

Cloudlet	Space Shared +Random Placement	Notre stratégie
10	40	14
20	128	63
40	251	134
80	499	251
200	1185	468
1000	6077	2379

TABLE 4.1 – *Impact du nombre du Cloudlets sur le nbr de données transférés*

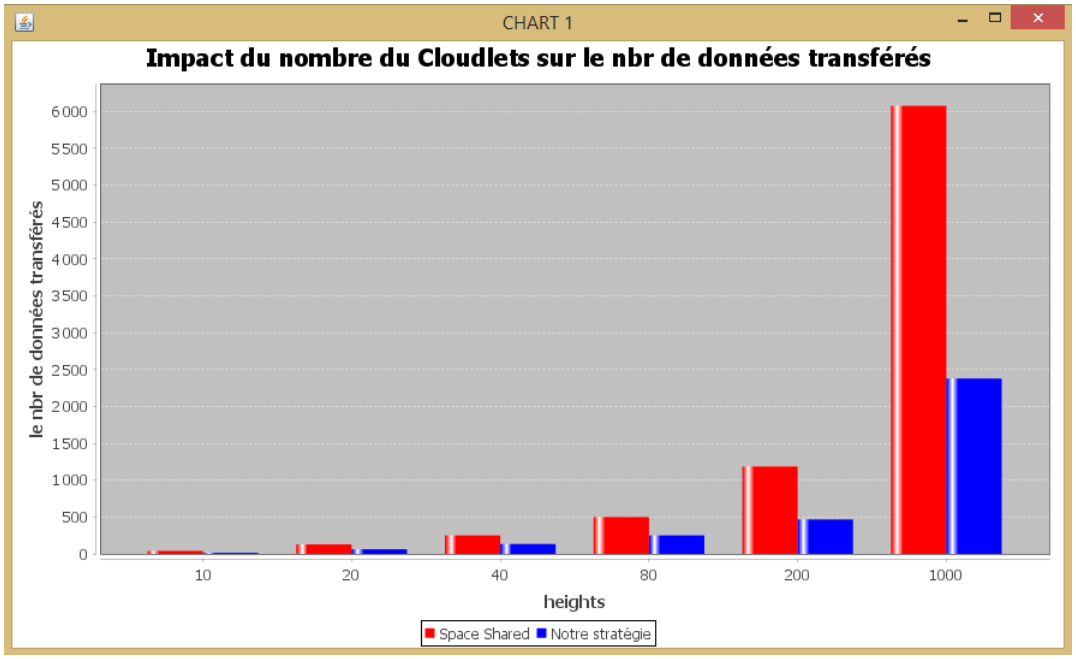

FIGURE 4.14 – *Impact du nombre du Cloudlets sur le nbr de données transférés*

Dans le tableau 4.1 consiste à faire une simulation en utilisant la variation du nombre de Cloudlets, nous remarquons une très grande différence dans le nombre des données transférés avec l'approche proposée par rapport aux autres

4.5.4.2. Simulation 2 :

Dans cette simulation, nous avons créé deux Data Center contenant 4 hôsts hétérogène, Chaque hôte posséde un seul processeur avec une vitesse variante en MIPS entre (20000,200000), bande passante entre(10000,20000),on fixes le nombre de cloudlets a 200 Coudlets. Cette simulation consiste à varier le nombre des données (10, 20, 30, 50, 80) ;

Data	Space Shared +Random Placement	Notre stratégie
10	181608	75060
20	364279	233521
30	428762	317326
50	851217	742656
80	1976867	1824719

Table 4.2 – *Impact du nombre du données sur la taille des données transférés*

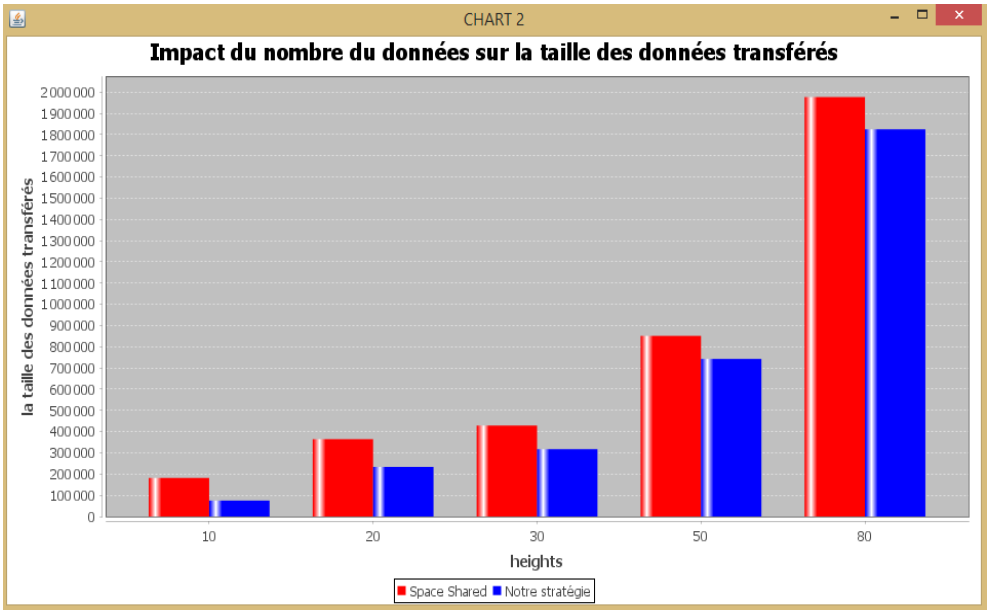

Figure 4.15 – *Impact du nombre du données sur la taille des données transférés*

Le tableau 4.2 est un tableau de comparaison qui se repose sur notre stratégie de Placement et la stratégie aléatoire pour une seule variante (la taille du données), ou nous pouvons déduire que notre stratégie réduit considérablement la taille des données transférées par rapport à l'approche Space Shared avec un placement aléatoire.

4.5.4.3. Simulation 3 :

Dans cette simulation, nous avons créé deux Data Center contenant 4 Host hétérogène, chaque Host posséde un seul processeur avec une vitesse variante en MIPS entre(20000, 200000), bande passante entre (10000,20000), le nombre de données est seulement 10 et le nmbre de Vm est 30 virtuelle machines . Cette simulation consiste à varier le nombre de Cloudlet :

Nombre des Cloudlets	Space Shared +Random Placement	Notre stratégie
10	54	39
20	127	78
40	246	137
80	460	230
200	1290	559
300	1553	675

Table 4.3 – *Impact du nombre de Cloudlets sur le temps de réponse (ms)*

Dans le tableau 4.3 consiste à faire une simulation en utilisant la variation du nombre de Cloudlets, nous remarquons une très grande différence dans le temps de réponse avec l'approche proposeé par rapport aux stratégié aléatoire.

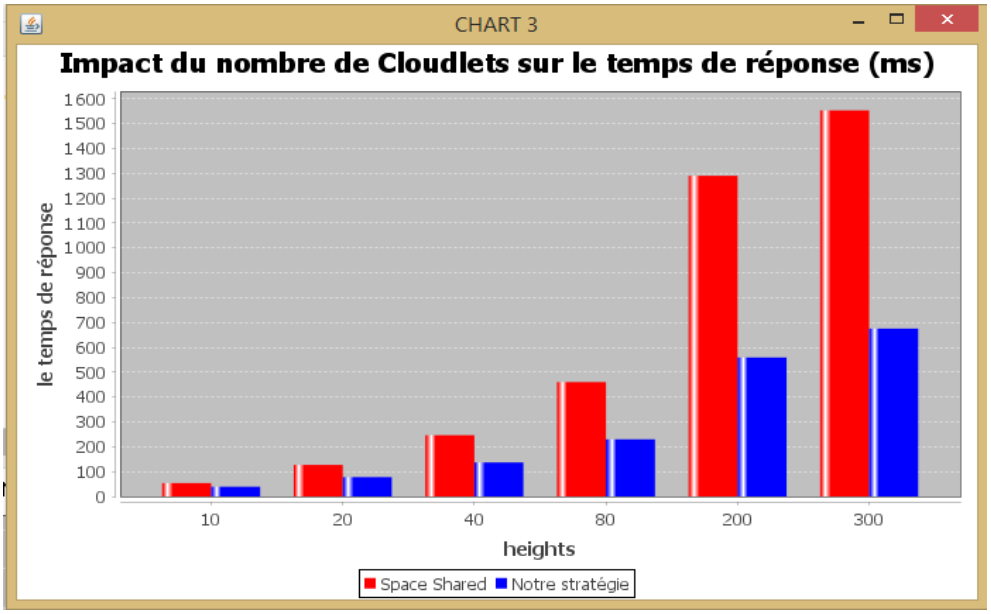

FIGURE 4.16 – *Impact du nombre de Cloudlets sur le temps de réponse (ms)*

4.6. CONCLUSION

Dans ce chapitre, nous avons présenté l'implémentation de notre application ainsi que les résultats obtenus. Aussi, nous avons réalisé plusieurs séries de simulations afin de comparer notre approche à une approche aléatoire en changeant divers paramètres tels que : nombre de données, nombre de Cloudlets. Les résultats de la comparaison ont montré que notre approche réduisait le trafic et donc réduisait le temps de réponse des Cloudlets.

Conclusion générale

A travers ce mémoire , nous avons étudié le problème d'ordonnancement de tâche d'un Workflow dans un environnement Cloud Computing. plusieurs stratégies d'ordonnancement ont été proposées dans la littérature : elles visent principalement de réduire le temps de réponse. En général, l'ordonnancement de tâches est le processus d'affectation des tâches aux ressources disponibles sur la base des caractéristiques et des conditions des tâches. Pour planifier et ordonner de ressources dans le Cloud Computing, nous avons proposé dans ce mémoire une stratégie d'ordonnancement basée sur la regroupement de données qui comporte cinq phases, nommée respectivement, phase de construction des ensembles des données ainsi que construction de la matrice de dépandance DM, et après ça on construit une matrice clustérisée nommée CM à base de l'algorithme BEA, puis, la phase de partitionnement et distribution des bloques construites aux différentes virtuelles machines, et la dernière phase c'est la phase de l'allocation et de l'exécution des tâches.

Dans ce travail, nous avons comparés notre approche avec l'approche de placement de données aléatoire. Notre stratégie d'ordonnancement basée sur le regroupement permet de réduire le temps de réponse et diminue le déplacement de données. Les résultats de simulation obtenus pour notre approche d'ordonnancement sont satisfaisants et trés encourageaux .

Perspectives

Ce travail a permis d'ouvrir quelques perspectives intéressantes que nous récapitulons dans les points suivants :

1. Dans le Cloud Computing, le placement des données est important. Dans ce but, nous proposons l'extension de notre stratégie par l'ajout d'un service de réplication pour diminuer encore plus le nombre déplacement de données.
2. Intèger le service de tolérance aus panne.
3. Traiter le coût de payement selon le contrat SLA.
4. Intèger notre approche dans un environnement de cloud réel.

Liste des acroynymes

As Accounting Service

AWS Amazon Web Services

CIS Cloud Information Service

Cloudlet réquet

CPU Central processing unit

DC Data Center

FCFS First Come First Served

IaaS Infrastructure as a service

IT Informatique Technology

Host Hôte

NIST The National Institute of Standards and Technology

PaaS Platform as a service

oS uality of Service

SaaS Software as a service

SDK Software Développement Kit

SLA Service Level Agreement

VM Virtual Machine

Bibliographie

[1] Ian T. Foster, Yong Zhao, Ioan Raicu, and Shiyong Lu. Cloud computing and grid computing 360-degree compared. CoRR, abs/0901.0131, 2009.

[2] DJEBBAR Esma Insaf,"optimisation d'ordonnancement et d'allocation de ressources dans les cloud computing",(thèse doctorat ,2016),

[3] Shruti Chhabra and V. S. Dixit. Cloud computing : State of the art and security issues. SIGSOFT Softw. Eng. Notes, 40(2) :1(11, April 2015).

[4] Luis M. Vaquero, Luis Rodero-Merino, Juan Caceres, and Maik Lindner. Abreak in the clouds : Towards a cloud definition. SIGCOMM Comput. Commun. Rev., 39(1) :50(55, December 2008).

[5] A. Ohri. R for Cloud Computing : An Approach for Data Scientists. Springer, New York Heidelberg Dordrecht London, 2014.

[6] Global Digital Vision. Cloud computing. http ://www.gdv.com.au/cloud computing.html, (Consulte juin 2022).

[7] DJEBBARA Mohamed Rédha, "gestion de repliques dans les clouds computing",(thèse doctorat ,2018-2019),

[8] Akbi khalil,Zehri mohammed, Etude et mise en place d'une solution cloud computing privé au sein de l'université de ouargla,(MASTER ACADEMIQUE,2013-2014)

[9] Liang Zhao, Sherif Sakr, Anna Liu, and Athman Bouguettaya. Cloud Data Management. Springer Editor, 2014.

[10] G. Oster,"Réplication Optimiste et Cohérence des Données dans les Environnements Collaboratifs Repartis",Nancy-Universite,France,Novembre 2005

[11] I. Foster and k Ranganathan,"Identifying dynamic replication strategies for high performance data grids". In 3rd IEEE/ACM International Workshop on Grid Computing, Lecture Notes on Computer Science, Denver, USA., 2002.

[12] A. Jebali, " Contrôle de divergence dans des environnements faiblementsconnectes." Thése de doctorat, Université de Versailles Saint¸uentin EnYvelines, France (Novembre 2003).

[13] K. Hakima, M. Kahina, « Etude et mise en place d'une solution Cloud Computing », Thèse de mémoire master, Université de Béjaia, 2013

[14] fa nomena fenohanita, implementation d'un cloud privé iaas et analyse big data dans un systeme distribue au sein d'acm , (diplome de master , 2019-2020),

[15] J. R. Rodrigues, L. Z. Zhou, L. M. Mendes, K. L. Lin, and J. L. Lloret. Distributed media-aware ow scheduling in cloud computing environment. Computer Communications, 35(1) :1819(1827, September 2012).

[16] C. T. Tsai and J. R. Rodrigues. Metaheuristic scheduling for cloud : A survey. IEEE Systems, 8(1) :279(291, March 2014).

[17] Setrag Khoshafian and Marek Buckiewicz. Groupware et workflow. 1998.

[18] GNIMASSOUN Ban'délé Jean Edgard, Ordonnancement d'Applications Parallèles (Workflows Scientifiques) sur les ressources IaaS du Cloud Computing,(thèse doctorat,2020-2021)

[19] Bessai, K. Gestion optimale de l'allocation des ressources pour l'exécution des processus dans le cadre du Cloud (thèse de Doctorat, Université Paris1 Panthéon-Sorbonne), 2014.

[20] FEMMAM Manel, Une approche formelle pour la planification des tâches pour la QoS dans le cloud-computing,(2017-2018).

[21] S., Mohapatra, S. Mohanty, and K.S. Rekha, Analysis of different variants in round robin algorithms for load balancing in cloud computing, International Journal of Computer Applications, vol. 69, no. 22, 2013.

[22] G. Ming, H. Li, "An Improved Algorithm Based on Max-Min for Cloud Task Scheduling," in Recent Advances in Computer Science and Information Engineering vol 125, pp. 217-223. Springer, Berlin, Heidelberg. https ://doi.org/10.1007/978-3-642-25789-632.

[23] Santhosh, and D.H. Manjaiah, "An improved task scheduling algorithm based on max-min for cloud computing," International Journal of Innovative Research in Computer and Communication Engineering, vol. 2, no. 2, pp. 84-88, May 2014.

[24] N. A. Mehdi, A. Amar, A. Amer, and Z. T AbdulMehdi, "Minimum Completion Time for Power-Aware Scheduling in Cloud Computing," in Developments in E-systems Engineering (DeSE), Dubai, 2011, pp. 484-489, doi : 10.1109/DeSE.2011.30.

[25] Rahul,M. "A Brief Review of Scheduling Algorithms in Cloud Computing". Asian Journal of Technology Management Research, vol.05, 2015.

[26] Akhtar, Muhammad AND Hamid, Bushra AND Ur-Rehman, Inayat AND Humayun, Mamoona and Hamayun, Maryam Khurshid, Hira. "An Optimized Shortest job first Scheduling Algorithm for CPU Scheduling". J. Appl. Environ. Biol. Sci , 5(12)42-46, 2015. 5. 42-46. 2015.

[27] Maheswaran, M., Ali, S., Siegal, H. J., Hensgen, D., and Freund, R. F. (1999). Dynamic matching and scheduling of a class of independent tasks onto heterogeneous computing systems. In Heterogeneous Computing Workshop, (HCW'99) Proceedings. Eighth .

[28] Z.AhLem,Imane,"Ordonnancement des workflow scientifiques dans environnement Cloud Computing" mémoire de master (soutenance en septembre 2022).

[29] Wiliam T .McCormick, Paul J. Schweitzer, and Thomas W. White. "Problem decomposition and data reorganization by a clustering Technique", Volume 20, chapter 1, Operations Research 1972

[30] M.Tamer Ozsu,Patrick Valduriez, Prinicipales of disributed Database Systems,Third Edition.

[31] Dong Yuan, Yun Yang, Xiao Liu, Jinjun Chen, A data placement strategy in scientific cloud workflows, Future Generation Computer Systems, Volume 26, Issue 8, 2010, Pages 1200-1214,

[32] https ://jmdoudoux.developpez.com/cours/developpons/eclipse/chap-eclipse-intro.php (consulte le 12/06/22)

[33] BOUAMAMA samah "gestion des ressources dans les cloud computing à base des modèles économiques"(thèse doctorat,Soutenu (e) le :06/03/2011).

Table des figures

Liste des tableaux