République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Abbes Laghrour Khenchela
Faculté des sciences et de la Technologie
Département de Mathématiques et Informatique

Mémoire de fin d'étude pour l'obtention de diplôme de Master en Informatique

Option : Sécurité et Technologies Web

Vers une Nouvelle Méthode à Base des Réseaux de Neurones Pour la Prédiction des Pannes dans le Cloud Computing

Présenté par :

BEZZA Youcef

Dirigée par :

Mme.HIOUAL Ouided

Devant le jury :

Président :	Dr. Sofiane HEMAM	MCA	Université Abbes Laghrour Khenchela
Rapporteur :	Mme.Ouided HIOUAL	MAA	Université Abbes Laghrour Khenchela
Examinateur :	Dr. Nawel TAKOUACHET	MCB	Université Abbes Laghrour Khenchela

Année universitaire :2017/2018

Remerciements

Tout d'abord, nous remercions le Dieu, notre créateur de nos avoir donné les forces, la volonté et le courage afin d'accomplir Ce travail modeste.

Je tiens à exprimer toute ma reconnaissance à mon Directeur de mémoire Madame « Ouided Hioual ». Je la remercie pour sa patience, sa disponibilité et surtout ses judicieux conseils, qui ont contribué à alimenter ma réflexion.

Veuillez bien recevoir mes remerciements pour le grand honneur que vous m´ avez Fait d'accepter l'encadrement de ce travail.

Aux membres de jury.

Et à tous ceux qui ont contribué de près ou de loin pour que ce projet soit possible, je vous dis merci.

Dédicace :

Je dédie ce modeste travail à :

À MES CHERS PARENTS

Aucune dédicace ne saurait exprimer mon respect, mon amour

Éternel et ma considération pour les sacrifices que vous avez consenti pour mon

instruction et mon bien être.

Je vous remercie pour tout le soutien et l'amour que vous me portez

Depuis mon enfance et j'espère que votre bénédiction m'accompagne toujours.

Que ce modeste travail soit l'exaucement de vos vœux tant

formulés, le fruit de vos innombrables sacrifices, bien que je ne vous en

acquitterai jamais assez.

Puisse Dieu, le Très Haut, vous accorder santé, bonheur et longue

vie et faire en sorte que jamais je ne vous déçoive.Mon frère Houssem-eddine, et mes sœurs Ikram et Nihal pour leur encouragement.

A tous mes Ami(e)s que j'aime tant : Aymen, Bachir, Djaber, Imène, Karim, Nourhanne, Zakaria, ...

Pour leur sincère amitié et confiance, et à qui je dois ma reconnaissance et mon attachement.

A tous ceux qui me sont chers et proches,

A mes collègues.

Table des matières :

Table des tableaux :

Table des figures :

INTRODUCTION GENERALE

1. Contexte du travail

De nos jours, le domaine des technologies de l'information et de la communication est devenu l'un des piliers de la société moderne. Le développement du concept de Cloud Computing (informatique en nuage) a constitué une avancée majeure vers la démocratisation de ce domaine. Cette technologie propose une combinaison de logiciels et de ressources qui montrent l'évolutivité dynamique dans la nature. Il fournit des services sophistiqués, de haute performance et de stockage de données évolutives à un large nombre croissant et quotidien des utilisateurs. Ces services sont accessibles n'importe où, n'importe quand et par n'importe qui. Les sociétés qui proposent ces services informatiques sont appelées fournisseurs de services Cloud. Elles facturent en général ces services en fonction de l'utilisation, un peu comme votre facture d'eau ou d'électricité chez vous. Le Cloud est composé de différents services qui peuvent être affectés à différents types de défauts. Donc, ces services nécessitent différents niveaux de tolérance aux pannes techniques afin de fournir un service sans faille. Le cloud computing est confronté à de nombreux problèmes pour détecter et prévoir l'échec de ses services, en particulier au niveau matériel .Le Cloud n'est pas infaillible, et le risque zéro n'existe pas .Les pannes de services Cloud peuvent avoir de fâcheuses conséquences. Selon Veritas, une grande majorité d'entreprises utilisant le Cloud Computing ne mesurent pas le coût d'une panne et n'ont pas conscience de leur part de responsabilité.

L'Intelligence Artificielle, partie de l'Informatique fondamentale s'est développée avec pour objectif la simulation des comportements du cerveau humain. Les premières essais de modélisation du cerveau précèdent même l'ère informatique. Les réseaux de neurones artificiels ont été développés avec pour objectifs principaux d'une part la modélisation et compréhension du

fonctionnement du cerveau et d'autre part pour réaliser des architectures ou des algorithmes d'intelligence artificielle.

Le neurone formel inspirait du neurone biologique , le neurone biologique est une cellule vivante spécialisée dans le traitement des signaux électriques. Un neurone artificiel est considéré comme un élément élémentaire de traitement de l'information, Il reçoit les entrées et produit un résultat à la sortie. Selon la topologie de connexion des neurones, on distingue deux grandes catégories : les réseaux de neurones non-bouclé et les réseaux de neurones bouclé. Il existe deux types d'apprentissages : l'apprentissage supervisé et l'apprentissage non supervisé, dans notre travail on travaille avec l'apprentissage supervisé.

Dans le domaine de la tolérance aux pannes beaucoup de recherches ont été fait, la tolérance aux pannes est la disposition d'un système informatique à réaliser son fonctionnement malgré la présence ou l'occurrence de pannes. Il y deux politiques de tolérance au panne (Fault Tolerance) dans le Cloud : tolérance aux pannes proactive et tolérance aux pannes réactive. Basé sur ces politiques de tolérance de pannes, différentes techniques sont utilisées pour fournir la tolérance de panne. Les techniques de tolérance aux pannes réactives sont utilisées pour réduire l'impact des défaillances sur un système lorsque les défaillances se sont réellement produites. La tolérance proactive aux pannes permet de prévoir les pannes et de remplacer les composants suspects par d'autres composants de travail, évitant ainsi la récupération des pannes et des erreurs.

2. Problématique et objectifs

Les pannes des services Cloud au niveau de la couche matérielle sont le grand défi face au fournisseurs des services , car leurs conséquences sont couteux, pour cela il faut prédire et éviter ces pannes.

Pour résoudre le problème de prédiction des pannes au niveau de Cloud computing on propose une technique probabiliste basée sur les réseaux de neurones artificiels en utilisant la technique proactive et l'apprentissage supervisé.

L'objectif de ce travail est l'utilisation des réseaux de neurones artificiels avec des fonctions probabilistes et l'apprentissage supervisé. Les réseaux de neurones utilisés sont les réseaux de neurones « Multi Couches », dont les nœuds sont les CPU. La couche cachée représente l'Algorithme de l'équilibrage de charge. Les Sorties représentent le temps d'exécution optimal et

la charge de chaque processeur. La Politique de tolérance de panne choisi est : « La Technique Proactive ».

3. Organisation de la thèse

Le manuscrit est structuré en quatre chapitres et une conclusion générale.

1- Tout d'abord, Le premier chapitre donne une introduction du Cloud Computing, y compris la définition, l'architecture, les méthodes de déploiement, les services de Cloud Computing et ses origines depuis l'informatique utilitaire en passant par les grilles de calculs, ainsi que les différents types de Cloud qui existent à l'heure actuelle et les différents types d'utilisateurs visés par les solutions de Cloud Computing et les besoins qui leur sont reliés.
2- Les bases sur les réseaux de neurones sont décrites dans le chapitre 2. Il sont abordés comme une méthodologie permettant de résoudre des problèmes de prédiction de panne.
3- Le chapitre 3 est consacré pour présenter la méthode proposée pour garantir une meilleurs utilisation du Cloud tout en garantissant qu'il n y aura pas des problèmes en niveaux de la couche matérielle.
4- Le chapitre 4 de ce manuscrit, montre la réalisation en détaille de notre méthode proposée.
5- Enfin nous terminons ce manuscrit par une conclusion générale qui récapitule le travail réalisé et propose quelques travaux en perspective.

Chapitre 1

Cloud Computing

1. Introduction

Le Cloud computing, ou informatique en nuages, a émergé comme un nouveau paradigme pour offrir des ressources informatiques à la demande et pour externaliser des infrastructures logicielles et matérielles.

Dans ce chapitre nous détaillons le modèle de Cloud computing qui permet à des demandeurs en ressources informatiques d'y accéder à la demande. Ce modèle a rencontré un fort succès et est utilisé par de nombreux acteurs très divers, aboutissant à une multitude de visions de ce qu'est le Cloud computing.

2. Historiques

La notion de Cloud fait référence à un nuage, tel que l'on a l'habitude de l'utiliser dans des schémas techniques lorsque l'on veut représenter Internet.

Il est communément admis que le concept de Cloud computing a été initié par le géant Amazon en 2002. Le cybermarchand avait alors investi dans un parc informatique afin de pallier les surcharges des serveurs dédiés au commerce en ligne constatées durant les fêtes de fin d'année. A ce moment-là, Internet comptait moins de 600 millions [AtulJha Johnson et al,2012] d'utilisateurs mais la fréquentation de la toile et les achats en ligne étaient en pleine augmentation. En dépit de cette augmentation, les ressources informatiques d'Amazon restaient peu utilisées une fois que les fêtes de fin d'année étaient passées. Ce dernier a alors eu l'idée de louer ses capacités informatiques le reste de l'année à des clients pour qu'ils stockent les données et qu'ils utilisent les serveurs. Ces services étaient accessibles via Internet et avec une adaptation en temps réel de la capacité de traitement, le tout facturé à la consommation.

Cependant, ce n'est qu'en 2006 qu'Amazon comprit qu'un nouveau mode de consommation de l'informatique et d'internet faisait son apparition.

Bien avant la naissance du terme de Cloud Computing, utilisé par les informaticiens pour qualifier l'immense nébuleuse du net, des services de Cloud étaient déjà utilisés comme le webmail2, le stockage de données en ligne (photos, vidéos,…) ou encore le partage d'informations sur les réseaux sociaux.

C'est en 2009 que la réelle explosion du Cloud survint avec l'arrivée sur le marché de sociétés comme Google (Google App Engine), Microsoft (Microsoft Azure), IBM (IBM Smart Business

Service), Sun (Sun Cloud) et Canonical Ltd (Ubuntu Enterprise Cloud). D'après une étude menée par Forrester [Ried. S, et al 2011].

La virtualisation a été la première pierre vers l'ère du Cloud Computing. En effet, cette notion permet regroupe l'ensemble des techniques matérielles ou logicielles permettant de faire fonctionner, sur une seule machine physique, plusieurs configurations informatiques (systèmes d'exploitation, applications, mémoire vive,…) de manière à former plusieurs machines virtuelles qui reproduisent le comportement des machines physiques [Hannachi.S,2015].

3. Définition du Cloud

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 ».

Même si les experts ne sont pas d'accords sur sa définition exacte, la plupart s'accordent à dire qu'elle inclue la notion de services disponibles à la demande, extensibles à volonté et à distance ou sur le net. En contradiction avec les systèmes actuels, les services sont virtuels et illimités et les détails des infrastructures physiques sur lesquels les applications reposent ne sont plus du ressort de l'utilisateur [Hannachi.S,2015].

"*Le Cloud Computing est un modèle qui permet d'accéder rapidement à un pool de ressources informatiques mutualisées, à la demande (serveurs, stockage, applications, bande passante, etc.), sans forte interaction avec le fournisseur de service* " [Peter.M and Timothy.G].

L'informatique en nuage s'appuie sur cinq caractéristiques essentielles :

- **Un accès libre à la demande :** l'utilisateur consomme des services (capacité de stockage, de calcul, plateforme de développement, etc.) dans le cloud selon ses besoins et ceci de façon automatique.
- **Le service doit être accessible via un réseau :** tous les services proposés aux utilisateurs doivent être disponibles sur le réseau et accessibles via des mécanismes standards favorisant l'utilisation des plateformes hétérogènes.
- **Mutualisation des ressources :** les ressources informatiques du fournisseur sont agrégées et mises à disposition des consommateurs sur un modèle multi-locataires, avec une attribution dynamique des ressources physiques et virtuelles en fonction de la demande.

Les consommateurs n'ont généralement aucune connaissance ni aucun contrôle de l'endroit exact où sont stockés les ressources fournies.

- **Redimensionnement rapide :** les consommateurs peuvent rapidement ajouter ou enlever des ressources informatiques en fonction de leurs besoins. Pour le consommateur, les ressources informatiques mises à disposition par le fournisseur semblent illimitées et peuvent être consommées en quantité à tout moment .
- **Le service doit être mesurable :** les systèmes contrôlent et optimisent de façon automatique l'utilisation des ressources par rapport à une moyenne estimée du service consommé.

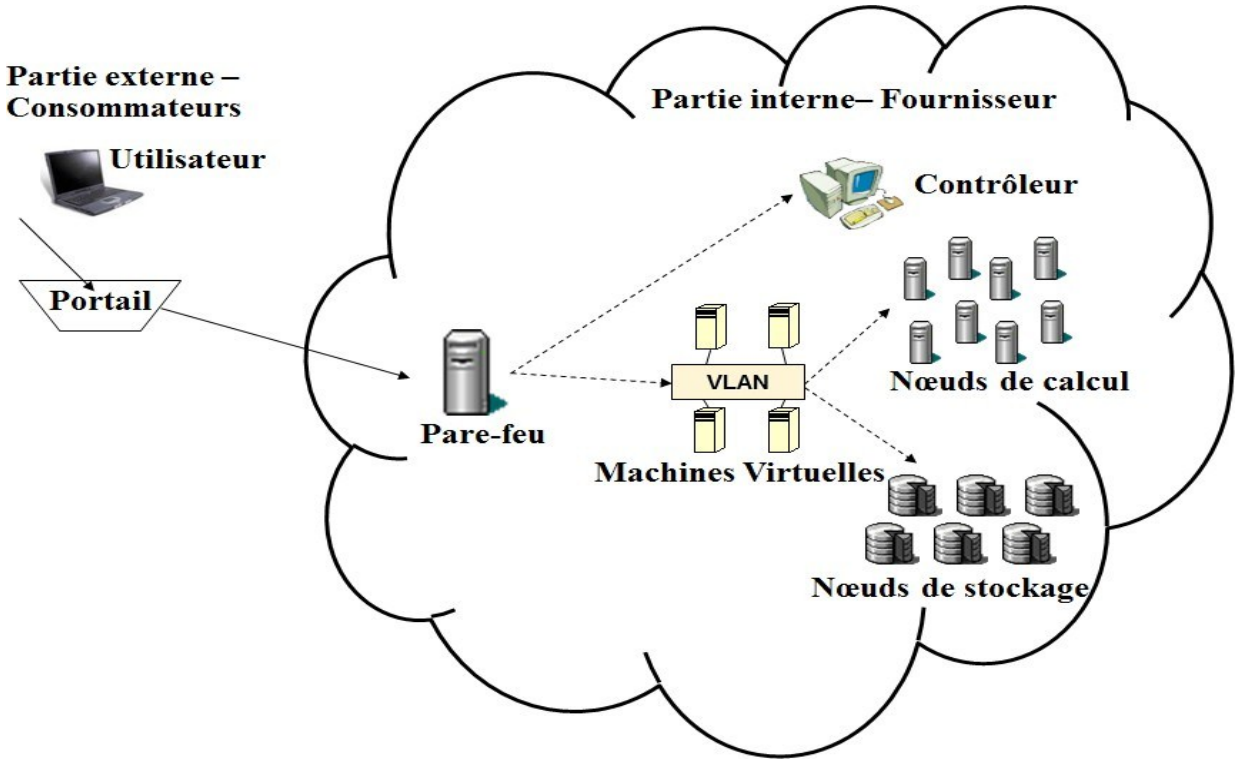

Figure1.1 : le Cloud computing(Informatique en nuage)
[Peter.M and Timothy.G].

4. Acteurs

Le Cloud computing est composé principalement par cinq acteurs majeurs (Cloud Provider, Cloud Consumer, Cloud Carrier, Cloud Broker, Cloud Auditor)

4.1. Cloud Provider

Le fournisseur des ressources Cloud computing. Il est responsable de fournir un service Cloud computing qui satisfait les caractéristiques définies dans la précédente section, tout en respectant les Service Level Agreements (SLAs) établies avec les autres acteurs (en particulier le Cloud Consumer). Le Cloud Provider a comme activité l'allocation, l'orchestration et la gestion des ressources qu'il offre tout en assurant le bon niveau de sécurité.

4.2. Cloud Consumer

L'utilisateur des ressources Cloud computing. Cet utilisateur peut être un utilisateur final ou un développeur selon le type du service Cloud alloué. Cet utilisateur peut être une personne, un groupe de personnes, les petites et moyennes entreprises, les multinationales ou les gouvernements.

4.3. Cloud Carrier ou Network Provider

Le fournisseur de réseau est l'intermédiaire qui assure principalement la connectivité entre les ressources Cloud computing et la liaison entre les acteurs du Cloud computing (en particulier entre le Cloud Provider et le Cloud Consumer).

Cet utilisateur peut jouer un simple rôle d'acheminements des paquets, comme il peut jouer un rôle plus important en offrant des fonctionnalités avancées dans le réseau. Ces fonctionnalités sont basées sur des SLAs établies avec les autres acteurs de l'écosystème.

4.4. Cloud Broker

Le courtier Cloud est un intermédiaire qui négocie la relation entre les Cloud Providers et les Cloud Consumers. Il peut offrir de nouveaux services qui simplifient les tâches de gestion du Cloud Consumer. Ce dernier peut demander les ressources Cloud Computing auprès du Cloud Broker au lieu du Cloud Provider directement.

4.5. Cloud Auditor

L'auditeur Cloud s'occupe de la vérification et l'audition des services Cloud computing. Il évalue les services offerts par les Cloud Providers, Cloud Carriers et Cloud Brokers du point de

vue performances et sécuritaires. Le but principal est de vérifier que les fournisseurs respectent bien les SLAs qu'ils proposent.

5. L'architecture cloud - Les 3 niveaux de service

Le Cloud computing peut être décomposé en trois couches :

- Applicative (SaaS, Software as a Service)
- Plate form (PaaS, Platform as a Service)
- Infrastructure (IaaS, Infrastructure as a Service)

La Figure 1.2 ci-dessous représente les différentes couches du cloud computing de la couche la moins visible pour les utilisateurs finaux à la plus visible. L'infrastructure as a Service (Iaas) est plutôt gérée par les architectes réseaux, la couche PaaS est destinée au développeurs d'applications et finalement le logicielle comme un service (SaaS) est le «produit final » pour les utilisateurs.[Akbi.K Zehri.M,2013]

5.1. Infrastructure as a Service (IaaS)

Seul le serveur est dématérialisé. Un prestataire propose la location de composants informatiques comme des espaces de stockages, une bande passante, des unités centrales et des systèmes d'exploitation. Les utilisateurs d'une IaaS peuvent donc utiliser à la demande des serveurs virtuels situés dans des Datacenter sans avoir à gérer les machines physiques(coûts de gestion, remplacement de matériel, climatisation, électricité….)L'IaaS offre une grande flexibilité, avec une administration à distance, et permet d'installer tout type de logiciel.

En revanche, cette solution nécessite la présence d'un administrateur système au sein de l'entreprise, comme pour les solutions serveur classiques. Parmi les prestataires d'IaaS, on peut citer : Amazon avec EC2 ou Orange Business Services avec Flexible Computing .[Akbi.K Zehri .M,2013].

5.2. Platform as a Service (PaaS)

La couche intermédiaire Platform as a Service regroupe des services qui ciblent les concepteurs de services logiciels. Les fournisseurs de PaaS fournissent des plates-formes logicielles aux concepteurs de service, qui sont livrées avec des interfaces de programmation

dédiées (API : application program interface) pouvant accueillir et exécuter des services de Cloud Computing. En utilisant les services de la couche PaaS, le travail des concepteurs est simplifié, car une grande partie de la complexité liée à la prise en main des aspects liés à l'infrastructure est masquée derrière l'API exposée par le fournisseur PaaS. De cette manière, le processus de développement d'un service logiciel est simplifié, car un concepteur qui utilise une plateforme PaaS ne fait que fournir du code source au fournisseur PaaS, qui en retour s'occupe de le déployer sous la forme d'un service, et de le rendre accessible aux utilisateurs au moyen d'un client léger. [Euro Cloud France , 2011]

5.3. Software as a Service (SaaS)

La couche haute Software as a Service (SaaS) regroupe des services rendus accessibles aux utilisateurs finaux au moyen d'un client léger (application web, application mobile, client bureau, . . .). Ces services sont hébergés à distance : La majeure partie de leur code source (en particulier la partie traitement), est exécutée à distance sur les infrastructures des fournisseurs de services.

Les avantages d'un tel modèle de service sont nombreux, comme le fait que la complexité liée au déploiement d'un service ainsi que sa maintenance sont masquées à l'utilisateur final. Les exigences en matière de matériel informatique sont aussi plus basses, car une partie de l'exécution du service est délocalisée sur l'infrastructure distante appartenant au fournisseur.

Les exemples connus de SaaS sont le service de messagerie Gmail (Google), le réseau social Facebook et l'outil de gestion de la relation client (CRM) Salesforces. [Euro Cloud France, 2011]

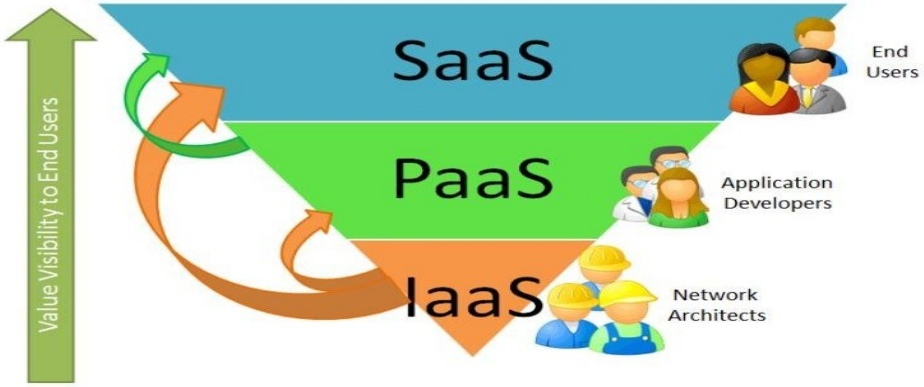

Figure1.2 : Les 3 couches du Cloud computing [Warin, 2011]

6. Les modèles de déploiement du Cloud Computing

Il existe différents modèles de déploiement du Cloud, chacun avec ses avantages et ses inconvénients.[Elwessabi.A Ahmed.Y,2014]

6.1. Cloud public

Dans un Cloud public les Fournisseur de services Cloud offrent leurs ressources comme services au grand public. Ce modèle peut être détenu, géré et exploité par une entreprise ou une organisation académique ou gouvernementale, ou une combinaison entre eux. Le Cloud public offre plusieurs avantages aux utilisateurs, y compris l'absence de coûts d'investissement élevés sur les infrastructures et le déplacement des risques vers les fournisseurs d'infrastructure. Mais, ces utilisateurs n'ont pas un contrôle fin sur les données, le réseau et les paramètres de sécurité, ce qui entrave l'efficacité de ce modèle de déploiement.

6.2 .Cloud prive

Le Cloud privé est conçu pour une utilisation exclusive par une seule organisation. Un Cloud privé peut être construit et géré par l'organisation, une tierce partie, ou une combinaison des deux. Il offre le plus haut degré de contrôle sur les performances, la fiabilité et la sécurité. Cependant, il est souvent critiqué car il est similaire aux serveurs propriétaires traditionnels qui ne fournissent pas les avantages du Cloud comme l'absence de coûts d'investissement élevés.

6.3. Cloud hybride

Un Cloud hybride est une combinaison de déploiement des modèles de Cloud (public, privé et communauté) qui tente de remédier aux limitations de chaque approche. Dans un Cloud hybride, une partie du service de l'infrastructure s'exécute dans des Clouds privés tandis que la partie restante est dans des Clouds publics.

Un Cloud hybride offre plus de flexibilité qu'un Cloud public ou privé, puisqu'il fournit un meilleur contrôle et une meilleure sécurité pour les données d'application des utilisateurs par rapport aux Clouds publics et une tarification avantageuse par rapport aux Clouds privés. Cependant, la conception d'un Cloud hybride nécessite une étude détaillée afin de déterminer la meilleure répartition entre les composantes de Cloud public et privé. Pour la plupart des organisations, le choix du modèle de Cloud dépend du cas d'utilisation et des exigences du CSU (sécurité, QoS, etc.).

6.4. Cloud communautaire

Un Cloud communautaire est une infrastructure IaaS ouverte à une communauté d'utilisateurs : à partir du moment où un utilisateur est membre de l'organisation en charge de l'infrastructure, celui-ci peut exploiter les ressources de l'infrastructure communautaire.

Ce modèle reprend de nombreux aspects du modèle de grille de calcul communautaire, et on le retrouve fréquemment dans le cadre des infrastructures IaaS dédiées à une utilisation scientifique. Ce modèle est à mi-chemin entre les Cloud privés et les Cloud publics.

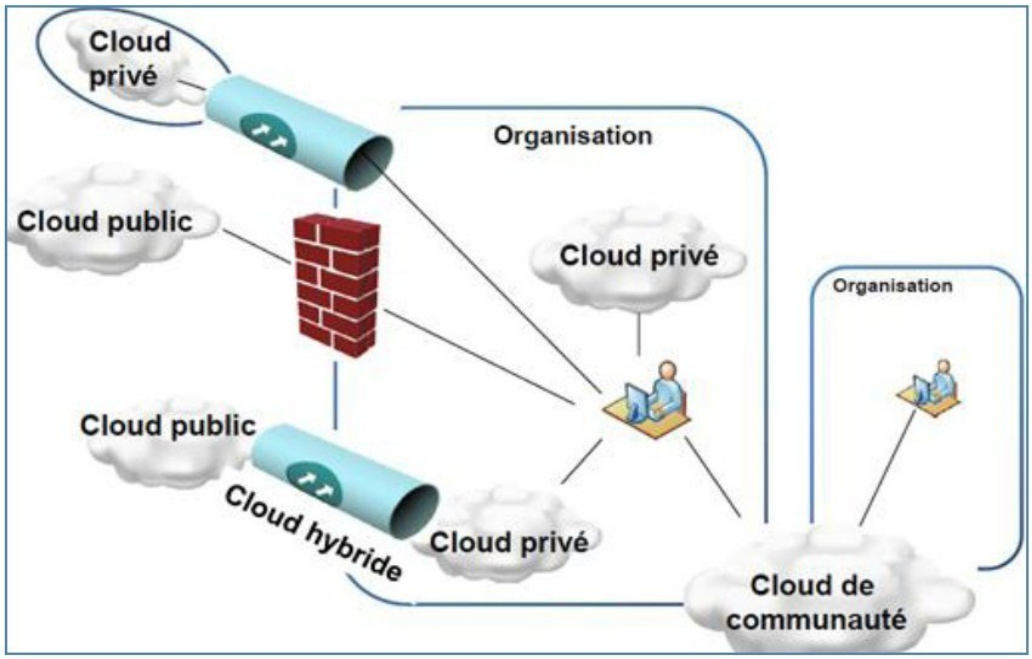

Figure1.3 : Modèles de déploiement du Cloud

7. les pannes au niveau de Cloud

Les pannes apparaissant dans le Cloud doivent être détectés et prédits efficacement afin de lancer des outils pour tolérer les défaillances survenues. Les techniques de prédictions de pannes sont classé en deux grandes classe L'approche « Proactive » et l'approche « Réactive » , ces deux techniques sont les plus utilisées pour prédire les pannes au niveau du Cloud .

7.1. L'approche Réactive

Les techniques de tolérance aux fautes réactives sont utilisées pour réduire l'impact des défaillances sur un système lorsque les défaillances se sont réellement produites. Les techniques basées sur cette politique sont check point / Restart et retry et ainsi de suite.

- Point de restauration : La tâche échouée est redémarrée depuis le dernier point de contrôle plutôt que depuis le début. C'est une technique efficace pour les grandes applications.
- Réplication: Afin de réussir l'exécution, diverses répliques de tâche sont exécutées sur des ressources différentes jusqu'à ce que la tâche répliquée entière ne soit pas écrasée. HAProxy, Haddop et AmazonEc2 sont utilisés pour implémenter la réplication.
- Migration des tâches: en cas d'échec, le travail est migré vers une nouvelle machine. HAProxy peut être utilisé pour migrer les travaux vers d'autres machines.
- SGuard: Il est basé sur la restauration de restauration et peut être exécuté dans HADOOP, Amazon Ec2.
- Réessayer: cette technique de niveau de tâche est la plus simple parmi toutes. L'utilisateur soumet à nouveau la tâche sur la même ressource de cloud.
- Nouvelle soumission de tâche: la tâche échouée est à nouveau soumise à la même machine sur laquelle elle était en fonctionnement ou à une autre machine.
- Gestion des exceptions définie par l'utilisateur: Ici, l'utilisateur définit l'action spécifique d'un échec de tâche pour les flux de travail
- Flux de travail de secours: Il permet au système de continuer à fonctionner après l'échec de n'importe quelle tâche jusqu'à ce qu'il ne puisse pas continuer sans rectifier le défaut

7.2. L'approche proactive

Cette approche de pannes permet de prévoir les pannes de manière proactive et de remplacer les composants suspects par d'autres composants de travail, évitant ainsi la récupération des pannes et des erreurs.

La migration préemptive, le rajeunissement logiciel, etc. suivent cette politique.

- Rajeunissement logiciel: le système est prévu pour des redémarrages périodiques et chaque fois que le système démarre avec un nouvel état.
- Tolérance de panne proactive utilisant l'auto-guérison: l'échec d'une instance d'une application s'exécutant sur plusieurs machines virtuelles est contrôlé automatiquement.
- Tolérance de panne proactive en utilisant la migration préemptive: Dans cette technique, une application est constamment observée et analysée.

La migration préemptive d'une tâche dépend du mécanisme de contrôle de la boucle de retour

8. Avantages et Inconvénients du Cloud Computing

Le Cloud computing a des avantages et des inconvénients [Mathur, P,2010]:

8.1. Avantage

Réduction des couts de gestion: avec le Cloud les entreprises non plus à se soucier de la gestion des ressources ou du personnel nécessaire à la supervision de leurs plateformes… il n'ont qu'à former le corps principal du personnel à utiliser les applications Cloud visés par l'entreprise.

- Moins de problèmes d'entretien.
- Gestion des mises à jour plus simple et rapide.
- infrastructure allouée et disponible juste à temps.
- Réduction du temps de mise sur le marché.
- Système anti désastre: la récupération des données et des applications après un désastre (séisme par exemple) est gérée par un backend qui stocke et relance le système à nouveau pour assurer une disponibilité permanente des services Cloud [Jian J.Z et al, 2011].
- Mobilité et accès facile: en stockant nos données et en déployant nos applications sur le Cloud l'accès à ces derniers devient seulement une question de connexion Internet.
- Réduction des couts d'utilisation: le modèle économique du Cloud permet aux clientsde réduire et de Controller leurs dépenses, puisque ils ne payent que ce qu'ils utilisent comme ressources Cloud.

8.2. Inconvénients

- Conformité réglementaire: lors du transfert des données du client vers le fournisseur du service Cloud, le client est seul responsable de l'intégrité et de la sécurité des données.
- Dépendance des services: un client n'a pas la possibilité de changer le type de services à consommer chez un fournisseur donné.
- Vie privée et lois de confidentialités: les lois qui régissent la préservation des données privées est différente d'un pays à un autre (par exemple les USA n'a pas les mêmes lois que L'union européenne), ce qui peut mettre les clients en situation de confusion par rapport aux services Cloud à utiliser, et cela peut limiter le transfert de données d'une ressource Cloud vers une autre qui se trouve dans un autre pays.
- Récupération de données: le fait de segmenter les taches et les données stockées par les clients et ensuite les éparpillés sur l'infrastructure Cloud, rend leur récupération et par la suite leurs rassemblement très difficile.
- Identification des clients: avec l'utilisation croissante du Cloud et l'utilisation multi location de ces ressources, il devient de plus en plus difficile d'identifier par qui et de quel endroit les données ont était modifiées?!
- Stockage de données: le stockage physique des données dans le Cloud est effectué par les fournisseurs des services ce qui limite la manipulation de ces dernier par les clients.

9. Conclusion

Dans ce chapitre, nous avons abordé le concept du Cloud computing, nous avons présenté la définition objective et générale de la notion de l'informatique dans le nuage, le modèle de déploiement et les caractéristiques essentielles. Cloud computing est encore un paradigme en évolution, il intègre de nombreuses technologies existantes dans le chapitre suivant en va présenter les réseaux de neurones

Chapitre 2

Réseaux de neurones

1. Introduction

Les réseaux de neurones sont constitués d'un ensemble de neurones artificiels ou nœuds qui sont analogues aux neurones biologiques. lls sont issus d'une tentative de conception d'un modèle mathématique très simplifié du cerveau humain en se basant sur notre façon d'apprendre et de corriger nos erreurs. Ce chapitre présente une introduction à la théorie de réseaux de neurone**.**

2. Historique sur les réseaux de neurones

Les recherches dans le domaine du connexionnisme ont démarré avec la présentation en 1943 par W. MC Culloch et W. Pitts d'un modèle simplifié de neurone biologique communément appelé neurone formel. Ils montrèrent également théoriquement que des réseaux de neurones formels simples peuvent réaliser des fonctions logiques, arithmétiques et symboliques complexes.

En 1949, D. Hebb initie, dans son ouvrage "The Organization of Behavior", la notion d'apprentissage. Deux neurones entrant en activité simultanément vont être associés (c'est-à-dire que leurs contacts synaptiques vont être renforcés). On parle de loi de Hebb et d'associationnisme.

En 1958, F. Rosenblatt développe le modèle du Perceptron. C'est un réseau de neurones inspiré du système visuel. Il possède deux couches de neurones : une couche de perception (sert à recueillir les entrées) et une couche de décision. C'est le premier modèle pour lequel un processus d'apprentissage a pu être défini. S'inspirant du perceptron, Widrow et Hoff, développent, dans la même période, le modèle de l'Adaline (Adaptive Linear Element). Ce dernier sera, par la suite, le modèle de base des réseaux de neurones multi-couches.

En 1969, Les recherches sur les réseaux de neurones ont été pratiquement abandonnées lorsque M. Minsky et S. Papert ont publié leur livre « Perceptrons » (1969) et démontré les limites théoriques du perceptron, en particulier, l'impossibilité de traiter les problèmes non linéaires par ce modèle.

En 1982, Hopfield développe un modèle qui utilise des réseaux totalement connectés basés sur la règle de Hebb pour définir les notions d'attracteurs et de mémoire associative. En 1984 c'est la découverte des cartes de Kohonen avec un algorithme non supervisé basé sur l'auto-organisation et suivi une année plus tard par la machine de Boltzman (1985).

Une révolution survient alors dans le domaine des réseaux de neurones artificiels : une nouvelle génération de réseaux de neurones, capables de traiter avec succès des phénomènes non- linéaires : le perceptron multicouche ne possède pas les défauts mis en évidence par Minsky.

Proposé pour la première fois par Werbos, le Perceptron Multi-Couche apparaît en 1986 introduit par Rumelhart, et, simultanément, sous une appellation voisine, chez Le Cun (1985). Ces systèmes reposent sur la rétropropagation du gradient de l'erreur dans des systèmes à plusieurs couches, chacune de type Adaline de Bernard Widrow, proche du Perceptron de Rumelhart. [Kadous Djamila, 2012][Dreyfus D.et al ,2004].

3. Les réseaux de neurones

3.1. Neurone artificiel

Un neurone artificiel est considéré comme un élément élémentaire de traitement de l'information, Il reçoit les entrées et produit un résultat à la sortie. x1, x2, ..., xn ; sont les entrées externes. y est la sortie. w1, w2, ..., wn sont les poids associés à chaque connexion. X est le vecteur d'entrée, W' est le vecteur poids, θ est appelé le biais.

La fonction φ est appelée fonction d'activation, c'est une fonction non linéaire. Différentes fonctions d'activation peuvent être utilisées, parmi lesquelles on peut citer : fonction signe, sigmoïde, tangente hyperbolique, Gaussienne [ZEMOURI.R,2003],[VIROLE.B.,2001].et le choix d'un type de fonction dépend de l'application.
La figure 2.1 montre un schéma comportant la structure générale d'un neurone artificiel.

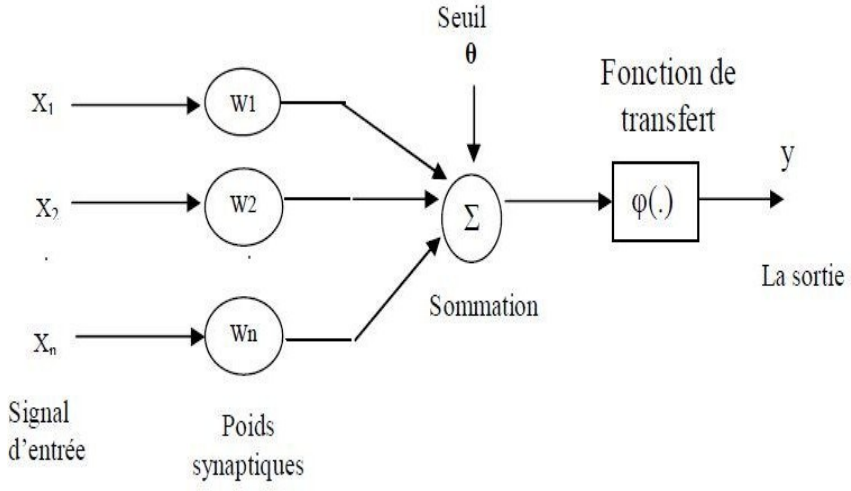

Figure 2.1: Neurone artificiel.[MERZOUKA.N,2009]

3.2. Les réseaux de neurones

Les réseaux de neurones artificiels sont des combinaisons de fonctions élémentaires. appelées neurones formels, ou simplement neurones associés en couches et fonctionnant en parallèle. Chaque processeur élémentaire calcule une sortie unique sur la base des informations qu'il reçoit. Toute structure hiérarchique de réseaux est évidemment un réseau.[MERZOUKA.N,2009]

Les réseaux de neurones artificiels sont des réseaux fortement connectés de processeurs élémentaires fonctionnant en parallèle. Chaque processeur élémentaire calcule une sortie unique sur la base des informations qu'il reçoit. Ils sont constitués d'un nombre fini de neurones qui sont arrangés sous forme de couches. Les neurones de deux couches adjacentes sont interconnectés par des poids. L'information dans le réseau se propage d'une couche à l'autre, on dit qu'ils sont de type « feed-forward ». Nous distinguons trois types de couches :

- **Couche d'entrée :** les neurones de cette couche reçoivent les valeurs d'entrée du réseau et les transmettent aux neurones cachés. Chaque neurone reçoit une valeur, il ne fait pas donc de sommation.
- **Couches cachées :** chaque neurone de cette couche reçoit l'information de plusieurs couches précédentes, effectue la sommation pondérée par les poids, puis la transforme selon sa fonction d'activation qui est en général une fonction sigmoïde. Par la suite, il envoie cette réponse aux neurones de la couche suivante.
- **Couche de sortie :** elle joue le même rôle que les couches cachées, la seule différence entre ces deux types de couches est que la sortie des neurones de la couche de sortie n'est liée à aucun autre neurone.

4. Propriétés des réseaux de neurones

Les réseaux de neurones artificiels possèdent une propriété fondamentale qui justifient l'intérêt croissant qui leur est accordé et que sont capable d'intervenir dans des domaines très divers, et qui les distingue des techniques classiques de traitement des données.

Les réseaux de neurones sont des approximateurs universels : Cette propriété peut être énoncée comme suit: Toute fonction bornée suffisamment régulière peut être approchée uniformément, avec bonne précision, dans un domaine fini de l'espace de ses variables, par un

réseau de neurones qui comporte une couche de neurones cachée en nombre fini, possédant tous la même fonction d'activation et un neurone de sortie linéaire [J.GHOUILI 2005.][B.GOSSELIN 1996.] .

Parcimonie : Lors de la modélisation d'un processus à partir de ses données, on cherche toujours à obtenir les résultats les plus satisfaisants possibles avec un nombre minimum de paramètres. On dit que l'on cherche l'approximation la plus parcimonieuse.

Pour obtenir un modèle non linéaire de précision donnée, un réseau de neurone a besoin de moins de paramètres ajustables que les méthodes de régression classiques (par exemple la régression polynomiale). Or le nombre de données nécessaires pour ajuster le modèle est directement lié au nombre de ses paramètres [G. DREYFUS et al,2002],[L.BAGHLI, 1999].

5. Architecture de réseaux de neurones

Selon la topologie de connexion des neurones, on peut les classer en deux grandes catégories: réseaux non bouclés (statique ou feed forward) et réseaux bouclés(dynamique, feed back ou récurrent).

5.1. Réseaux statiques "feed-forward "

Un réseau de neurones non bouclé (appelé aussi statique) est représenté comme un graphe dont les nœuds sont les neurones. L'information circule des entrées vers les sorties sans retour en arrière (Figure 2.2). Ce type de réseaux est utilisé pour effectuer des taches d'approximation de fonction non linéaire, de la classification ou de la modélisation de processus statiques non linéaires [O. NERRAND et al 1993.].

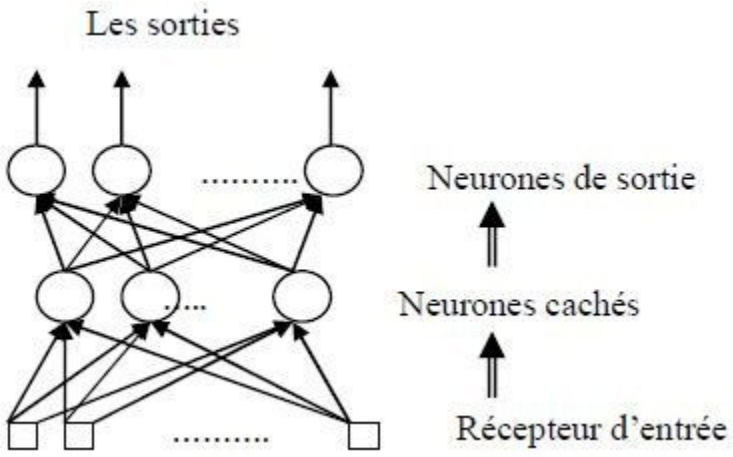

Figure: 2.2 : Exemple d'un réseau de neurones non bouclé [MERZOUKA .N,2009]

5.2. Réseaux multicouches (ou Perceptron Multi Couche PMC)

C'est le réseau de neurones statique le plus utilisé. Les neurones sont arrangés par couche. Les neurones de la première couche reçoivent le vecteur d'entrée, ils calculent leurs sorties qui sont transmises aux neurones de la seconde couche qui calculent eux même leurs sorties et ainsi de suite de couche en couche jusqu'à celle de sortie.

Chaque neurone dans la couche cachée est connecté à tous les neurones de la couche précédente et de la couche suivante, et il n'y a pas de connexions entre les cellules d'une même couche. Il peut résoudre des problèmes non linéairement séparables et il suit un apprentissage supervisé avec la règle de correction de l'erreur.[MERZOUKA NOURESSADAT,2009]

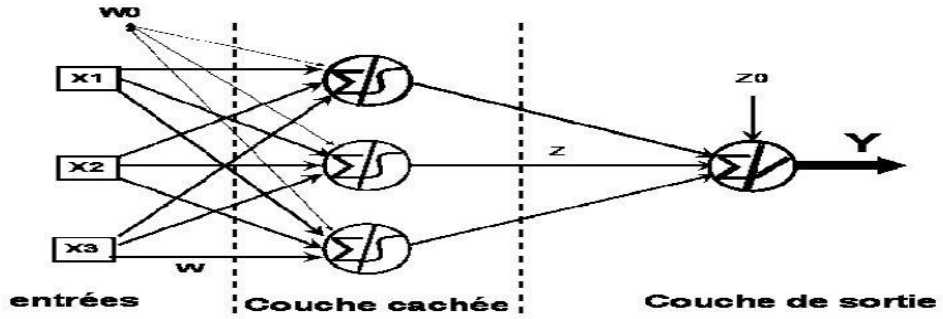

Figure 2.3 : Perceptron Multi Couche PMC [MERZOUKA.N,2009]

La sortie du réseau a pour expression :

$$Y = g\left[Z\left[f\left(WX + W_0\right)\right] + Z_0\right]$$

Avec :

f et g les fonctions de transfert, des couches cachées et de sorties respectivement, Y sortie du réseau, X vecteur des entrées. W : matrice des poids de connexions liant la couche d'entrée à la couche cachée. W0 : vecteur des biais des cellules de la couche cachée.

Z : matrice des poids des connexions liant la couche cachée à la couche de sortie.

Z0 : vecteur des biais des cellules de la couche de sortie.

5.3. Réseau récurrents "Feed-back"

Un réseau bouclé (récurrent), régi par une ou plusieurs équations différentielles, résulte de la composition des fonctions réalisées par chacun des neurones et des retards associés à chacune des connexions. Ces réseaux sont utilisés pour effectuer des taches de modélisation des systèmes dynamiques, de commande de processus ou de filtrage [NERRAND.O.et al, 1993.]-[SRINIVASA.V et al, 1996.].

Le comportement dynamique d'un réseau de neurones bouclé peut être décrit par une représentation d'état paramétrée par les coefficients C, représentée sur la figure 2.4.

$$\begin{cases} \mathbf{S(k+1) = \psi\ (\ S(k), I(k); C\ )} \\ \mathbf{Y(k) = \xi(\ S(k), I(k); C\ )} \end{cases}$$

Où I(k) est le vecteur des entrées externes, S(k) le vecteur des variables d'état, Y(k) le vecteur des sorties, ψ (., .; C) et ξ (., .; C) représentent les fonctions réalisées par le réseau de neurones statiques de la forme canonique interconnectés avec les coefficients C [RIVALS.I et al,1995.] .

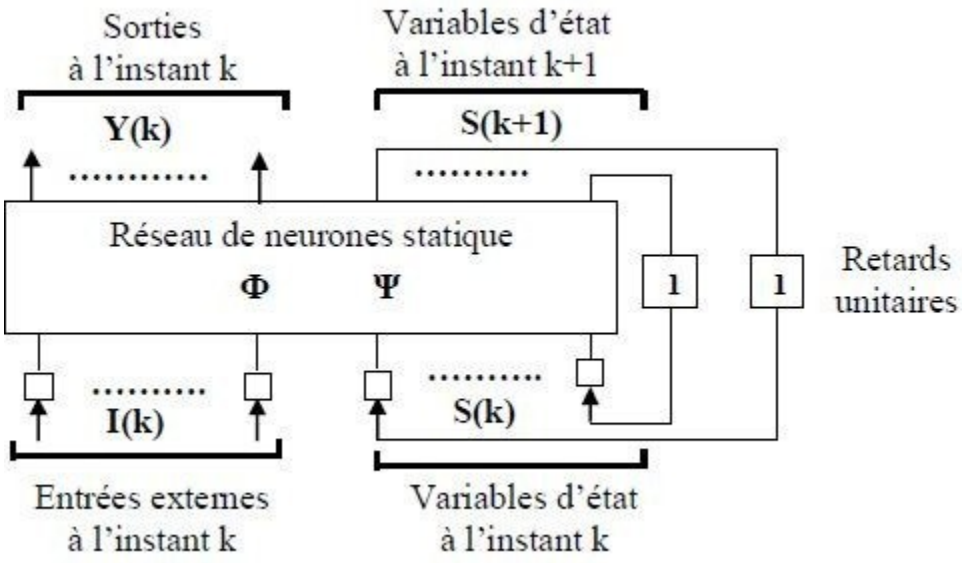

Figure 2.4 : Forme canonique d'un réseau de neurones bouclé [MERZOUKA.N, 2009]

6. L'apprentissage des réseaux de neurones

On appelle apprentissage des réseaux de neurones la procédure qui consiste à estimer les paramètres (poids, biais,…) des neurones du réseau, afin que celui-ci remplisse au mieux la tâche qui lui est affectée. il existe plusieurs paradigmes d'apprentissage.[Guesbaya Tahar,2012].

6.1. Apprentissage supervisé

Un apprentissage est dit supervisé lorsque l'on force le réseau à converger vers un état final précis, en même temps qu'on lui présente un motif. Ce genre d'apprentissage est réalisé à l'aide d'une base d'apprentissage, constituée de plusieurs exemples de type entrées-sorties (les entrées du réseau et les sorties désirées ou encore les solutions souhaitées pour l'ensemble des sorties du réseau).

La procédure usuelle dans le cadre de la prévision est l'apprentissage supervisé (ou à partir d'exemples) qui consiste à associer une réponse spécifique désirée à chaque signal d'entrée. La modification des poids s'effectue progressivement jusqu'à ce que l'erreur (ou l'écart) entre les sorties du réseau (ou résultats calculés) et les résultats désirés soient minimisés. Cet apprentissage n'est possible que si un large jeu de données est disponible et si les solutions sont connues pour les exemples de la base d'apprentissage. [Marc.P, 2004].

6.2. Apprentissage non supervisé (auto-organisationnel)

L'apprentissage non supervisé consiste à ajuster les poids à partir d'un seul ensemble d'apprentissage formé uniquement de données. Aucun résultat désiré n'est fourni au réseau. Qu'est-ce que le réseau apprend exactement dans ce cas ? L'apprentissage consiste à détecter les similarités et les différences dans l'ensemble d'apprentissage. Les poids et les sorties du réseau convergent, en théorie, vers les représentations qui capturent les régularités statistiques des données.

Ce type d'apprentissage est également dit compétitif et (ou) coopératif. L'avantage de ce type d'apprentissage réside dans sa grande capacité d'adaptation reconnue comme une Auto organisation, «self-organizing» [Kohonen, 1987]. L'apprentissage non supervisé est surtout utilisé pour le traitement du signal et l'analyse factorielle. [Jean-François.J , 1994].

6.3. Apprentissage hybride

Dans ce type on utilise les deux apprentissages supervisé et non-supervisé.

6.4. Apprentissage par renforcement

L'apprentissage renforcé est une technique similaire à l'apprentissage supervisé à la différence qu'au lieu de fournir des résultats désirés au réseau, on lui accorde plutôt un grade (ou score) qui est une mesure du degré de performance du réseau après quelques itérations. Les algorithmes utilisant la procédure d'apprentissage renforcé sont surtout utilisés dans le domaine des systèmes de contrôle [Marc.P, 2004].

6.5. Apprentissage compétitif

L'apprentissage compétitif, consiste à faire compétitionner les neurones d'un réseau pour déterminer lequel sera actif à un instant donné. Contrairement aux autres types d'apprentissage, où généralement tous les neurones peuvent apprendre simultanément et de la même manière, l'apprentissage compétitif produit un « vainqueur »ainsi que parfois, un ensembles de neurones « voisins » vainqueurs, et seuls se vainqueur et son voisinage bénéficient d'une adaptation de leurs poids, alors que le neurone qui ne gagne pas la compétition ne modifiera aucunement ses poids. Ainsi, les neurones individuels peuvent apprendre à se spécialiser sur des sous-ensembles de données pour devenir des détecteurs de caractéristiques [Dreyfus.G et al, 2002].

7. Avantages et Inconvénients

7.1. Avantage

Les principales qualités des réseaux de neurones sont leur capacité d'adaptabilité et d'auto-organisation et la possibilité de résoudre des problèmes non-linéaires avec une bonne approximation [Anand et al, 1992][Watrous.R.L., 1987]. Ils ont une bonne immunité aux bruits et se prêtent bien à une implantation parallèle. La rapidité d'exécution est une qualité importante et elle justifie souvent à elle seule le choix d'implanter un réseau de neurones. Ces qualités ont permis de réaliser avec succès, plusieurs applications : classification, filtrage, compression de données, contrôleur, etc.[HICHAM.C,2002].

7.2. Inconvénients

La difficulté d'interpréter le comportement d'un réseau de neurones est un inconvénient pour la mise au point d'une application. Il est souvent impossible d'utiliser les résultats obtenus pour améliorer ce comportement. Il est également hasardeux de généraliser à partir d'expériences antérieures et de conclure ou de créer des règles sur le fonctionnement et le comportement des réseaux de neurones.

Plusieurs paramètres doivent être ajustés et aucune méthode ne permet de choisir des valeurs optimales. Beaucoup d'heuristiques sont utilisées, mais elles se contredisent parfois et elles ne permettent pas toujours de trouver des valeurs optimales.[HICHAM.C,2002].

8. Conclusion

Dans ce chapitre nous avons présenté un état de l'art sur les réseaux de neurones. On a commencé par une introduction générale, puis l'historique des réseaux de neurones et fondement biologique.

Ensuite nous avons discuté différents architecture et apprentissage des réseaux de neurones et on a conclure par les avantages et les inconvénients. Dans le chapitre suivant en va présenter notre modèle.

Chapitre 3

Conception de la méthode proposée

1. Introduction

Le Cloud computing est confronté à de nombreux problèmes pour détecter et anticiper l'échec de ses services, notamment dans la couche matérielle, car la tolérance aux pannes est le plus grand défi du Cloud computing .La tolérance aux pannes est un point important pour garantir la disponibilité et la fiabilité des services sérieux de l'application.

Dans ce chapitre On va présenter notre méthode proposer pour garantir une meilleurs utilisation du Cloud tout en garantissant qu'il n y aura pas des problèmes en niveaux de la couche matérielle. Nous introduisons, tout d'abord, la problématique et les objectifs visés par cette étude. après avoir survoler quelques travaux existants. Ensuite, nous présentons notre méthode et nous terminerons ce chapitre par une conclusion.

Le Cloud computing est rapidement devenu l'une des technologies importantes dans le monde de l'ingénierie et de l'informatique. Jours après jours, l'utilisation du Cloud computing augmente car il fournit des services élastiques aux utilisateurs et aux entreprises.

Le Cloud computing est une technologie innovante dans le domaine de l'informatique distribuée qui fournit un service à la demande dans la technologie informatique. Cette technologie offre l'intégration de logiciels et de ressources qui présentent une évolutivité dynamique dans la nature. Ces systèmes sont bénéfiques, mais il y a aussi un inconvénient pour le même.

Le majeur problème dans le Cloud Computing est la tolérance aux pannes. La tolérance aux pannes est une préoccupation majeure pour garantir la disponibilité et la fiabilité des services critiques ainsi que l'exécution de l'application. Afin de minimiser l'impact de l'échec sur le système et l'exécution de l'application

Comme la tolérance aux pannes est très importante, nous proposons une méthode de prédiction de pannes on utilisant des réseaux de neurones probabilistes supervisés dans le Cloud computing.

2.Travaux connexes

Il y a beaucoup de travail sur la tolérance aux pannes dans le Cloud computing, nous examinons ceci:

Dans [Karahroudy, 2011], les auteurs ont proposé un middleware de tolérance aux pannes qui implémente un plan de réplication de serveur synchronisé, dans lequel un serveur défaillant est maintenu avec un état cohérent.

Dans [Labaf, 2007], les auteurs ont proposé un nom de méthode AFTRC (Adaptive FaultTolerance in Cloud Computing en temps réel). Ce schéma tolère les défauts sur la base de la fiabilité de chaque nœud de calcul. Le schéma proposé est une bonne option à utiliser en tant que mécanisme de tolérance aux pannes pour le calcul en temps réel sur l'infrastructure Cloud.

Les auteurs de [Rajasekaran et VijayalakshmiPai, 2011] ont proposé une tolérance aux fautes de système autonome. Les résultats expérimentaux démontrent que le système proposé peut traiter divers défauts de logiciel pour des applications de serveur dans un environnement virtualité en Cloud.

Dans [Zhao et al, 2010], les auteurs ont proposé un mécanisme adaptatif pour la distribution des répliques pour une tolérance efficace aux pannes dans le Cloud computing, qui peut être efficacement utilisé pour atteindre une disponibilité de données de plus haut niveau. L'instrument de distribution de réplique proposé reconnaît la machine à prendre la sauvegarde ou à faire la récupération dans la situation des données de nuage néglige de charger sur les machines clientes finales.

Les auteurs de [Malik et Huet, 2011] ont proposé une méthode combinant les algorithmes EIPR et SBA pour l'ordonnancement des tâches et le processus de réplication dans le nuage avec des performances efficaces et efficientes.

Dans [Singh et al, 2013], la technique proposée par les auteurs peut fournir de meilleures performances en termes de précision et de vitesse de détection, ce qui est essentiel pour le système de Cloud.

Les auteurs de [Arunkumar et Kesavamoorthi, 2016] ont proposé une approche de tolérance aux pannes optimisée où un méthode est conçu pour tolérer les fautes basées sur la fiabilité de chaque nœud de calcul (machine virtuelle) et peut être remplacé si la performance n'est pas optimale. Les résultats obtenus suggèrent une bonne performance de notre méthode par rapport aux approches actuelles existantes.

A notre connaissance, ils n'existent pas encore de travaux qui utilisent les réseaux de neurones pour prédire les pannes au niveau de la couche matérielle du Cloud. Dans la section qui suit, nous expliquons en détail notre contribution.

3. Méthode Proposée

L'un des principaux canons du Cloud Computing est le paradigme «en tant que service» dans lequel certains services sont offerts par un fournisseur de services à un utilisateur. Ce service peut également être classé en fonction du domaine d'application de son déploiement [Muijnck.H and Hons, 2011], le Cloud computing possède trois couches principales: logiciel en tant que service (SaaS), plateforme en tant que service (PaaS), infrastructure en tant que service (IaaS).Infrastructure en tant que service (IaaS), l'utilisateur est livré avec la capacité de traitement, de stockage et tous les logiciels dont ils ont besoin pour exécuter et le système d'exploitation qu'ils sélectionnent sur l'infrastructure de Cloud. L'utilisateur ne contrôle pas l'infrastructure du Cloud, mais les composants réseau tels que le pare-feu hôte, le stockage, les systèmes d'exploitation et les applications déployées sont contrôlés par le consommateur [Yang et Ma, 2008], IaaS est mentionné infrastructure en tant que service. Il s'agit d'un méthode de distribution dans lequel une organisation sous-traite l'équipement utilisé pour soutenir les opérations, y compris le stockage, le matériel, les serveurs et les composants de réseau [Karahroudy, 2011].

Notre technique de prédiction de tolérance aux pannes est appliquée dans cette couche (IaaS). La tolérance aux pannes est l'action de recherche des fautes et d'affaiblissement dans un système. En cas de panne ou de panne matérielle ou logicielle, le système doit également fonctionner correctement. Les échecs doivent être manipulés de manière dynamique pour un bon Cloud Computing. Nous nous intéressons dans notre travail à la tolérance aux pannes au niveau de la couche matérielle du Cloud en utilisant des réseaux de neurones probabilistes selon la technique proactive. Cette méthode consiste à éviter un effort supplémentaire pour récupérer les tâches échouées, en prédisant le défaut auparavant et les remplacer par d'autres parties actives [Tchana et al, 2012].

La figure 3.1 illustre l'architecture générale de notre système qui comprend deux composants essentiels qui sont : le Cloud et le réseau de neurones utilisé. [BEZZA.Y Hioual.O, 2018]

- Le Cloud Computing est une plate-forme qui rend le traitement et le stockage disponibles aux utilisateurs finaux en tant que services. Dans notre travail, nous nous intéressons à la prédiction au niveau de la couche matérielle du Cloud, en supposent que cette couche contient un ensemble de processeurs de machine différentes qui travaille avec se Cloud.

- Réseau neuronal artificiel (ANN) : Un réseau de neurones est une structure composite qui contient une collection de neurones interconnectés qui offre une alternative très excitante pour la réponse aux problèmes complexes et autres application qui peut jouer un rôle important dans le domaine de l'informatique [Kumar1 et Sharma, 2014], réseau dans le réseau neuronal signifie l'interconnexion entre les neurones présent dans diverses couches d'un système. Chaque système est basé sur une couche d'entrée et de sortie et une ou plusieurs couches cachées. La couche d'entrée à des neurones d'entrée qui transfèrent des données via des synapses à la couche cachée, et de même la couche cachée transfère ces données à la couche de sortie via plus de synapses. Les synapses stockent des valeurs appelées pondérations qui les aident à manipuler l'entrée et la sortie de différentes couches [Mehtani.R, 2011].

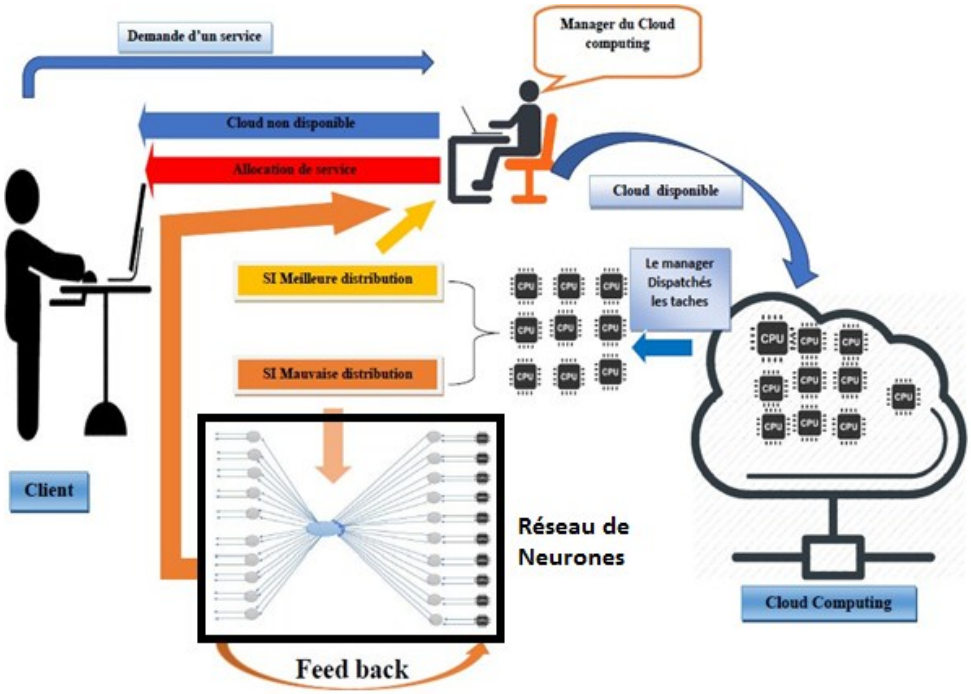

Figure 3.1 : Architecture de la méthode proposée [BEZZA.Y, HIOUAL.O, 2018]

4. Fonctionnalité du méthode proposée

Dans notre travail, nous nous intéressons à la prédiction au niveau de la couche matérielle du Cloud, on suppose que cette couche contienne un ensemble de processeurs différents qui fonctionnent avec ce dernier, la prédiction est faite en fonction du temps d'exécution et de la probabilité de tombe en panne. Initialement, lorsque l'utilisateur demande un service (dans notre cas, on suppose que les tâches sont indépendantes et unitaires),

Si le Cloud est libre, le gestionnaire du Cloud dispatches ces tâches aux différentes machines. Donc, si un processeur a un temps d'exécution plus long et beaucoup de tâches, sa probabilité de tombé en panne est élevée, et de même si un processeur a un temps d'exécution plus petit et beaucoup de tâches de sorte que sa probabilité de tombé en panne est aussi plus élevée. La probabilité que le Cloud ne répond pas aux services d'utilisateur est élevé parce que ces processeurs vont probablement tombé en pannes.

Pour soulever ce problème et prévoir les défaillances au niveau de la couche matériel du Cloud. Nous avons utilisé un réseau de neurones probabiliste avec apprentissage supervisé, la couche d'entrée possède comme paramètres les caractéristique du processeur (temps d'exécution, probabilité de tombé en panne) la couche de sortie est le temps d'exécution optimal et la charge final de chaque processeur. [BEZZA.Y, HIOUAL.O, 2018]

La couche d'entrée transfère les données à la couche cachée, cette dernière applique une fonction probabiliste sur ces données et transfère le résultat à la couche de sortie, le comportement souhaité est le temps d'exécution optimal qui est calculé par l'algorithme optimal de [Legrand et Robert, 2005].

Si le temps d'exécution optimal est plus grand que le temps d'exécution du processeur aucune panne ne sera dans le futur, mais si le temps d'exécution optimal est inférieur au temps d'exécution du processeur nous faisons une propagation en arrière en utilisant la technique proactive jusqu'à l'obtention du meilleurs temps ainsi qu'une meilleure distribution de charge.

L'apprentissage supervisé est une phase dans le développement d'un réseau de neurones pendant laquelle le comportement du réseau est modifié jusqu'à l'obtention du comportement désiré, Cela signifie modifier le poids des connexions entre les neurones et pour modifier les poids, nous utilisons un algorithme qui calcule la charge de chaque processeur.

On suppose que les tâches soient indépendantes et unitaires, nous avons utilisé deux algorithmes : un algorithme d'équilibrage de charge statique modifier [Dongarra et al, 2007]. Basé sur un algorithme optimal qui existe déjà [Legrand et Robert, 2005].

L'algorithme optimal repose sur le principe de distribution d'un ensemble de tâches indépendantes sur un ensemble de processeurs en respectant le temps d'exécution de chacun, qui vise à affecter au processeur ayant la vitesse de traitement la plus rapide un nombre maximum de tâches à exécuter afin d'obtenir un temps d'exécution optimal et ainsi un meilleur équilibrage de da charge pour prédire les pannes c'est à dire la capacité d'un algorithme pour effectuer l'équilibrage de la charge uniforme en dépit de nœud arbitraire ou défaillance de la liaison. L'équilibrage de charge devrait être une bonne technique à tolérance de pannes.

Le gestionnaire d'équilibrage de charge distribue les tâches sans connaître la charge actuelle sur les processeurs système. L'algorithme utilisé ici doit donc être capable de calculer les charges à l'avance, afin de permettre un équilibrage de charge efficace. Ainsi, la charge produite par une tâche doit être fournie par la tâche elle-même ou par d'autres sources. En outre, les différentes capacités des processeurs ou des nœuds de réseau doivent être connues.

4.1.Modélisation avec des diagrammes UML

UML (Unified Modeling Language): se définit comme un langage de modélisation graphique et textuel destiné à comprendre et décrire des besoins, spécifier, concevoir des solutions et communiquer des points de vue.

UML unifie à la fois les notations et les concepts orientés objet. Il ne s'agit pas d'une simple notation, mais les concepts transmis par un diagramme ont une sémantique précise et sont porteurs de sens au même titre que les mots d'un langage. UML permet de modéliser de manière claire et précise la structure et le comportement d'un système indépendamment de toute méthode où de tout langage de programmation.

4.1.1 Diagramme de séquences du scénario de la méthode proposée

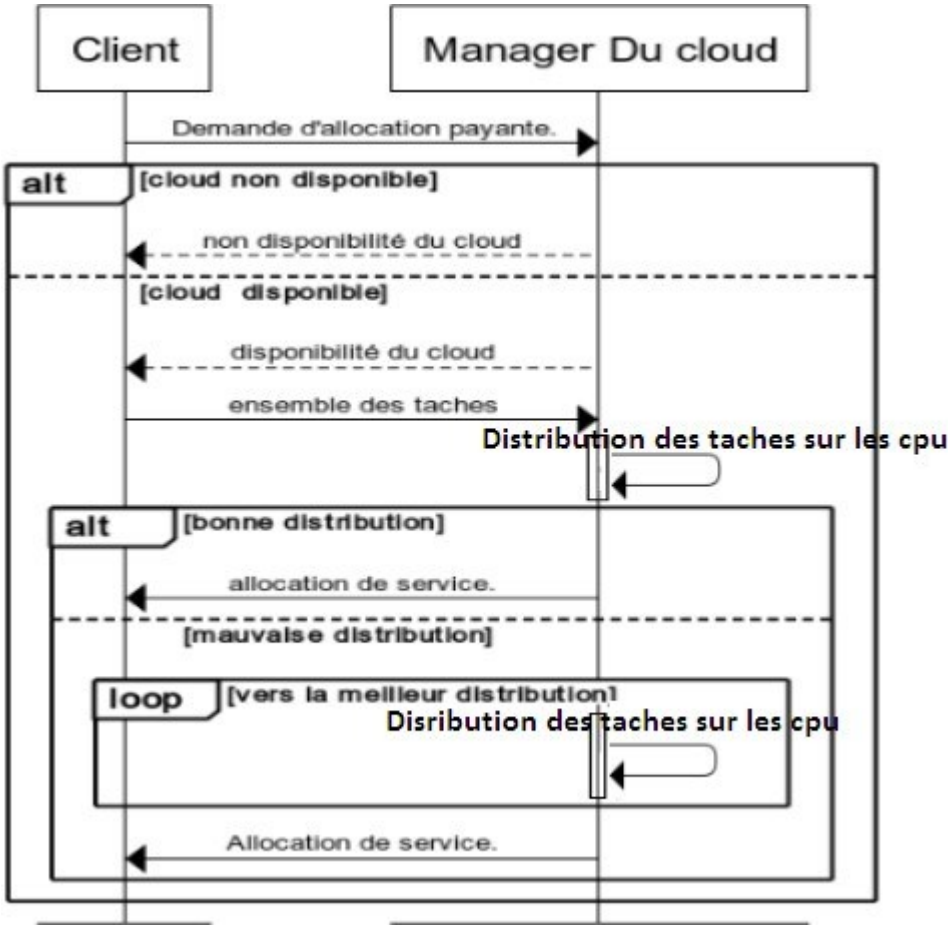

Figure 3.1 : Diagramme de séquences de la méthode proposée

Le scénario :

1. Le client passe une demande d'allocation payante d'un service Cloud.
2. Le manager du Cloud vérifie la disponibilité .
3. Si le Cloud n'est pas disponible le client va chercher un autre Cloud.
4. Si le Cloud est disponible le client envoie les taches au manager de Cloud.
5. Le manager dispatches les taches sur les CPU.
6. On fait une étude sur la distribution initiale du manager
7. Si la distribution est bonne on envoie le service au client

8. Sinon on redistribué les taches on utilisant l'algorithme de l'équilibrage de charge suivant :

Algorithme 1 : Allocation Optimal [Legrand et Robert, 2005]

Distribution ($(t_1,….,t_p)$,M)

{Initialisation: calculer les valeurs de ni tel que $n_i \times t_i \approx$constant et n_1+ + $n_m \leq N$}

1-Pour i=1 à m :

2- $n_i = \left\lfloor \frac{\frac{1}{t_i}}{\sum_{i=1}^{m} \frac{1}{t_i}} \times N \right\rfloor$

{Itérativement incrémenter ni qui minimise le temps d'exécution tant que $\sum_{i=1}^{m} ni < N$ **}**

-

3- Tant que $\sum_{i=1}^{m} ni < N$

4- Trouver kϵ{1,…..,N} telque $t_k * (n_k+1) =$min $\{t_i*(n_i+1)\}$

5- $n_k = n_k+1$

6- retourner $(n_1+n_2+…….+n_K)$

M : Le nombre des processeurs.

N : Le nombre des taches.

n_i : La charge de I éme processeur.

t_i : Le temps d'exécution optimale de I éme processeur.

9. Puis on utilisant un deuxième algorithme pour minimiser le temps d'exécution optimal .

Algorithme 2 : Algorithme d'équilibrage de charge statique modifier [Dongarra et al, 2007].

Distribution ((t1,….,tp),M)

```
Distribution ((b1,….,bp),M)
entrée: qϵ [0, 1[
Calculer Top1= q *Top en utilisant l'algorithm1
Trier les processeur en augmentant t1[i]
Trier les processeurs en diminuant b[i]
Trier les processeur en augmentant t1[i]* b[i]
 X → 0
pour i=1:m
    Si ( X < N)
        n[i] → min( N-X, int (Top/t[i])
    Sinon
        n[i]=0
X=X+n[i]
```

n[i] : La charge de iéme processeur.

t[i] : Le temps d'exécution de iéme processeur

b[i] : La probabilité de tomber en panne.

Top:Le temps d'exécution optimale.

10. Après l'utilisation des ces deux algorithme le manager alloue le service au client.

5. Conclusion

Le travail présenté dans ce chapitre concerne une technique probabiliste de prédiction de panne utilisant des réseaux neuronaux probabilistes supervisés dans le Cloud computing et la technique proactive, nous avons fait la prédiction au niveau de la couche matérielle du Cloud, nous avons utilisé deux algorithmes d'équilibrage de charge au niveau de la couche intermédiaire du réseau neuronal afin d'avoir un meilleur équilibrage de charge pour prédire les pannes,

Le chapitre suivant présente les aspects implémentation et mise en œuvre de notre méthode.

Chapitre 04

Implémentation de la méthode proposée

1. Introduction

Dans le chapitre précédent, nous avons proposé notre méthode de tolérance aux pannes basé sur les réseaux de neurones probabilistes et la technique proactive.

Dans ce chapitre on va essayer de donner un bref aperçu sur quelques outils utilisés dans la réalisation de notre application, puis présenter les résultats de notre travail et finir par une petite conclusion.

2. Environnement et outils utilisés

Nous présentons dans cette partie l'environnement de programmation utilisée, ainsi, tous les outils.

2.1. Eclipse

Eclipse est un projet, décliné et organisé en un ensemble de sous projets de développements logiciels, de la Fondation Eclipse visant à développer un environnement de production de logiciels libre qui soit extensible, universel et polyvalent, en s'appuyant principalement sur Java.

Les principaux modules fournis en standard avec Eclipse concernent Java mais des modules sont en cours de développement pour d'autres langages notamment C++, Cobol, mais aussi pour d'autres aspects du développement (base de données, conception avec UML, ...). Ils sont tous développés en Java soit par le projet Eclipse soit par des tiers commerciaux ou en open source. dont le but est de fournir une plate−forme modulaire pour permettre de réaliser des développements informatiques.

Eclipse possède de nombreux points forts qui sont à l'origine de son énorme succès dont les principaux sont :

- Une plate−forme ouverte pour le développement d'applications et extensible grâce à un mécanisme deplug−ins

- Plusieurs versions d'un même plug−in peuvent cohabiter sur une même plate−forme.

- Un support multi langage grâce à des plug−ins dédiés : Cobol, C, PHP, C# , ...

- Support de plusieurs plate−formes d'exécution : Windows, Linux, Mac OS X, ...

2.2. Java

Le langage Java est un langage généraliste de programmation synthétisant les principaux langages existants lors de sa création. Il permet une programmation orientée-objet (à l'instar de SmallTalk et, dans une moindre mesure, C++), modulaire (langage ADA) et reprend une syntaxe très proche de celle du langage C. Outre son orientation objet, le langage Java a l'avantage d'être modulaire (on peut écrire des portions de code génériques, c-à-d utilisables par plusieurs applications), rigoureux (la plupart des erreurs se produisent à la compilation et non à l'exécution) et portable (un même programme compilé peut s'exécuter sur différents environnements). En contre-partie, les applications Java ont le défaut d'être plus lentes à l'exécution que des applications programmées en C par exemple.[Gauthier.P, Laurent.V].

Voici les caractéristiques de Java en quelques mots :

1- Java est un langage de programmation moderne développé par Sun Microsystems, aujourd'hui racheté par Oracle.

2- Une de ses plus grandes forces est son excellente portabilité : une fois votre programme créé, il fonctionnera automatiquement sous Windows, Mac, Linux, etc.

3- On peut faire de nombreux types de programmes avec Java .

4- Des applications, sous forme de fenêtre ou de console .

5- Des sites web dynamiques, avec J2EE (Java 2 Enterprise Edition, maintenant JEE) ;

- On peut distinguer trois grandes phases dans la vie d'un code Java :

- La phase d'écriture du code source, en langage Java ;

- La phase de compilation de votre code ;

- La phase d'exécution. [Cyrille.H,2011]

3. Implémentations

Notre méthode est constitué de deux parties, la première partie est la partie Cloud et la deuxième partie est la partie réseaux de neurones. Notre Cloud est composé de 10 processeurs de différentes technologies et générations, aussi il y'a un manager qui va gérer l'allocation de ces

processeurs. Ces derniers sont les nœuds de notre réseau de neurones dont chaque neurone a deux paramètres le temps d'exécution et la probabilité de tomber en panne , la couche caché du réseau travaille avec deux algorithmes d'équilibrage de charge .

Figure 4.1 : Fenêtre principale

La figure 4.1 montre L'interface Principale de notre application

Quand on appuit sur le bouton « Gestionnaire » une petite fenêtre s'affiche pour faire entrer le mot de passe et le nom utilisateur.

Figure 4.2 : Fenêtre Login

Si le nom d'utilisateur et le mot de passe sont corrects donc on passe à la fenêtre ou le client doit entrer le nombre des taches sinon un message d'erreur s'affiche.

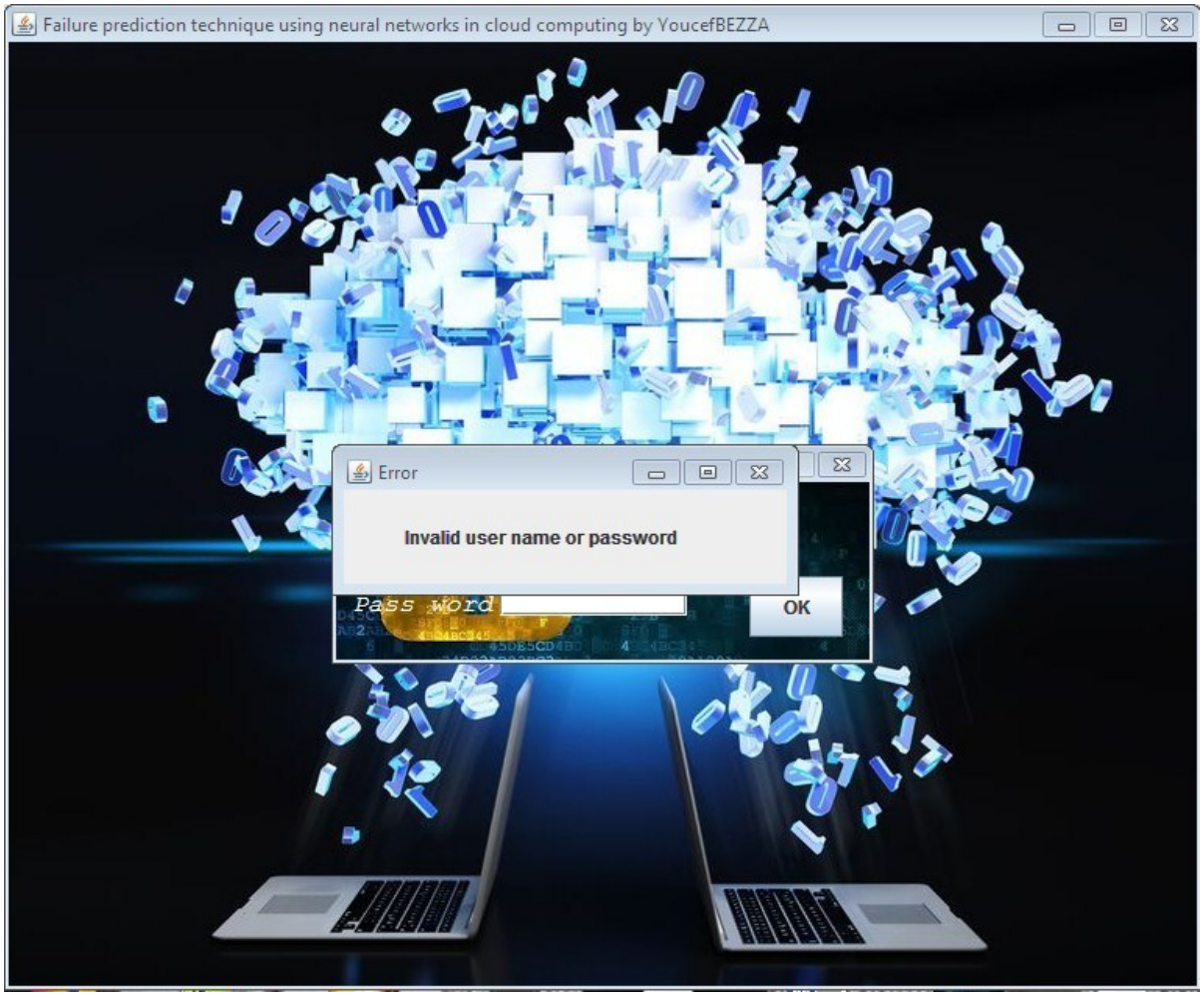

Figure 4.3 : Message d'erreur

Les processeurs utilisés dans notre étude sont présentés dans le tableau suivant :

Numéro de CPU	**Caractéristiques**	**Temps d'exécution(×〖10〗^(-4)) Secondes**
01	I7|2.20GHZ	0.88057
02	Duel_core|2.10 GHZ	2.17386
03	Duel_core|2.00 GHZ	3.86723
04	I3|2.00GHZ	2.82731
05	Celeron|2.10 GHZ	4.64548
06	I7|2.10GHZ	2.72795
07	Duel_core|2.00GHZ	3.85982

08	I5|	2.29233
09	I5|	1.85128
10	I3|2.40 GHZ	4.03259

Tableau 4.1 : Liste des processeurs

La première colonne représente les numéros de processeur, la deuxième colonne représente la technologie et la fréquence de chaque processeur, la dernière colonne représente le temps d'exécution de chaque processeur, ce dernier est calculé par un algorithme proposé par nous même. On n'est pas dans un réseau pour cela on a exécuté cet algorithme sur les 10 CPU un par un, le temps d'exécution est en secondes, le code java suivant présente le code source de l'algorithme. Pour exécuter cet algorithme il faut mettre une tache simple entre les deux instructions :

- longstarttime:=system.nanoTime() : qui marque le temps de début d'exécution.
- longendtime=system.nanoTime(): qui marque le temps de fin d'exécution.

Le temps d'exécution de la tache est donc la différence entre le temps de fin d'exécution et le temps de début d'exécution.

```
Package Casestudy;
Public class Runtime {
Static int x=5;
        static int y=6;
        static int s;
        public static void main(String[] args){
                long tempsDebut = System.nanoTime();
                s=x+y;
                System.out.println(s);
                Long tempsFin = System.nanoTime();
                Double seconds = (tempsFin - tempsDebut) / 1e9;
                System.out.println("Opération effectuée en: "+ seconds + " secondes.");
                }}
```

Le client fait une demande d'allocation payante , le manager de Cloud vérifie la disponibilité des processeurs et il envoie une réponse au client .si la réponse est : « non » le client cherche un

autre Cloud, si la réponse est oui ,le client envoie les taches au manager ,sachant que les taches sont indépendantes et unitaires, et le nombre entre 1000 et 10000.

Figure 4.4 :Fenêtre pour faire entrer le nombre des taches

Premièrement le manager dispatche les taches on utilisant la fonction Math.random() ; le nombre des taches est entre 1000 et 10000, si les taches affectées à chaque processeurs sont dans l'interval donc on a pas besoin d'utiliser le réseaux de neurone.

Ce code montre comment distribuer les taches sur les différents processeurs :

```
Int re;
Double ta=10000;
        Double temp=0;
        doublete;
        for(i=0;i<10;i++){
double x= Math.random()*ta;
        re=(int)x;
        M1[i][2]=re;
        M1[i][2]=(int)M1[i][2];
        temp=M1[i][2];
```

```
ta=te;}
```

Sinon en doit utiliser le réseaux de neurone pour prédire la panne au niveau du Cloud , chaque neurone représente un processeur, les neurones de la couche d'entrée travaillent aves deux paramètres qui sont le temps d'exécution et la probabilité de tombé en panne.

La couche intermédiaire du notre réseau travaille avec deux algorithmes d'équilibrages de charge. Le premier algorithme est celui de (Legrand and Robert ,2005), cet algorithme travail avec un seul paramètre qui est le temps d'exécution. Initialement on calcul la charge de chaque processeur suivant son temps d'exécution c-à-d calculer le nombre de taches affecter a chacun. En suite on calcul la somme des taches affectées et on la compare avec la charge initial, si on obtient une égalité qui est impossible dans ce cas puisque dans la formule utilisé dans l'algorithme en prend la partie entière seulement, donc en recommence l'opération en affectant les taches restantes aux processeur qui vérifie cette formule (min { t_i (+1)). × n_i

Enfin en arrivant au résultat $\sum_{i=1}^{m} n_i$<N) en calcul le temps optimal qui correspond au processeur le plus long. .(voir le code ci dessous)

```
for(int a=0;a<1;a++){
for( i=0; i<m; i++){
t[i]= M1[i][1];}
double som1=0;
for( i=0;i<m;i++){
som1=(1/t[i])+som1;}
double n[]=new double [m];
for( i=0;i<m;i++){
double o=((1/t[i])/som1)*N;
double p=(int)o;
n[i]=p;}
double som2=0;
for( i=0;i<m;i++){
som2=n[i]+som2;}
while (som2<N){
```

```
for( i=0;i<m;i++){
tab[i] = t[i]*(n[i]+1);
double min=tab[0];
intk=0;
for( i=1;i<m;i++){
if(tab[i]<min){
min=tab[i];
k=i;} }
n[k]=n[k]+1;
som2=0;
for( i=0;i<m;i++){
som2=n[i]+som2;}}
for( i=0;i<m;i++){
d[i]=n[i]*t[i];}
double max2=d[0];
double Top2=max2;
for(intW=1;W<m; W++){
if(d[W]>max2){
max2=d[W];
Top2=max2;
s=W;}}
Top=Top2;}
```

Et pour optimiser le temps d'exécution plus on a utilisé un deuxième algorithme modifié cité dans les travaux de « Dongarra et al ,2007 » , Cet algorithme a aussi pour but l'équilibrage de charges entre plusieurs processeurs, il se base sur deux paramètres qui sont : le temps d'exécution et la probabilité de tombé en panne d'un processeur ainsi que le résultat de l'algorithme optimal cité ci-dessus.

Initialement on tri les processeur par ordre croissant selon le produit (b[i]t[i]), puis en calcul le temps d'exécution optimal qui se base sur le résultat de l'algorithme précédant suivant la formule (Top1 = q Top) avec 'q' nombre aléatoire compris entre 1.1 est + ∞. Puis on refait le même travail de l'algorithme précédant qui est de calculer la charge de chaque processeur.

```
for(i=0; i<m; i++)
{ t1[i]= M1[i][1];}
for( i=0; i<m; i++)
{b[i]= M1[i][3];}
for( i=0; i<m; i++) {
for(j=i+1; j<m; j++){
if(b[j]>b[i]){
double h=b[i];double h1;
b[i]=b[j];
b[j]=h;
h1=M1[i][3];M1[i][3]=M1[j][3];M1[j][3]=h1;}}}
doubleg2;
doubleg3,g4,g5;
for( i=0; i<m; i++)
{ pro[i]=t1[i]*b[i]; }
for( i=0; i<m; i++) {
for( j=i+1; j<m; j++){
if(pro[j]<pro[i]){
doubleh=pro[i];
pro[i]=pro[j];
pro[j]=h;
doubleg=t1[i];
t1[i]=t1[j];
t1[j]=g;
g2=M1[i][1];
M1[i][1]=M1[j][1];M1[j][1]=g2;
doublep=b[i];
b[i]=b[j];
b[j]=p;
g5=M1[i][0];M1[i][0]=M1[j][0];M1[j][0]=g5;
g3=M1[i][3];M1[i][3]=M1[j][3];M1[j][3]=g3;
```

```
g4=M1[i][2];M1[i][2]=M1[j][2];M1[j][2]=g4;}} }
double x;
double Top1;
double q= Math.random();
Top1=Top*q;
x=0;
for( i=0; i<m; i++) {
if (x<N){
double z=Top1/(t1[i]);
double z2=(int)z;
double z1=N-x;
if(z1<z2)
{n1[i]=z1;}
else
{n1[i]=z2;}}
else
{n1[i]=0;}
x=x+n1[i];}
double som4=0;
for( j=0; j<m; j++)
{ som4= som4+n1[j]; }
while (som4<N){
for( i=0;i<m;i++){
tab1[i] = t1[i]*(n1[i]+1); }
double min=tab1[0];
int k=0;
for( i=1;i<m;i++){
if(tab1[i]<min){
min=tab1[i];
k=i;} }
n1[k]=n1[k]+1;
```

```
som4=0;
for( i=0;i<m;i++){
som4=n1[i]+som4;}}
for( i=0;i<m;i++){
d1[i]=n1[i]*t1[i]; }
doublemax1=d1[0];
Top1=max1;
for( i=1;i<m;i++){
if(d1[i]>max1){
max1=d1[i];
Top1=max1;s=i; }}
for( i=0;i<m;i++)
{M2[i][2]=n1[i];
M2[i][4]=Top1;
M2[i][0]=M1[i][0];
M2[i][1]=M1[i][1];
M2[i][3]=M1[i][3];}
for(i=0;i<M2.length;i++){
        for(u=0;u<5;u++){
                System.out.print(M2[i][u]);
                System.out.print("     ");}
                        System.out.print("\n");}}
```

Après exécution de ces deux algorithmes on obtient deux résultats qui sont la charge de chaque processeur et le temps d'exécution optimal, ces deux résultats sont les données de la couche de sortie de notre réseau de neurone.

Puis on compare les résultats obtenus, si le temps d'exécution de chaque processeur est inferieur au temps d'exécution optimal on arrête, sinon on continu notre apprentissage jusqu'a le contraire

Comme nous avons déjà dit précédemment on a utilisé la technique proactive pour la prédiction des pannes, le principe de cette technique est inclus avec le principe de l'apprentissage de notre

réseau de neurone c-à-d tant que le temps d'exécution de chaque processeur est supérieur a celui du temps optimal on refait l'apprentissage donc on fait des retours en arrière.

Notre code source pour réaliser la méthode proposée est le suivant :

```
import java.io.*;
importjava.lang.Math;
importjava.util.Random;
public class technique {
    public static void main (String args[]){
                int i=10;
                int j=5;
                int u=6;
                int m=10;
                int N=100;
                double t1[] = newdouble [m];
                double t[] = newdouble [m];
                double tab[] = newdouble [m];
                double tab1[] = newdouble [m];
                double d[] = newdouble [m];
                double d1[] = newdouble [m];
                double n1[] = newdouble [m];
                double b[] = newdouble [m];
                double pro[] = newdouble [m];
                double Top=0;
                        double[][]M1=newdouble [i][j];
                        double[][]M2=newdouble [i][u];
                                int s=0;
                                //numdecpu
                                        for( i=0;i<10;i++){
                                                M1[i][0]=s;
                                                        s++; }
                        double tmp;
```

```
                    //prob de tomber en panne
                    for(i=0;i<10;i++){
                            M1[i][3]=Math.random();
                            tmp=M1[i][3]*100;
                            M1[i][3]=tmp;
                            }
                    //charge initialeenentre 0et 100
                            Int re;
                            double ta=100;
                            double temp=0;
                            double te;
                            for(i=0;i<10;i++){
                            double x= Math.random()*ta;
                            re=(int)x;
                            M1[i][2]=re;
                            M1[i][2]=(int)M1[i][2];
                            temp=M1[i][2];
                            }

            M1[0][1]=0.88057;
            M1[1][1]=1.85128;
            M1[2][1]=2.17386;
            M1[3][1]=2.29233;
            M1[4][1]=2.72795;
            M1[5][1]=2.82731;
            M1[6][1]=3.85982;
            M1[7][1]=3.86723;
            M1[8][1]=4.03259;
            M1[9][1]=4.64548;
for(inta=0;a<1;a++){
for( i=0; i<m; i++){
```

```
t[i]= M1[i][1];}
double som1=0;
for( i=0;i<m;i++){
som1=(1/t[i])+som1;}
double n[]=new double [m];
for( i=0;i<m;i++){
double o=((1/t[i])/som1)*N;
double p=(int)o;
n[i]=p;}
double som2=0;
for( i=0;i<m;i++){
som2=n[i]+som2;}
while (som2<N){
for( i=0;i<m;i++){
tab[i] = t[i]*(n[i]+1);
double min=tab[0];
int k=0;
for( i=1;i<m;i++){
if(tab[i]<min){
min=tab[i];
k=i;} }
n[k]=n[k]+1;
som2=0;
for( i=0;i<m;i++){
som2=n[i]+som2;}}
for( i=0;i<m;i++){
d[i]=n[i]*t[i];}
double max2=d[0];
double Top2=max2;
for(int W=1;W<m; W++){
if(d[W]>max2){
```

```
max2=d[W];
Top2=max2;
s=W;}}
Top=Top2;}
for( i=0; i<m; i++)
{ t1[i]= M1[i][1];}
for( i=0; i<m; i++)
{b[i]= M1[i][3];}
for( i=0; i<m; i++) {
for(j=i+1; j<m; j++){
if(b[j]>b[i]){
double h=b[i];double h1;
b[i]=b[j];
b[j]=h;
h1=M1[i][3];M1[i][3]=M1[j][3];M1[j][3]=h1;}}}
double g2,g3,g4,g5;
for( i=0; i<m; i++)
{ pro[i]=t1[i]*b[i]; }
for( i=0; i<m; i++) {
for( j=i+1; j<m; j++){
if(pro[j]<pro[i]){
double h=pro[i];
pro[i]=pro[j];
pro[j]=h;
double g=t1[i];
t1[i]=t1[j];
t1[j]=g;
g2=M1[i][1];
M1[i][1]=M1[j][1];M1[j][1]=g2;
double p=b[i];
b[i]=b[j];
```

```
b[j]=p;
g5=M1[i][0];M1[i][0]=M1[j][0];M1[j][0]=g5;
g3=M1[i][3];M1[i][3]=M1[j][3];M1[j][3]=g3;
g4=M1[i][2];M1[i][2]=M1[j][2];M1[j][2]=g4;}} }
double x;
doubleTop1;
double q= Math.random();
Top1=Top*q;
x=0;
for( i=0; i<m; i++) {
if (x<N){
double z=Top1/(t1[i]);
double z2=(int)z;
double z1=N-x;
if(z1<z2)
{n1[i]=z1;}
else
{n1[i]=z2;}}
else
{n1[i]=0;}
x=x+n1[i];}
double som4=0;
for( j=0; j<m; j++)
{ som4= som4+n1[j]; }
while (som4<N){
for( i=0;i<m;i++){
tab1[i] = t1[i]*(n1[i]+1); }
double min=tab1[0];
int k=0;
for( i=1;i<m;i++){
if(tab1[i]<min){
```

```
min=tab1[i];
k=i;} }
n1[k]=n1[k]+1;
som4=0;
for( i=0;i<m;i++){
som4=n1[i]+som4;}}
for( i=0;i<m;i++){
d1[i]=n1[i]*t1[i]; }
double max1=d1[0];
Top1=max1;
for( i=1;i<m; i++){
if(d1[i]>max1){
max1=d1[i];
Top1=max1;s=i; }}
System.out.println(s+1);
for( i=0;i<m;i++)
{M2[i][2]=n1[i];
M2[i][4]=Top1;
M2[i][0]=M1[i][0];
M2[i][1]=M1[i][1];
M2[i][3]=M1[i][3];}
for(i=0;i<M2.length;i++){
        for(u=0;u<5;u++){
                System.out.print(M2[i][u]);
                System.out.print("     ");}
                System.out.print("\n");}
        }}}
```

4. Etude de cas

On Suppose qu'un client demande un service pour exécuter 9000 taches indépendantes et unitaire , et le Cloud est disponible .

Initialement, On entre le mot de passe et le nom d'utilisateur.(voire figure 4.5)

Figure 4.5 : Fenêtre d'identification.

On entre le nombre des taches : 9000 taches (Figure 4.6)

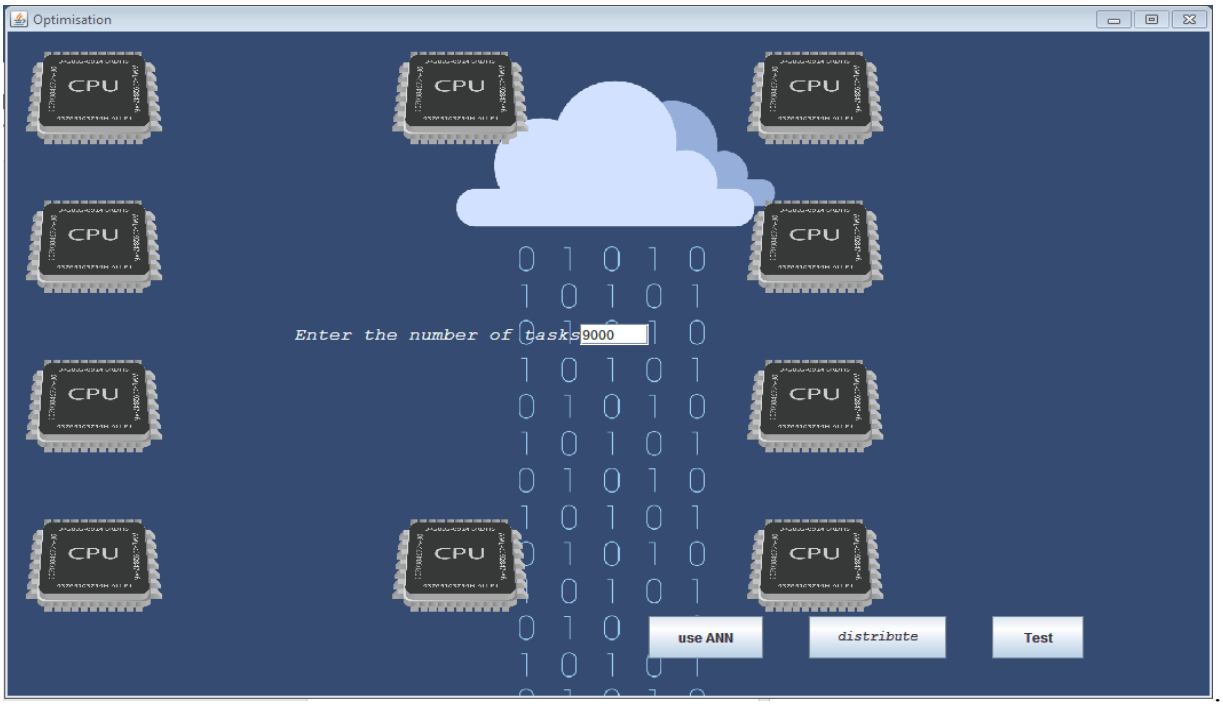

Figure 4.6 : Fenêtre d'envoie des taches

Puis on click sur le bouton distribute (Figure 4.7)

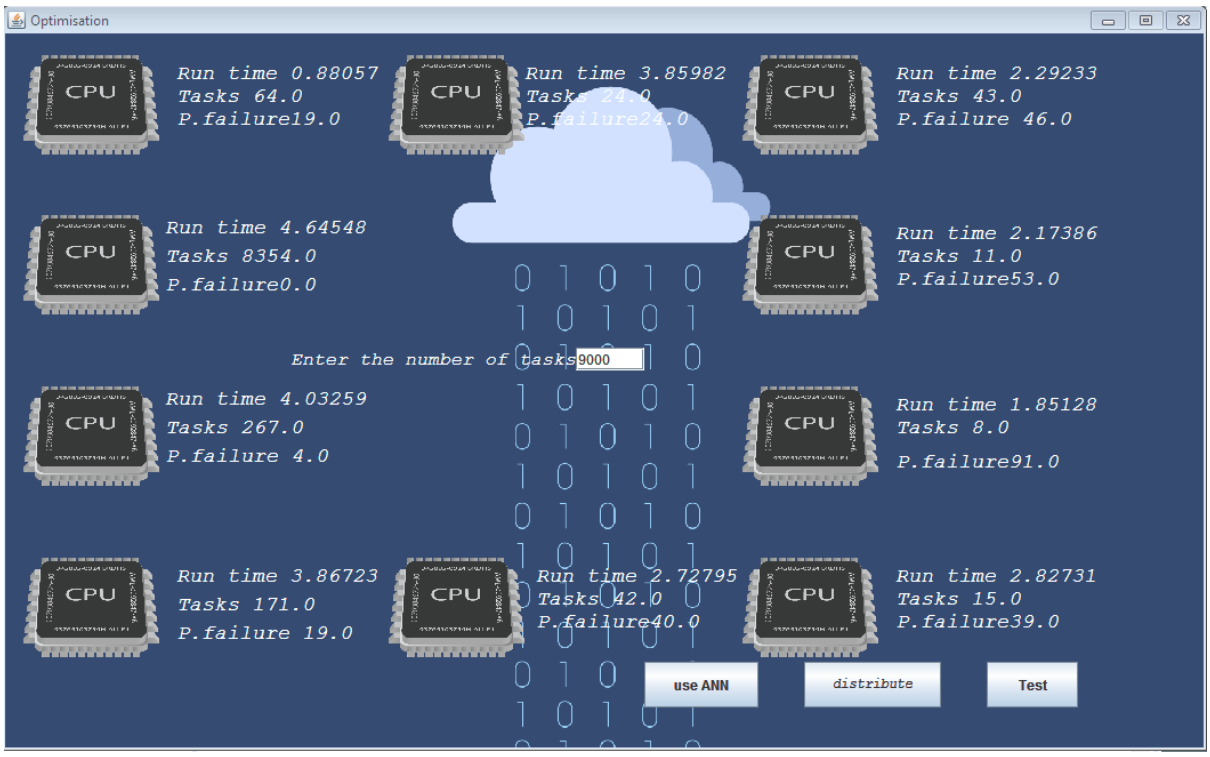

Figure 4.7 : Distribution des taches

Cette figure montre la distribution initial des charges sur les processeurs, pour vérifier la charge de chaque processeur, on click sur le bouton test,

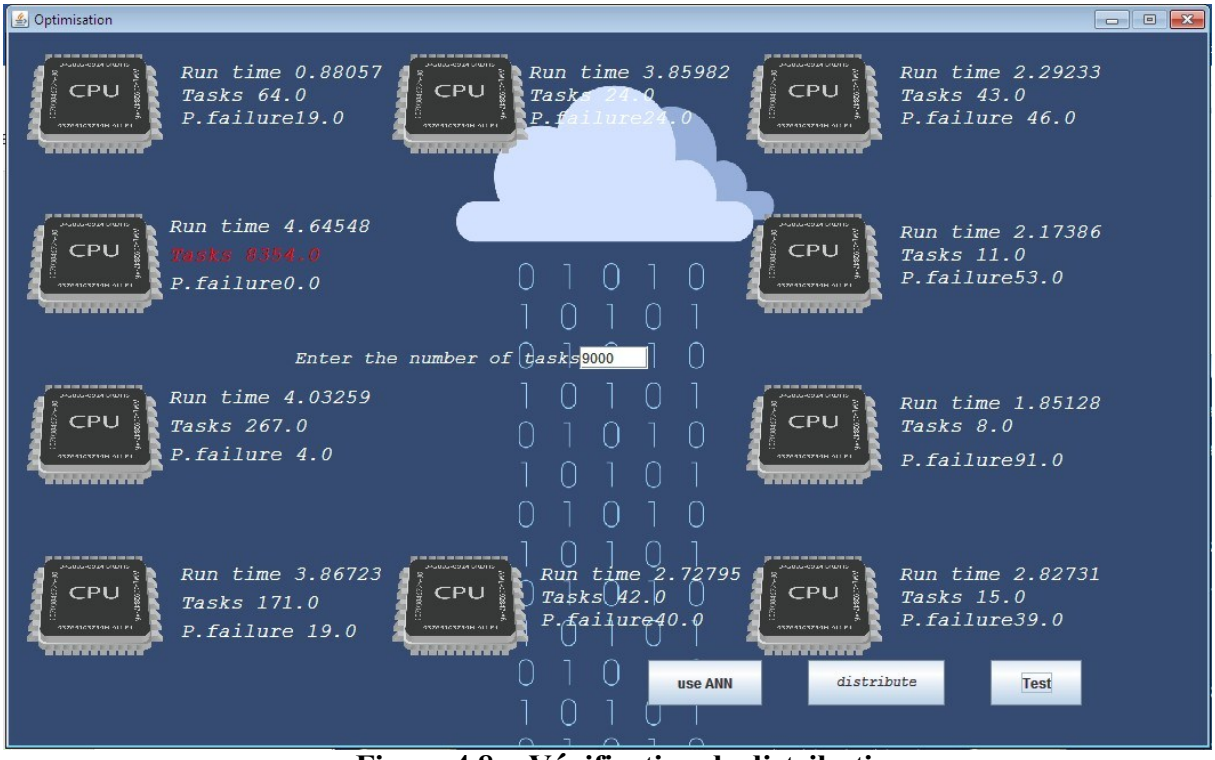

Figure 4.8 : Vérification de distribution

la couleur rouge signifie que la charge affectée au processeur est supérieure au charge maximale .(figure 4.8) pour une meilleurs utilisation du cloud en utilise le réseaux de neurone

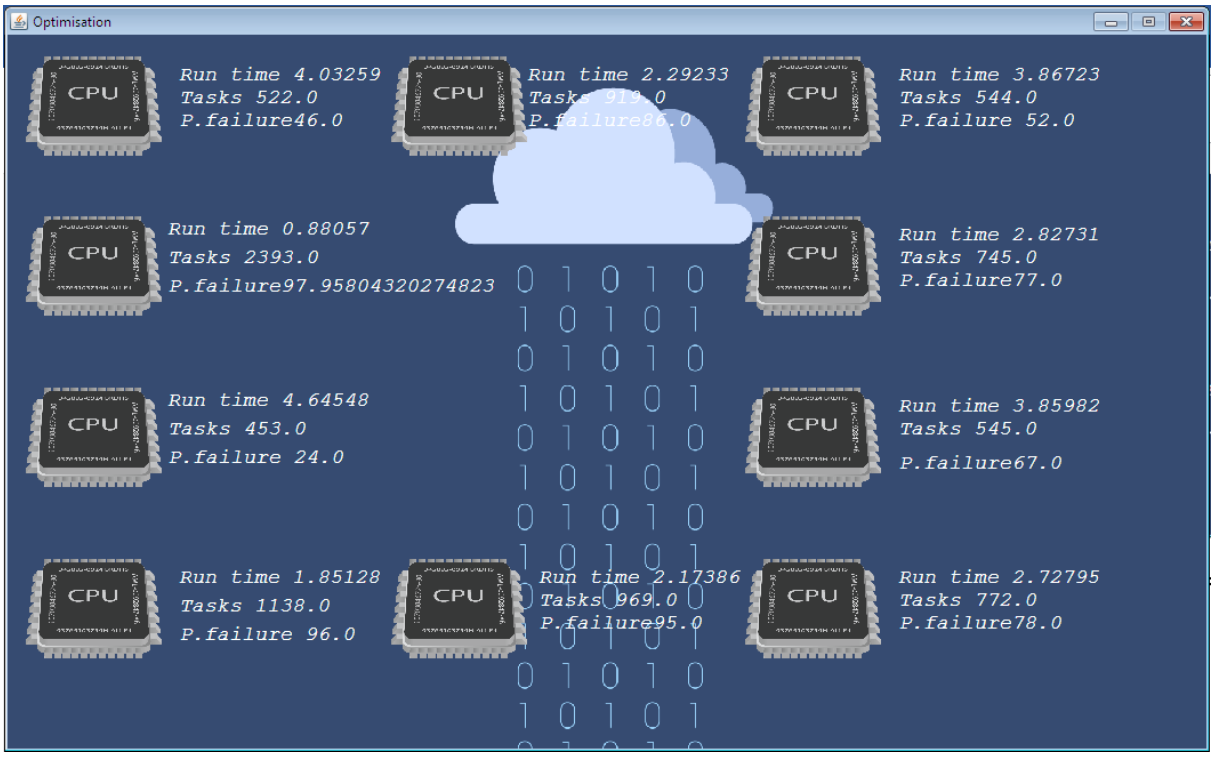

Figure 4.9 : Résultats obtenus

La meilleure distribution des taches pour garantir que le Cloud ne tombe pas en panne(figure 4.9).

5. Conclusion

La prédiction de panne est utilisée pour fournir la disponibilité et la robustesse du système lorsque le système présente une défaillance matérielle ou logicielle. Dans ce chapitre, nous avons essayé de mettre en œuvre l'ensemble des idées qui caractérise le méthode que nous avons proposé. Notre méthode a été implémentée en utilisant le langage Java. Les résultats obtenus, basés sur une étude de cas réelle, montrent que cette méthode nous donne de bons résultats.

CONCLUSION GENERALE ET PERSPECTIVES

Le Cloud Computing, est un méthode informatique récent qui a pour objectif de proposer les services informatiques sous forme de services à la demande, accessibles de n'importe où, n'importe quand et par n'importe qui. Le Cloud Computing est basé sur trois services qui ne s'adresse pas aux mêmes types d'utilisateurs : IaaS, PaaS, SaaS. Les Méthodes de déploiement de cloud sont : Cloud Privé, Cloud Hybride, Le Cloud public,le cloud hyrbid, le cloud, Le Cloud communautaire.

Dans un réseau Cloud, Il y a deux politiques de tolérance aux pannes (Fault Tolerance) disponibles, tolérance aux pannes proactive et Politique de tolérance aux pannes réactive. Basé sur ces politiques de tolérance de pannes, différentes techniques sont utilisées pour fournir la tolérance de panne. Les techniques de tolérance aux fautes réactives sont utilisées pour réduire l'impact des défaillances sur un système lorsque les défaillances se sont réellement produites. La tolérance proactive aux pannes permet de prévoir les pannes de manière proactive et de remplacer les composants suspects par d'autres composants de travail, évitant ainsi la récupération des pannes et des erreurs.

Un Réseau de neurones artificiels est un ensemble de neurones formels interconnectés permettant la résolution de problèmes complexes, grâce à l'ajustement des coefficients de pondération dans une phase d'apprentissage. Un neurone artificiel est considéré comme un élément élémentaire de traitement de l'information, il reçoit des entrées et réaliser un résultat à la sortie. Selon la topologie de connexion des neurones, on peut les classer en deux grandes catégories : réseaux non bouclés (statique ou feed forward) et réseaux bouclés (dynamique, feed back ou récurrent).

Nous avons dans ce mémoire exprimer nos recherches sur le sujet, ce qui nous guidera et nous aidera à créer notre opinion personnelle. Pour cela, Dans un premier temps, nous avons réalisé une présentation générale du paradigme de Cloud Computing, ses caractéristiques, ses méthodes de services et de déploiement, et les différents politiques de tolérances aux pannes . Dans une deuxième partie on a défini les réseaux de neurones artificiels, leurs avantages et inconvénients, les différents types , et les différentes techniques l'apprentissage. Pour la partie pratique on a commencé par introduisé, tout d'abord, la problématique et les objectifs visés par cette étude. après avoir survoler quelques travaux existants. Ensuite, nous avons présenté notre méthode . Enfin dans une dernière partie, nous avons implémenté une technique probabiliste de prédiction des pannes au niveau de Cloud computing.

Dans notre travail on a proposé une méthode de prédiction des pannes probabiliste en utilisant les réseaux de neurones artificiels, notre méthode est composé d'un réseau de neurones artificiels multi couches, et un Cloud de 10 processeurs de différents technologie et générations, on a utilisé l'apprentissage supervisé et la politique de tolérance au pannes proactive. Notre générateur nous donne des bons résultats en le testant sur des cas réels.

Comme future travail, nous envisageront d'améliorer notre méthode en utilisant des taches dépendantes et non-unitaires avec le multi Cloud.

Bibliographie:

- Akbi.K et Zehri.M , Etude et mise en place d'une solution cloud computing privé au sein de l'université deouargla,2013

- Anand, Mehrotra, Mohan et Ranka «Intelligent Control Using Neural Networks», IEEE Control System mag; Avril, pp 11-18, 1992.

- Arunkumar.B, and M. Kesavamoorthi .Task Scheduling and Seedblock Based Fault Tolerance in Cloud, Volume 11, Number 6,2016.

- Atul.J Johnson.D Kiran Murari Murthy Raju Vivek Cherian Yogesh Girikumar,OpenStack Beginner's Guide(for Ubuntu - Precise), v3.0, 7, 83 pages, May 2012.

- Baghli.L« Contribution à la commande de la machine asynchrone, l'utilisation de la logique floue, des réseaux de neurones et des algorithmes génétiques » Thèse de Doctorat, université Nancy I , 1999.

- Benoît.V ,Etude prospective des applications possibles des réseaux de neurones formels dans le traitement des données psychométriques,2001.

- Bernard.G, Application de réseaux de Neurones Artificiels a la reconnaissance automatique de caracteres manuscrit,1996.

- Bezza.Y et Hioual.O," Probabilistic failure prediction technique using neural networks in cloud computing", 12th edition of the Conference on Advances of Decisional Systems, Marrakech,Maroc,2018.

- Cyrille.H,2011

- Dreyfus D., Martinez J.-M., Samuelides M., Gordon M. B., Badran F., Thiria S. et Hérault L., Réseaux de neurones, méthodologie et applications, Eyrolles, 2ème édition, (2004).

- Elwessabi.A Ahmed.Y, Une approche basée agent mobile pour le cloud computing,2014.

- EuroCloud France. L'évolution maitrisée vers le IaaS/PaaS . novembre 2011.

- Dreyfus.G , Martinez.J-M , Samuelides.M , Gordon.M.B , Badran.F , Thiria.S et Herault.L « Réseaux de neurone, Méthodologies et Application » , Paris, Edition Eyrolles.2002.

- Gauthier.P et Laurent.V.

- Hannachi.S , Etude et Mise en Place d'une Solution Cloud Computing Privé au sein de Tunisie Télécom ,2015.

- Hicham.C , conception et comparaison de lois de commande adaptative à base de réseaux de neurones pour une articulation flexible avec non-linéarité dure ,2002.

- Ghouili.J « Commande sans capteur d'une machine asynchrone avec estimation de la vitesse par les réseaux de neurones » Thèse de Doctorat, Université de Québec. Avril 2005

- Jean-François.J , Les Réseaux neuromimétiques,1994

- Jian J.Z , Chengdu.C et Nan.Z 'Cloud Computing-based Data Storage and Disaster Recovery', Proceediongs of the International Conference on Future computer science and education, Xi'an, China, pp. 629-632,2011.

- Kadous. D , Utilisation des réseaux de neurones comme outil du datamining :Génération de modèle comportemental d'un processusphysique à partir de données ,2012.

- Karahroudy.A ,Security Analysis and Framework of Cloud Computing with ParityBased Partially Distributed File System,2011.

- Kohonen.T , Self organized formation of topologically correct feature maps, Biol Cybernetics, Vol. 43, pp. 59-69, 1982.

- Labaf.M , Esfahan University, Rule Extraction From Artificial Neural Networks Under Background Knowledge,2007.

- Rajasekaran.S , et G.A.VijayalakshmiPai .Neural Network, Fuzzy Logic and Genet ic Algorithm, Prentice Hall of India, pg-13-20,2011.

- Marc.P ‘Réseaux de neurones’, université Laval, 2004.

- Malik.S , Huet.F , Adaptive Fault Tolerance in Real Time Cloud Computing ,IEEE World Congress on Services,2011.

- Mathur.P , ‘Cloud Computing: New challenge to the entire computer industry’,Proceediongs of the 1st International Conference on Parallel, Distributed and Grid Computing, Solan, India, pp. 223-228,2010.

- Merzouka.N , Etude des performances des réseaux de neurones dynamiques à représenter des systèmes réels : une approche dans l'espace d’état,2009.

- Muijnck-Hughes.J , et Hons.B .Data Protection in the Cloud,2011.

- Nerrad.O Roussel-Ragot.P , Urbani.D, Personnaz.L et Dreyfus.G « Training Recurrent Neural Networks : Why and How? An Illustration in Process Modeling » IEEE Trans. on Neural Networks, Vol. 5, No. 2, pp. 178-184, 1994.
- Peter.M and Timothy.G. NIST : The NIST Definition of Cloud Computing. http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145. pdf.

- Ried.S , Kisker.H , Matzke.P , Bartels.A et Lisserman.M Understanding and quantifying the future of Cloud Computing. Technical report, 2011.

- Singh.T , TarakaramaRaviTeja.Gv, et SrinivasaPappala.P Fault Tolerance- Challenges, Techniques and Implementation in Cloud Computing, International Journal of Scientific and Research Publications, Volume 3, Issue 6,2013.
- Srinivasa.V , Chakavathy et Joydeep.G » Scale-based clustering using the radial basis function network » IEEE Trans. on Neural Networks, Vol.7 , No. 5, pp. 1250-12617, 1996.

- Soumya.R“Execution Analysis of Load Balancing Algorithms in Cloud Computing Environment,” Int. J. Cloud Comput. Serv. Archit., vol. 2, no. 5, pp. 1–13, Oct. 2012.

- Tchana.A , Broto.L, et D. Hagimont . Approaches to cloud computing fault tolerance, pp. 1–6,2012.

- Watrous.R.L « Learning Algorithms for Connectionist Networks: Applied Gradient Methods of Nonlinear Optimization», IEEE First International Conference on Neural Networks, 2, pp 619-627, 1987.

- Webbing.Z et Al . Fault Tolerance Middleware for cloud computing .Third International Conference on Cloud Computing,2010.

- Yang.K , et Ma.J Implementation of IEEE802.1x in OPNET, in Proc. 7th Asia Simulation Conference on System Simulation and Scientific Computing (ICSC),pp.1390-1394,2008.

- Zemouri.R , Contribution à la surveillance des systèmes de production à l'aide des réseaux de neurones dynamiques : Application à la e-maintenance,2003.