الجمهـوريــة الجزائريـة الديمقراطيـة الشعبيــة

وزارة التعليم العالي والبحث العلمي

جامعة سعيدة د. مولاي الطاهر

كلية العلوم

قسم: الإعلام الآلي

Mémoire de Master

Spécialité : **Réseaux Informatique et Systèmes Répartis**

Thème

Une stratégie de réplication dynamique basée sur l'apprentissage automatique.

Présenté par :

- DHAIF Mouna
- KADA Ahlem

Dirigé par :

- Dr. LIMAM SAID

Promotion : 2023 - 2024


بسم الله الرحمن الرحيم

Sommaire

Table des figures

Liste des tableaux

Liste des Abréviations

ADR	Adaptive Data Replication
CDRM	Cost-Effective Dynamic Resource Management
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
FR	Frequency Ratio
GB	Gigabyte
IaaS	Infrastructure as a Service
IoT	Internet of Things
NIST	National Institute of Standards and Technology
NP-complet	Non-deterministic Polynomial-time complete
PaaS	Platform as a Service
PC	Personal Computer
RTRM	Real-Time Resource Management strategy
SaaS	Software as a Service
SGBD	Système de Gestion de Base de Données
SLA	Service Level Agreement
TB	Terabyte
TRI	Temps de Réponse Intermédiaire

Abstract

Cloud Computing is renowned for its high performance and cost-effectiveness, while Fog Computing stands out for its proximity to connected devices, facilitating data storage and analysis. Data replication emerges as a crucial solution to enhance both availability and performance.

In this study, we propose a data replication strategy integrating machine learning, optimizing the management of distributed resources and data. We offer two distinct solutions : the first is centralized, leveraging the cloud for global data classification ; the second is distributed, sharing responsibilities among fog nodes, enhancing system resilience against failures and reducing network latencies.

We expanded the iFogSim simulator to evaluate our strategy. Simulation results on iFogSim confirm the effectiveness of our approach, showing significant improvements not only in execution times but also in energy consumption and bandwidth usage

Keywords : Fog Computing, Cloud Computing, Machine Learning, Replication, iFogSim.

Résumé

Le Cloud Computing est renommé pour ses performances élevées et ses coûts réduits, tandis que le Fog Computing se distingue par sa proximité avec les objets connectés, facilitant ainsi le stockage et l'analyse des données. La réplication des données se présente comme une solution cruciale pour améliorer à la fois la disponibilité et la performance.

Dans ce travail, nous proposons une stratégie de réplication des données intégrant l'apprentissage automatique. Cette approche permet d'optimiser la gestion des ressources et des données distribuées. Deux solutions distinctes sont proposées : la première est centralisée, exploitant le cloud pour la classification globale des données ; la seconde est distribuée, répartissant les responsabilités entre les nœuds de fog, permettant une meilleure résilience du système face aux pannes et une réduction des latences réseau.

Nous avons étendu le simulateur iFogSim pour évaluer notre stratégie. Les résultats de l'évaluation par simulation sur iFogSim confirment l'efficacité de notre approche, en montrant des améliorations significatives des temps d'exécution, mais aussi de la consommation d'énergie et de l'utilisation de la bande passante

Mots clés : Fog Computing, Cloud Computing, Apprentissage automatique, Réplication, iFogSim.

ملخص

الحوسبة السحابية مشهورة بأدائها العالي و تكاليفها المنخفضة، بينما تتميز الحوسبة الضبابية بقربها من الأجهزة المتصلة، مما يسهل تخزين البيانات وتحليلها.يعتبر تكرار البيانات حلاً حاسماً لتحسين كل من الأداء والتوفر. في هذا العمل، نقدم استراتيجية تكرار البيانات تتضمن التعلم الآلي. هذا النهج يسمح بتحسين إدارة الموارد والبيانات الموزعة. تم اقتراح حلين متميزين: الأول مركزي، يستغل السحابة لتصنيف البيانات على نطاق واسع؛ الثاني موزع، يوزع المسؤوليات بين عقد الضباب، مما يتيح مرونة أكبر للنظام في مواجهة الأعطال وتقليل تأخيرات الشبكة.

لقد قمنا بتوسيع محاكي iFogSim لتقييم استراتيجيتنا. تؤكد نتائج التقييم بالمحاكاة على iFogSim فعالية نهجنا، حيث أظهرت تحسينات كبيرة في أوقات التنفيذ، بالإضافة إلى تقليل استهلاك الطاقة واستخدام النطاق الترددي.

الكلمات المفتاحية : حوسبة الضباب، الحوسبة السحابية، التعلم الآلي، النسخ المتماثل iFogSim,

Remerciements

Nous remercions **DIEU** le tout puissant de nous avoir donné la santé et la volonté d'entamer et de terminer ce travail.

Nous tenons à exprimer nos reconnaissances et nos gratitudes envers notre encadreur **M. SAID LIMAM** qui nous a soutenue et guidée dans l'élaboration de notre travail,Nous le remercions grandement pour sa disponibilité durant notre préparation de mémoire, ses encouragements permanents et pour tous ses précieux conseils et critiques constructives qui ont contribué à améliorer la qualité de cette thèse et l'enrichir davantage.

Nous remercions tous nos enseignants pour leurs savoirs et leurs efforts prodigués pendant notre formation.

Nous remercions vivement les membres des jurys pour les efforts qu'ils auront fournis à lire notre travail.

Enfin, nous remercions nos familles et nos proches des moments de détente partagés, mais aussi d'avoir patiemment enduré les oscillations de nos humeurs lors de l'élaboration de ce travail.

" Merci infiniment à vous tous "

Dédicace

Je dédie ce travail avec l'amour et le respect comme un geste de gratitude a :

Mon cher père **Aounallah** et Ma chère mère **Fatima**, les parents les plus gentils de terre qui ont éclairé mon chemin qui m'ont encouragé et soutenu tout au long de mes études.

A mes sœurs : **Lamis, Djoumana et marwa**

A mes petites chères : **Aous et mayes**

A mon binôme et mon meilleur ami : **Ahlem**

A toute la famille : **DHAIF** et **CHEBIL** qui ont été toujours à mon côté pour m'aider, soutenu et encourager.

A toute personne qui occupe une place dans mon cœur.

A tous ceux qui ont contribué à la réalisation de ce travail.

MOUNA

Dédicace

À mon très cher **père**, qui est la source de ma fierté, qui a toujours répondu présent dans les moments les plus difficiles, toujours près de moi pour m'écouter, me soutenir, m'encourager et me donner la force de poursuivre mes études.

À ma chère **maman**, qui m'a encouragé avec une confiance indéfectible tout au long de mes études, qui m'a donné l'aide et l'amour nécessaires pour écrire ce mémoire dans les meilleures conditions.

Aucune dédicace ne pourra compenser les sacrifices de mes parents, sans qui ma réussite n'aurait pas été possible.

À mes adorables **sœurs** et mon cher **frère**, qui m'ont toujours soutenu.

À mon amie et binôme **Mouna**, que Dieu nous garde toujours unis.

À toute personne qui nous a aidés à réaliser ce mémoire.

AHLEM

Introduction générale

La révolution technologique a entraîné une explosion du volume de données, marquant un changement d'échelle significatif en termes de tailles, de nombres et de types de données. Cette croissance massive a transformé la manière dont les données sont gérées, stockées et accessibles, les dernières étapes du développement informatique ouvrant la voie à de nouvelles technologies, notamment : Cloud Computing, Fog computing et réplication, en particulier dans les systèmes distribués.

le Cloud Computing est un paradigme majeur d'utilisation des infrastructures informatiques, dans lequel les ressources informatiques sont abstraites, virtualisées, dynamiques, évolutives, hautement disponibles et configurables. Il existe généralement trois types de services dans le Cloud Computing : Infrastructure (IaaS), Plateforme (PaaS) et Logiciel (SaaS).Ces services sont offerts sur demande à des clients externes via une connexion internet haut débit. Le Cloud Computing offre aux organisations qui gèrent d'immenses volumes de données et qui disposent de ressources internes limitées, la possibilité d'acquérir les ressources selon leurs besoins sur demande, tout en minimisant les coûts. Elles n'auront pas besoin d'investir dans l'installation de nouvelles infrastructures informatiques, ni d'investir dans des ressources humaines pour maintenir et gérer de telles infrastructures.

malgré les nombreux avantages du Cloud, son principal défi réside dans la latence due à sa distance considérable par rapport aux utilisateurs finaux, ainsi que la gestion des vastes quantités de données générées par les appareils, particulièrement avec l'avènement et la croissance de l'IoT. Cela a conduit à l'émergence d'une nouvelle architecture plus proche des utilisateurs finaux : le Fog Computing.

le Fog est une extension du Cloud, très proche des utilisateurs, ce qui permet d'éliminer le problème de latence et de répondre aux besoins des utilisateurs finaux en temps réel.Il se compose de nœuds et chaque nœud couvre une zone géographique, ce qui signifie que l'analyse, le traitement et la réponse aux données reçues se font en temps réel. Le Fog Computing offre des services hébergés similaires au Cloud Computing, comme des ressources de traitement, des espaces de stockage ou encore des applications. Cela permet de réduire le temps de réponse des applications et d'analyser et traiter les données reçues en temps réel.

malgré les avantages et l'utilité du Fog, mais avec le développement d'IoT et l'émergence des villes intelligentes, il est devenu crucial de considérer l'évolutivité des infrastructures Fog pour répondre à l'augmentation massive du nombre d'appareils connectés. De plus, la réplication des données devient essentielle pour

assurer une disponibilité continue des services locaux, minimiser les temps de latence et renforcer la résilience face aux pannes potentielles des nœuds Fog dispersés.

dans ce contexte, la réplication de données est une technique largement utilisée dans environnements distribués qui permet de créer plusieurs copies de certaines données et les stocke sur plusieurs sites donc,si un site de stockage tombe en panne, le système peut continuer à fonctionner en utilisant des données répliquées, ce qui augmente la disponibilité et la tolérance aux pannes. En même temps, et permet de rapprocher les données des utilisateurs pour diminuer la latence , Les principaux objectifs de la réplication sont d'améliorer les performances des demandes de données locales et d'augmenter la disponibilité des données.

l'intégration de l'apprentissage automatique, un sous-domaine de l'intelligence artificielle, représente une avancée significative dans l'optimisation de la réplication des données pour le fog computing. En exploitant les capacités prédictives de l'apprentissage automatique, les systèmes peuvent anticiper les besoins futurs et ajuster dynamiquement la distribution des données. Cette approche permet non seulement de réduire la latence et d'améliorer la disponibilité des services, mais aussi d'optimiser l'utilisation des ressources, d'augmenter la qualité de service et de réduire les coûts opérationnels grâce à une gestion proactive du trafic réseau. Cependant, malgré son potentiel, les travaux de recherches sur la réplication dynamique des données utilisant l'apprentissage automatique reste relativement limitée, offrant ainsi un vaste terrain pour de futures études et innovations dans ce domaine.

Contribution

Nous avons développé une stratégie innovante de réplication des données basée sur l'apprentissage. Cette approche utilise deux algorithmes de classification : DBSCAN pour la classification locale et K-means pour la classification globale. Notre étude propose deux solutions distinctes pour optimiser la gestion de la réplication des données dans le contexte du Fog Computing. La première approche est centralisée, où l'ensemble du processus de classification et de répartition est effectué au niveau du Cloud. Initialement, chaque Fog effectue une classification locale des fichiers en fonction de leur dépendance.

Ensuite, ces clusters sont envoyés au Cloud, qui effectue une seconde classification en fonction de la dépendance et de la fréquence d'accès, puis envoie cette classification à chaque Fog pour guider la réplication des fichiers. Cette approche tire parti de la vue d'ensemble du Cloud pour optimiser la distribution des données.

en revanche, la deuxième approche est distribuée et se concentre sur la division des responsabilités entre les Fog. Dans cette approche, les Fog sont divisés en deux niveaux, le niveau inférieur est chargé de l'envoi des clusters de données qui sont calculé par l'algorithme DBSCAN vers le niveau supérieur, Le niveau supérieur est responsable de la classification globale des données pour la réplication, ce qui réduit la dépendance vis-à-vis du Cloud. Cette approche distribuée présente plusieurs avantages supplémentaires. Elle permet une meilleure résilience du système en cas de panne locale d'un nœud Fog, car les autres nœuds peuvent continuer à fonctionner de manière autonome. De plus, elle offre une plus grande flexibilité dans la gestion des ressources locales et une capacité d'adaptation plus rapide aux variations de charge de travail. Enfin, en répartissant la charge de traitement entre plusieurs nœuds Fog, cette approche contribue à réduire les goulets d'étranglement potentiels et à améliorer la réactivité globale du système.

Plan du manuscrit

Les travaux que nous avons menés dans le cadre de la problématique d'une stratégie de réplication Dynamique basée sur L'apprentissage Automatique, Sont résumés dans ce manuscrit qui est composé, en plus d'une introduction et d'une conclusion, de 4 chapitres comme suit :

- **Le premier chapitre** : introduit les notions essentielles et les concepts fondamentaux du Cloud et du Fog Computing, en détaillant leurs services, leurs différentes typologies ainsi que leurs objectifs spécifiques
- **Le deuxième chapitre** :nous avons présenté quelques méthodes développées dans la littérature traitant la réplication des données dans le Cloud et le Fog computing.
- **Le troisième chapitre** :destiné à l'étape de conception de notre contribution qui prendra en compte la stratégie proposée pour résoudre notre problème.
- **Le quatrième chapitre** : est destiné à l'implémentation de la stratégie de réplication automatique proposée. Il décrit l'environnement de simulation ainsi que les différents critères d'évaluation des stratégies de réplication.

Chapitre 1

Le Cloud ET Le Fog Computing

1.1 Introduction

Aujourd'hui, les systèmes d'information recherchent constamment des technologies novatrices pour améliorer la performance, l'efficacité économique et l'impact écologique de leurs infrastructures informatiques.

L'émergence du **Cloud Computing**, initiée par des géants tels qu'Amazon et Microsoft, a révolutionné la gestion des ressources en fournissant une infrastructure informatique partagée. Cependant, de nombreuses entreprises, en particulier celles à capital réduit, peinent à adopter ces nouvelles technologies.

Ce chapitre explorera le **Cloud Computing**, ses caractéristiques et ses domaines d'application, avant d'aborder le concept émergent du **Fog Computing**.

1.2 Cloud computing

1.2.1 Définition

Le cloud computing, également appelé informatique en nuage, est la mise à disposition à la demande de ressources informatiques telles que le stockage et l'infrastructure, sous forme de services via Internet [2].

Le NIST donne la définition du cloud comme cette conception d'architecture informatique, qui fournit des ressources très puissantes pour le calcul, le stockage, l'environnement de développement d'applications avec plusieurs plateformes et une facilité de gestion et de coordination de tous les dispositifs de ressources en un seul endroit [3].

En d'autres termes, le cloud computing permet aux individus et aux entreprises d'accéder à un pool virtuel de ressources partagées (calcul, stockage, services, réseau) situées sur des serveurs distants gérés par des fournisseurs de services. L'un des avantages majeurs du cloud computing est que vous ne payez que pour ce que vous utilisez, ce qui permet aux organisations de s'adapter rapidement aux fluctuations de la demande sans avoir à gérer leurs propres centres de données physiques et serveurs [2].

FIGURE 1.1 – Cloud Computing

1.2.2 Caractéristiques du Cloud computing

NIST a défini cinq caractéristiques principales comme suit :

- **Libre-service à la demande** : Les clients ou les utilisateurs peuvent demander n'importe quelle ressource informatique à tout moment et la ressource est fournie automatiquement sans aucune interaction avec le fournisseur de service.
- **L'accès universel** : Les services fournis par le cloud sont accessibles via le réseau. L'accès à ces services est basé sur des mécanismes standards et peut être activé par différents appareils : mobile, tablette, ordinateur de bureau, ordinateur portable.
- **La mise en commun de ressources** : Les ressources mises à disposition sont communes à tous les clients et sont partagées dynamiquement entre eux en fonction de leurs besoins et l'emplacement exact des ressources concernées leur est inconnu.
- **Une élasticité rapide** : Les ressources sont mises à disposition et libérées à la demande, et à tout moment, le consommateur doit disposer exactement de la quantité de ressources dont il a besoin pour le produit. En fait, le consommateur doit être en mesure d'augmenter ou de diminuer la capacité de son système. En d'autres termes, elle définit la capacité d'une infrastructure donnée à s'adapter de manière dynamique avec les changements de demandes des utilisateurs. Cette caractéristique permet aux utilisateurs de provisionner rapidement de nouvelles ressources afin de pouvoir répondre à une augmentation ou une diminution de charge.
- **La mesure du service fourni** : Les ressources sont constamment surveillées et optimisées, et leur utilisation est gérée et communiquée aux clients.

1.2.3 Types de services Cloud (SaaS, PaaS, IaaS)

Il existe trois principaux types de services cloud, Chacun de ces types offre un certain niveau de gestion et répond à des besoins spécifiques :

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

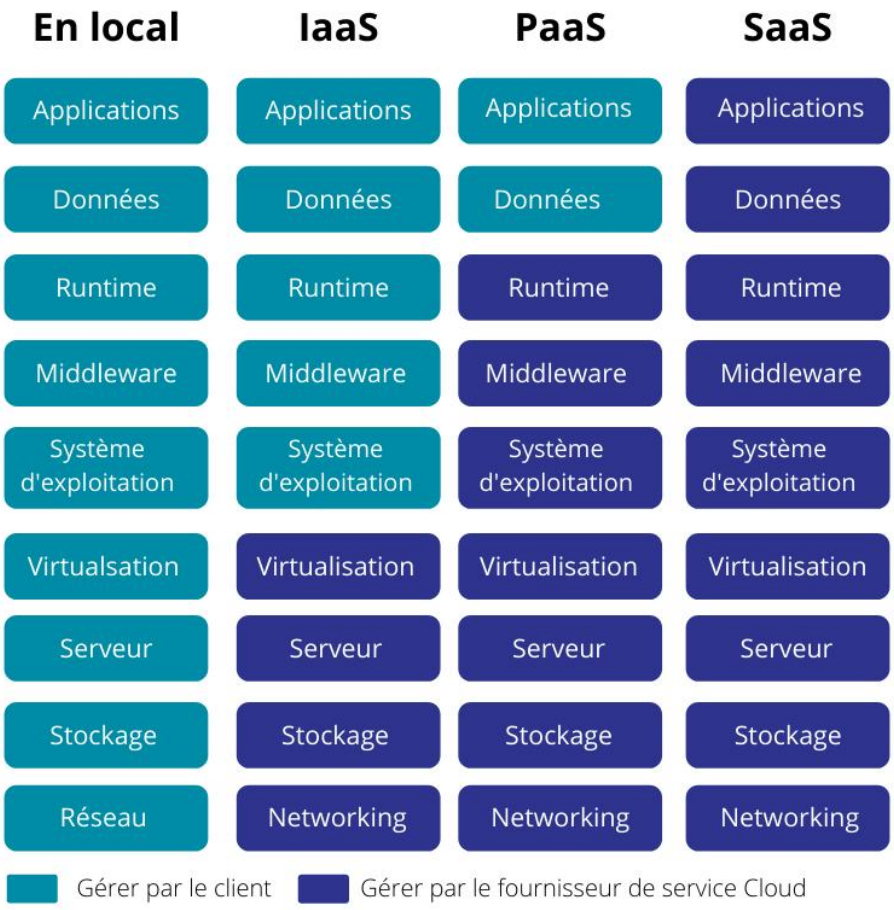

FIGURE 1.2 – Types de services Cloud (SaaS, PaaS, IaaS)

- **Infrastructure as a service (IaaS)** Infrastructure as a service (IaaS) est une forme de Cloud Computing offrant des ressources informatiques au sein d'un environnement virtualisé (le Cloud) par le biais d'internet ou d'une autre connexion. L'infrastructure en tant que service c'est le service de plus bas niveau. Il consiste à offrir un accès à un parc informatique virtualisé. Des machines virtuelles sur lesquelles le consommateur peut installer un système d'exploitation et des applications. Le consommateur est ainsi dispensé de l'achat de matériel informatique. Ce service s'apparente aux services d'hébergement classiques des centres de traitement de données, et la tendance est en faveur de services de plus haut niveau, qui fait davantage abstraction de détails techniques [4].
- **Platform as a service (PaaS)** Platform as a service (PaaS) : Il s'agit de l'offre intermédiaire dans le portefeuille Cloud Computing. Le fournisseur met à disposition une plateforme middleware opérationnelle, incluant des serveurs d'applications, des bases de données et les outils permettant au client de développer et de déployer ces propres applications. Cette configuration est très employée pour disposer de plateformes de développement ou de tests disposant de l'ensemble des outils et middleware nécessaires, en évitant ainsi les tâches de construction et de maintenance de ces plateformes non critiques. Elle se destine donc naturellement avant tout aux développeurs. Des services comme Google App Engine, Bungee Connect et Force.com sont des exemples PaaS [5]. Les principaux fournisseurs de PaaS sont : Microsoft avec Azure, Google avec Google App Engine et Orange Business Services.

- **Software as a service (SaaS)** Software as a service (SaaS) : Il s'agit d'une offre « tout compris ». Le prestataire met à disposition une application qu'il administre et configure en majeure partie. Le client externalise ainsi ses applications auxquelles il accède à la demande. Il paie à l'usage, selon le nombre d'utilisateurs et/ou le temps d'utilisation du logiciel. Les prestataires de solutions SaaS les plus connus sont : Google avec Gmail et YouTube ou encore les réseaux sociaux Facebook et Twitter [5].

1.2.4 Avantages et Inconvénients des services Cloud (SaaS, PaaS, IaaS) [1]

	Avantage	Inconvénient
SaaS	✓ Pas d'installation ✓ Plus de licence ✓ Migration ✓ Accessible via un abonnement	✓ 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 ✓ Capacité de stockage infini	✓ Sécurité ✓ Besoin d'un administrateur système ✓ Demande pour les acteurs du Cloud des investissements très élevés

FIGURE 1.3 – Avantages et Inconvénients des services Cloud **(SaaS, PaaS, IaaS)**

1.2.5 Les modèles de déploiement

1. **Cloud public :**Aussi appelé cloud Externe, il comprend toutes les offres cloud où les fournisseurs et les utilisateurs potentiels n'appartiennent pas à la même unité organisationnelle. Dans ce modèle, les fournisseurs rendent leur cloud accessible au public et proposent généralement un portail Web en libre-service.[6]

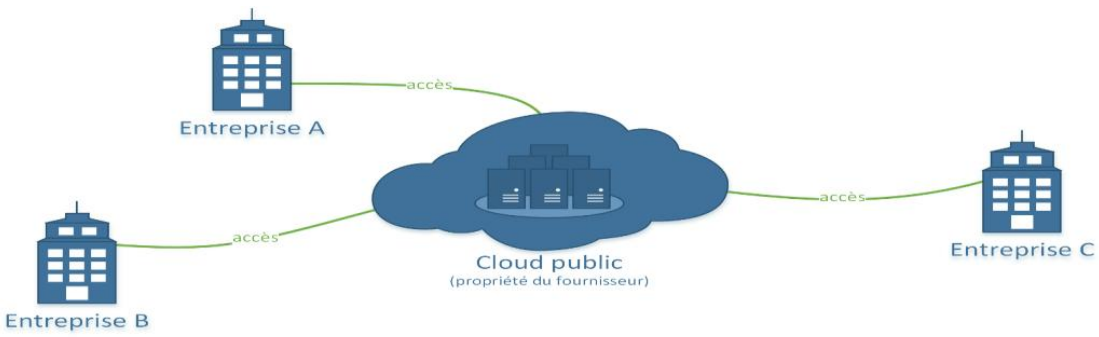

FIGURE 1.4 – Cloud public

2. **Cloud Privé :**Aussi appelé cloud Internet. Les Fournisseurs et utilisateurs appartiennent à la même unité organisationnelle (Entreprise, université, gouvernement, etc.) Dans ce modèle, le contrôle sur les données reste avec les utilisateurs ou leur organisation. Le Cloud privé est un Cloud dans lequel les services et l'infrastructure se trouvent sur un réseau privé.[6]

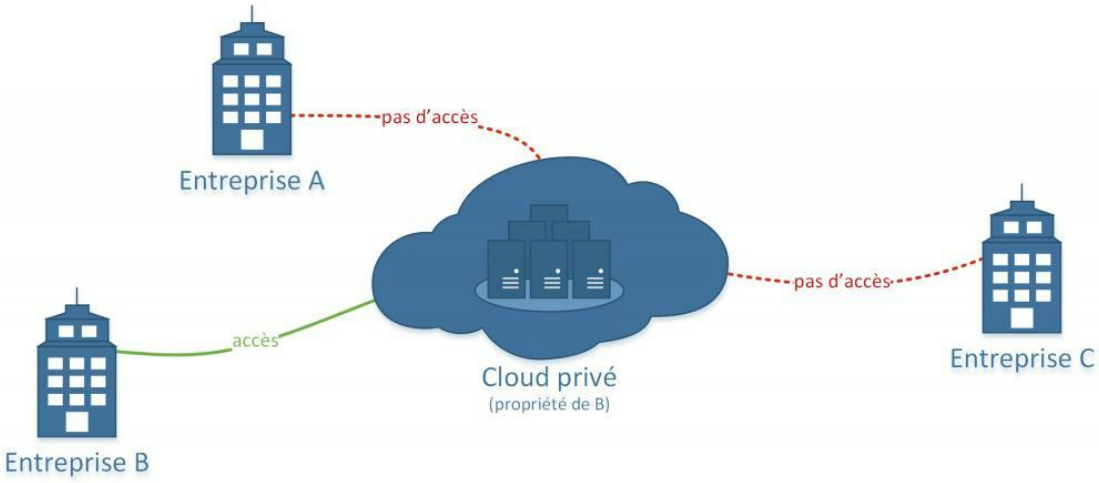

FIGURE 1.5 – Cloud Privé

3. **Cloud communautaire :**Ce cloud est partagé par plusieurs organisations regroupées au sein d'une communauté. Cette communauté a des intérêts communs (réglementaire, sécurité des données). Ce cloud peut être géré par les organisations ou un tiers.[6]

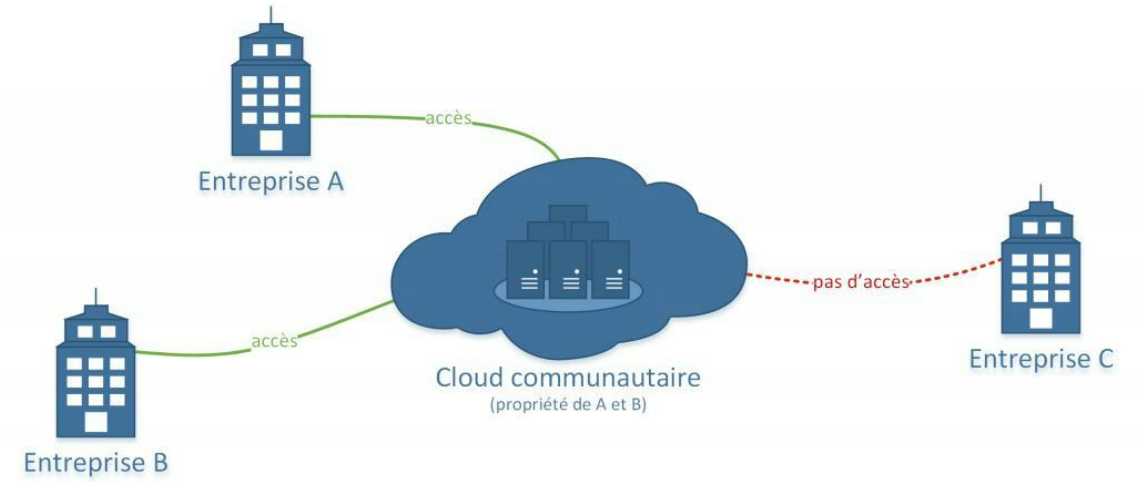

FIGURE 1.6 – Cloud communautaire

4. **Cloud Hybride :**L'infrastructure cloud est une composition de deux ou plusieurs infrastructures cloud distinctes (privées, communautaires ou publiques) qui restent des entités uniques mais sont liées par une technologie standardisée ou propriétaire qui permet la portabilité des données et des applications (par exemple, l'éclatement du cloud pour l'équilibrage de charge entre les cloud ,le Cloud hybride offre aux entreprises un plus grand niveau de flexibilité et plus d'options de déploiement.[6]

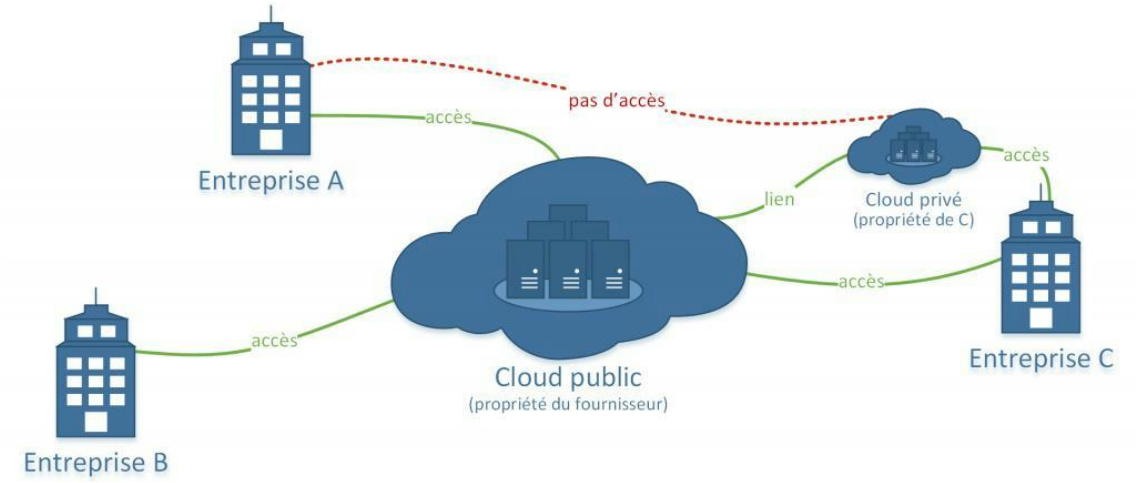

FIGURE 1.7 – Cloud Hybride

1.2.6 Architecture globale du Cloud Computing

En faisant abstraction de détails de plus haut niveau, l'architecture globale du cloud computing comporte essentiellement :

- **Des clients :** personne, entreprise, groupe qui accède aux différents services offerts par le cloud.
- **Des services :** Différents niveaux de services et données sont gérés par des fournisseurs afin de les offrir à la demande des clients du cloud.

- **Un réseau :** Intermédiaire entre le client et le fournisseur, qui permet de transiter les services (chemin que les services entreprennent) c'est le réseau Internet.
- **Des fournisseurs :** Ce sont des entités chez lesquelles on alloue les services cloud.
- **Des serveurs :** Où sont stockés les services cloud, ils sont éparpillés partout dans le monde et constituent le cœur hardware du cloud computing. Les entités citées ci-dessus sont connectées selon le modèle de déploiement du cloud privé et public.

1.2.7 Avantages du Cloud computing

Le Cloud computing présente de nombreux avantages tels que :

- **Accessibilité facile :** accéder à tous les services de Cloud computing à tout moment, sur tous les supports, via une connexion internet.[7]
- **Disponibilité et fiabilité :** La raison pour laquelle l'environnement cloud est très fiable est que le fournisseur de services cloud utilise une sauvegarde complète. Tant que l'utilisateur est connecté à Internet, le service cloud est toujours disponible pour l'utilisateur et le client peut accéder à ses données de n'importe où.[7]
- **Capacité de stockage illimitée :** À l'aide de services Cloud, le client peut utiliser la capacité de stockage illimitée. Si les besoins de stockage du client augmentent, le client doit payer un peu plus pour utiliser une plus grande capacité de stockage fournie par le serveur Cloud. Ainsi, avec un faible coût d'installation, un utilisateur peut utiliser une capacité de stockage illimitée.[8]
- **Facilité de mise à l'échelle :** Les services Cloud peuvent facilement être mis à l'échelle selon les souhaits du consommateur. Par exemple, la capacité de stockage dans le cloud peut être augmentée jusqu'à des To ou elle peut être aussi faible que quelques Go.[8]
- **Aucun coût de maintenance :** Réduisant les coûts de maintenance par la possibilité de ne pas installer d'applications sur le PC, le consommateur n'a qu'à payer pour le service qu'il a utilisé, et lorsque le serveur ou le lecteur de données s'use et tombe en panne, le fournisseur de services doit supporter le coût de remplacement.[8]

1.2.8 Limites du Cloud Computing

Le Cloud computing présente également certains inconvénients tels que :

- **Sécurité des Données :** C'est le plus gros inconvénient du cloud, les données sont mises en ligne là où d'autres personnes peuvent potentiellement y accéder en cas de violation. C'est la principale raison pour laquelle de nombreuses entreprises hésitent à utiliser le cloud et malgré le succès du cloud à conquérir le marché des petites et moyennes entreprises, il a peu de part dans les grandes entreprises.[8]
- **Dépendance du réseau :** Un autre inconvénient majeur du Cloud est sa dépendance à Internet, ou à un certain autre réseau local en cas de Cloud privé. Même si Internet est devenu omniprésent dans le monde développé, il trouve encore ses marques dans les pays en développement. Donc, utiliser le Cloud pour des produits signifie essentiellement ignorer cette partie de la population mondiale qui est sans Internet, et pour certains produits, ils peuvent représenter une population importante.[8]

- **Contrôle limité :** Les consommateurs ont très peu de contrôle sur leurs produits dans le Cloud. Le plus grand contrôle dont ils disposent se trouve dans le modèle IaaS, où ils peuvent provisionner des machines virtuelles entières et les personnaliser en fonction de leurs besoins. Par contre, dans le modèle SaaS, ils ont le moins de contrôle, puisqu'ils peuvent seulement configurer certaines parties de l'application, mais n'ont aucun contrôle sur le reste.[8]
- **Problèmes techniques :** Bien que les informations et les données stockées dans le cloud soient accessibles à tout moment et de n'importe où, le système connaît parfois de graves défaillances. Les entreprises doivent être conscientes que cette technologie est toujours sujette à des défaillances et à d'autres problèmes techniques. Malgré les normes de maintenance élevées, même les meilleurs fournisseurs de services de cloud computing sont confrontés à ces problèmes.[9]
- **Configurations limitées :** Les fournisseurs de solutions de cloud public disposent d'un agencement standard de configurations de base qui répondent aux besoins de la population globale. Dans certains cas, un matériel exceptionnellement spécialisé est nécessaire pour traiter des problèmes informatiques majeurs. Dans de tels cas, l'utilisation de solutions de cloud public est souvent impossible, car les fonctionnalités requises ne sont tout simplement pas proposées par le fournisseur.[10]

1.3 Fog computing

1.3.1 Définition

Le terme « Fog computing » a été proposé en 2012 par des chercheurs de Cisco System[11]. Le Fog computing, également appelé Edge computing ou fogging, est un complément et une extension du modèle de base cloud traditionnel. C'est une architecture qui étend le cloud computing en déployant une quantité significative de calcul, de stockage et de communication au niveau des périphériques (ou bords) du réseau, plutôt que de centraliser ces opérations dans des centres de données distants[12].

Le Fog computing a pour but d'optimiser les communications entre un grand nombre d'objets connectés (IoT) et les services de traitement distants (Cloud), en tenant compte d'une part des volumes de données considérables engendrés par ce type d'architecture (mégadonnées) et d'autre part de la variabilité de la latence dans un réseau distribué, tout en donnant un meilleur contrôle sur les données transmises[13].

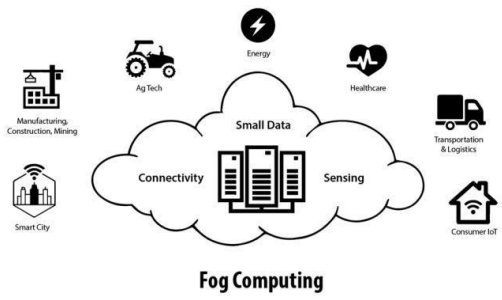

FIGURE 1.8 – Fog computing

1.3.2 Caractéristiques du Fog Computing

Les caractéristiques du Fog computing peuvent être résumées comme suit :[14][15][16]

- **La détection de l'emplacement et de la faible latence :** Le Fog computing prend en charge la connaissance de l'emplacement et peut déployer des nœuds de reconnaissance à différents endroits. De plus, comme le Fog est plus proche du terminal, il permet de réduire la latence lors du traitement des données du terminal[15].
- **La distribution géographique :** Contrairement aux cloud centralisés, les services et applications fournis par le Fog sont distribués et peuvent être déployés n'importe où.
- **L'évolutivité :** Il existe de vastes réseaux de capteurs qui surveillent l'environnement environnant, et le Fog computing fournit des ressources de calcul et de stockage distribuées qui peuvent fonctionner avec ces périphériques à grande échelle[16].
- **L'interaction en temps réel :** Les applications de calcul du Fog fournissent des interactions en temps réel entre les nœuds du Fog plutôt que le traitement par lots utilisé dans le cloud[17].
- **L'hétérogénéité :** Les nœuds du Fog, dans lesquels les terminaux sont conçus par différents fabricants, ont des formes différentes, et doivent être déployés en fonction de leurs plateformes.
- **L'interopérabilité :** Les composants du Fog peuvent interagir et travailler avec différents fournisseurs de services dans différents domaines.
- **L'accompagnement à la mobilité :** Un aspect important des applications du Fog est la capacité de communiquer directement avec les appareils mobiles et donc d'activer des méthodes de navigation.

1.3.3 Les modèles de déploiement

Il existe quatre modèles différents de Fog computing :

- **Fog privé** : Le Fog privé est développé, acheté, entretenu et contrôlé par une entité, une entité tierce ou une variante de celle-ci. Il peut être installé sur site ou hors site. Les services de Fog privés sont vendus pour un usage exclusif par une seule entreprise.
- **Fog public** : Un Fog public est détenu, développé et géré par, ou une combinaison de, l'entreprise, établissement universitaire ou organisme gouvernemental. Il est déployé dans les locaux des fournisseurs de Fog. Les services publics de Fog sont fournis en libre accès par les services communs ou généraux publics.
- **Fog communautaire** : Le Fog communautaire est développé, entretenu et contrôlé par différents organismes communautaires, qui peuvent également inclure un tiers, ou une consolidation de ceux-ci. Ils peuvent être installés sur site ou hors site et les services sont fournis pour un usage exclusif, généralement par des clients dans un groupe particulier d'organisations ayant des intérêts communs.
- **Fog hybride** : Le Fog hybride est un type de cloud computing qui intègre l'utilisation de Fog public/privé/communautaire avec le cloud computing public/privé (c'est-à-dire le cloud hybride). Cela pourrait être utile en raison des inconvénients des ressources physiques dans le Fog. En conséquence,

la plateforme est appliquée au cloud hybride pour faire évoluer les performances. Le cloud hybride est flexible, modulaire et les services sont accessibles sur demande.

1.3.4 Architecture du Fog Computing

En général, l'architecture du Fog comprend trois couches : la couche dispositif, la couche Fog et la couche cloud. Ces couches sont liées aux autorités publiques comme le centre de génération de clés et l'autorité de certification [17].

Les trois couches sont définies comme suit :

1. **La couche périphérique** : Il s'agit de la couche la plus proche des utilisateurs finaux. Cette couche inclut tous les appareils IoT (mobiles ou statiques) tels que les téléphones mobiles, les capteurs, etc. [17]. De tels appareils diffusent généralement des informations géographiques et sensibles provenant de divers objets ou incidents. Par la suite, les informations collectées sont transférées vers la couche supérieure, spécifiquement pour un traitement ultérieur ainsi que pour le stockage [18].
2. **Couche de Fog** : Elle se situe entre les appareils de l'IoT et le cloud et elle joue un rôle important dans la transmission des données entre la couche périphérique et la couche de cloud computing. Elle est composée d'un grand nombre de nœuds Fog distribués [17]. Les périphériques réseau tels que les commutateurs, les points d'accès, les routeurs et les passerelles constituent la couche de Fog [18].
3. **Couche Cloud** : La couche Cloud implique des serveurs hautes performances et un stockage permanent des données. De nombreux services sont fournis par la couche Cloud tels que les applications de transport et de distribution d'énergie [17]. Elle peut également attribuer une partie de la fonction au nœud du Fog, car le Fog est utile pour l'évaluation locale et la prise de décisions en peu de temps [9].

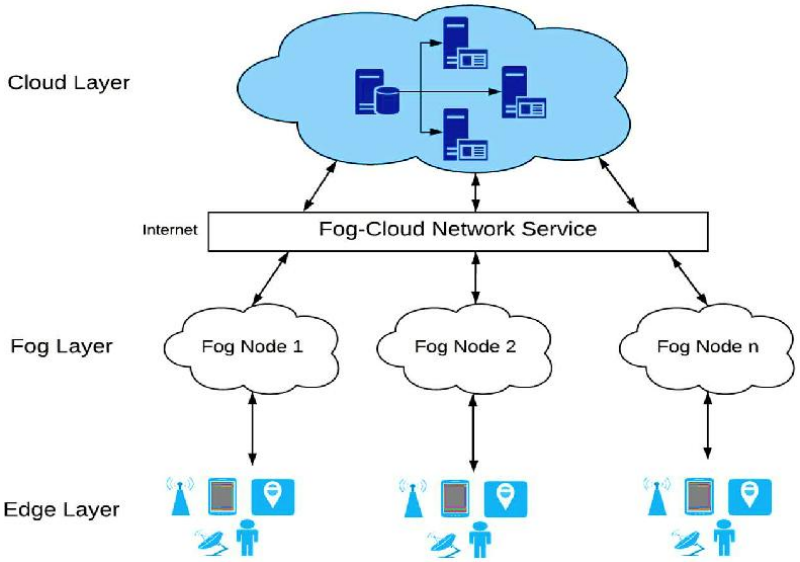

FIGURE 1.9 – Architecture du Fog Computing

- **La figure 9** montre l'idée du Fog computing. Il structure une manière de gérer les appareils, ces appareils peuvent communiquer entre eux ou directement avec le cloud dans certains cas [19]. Le Fog agit comme une couche intermédiaire entre les appareils Edge et le Cloud. Les appareils Edge demandent des services de calcul, de stockage et de communication au Fog. Le Fog fournit une réponse locale à faible latence à ces demandes et transmet les données pertinentes pour un traitement intensif en calcul, des analyses à long terme et un stockage persistant vers le cloud.

1.3.5 les Avantage du Fog computing

1. **Proximité géographique** : Contrairement au cloud computing, qui centralise le traitement des données dans des centres de données éloignés, le Fog Computing déplace une partie de ce traitement vers des dispositifs situés au plus près des utilisateurs et des objets connectés.
2. **Faible latence** : En rapprochant le traitement des données du lieu où elles sont générées, le Fog Computing réduit la latence, permettant des temps de réponse plus rapides pour les applications sensibles à la latence, telles que les applications de réalité virtuelle et les voitures autonomes.
3. **Mobilité** : En raison de la communication directe entre le brouillard et les appareils, elle offre une meilleure mobilité [12].
4. **Interaction en temps réel** : En raison de la proximité des appareils et des nœuds de brouillard, les interactions peuvent se produire en temps réel, contrairement au traitement par lots dans le cas du cloud computing [12].
5. **Hétérogénéité** : Les nœuds de Fog sont disponibles dans différents facteurs de forme et seront déployés dans une grande variété d'environnements.
6. **Évolutivité et agilité des clusters fédérés à nœuds de Fog** : Le Fog Computing est de nature adaptative, au niveau d'un cluster ou d'un cluster de clusters, prenant en charge le calcul élastique, la mise en commun des ressources, les modifications de la charge de données et les variations de l'état du réseau, pour répertorier quelques-unes des fonctions adaptatives prises en charge [20].
7. **Améliore la sécurité** : Bien que le brouillard crée de nouveaux défis de sécurité, il améliore la sécurité globale. Cela signifie moins de chances d'écoute clandestine car les informations parcourent moins de distance [17].

1.3.6 Inconvénients du Fog computing

1. **Coûteux** : le Fog Computing est très coûteux car l'organisation doit acheter des appareils tels que des hubs, des routeurs et des passerelles [21].
2. **Consommation d'énergie** : Il existe de nombreux nœuds de Fog disponibles dans l'environnement de brouillard qui sont directement proportionnels à leur consommation d'énergie. Cela implique que ces nœuds de Fog nécessitent un volume élevé d'énergie pour fonctionner. Si l'infrastructure de Fog nécessite plus de nœuds de Fog, il y a encore plus de consommation d'énergie [12].
3. **Sécurité** : L'accès authentique aux services et la confidentialité dans le Fog Computing sont difficiles à garantir. Il existe divers appareils et différents nœuds de Fog dans l'environnement de brouillard informatique. Ces nœuds de Fog sont susceptibles de se trouver dans un environnement moins sé-

curisé. Les pirates peuvent facilement utiliser de fausses adresses IP pour accéder au nœud de brouillard correspondant [12].

4. **Système complexe** : Le Fog Computing utilise de nombreux nœuds et constitue une couche supplémentaire des systèmes de traitement et de stockage des données, il s'agit donc d'un processus informatique très compliqué [21].
5. **Authentification** : Le service fourni par le Fog Computing est à grande échelle. Le réseau Fog est composé d'utilisateurs finaux, de fournisseurs de services Internet et de fournisseurs de cloud. Cela posera également des problèmes de confiance et d'authentification dans le Fog [12].

1.3.7 Différences entre le Cloud et le Fog Computing

Le Fog computing, bien que partageant des similitudes avec le Cloud computing, se différencie par certains paramètres, comme indiqué dans le tableau 1.1 qui synthétise la comparaison entre ces deux concepts apparentés [22, 23].

Fonctionnalité	Cloud Computing	Fog Computing
Latence	Élevée	Faible
Localisation Du Service	Dans l'Internet	Aux Frontières Du Réseau
Distance Client-Serveur	Plusieurs Sauts	Un Seul Saut
Sécurité	Moins Sécurisé	Plus Sécurisé
Temps de réponse	Faible	Haute
Support Mobilité	Limité	Supporté
Géodistribution	Centralisée	Distribuée

TABLE 1.1 – Comparaison entre le Cloud et le Fog Computing

1.4 Conclusion

Dans ce premier chapitre, nous avons défini le Cloud Computing et le Fog Computing, exposant leurs caractéristiques et leur architecture globale, ainsi que leurs avantages et inconvénients. Ces avancées majeures vers une infrastructure informatique dématérialisée offrent des ressources en tant que service, mais leur adoption soulève des défis, particulièrement en ce qui concerne la disponibilité des ressources. Ce chapitre établit ainsi le socle nécessaire pour aborder les enjeux liés à la réplication dynamique dans le Cloud et le Fog.

Chapitre 2

La Réplications Dans Le Cloud Et Le Fog Computing

2.1 Introduction

Les systèmes géographiquement distribués à grande échelle deviennent de plus en plus populaires dans les applications scientifiques à forte intensité de données [24]. De nouvelles stratégies de gestion de ces données sont nécessaires. Dans ces environnements, des millions de fichiers sont générés, et des milliers de clients à travers le monde accèdent à ces données. Cet énorme volume d'ensembles de données nécessite des approches innovantes pour garantir leur disponibilité, leur fiabilité et leur efficacité [25].

La réplication des données est généralement considérée comme la solution permettant d'améliorer le temps d'accès et la fiabilité des fichiers. Les schémas de réplication de données impliquent de déterminer le nombre de répliques nécessaires et l'emplacement des copies de données dans les nœuds distribués. Cependant, dans un contexte dynamique où les besoins en données évoluent constamment, une approche statique de la réplication n'est plus suffisante [25].

L'intégration de l'apprentissage automatique dans les mécanismes de réplication offre une solution adaptative. L'apprentissage automatique permet d'analyser les schémas d'utilisation des données en temps réel, d'anticiper les changements de demande et de mettre à jour dynamiquement les stratégies de réplication en conséquence. Ainsi, l'apprentissage automatique sert de catalyseur pour une gestion plus intelligente des répliques, améliorant ainsi l'efficacité du système en temps réel tout en répondant de manière proactive aux fluctuations des besoins en données.

Dans cet esprit, nous présenterons une approche concernant la réplication dynamique basée sur l'apprentissage automatique. Notre objectif est d'utiliser des techniques d'apprentissage automatique pour déterminer de manière adaptative le nombre optimal de répliques et leur emplacement, en fonction des caractéristiques des données et des besoins des utilisateurs.

2.2 La réplication de données

2.2.1 Définition

La réplication de données est une technique bien connue utilisée dans les systèmes distribués, qui permet d'améliorer la fiabilité, la tolérance aux pannes, l'accessibilité et d'augmenter la disponibilité des données. La réplication consiste à créer des copies d'une donnée et à les stocker dans des endroits différents, en mettant en œuvre un processus de création et de placement des copies d'entité logiciel.

La phase de création consiste à reproduire la structure et l'état des entités répliquées, tandis que la phase de placement consiste à choisir, en fonction des objectifs de la réplication, le bon emplacement de cette nouvelle reproduction. Les lectures sont exécutées sur le site disposant de la copie la plus proche du client, ce qui diminue le nombre de transferts de données. Quant à la disponibilité des données, la réplication permet de ne plus dépendre d'un site central unique et d'utiliser une copie sur un site lorsqu'elle est indisponible sur un autre [26, 27]. De plus, la réplication permet de distribuer la charge de travail sur différents nœuds possédant une réplique.

2.2.2 Objectif de la réplication

Les objectifs visés peuvent être résumés en :

- **Disponibilité** : Les ressources sont disponibles et le système est prêt à fonctionner et apte à accomplir sa fonction de manière fiable. Si une donnée se trouve localisée sur plusieurs sites, il n'est pas forcément nécessaire de s'adresser à un site distant afin de la consulter (localité des accès en consultation).
- **Fiabilité** : Aptitude à accomplir sa fonction sans défaillance dans des conditions données pour une durée déterminée. En répliquant les composants, le système sera capable de continuer à rendre un service malgré la panne d'un ou de plusieurs de ses composants.
- **Maintenabilité** : Possibilité d'être maintenu ou rétabli en un temps donné dans un état d'aptitude à accomplir sa fonction.

2.2.3 Formes de réplication

La réplication peut prendre plusieurs formes :

- **Aucune réplication** : les données ne sont pas dupliquées entre les sources du système (centralisation du système).
- **Réplication partielle** : les données sont répliquées sur quelques sources du système.
- **Réplication totale** : les données sont répliquées sur toutes les sources du système [28].

2.2.4 Méthodes de réplication

1. **Réplication statique :** Après qu'une réplique soit créée, elle existera dans le même lieu jusqu'à ce qu'elle soit effacée manuellement par des utilisateurs ou sa durée est expirée. L'inconvénient de la réplication statique est évident, quand les configurations d'accès du client changent considérablement dans la grille de données, les avantages apportés par la réplication diminueront brusquement.
2. **Réplication dynamique :** Elle prend en compte les modifications des environnements de grille et crée automatiquement de nouvelles réplications pour les fichiers de données populaires ou déplace les réplications vers d'autres sites si nécessaire pour améliorer la performance. Le but principal de la réplication dynamique est de diminuer le temps de réponse moyen qui est remarqué par l'utilisateur final. En même temps, de la vue du système entier, la métrique de performance de la consommation de largeur de bande et la fréquence de réplication doivent être estimées pour s'assurer que les réplications dynamiques n'entraînent pas une lourde charge sur le système.

2.2.5 Modes de Réplication

Lors du traitement des données répliquées sur un système distribué, en fonction de la propagation des mises à jour et du traitement des copies des données, on distingue en général deux modes différents de réplication.

1. **Réplication Symétrique :** Le premier est défini par la symétrie des comportements des copies d'une donnée répliquée. Chaque copie est traitée d'une façon identique aux autres, c'est le cas de la réplication symétrique.
2. **Réplication asymétrique :** Le deuxième mode distingue deux comportements d'une donnée répliquée : la copie primaire et les copies secondaires. La copie primaire est la seule à effectuer tous les traitements. Les copies secondaires surveillent la copie primaire. En cas de défaillance de la copie pri-

maire, une copie secondaire devient la nouvelle copie primaire, c'est le cas de la réplication asymétrique.

2.2.5.1 Réplication asymétrique : La réplication asymétrique distingue un site maître chargé de centraliser les mises à jour, il est aussi appelé site primaire. Il est le seul autorisé à mettre à jour les données et est chargé de diffuser les mises à jour aux autres copies dites secondaires (esclaves). Dans ce mode de réplication, le site primaire effectue les contrôles et garantit l'ordonnancement correct des mises à jour.

Le problème de la gestion de copie asymétrique est la panne du site maître. Dans ce cas, il faut choisir un remplaçant si l'on veut continuer les mises à jour. On aboutit alors à une technique asymétrique mobile dans laquelle le site maître change dynamiquement. Il devient nécessaire de gérer à la fois les problèmes de pannes qui provoquent des échecs de transactions et l'évolution des travaux [29].

Il existe la réplication asymétrique synchrone et asynchrone :

- **Réplication asymétrique synchrone** : Utilise un site maître qui pousse les mises à jour en temps réel vers un ou plusieurs sites esclaves (Figure 2.1).

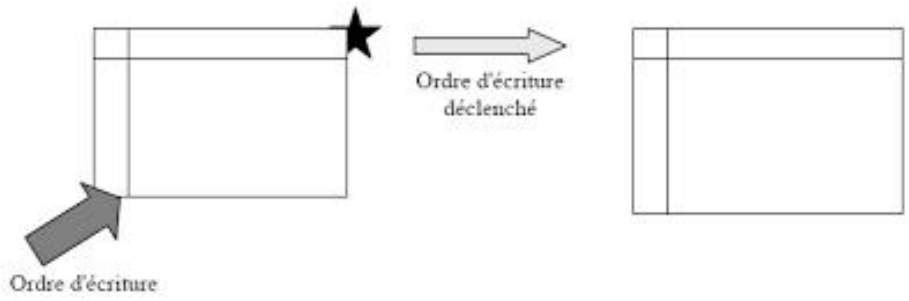

FIGURE 2.1 – Réplication asymétrique synchrone

- **La réplication asymétrique asynchrone** : Les mises à jour sont poussées en temps différé via une file d'attente. Les mises à jour seront exécutées ultérieurement, à partir d'un déclencheur externe, une horloge par exemple (voir Figure 2.2).

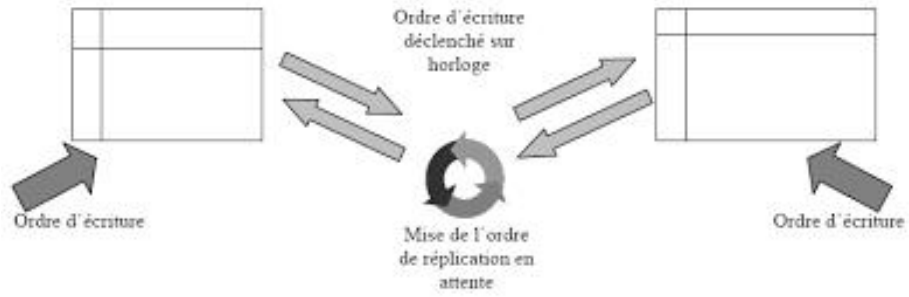

FIGURE 2.2 – Réplication asymétrique asynchrone

2.2.5.2 Réplication symétrique : À l'opposé de la réplication asymétrique, la réplication symétrique ne privilégie aucune copie, elle permet des mises à jour simultanées de toutes les copies par des transactions différentes. Dans ce cas, chaque copie peut être mise à jour à tout instant et assure la diffusion des mises à

jour aux autres copies. Ce mode pose le problème de concurrence d'accès, en risquant de faire diverger les copies. Un algorithme global de résolution des conflits doit donc être mis en œuvre [29].

Pour la réplication symétrique aussi, on distingue deux modes : **la réplication symétrique synchrone** et **asynchrone**.

1. La réplication symétrique synchrone : Gère des accès concurrentiels aux écritures, toutes les copies sont accessibles à la fois en lecture et en écriture. Chaque copie possède ses propres triggers qui lors de la réception d'une requête de mise à jour déclenchent le déploiement de cette dernière vers toutes les autres copies. À ce moment entre en jeu le protocole de validation atomique qui doit s'assurer que la transaction est bien validée sur toutes les copies ou sur aucune [29] (Figure 2.3).

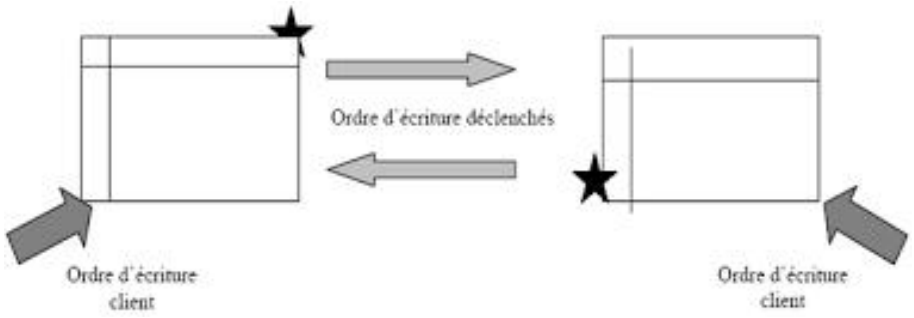

FIGURE 2.3 – Réplication symétrique synchrone

2. La réplication symétrique asynchrone : Le principe est basé sur trois éléments :

- **Écritures sur le maître** : les écritures initiées par les utilisateurs sont orientées vers un site unique (le site maître), les esclaves n'acceptent que les requêtes de lecture.
- **Mise en attente** : chaque écriture effectuée sur le site maître est sauvegardée dans une file locale ou dans un journal. C'est un processus extérieur au SGBD qui scrute en permanence la file (ou le journal) afin de détecter les modifications apportées au site maître. Ce dernier se charge par la suite de propager ces changements vers les copies secondaires (esclaves) via un déclencheur tel que l'horloge.
- **Propagation des changements sur l'esclave** : les seules écritures admises par les esclaves sont celles effectuées par les utilisateurs sur le maître. Une panne qui intervient sur le site maître avant qu'il n'ait eu le temps d'enregistrer la requête dans une file d'attente ou un journal, entraîne la perte de la mise à jour (Figure 2.4).

FIGURE 2.4 – Réplication symétrique asynchrone

2.2.6 La stratégie de réplication

Un système a besoin des stratégies de réplication dynamiques (où la création, la suppression et la gestion des répliques sont effectuées automatiquement) qui ont la capacité de s'adapter aux changements dans le comportement des utilisateurs. Les trois questions fondamentales auxquelles n'importe quelle stratégie de réplication doit répondre sont :

- Quand les répliques doivent être créées ? Moment de la réplication.
- Où les répliques doivent être placées ? Placement des répliques.
- Quelles données doivent être reproduites ? Choix de l'entité à répliquer.
- Combien de répliques doivent être créées ? Nombre de la réplication.

a- Quand les répliques doivent être créées ?

Il est crucial de déterminer le moment optimal pour créer des répliques afin de garantir la disponibilité continue des données en cas de défaillance du système principal. Les répliques peuvent être créées en temps réel, de manière synchrone ou asynchrone, en fonction des besoins spécifiques du système. La création des répliques constitue ainsi un aspect important dans une stratégie de réplication dynamique. Le processus de création de réplique peut être divisé en deux sous-processus :

- Vérifier la nécessité de nouvelles répliques,
- Puis procéder à leur création effective en conséquence.

b- Où les répliques doivent être placées ?

Un autre aspect important dans une stratégie de réplication est où placer des répliques. Un placement différent d'une réplique a un effet différent sur l'amélioration du système. Généralement, les stratégies de réplication placent la nouvelle réplique suivant des critères choisis au préalable, pour déterminer le site candidat le plus approprié au placement de la réplique. L'ensemble des sites candidats est l'ensemble des sites potentiels qui répondent à ces critères :

- Le site ne contient pas de copie du fichier à répliquer.
- La capacité de stockage du site est suffisante.
- Le temps de transfert entre le site et les utilisateurs potentiels est raisonnable.

c- Quelles données doivent être reproduites ?

Pour répondre à la question quoi : les données répliquées sont généralement de deux types : des fichiers ou des objets. Les objets peuvent être composés d'un ensemble de fichiers distribués (on les appelle aussi collection). Selon les stratégies de réplication, les données à répliquer peuvent être les plus populaires ou encore les plus fréquemment accédées.

d- Combien de répliques doivent être créées ?

Combien de répliques sont nécessaires ? Le nombre de répliques dépend de la fréquence d'accès et des exigences en matière de disponibilité, de performance et de tolérance aux pannes. Il peut être nécessaire d'avoir plusieurs copies des données pour garantir une haute disponibilité et une récupération rapide en cas de défaillance d'un serveur ou d'un site.

2.2.7 Protocole De Réplication

2.2.7.1 Réplication Passive

Dans un protocole de réplication passive (passive réplication ou primary/backup réplication), une seule copie, appelée copie primaire, reçoit la requête d'un client et l'exécute. Les autres copies, nommées copies secondaires ou copies de sauvegardes, ne sont là que pour prendre le relais en cas de défaillance de la copie primaire. Afin d'assurer la cohérence des copies secondaires, la copie primaire leur diffuse régulièrement son nouvel état (un point de reprise). Cette méthode assure que toutes les copies ont la même valeur même si les traitements sont non déterministes, car il y a diffusion de l'état de la copie primaire vers les copies secondaires et non-exécution des requêtes [30].

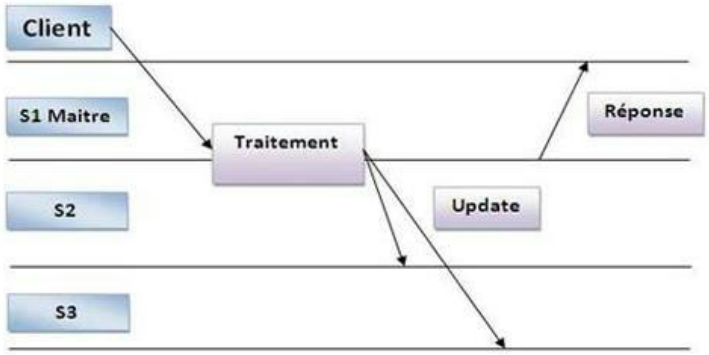

FIGURE 2.5 – Protocole de réplication passive

2.2.7.2 Réplication Active Dans ce protocole, toutes les copies jouent le même rôle. Elles reçoivent toutes la même séquence totalement ordonnée de requêtes de la part des clients, les exécutent de manière déterministe et renvoient la même séquence totalement ordonnée des réponses (Figure 2.6). La condition pour que toutes les copies reçoivent et exécutent toutes les requêtes dans le même ordre peut être décomposée en deux conditions distinctes :

1. Toutes les copies reçoivent les mêmes requêtes.
2. Toutes les copies exécutent les requêtes dans le même ordre relatif.[30]

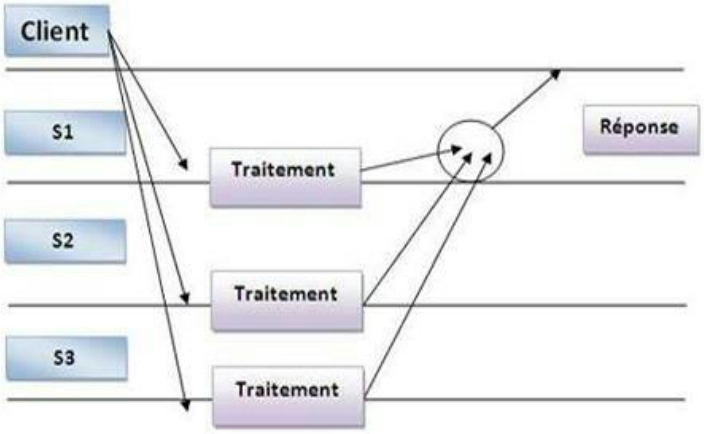

FIGURE 2.6 – Protocole de réplication active

2.2.7.3 Réplication Semi-active C'est un protocole hybride qui se situe entre les deux protocoles précédents, où toutes les copies exécutent en même temps les requêtes des clients, mais une seule copie (leader) d'entre elles émet la réponse. Les autres copies (suiveurs) mettent à jour leur état interne et sont donc étroitement synchrones avec le leader.[30]

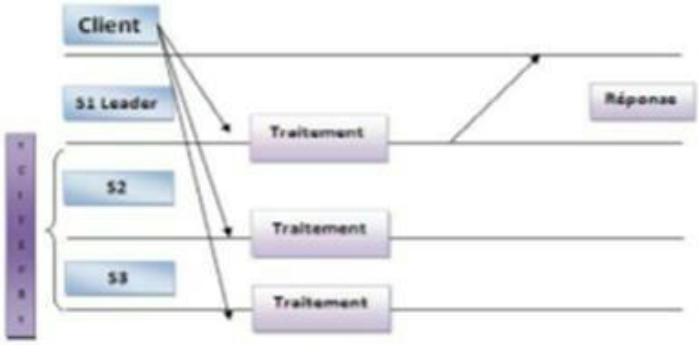

FIGURE 2.7 – Protocole de réplication semi-active

2.2.8 Les méthodes de réplication existantes dans la littérature

1-Stratégies de réplication dans les grilles

De nombreux travaux concernent l'ordonnancement de tâches indépendantes sur les plates-formes hétérogènes et distribuées. La complexité de la réplication dans les grilles de données a été présentée dans [31]. Les auteurs montrent que le problème est NP-complet et non-approximable et donnent deux approches pour le résoudre grâce à une simplification du problème.

Dans [32], les auteurs comparent plusieurs stratégies dynamiques de réplication dans un environnement de grille hiérarchique. Les stratégies sont comparées en mesurant (par simulation) le temps de réponse moyen et la bande-passante totale utilisée. Les deux stratégies qui obtiennent les meilleurs résultats sont *cascade* qui inonde l'arbre modélisant la grille un peu comme une fontaine en utilisant des mesures des besoins à chaque niveau. Cette stratégie fonctionne bien dans le cas où il n'y a pas beaucoup de localité. La seconde stratégie qui a obtenu de bons résultats dans le cas de requêtes aléatoires est *diffusion rapide* dans laquelle une donnée est copiée à chaque niveau sur le chemin vers le client.

Dans [33], les auteurs présentent l'algorithme *ADR* (Adaptive Data Réplication), conçu pour ajuster dynamiquement la réplication des données en fonction de leur utilisation. L'objectif principal de cet algorithme est d'optimiser les accès distants en lecture et en écriture en répliquant les données de manière stratégique, minimisant ainsi les copies inutiles tout en les rendant facilement accessibles. Lorsqu'il existe plusieurs répliques d'une même donnée, *ADR* permet la lecture à partir de n'importe quelle réplique, mais exige que toute opération d'écriture soit propagée à l'ensemble des répliques. À des intervalles réguliers, les nœuds stockant des données analysent les schémas d'accès en lecture et écriture pour déterminer s'ils doivent propager, supprimer ou échanger des données avec les nœuds voisins, en fonction des accès qui ont été effectués sur les données qu'ils stockent. Les auteurs ont démontré que dans des conditions d'accès régulières, l'algorithme *ADR* converge vers une réplication optimale. Cette approche adap-

tative offre des perspectives intéressantes pour l'efficacité des systèmes de stockage de données.

Dans l'article [34], un modèle économique de gestion des réplicas, incluant leur création et destruction, est présenté et évalué expérimentalement dans [35] . Ce modèle repose sur l'utilisation d'un protocole d'enchères par des agents situés sur chaque nœud de stockage pour choisir le réplica d'un fichier à utiliser. Lorsqu'une donnée est demandée sur un site, l'agent correspondant interroge les serveurs de stockage. Le serveur remportant l'enchère est celui proposant le prix le plus bas et reçoit le prix de la deuxième enchère la moins élevée. Si la donnée est présente chez un serveur interrogé, le prix est proportionnel au temps estimé pour le transfert de fichier entre ce serveur et le site demandeur. Si la donnée n'est pas présente, le serveur de stockage peut déclencher une enchère pour l'acquérir s'il estime que les revenus qu'elle va générer dépasseront son coût d'acquisition. Ce processus peut entraîner des enchères en cascade, supposant que les serveurs de stockage disposent de moyens de prédiction de l'utilisation des données pour estimer les revenus potentiels.

2-Stratégies de réplication dans le cloud

Les performances d'une stratégie de placement de répliques peuvent être affectées par des décisions telles que :

- Le modèle du système utilisé
- Le modèle de coût adopté
- Les critères utilisés comme métriques pour améliorer les performances dans un environnement de Cloud, où les demandes d'accès fréquentes sont placées sur des données à grande échelle, un temps de réponse réduit et une haute disponibilité sont essentiels pour les clients du Cloud.

Comme dans de nombreux systèmes, un certain nombre de stratégies de réplication de données ont été proposées [36] pour améliorer la qualité de service grâce à des contrats SLA (Service Level Agreement) dans les systèmes Cloud. Une partie des stratégies de réplication de données proposées à base de SLA se concentrent sur la minimisation de la consommation d'une certaine ressource de Cloud [37], par exemple : la bande passante, tandis que d'autres traitent de la réplication de données pour satisfaire le SLA sans tenir compte de l'impact monétaire des décisions de réplication [38]. Les garanties de performance, par exemple : temps de réponse, ne sont souvent pas offertes aux utilisateurs par les fournisseurs de Cloud dans le cadre du SLA. La plupart des stratégies de réplication de données dans le Cloud se concentrent uniquement sur la satisfaction de la disponibilité [39].

Il y a aussi un manque de garanties de performance dans les SLA pour les Cloud commerciaux proposés par Amazon [1], Google [2] et Microsoft [3]. Afin de combler cette lacune, plusieurs travaux ont été proposés [40] dans la littérature pour proposer des mécanismes permettant d'inclure des garanties de temps de réponse dans la SLA. En ce qui concerne la réplication des données, seules quelques études s'intéressent particulièrement à l'amélioration du temps de réponse [41]. En outre, encore moins de ces études [42] prennent en compte l'impact économique de la réplication des données.

Les sous-sections suivantes classent les stratégies de réplication de données existantes dans les systèmes de Cloud selon leurs objectifs. Étant donné que le temps de réponse et/ou la disponibilité sont des critères les plus considérés dans chaque

1. https ://aws.amazon.com
2. https ://cloud.google.com
3. https ://azure.microsoft.com

processus de réplication de données et que la contribution de cette thèse porte sur la satisfaction de ces critères, la classification est délibérément basée sur la satisfaction de l'objectif de la disponibilité ou la satisfaction de l'objectif du temps de réponse.

[43] ont proposé une stratégie de clustering k-means basée sur une matrice pour le placement de données dans des systèmes cloud. Dans leur stratégie, ils tentent de conserver les ensembles de données dans un seul centre de données, dans le but que lorsque les tâches sont ordonnancées dans ce centre de données, la plupart, sinon la totalité, des données dont elles ont besoin sont stockées localement. Leur idée était d'étudier et de déterminer la matrice de dépendance pour toutes les données d'application, ce qui montre les dépendances entre tous les ensembles de données, y compris les ensembles de données qui peuvent avoir des emplacements fixes.

Ensuite, ils ont appliqué une technique de clustering [44] pour grouper la matrice et la partitionner de telle sorte que les ensembles de données dans chaque partition soient fortement dépendants les uns des autres. Les résultats de la simulation indiquent qu'avec cette stratégie, le déplacement de données entre les centres de données est considérablement réduit par rapport à un placement de données aléatoire.

La gestion de la réplication dynamique à base de coût (cost-effective Dynamic Replication Management, CDRM) [45] souligne que le fait d'avoir trop de répliques n'augmente pas la disponibilité mais entraîne des rendements réduits. Le CDRM calcule et maintient un nombre minimum de répliques pour satisfaire un niveau de disponibilité donné. Dans le stockage en nuage considéré, les ensembles de données sont fragmentés en plusieurs blocs. Les auteurs calculent la disponibilité des ensembles de données en fonction de la disponibilité de chacun de leurs fragments.

Dans un autre travail similaire, Skute est présenté par [46] en tant que stratégie de réplication de données basée sur une économie virtuelle prenant en compte l'utilité marginale, l'utilisation du stockage et la charge de requêtes. Les nœuds virtuels agissent de manière autonome et annoncent périodiquement leur loyer aux autres nœuds. De plus, les nœuds accumulent également de la richesse en répondant à des requêtes et dépensent cette richesse sur d'autres nœuds pour y stocker des répliques, en fonction de leur loyer. Skute auto-organise la configuration de la réplique parmi les nœuds virtuels afin de minimiser les coûts de communication tout en maximisant les bénéfices économiques. Même si les stratégies de ce type utilisent une forme d'économie virtuelle, ce n'est pas un modèle de coût monétaire réel de la relation fournisseur-locataire [47].

L'augmentation de la disponibilité en répliquant les données populaires plus près des emplacements qui génèrent le plus de demandes est également un problème fréquemment étudié. Certaines stratégies proposent même des mécanismes pour prédire les futurs accès à partir des enregistrements d'accès historiques afin de répliquer les données de manière préventive afin de répondre à l'augmentation de la demande [48].

De plus, d'autres stratégies traitent de la popularité des données qui atteint un sommet pendant une courte durée, puis diminue progressivement [49]. Afin de faire face à l'explosion rapide de la popularité, ces stratégies de réplication de données répondent rapidement et modifient la configuration de la réplique en conséquence. L'augmentation de la localisation des données en fonction de la popularité diminue également la consommation du réseau [50].

Les stratégies de cette sous-section prennent généralement en compte la qualité

de service et le temps de réponse et ajustent de manière élastique le nombre de répliques d'une manière ou d'une autre pour garantir des garanties de performance à l'utilisateur. Certaines stratégies telles que RepliC [51] effectuent cette tâche dans un environnement de base de données élastique à de multiples utilisateurs. RepliC surveille l'utilisation du système par rapport aux changements de la charge de travail afin de gérer la variation en dirigeant les transactions vers les réplicas avec les ressources disponibles. Par conséquent, la stratégie RepliC satisfait la qualité de service avec des violations de SLA minimales. L'augmentation de la localisation des données, plus précisément la localisation spatiale grâce aux rendements de la réplication des données, permet d'améliorer les performances en plaçant stratégiquement les répliques plus près des sites demandeurs. Même si certaines stratégies ne considèrent pas directement la performance comme un objectif, une augmentation de performance est observée en conséquence de la réplication des données [52].

Une autre forme de croissance de localisation des données pour améliorer les performances se base sur la localité temporelle. De cette manière, certaines stratégies font valoir que les ensembles de données fréquemment consultés resteront probablement populaires à l'avenir [53] et décideront ce qu'il faut répliquer selon l'évaluation de la popularité mentionnée. C'est une stratégie simple mais efficace pour satisfaire les garanties de performance comme en témoigne la stratégie RTRM [54]. Lorsque l'accès à des données populaires entraîne un temps de réponse moyen supérieur à celui souhaité, ces ensembles de données sont répliqués pour résoudre le problème de performances. Ce qui est intéressant à propos de RTRM, c'est qu'il ne reproduit pas toutes les données, sauf les données populaires, afin de conserver l'utilisation des ressources.

3-Stratégies de réplication dans les Fog

Les stratégies de réplication dans le Fog Computing sont un domaine de recherche dynamique et crucial, visant à optimiser la disponibilité, la performance et la résilience des systèmes décentralisés. Le Fog Computing, qui étend le Cloud Computing vers les périphéries du réseau, présente des défis uniques en matière de gestion des données et de garantie de services fiables. Dans ce contexte, les chercheurs se penchent sur différentes approches de réplication pour répondre aux besoins variés des applications déployées dans des environnements Fog. Une des stratégies couramment étudiées est la réplication proactive, où les données sont promptement répliquées sur plusieurs nœuds du Fog pour réduire les temps de réponse et améliorer la résilience face aux pannes. Des travaux de recherche tels que ceux menés par **Gupta et al. (2020)** ont exploré l'impact de la réplication proactive sur la latence des requêtes et la disponibilité des données dans les environnements Fog. Leur étude a montré que la réplication proactive peut considérablement réduire les temps de réponse en distribuant les données de manière anticipée.[55] Parallèlement, la réplication réactive est une autre stratégie importante, où les données sont répliquées en réponse à des événements spécifiques tels que des pannes de nœuds ou des pics de charge.

Des chercheurs comme [56] ont examiné comment la réplication réactive peut être optimisée pour garantir une disponibilité élevée tout en minimisant les coûts de stockage et de traitement dans les environnements Fog à ressources limitées [57]. Leur travail met en lumière l'importance de synchroniser les mécanismes de réplication réactive avec la dynamique du réseau et des charges de travail.

En outre, la réplication hybride combine des éléments de réplication proactive et réactive pour tirer parti des avantages de chaque approche. Les recherches menées par [58] ont examiné les stratégies de réplication hybride dans le contexte du

Fog Computing, en mettant l'accent sur l'optimisation des performances tout en garantissant la cohérence des données et la réduction des coûts de gestion [59]. Leur étude montre comment une approche hybride peut s'adapter de manière dynamique aux conditions changeantes du réseau et aux exigences des applications.

En outre, les chercheurs explorent des techniques avancées telles que la réplication basée sur les politiques, où les décisions de réplication sont guidées par des règles définies par les utilisateurs ou les administrateurs.

Des travaux comme ceux de [60] ont examiné comment les politiques de réplication peuvent être personnalisées pour répondre aux besoins spécifiques des applications Fog [61], en prenant en compte des facteurs tels que la confidentialité des données, les exigences de performance et les contraintes de ressources. En complément à ces travaux, [62] ont introduit une stratégie de réplication dynamique dans le Fog Computing basée sur des considérations de coût et de disponibilité. Leur approche, appelée Cost-Effective Dynamic Réplication Management (CDRM), vise à maintenir un nombre optimal de répliques pour garantir un niveau de disponibilité spécifié tout en minimisant les coûts associés à la réplication [63]. En utilisant des modèles de calcul de disponibilité et des algorithmes d'optimisation des coûts, ils ont démontré une gestion efficace des répliques dans des environnements Fog complexes.

En conclusion, les stratégies de réplication dans le Fog Computing sont au cœur des efforts visant à garantir la disponibilité, la performance et la résilience des systèmes décentralisés. Les travaux de recherche continuent d'explorer de nouvelles approches, en intégrant des aspects tels que la réactivité aux événements, l'optimisation des coûts et la personnalisation des politiques pour répondre aux défis complexes posés par les environnements Fog. Ces avancées contribuent à l'évolution constante des pratiques de gestion des données et des services dans le domaine du Fog Computing.

2.2.9 Avantages de la réplication

Selon le type de réplication et les options sélectionnées, la réplication présente divers avantages. Toutefois, l'avantage principal de cette technique réside dans la disponibilité constante des données, accessibles à tout moment et en tout lieu. Les bénéfices liés à l'utilisation de la réplication peuvent être récapitulés de la manière suivante :

- **Disponibilité des données** : les données sont disponibles localement même en l'absence de toute connexion à un serveur central. En conséquence, l'utilisateur n'est pas coupé de ses données en cas de défaillance d'une connexion réseau.
- **Amélioration du temps de réponse** : l'amélioration du temps de réponse des requêtes d'interrogation est due à une meilleure bande passante suite à la réplication de données distantes.
- **Fiabilité** : la réplication permet aux applications de se prémunir contre les problèmes de pannes subites, volontaires ou involontaires, et contre le partitionnement. L'allocation de plusieurs copies d'une donnée offre une plus grande fiabilité aux applications.
- **Répartition de charge** : la présence d'un serveur unique pour la copie originale peut mener à une surcharge du serveur et de ses voies de communication. Cette congestion peut être une cause de temps de réponse de plus en plus longs jusqu'à l'écroulement du système. En se partageant la réception et le traitement des requêtes, des serveurs multiples distribués sur le réseau de manière appropriée évitent ce problème de surcharge.

2.2.10 Inconvénients de la réplication

La réplication des données dans les systèmes distribués présente plusieurs inconvénients, notamment :

- **Cohérence des répliques** : Le problème de la cohérence est le défi le plus complexe lié à la réplication des données. Les techniques de réplication ne garantissent pas automatiquement la cohérence entre les répliques [64], ce qui peut entraîner des copies différentes d'un ensemble de données à un instant donné sur différents nœuds. Pour éviter cela, il est nécessaire de définir un protocole de cohérence qui puisse garantir que toutes les répliques d'une même donnée sont cohérentes [65, 66].
- **Choix d'une réplique** : Il s'agit ici de sélectionner, parmi toutes les répliques d'une donnée, celle qui est la meilleure du point de vue de la consistance [67, 68].
- **Placement des répliques** : Ce problème consiste à choisir des emplacements physiques pour les répliques, alignés sur les objectifs des applications et de la réplication, tout en minimisant les coûts de stockage et d'accès aux données [69, 70, 71, 72]. Un modèle de coût est nécessaire pour évaluer les coûts des différentes options de placement [73, 74].
- **Coût** : La mise à jour ou la modification de la copie originale engendre la mise à jour de l'ensemble des copies. Ce genre de coût dépend du nombre de copies [75].

2.3 Conclusions

Ce chapitre représente un diaporama des méthodes de réplication existantes dans l'environnement de cloud et le Fog Computing. Les stratégies de réplication de données existantes sont orientées vers une plus large sélection de scénarios. De nombreuses stratégies existantes traitent de la disponibilité avec une minorité d'entre elles considérant également la performance. En ce qui concerne le fog, nous présentons notre contribution pour résoudre le problème de réplication dans les environnements Fog computing. Nous détaillons la stratégie mise en œuvre pour la création de ces répliques, ainsi que leur placement.

Chapitre 3

Stratégie Proposée

3.1 Introduction

Dans les chapitres précédents, nous avons exposé de manière exhaustive le concept de réplication dans les environnements Cloud/Fog Computing, ainsi que ses différents aspects et principes sous-jacents. Par ailleurs, nous avons réalisé une revue approfondie de la littérature existante, mettant en évidence divers travaux pertinents qui partagent des similarités contextuelles avec notre projet. L'objectif principal de notre recherche consiste à concevoir et à mettre en œuvre une nouvelle stratégie de réplication dynamique basée sur l'apprentissage automatique, spécifiquement adaptée à l'environnement du Fog et Cloud Computing.

Nous avons proposé deux variantes pour notre stratégie de réplication, la première est une solution centralisée et la deuxième est une solution totalement distribuée.

Ce chapitre est consacré à la conception détaillée de notre stratégie, dans un premier temps nous présentons l'architecture que nous avons élaborée,ensuite nous exposons en détail les différents concepts, phases et éléments nécessaires à au bon fonctionnement de nos deux solutions.

3.2 Description de la stratégie proposée

Dans le contexte du ***fog computing***, la réplication des données est essentielle pour garantir une disponibilité élevée des services dans les environnements distribués et centralisés. En dispersant les copies des données sur différents nœuds du réseau ***fog***, on réduit la dépendance à l'égard d'un seul point de défaillance, assurant ainsi une continuité de service même en présence de pannes locales ou de connexions intermittentes. De plus, la réplication peut contribuer à améliorer les performances en réduisant la latence et en offrant un accès plus rapide aux données pour les utilisateurs locaux.

L'apprentissage automatique est une branche de l'intelligence artificielle permettant aux ordinateurs d'apprendre à partir de données. Il existe trois types principaux : supervisé, non supervisé et par renforcement. L'apprentissage supervisé utilise des algorithmes comme la régression linéaire et les forêts aléatoires pour prédire des résultats à partir de données étiquetées. L'apprentissage non supervisé, avec des algorithmes comme ***K-means*** et ***DBSCAN*** qui sont utilisés dans notre stratégie, permet de découvrir des structures cachées dans des données non étiquetées. Pour le troisième type, l'apprentissage par renforcement, utilise des techniques comme ***Q-Learning*** et les ***Deep Q-Networks*** et permet aux agents de prendre des décisions en fonction de récompenses reçues.

L'apprentissage automatique peut être utilisé dans le domaine de la réplication des données en ***fog computing*** en prédisant les besoins futurs et en ajustant dynamiquement la distribution des données. Cela permet de réduire la latence et d'améliorer la disponibilité des services. Les avantages incluent une utilisation optimisée des ressources, une qualité de service accrue et une réduction des coûts opérationnels grâce à une gestion proactive du trafic réseau. Ainsi, l'intégration de l'apprentissage automatique rend le ***fog computing*** plus adaptatif et efficace.

À ce jour, il existe peu de travaux de recherche consacrés à la réplication dynamique des données basée sur l'apprentissage automatique. Bien que des études aient été réalisées sur la gestion adaptative des ressources et l'optimisation des bases de données distribuées, la réplication dynamique utilisant des techniques d'apprentissage automatique reste un domaine largement inexploré. Cette lacune dans la littérature souligne l'innovation et la pertinence de notre strategie.

Nous proposons une stratégie de réplication dynamique basée sur l'apprentissage. Cette approche utilise des algorithmes d'apprentissage automatique pour analyser et prédire les schémas de demande et d'accès aux données et repose sur un déclenchement intelligent du processus de réplication, conditionné par le volume de requêtes.La stratégie proposée est composée de plusieurs phases, la première phase traite les requêtes reçus par les utilisateurs, cette phase gère la vérification de la disponibilité du service demandé et , elle fait la recherche des données au sein du fog local ou des fogs voisins. La phase de réplication des données, et la phase de Suppression des données.

Dans ce travail, nous avons proposé une stratégie de réplication de donnée basée sur l'apprentissage où nous avons utilisé deux classifications : la classification locale avec l'algorithme ***DBSCAN*** et pour la classification globale nous avons utilisé l'algorithme ***K-means***. Nous avons élaboré deux solutions : la première est centralisée et la seconde est distribuée dans la solution centralisée. Initialement, chaque Fog effectue une classification locale, les nœuds Fog locaux utilisent l'algorithme ***DBSCAN*** pour regrouper les fichiers en clusters, il tient en compte des interactions locales entre les fichiers, ce qui permet une gestion fine des données au niveau du nœud Fog. Ensuite, ces clusters sont envoyés au Cloud, qui effectue une seconde classification à l'aide de l'algorithme ***K-means*** en fonction de la dépendance et de la fréquence d'accès, puis envoie cette classification à chaque Fog pour guider la réplication des fichiers. Cette approche capitalise sur la vision globale du Cloud pour optimiser la répartition des données.

En revanche, la deuxième approche est distribuée et se concentre sur la division des responsabilités entre les Fog. Dans cette approche, les Fog sont divisés en deux niveaux, le niveau inférieur est le niveau supérieur, Au niveau inférieur du réseau Fog, les nœuds utilisent l'algorithme ***DBSCAN*** pour regrouper les données en clusters et l'algorithme d'Arbre pour assurer un flux efficace et sécurisé des clusters de chaque nœud Fog vers le niveau supérieur pour une deuxième classification, qui est la responsable de la classification globale des données en exploitant l'algorithme ***K-means***. En fonction de leurs fréquences d'accès respectives.

Les avantages de notre stratégie incluent une meilleure utilisation des ressources, une réduction des temps de latence, et une adaptation continue aux variations des charges de travail. En exploitant les capacités de l'apprentissage, la stratégie assure une gestion plus réactive et précise des processus de réplication, ce qui se traduit par une amélioration globale des performances du système.

3.3 Les Étapes de la Stratégie

Notre stratégie est composée de plusieurs phases : la phase de traitement d'une requête, la phase de réplication des données et de suppression des données.

Dans la section suivante, nous allons détailler les différentes phases de notre solution :

3.3.1 La phase de Traitement d'une requête

Dans cette section, nous présentons notre première contribution, qui consiste en une stratégie détaillée couvrant l'ensemble du processus, depuis la réception d'une requête jusqu'à l'exécution des données. Les clients se connectent aux Fogs les plus proches géographiquement.

À la réception d'une requête q d'un client, le Fog correspondant vérifie si des

fichiers sont nécessaires pour son exécution. Dans le cas où aucun fichier n'est requis, l'accès est autorisé pour traiter la demande du client. En revanche, si des fichiers sont nécessaires, le Fog lance une recherche pour les trouver. Si les fichiers sont disponibles localement, le Fog évalue le temps de réponse (TR_i). Si le TR_i est inférieur au seuil défini dans le contrat SLA pour garantir la qualité de service, le traitement de la demande du client est approuvé. Si le TR_i dépasse ce seuil, le Fog déclenche un processus de réplication local pour réduire le temps de réponse.

Dans le cas où les fichiers ne sont pas disponibles localement, le Fog explore la liste des autres Fogs pour les localiser. Une fois les fichiers trouvés, l'estimation du temps de réplication des fichiers est calculée. L'algorithme va vérifier le temps TR_i avec le seuil , si le temps TRi est inférieure, le traitement de la demande du client doit être approuvé. Sinon, le Fog déclenche un processus de réplication local pour réduire le temps de réponse.

Dans le cas contraire, où les fichiers ne peuvent pas être trouvés dans la liste des Fogs voisins, le Fog va créer des répliques à partir du Cloud.

La Figure suivante illustre un diagramme d'activité qui détaille le fonctionnement du traitement d'une requête :

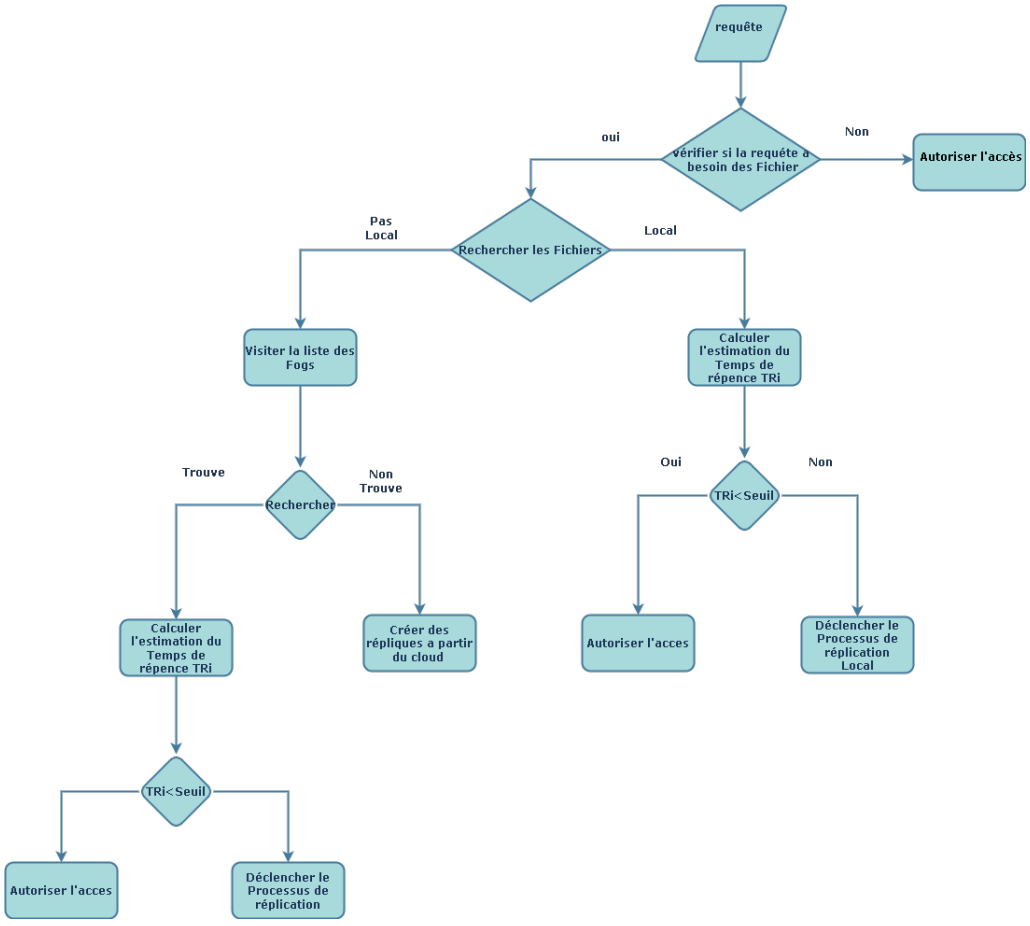

FIGURE 3.1 – Diagramme d'activité du traitement d'un requête

Pour le calcul du temps de réponse d'une requête (TR_i), Noun avons utilisé une équation qui prend en compte divers facteurs tels que le temps de traitement local, le temps de transmission des données, et le temps d'attente. Voici une équation générale pour estimer le temps de réponse (TR_i) :

$$TR_i = T_{\text{traitement}} + T_{\text{transmission}} + T_{\text{attente}} \tag{3.1}$$

Avec :

- $T_{\text{traitement}}$: c'est le temps nécessaire pour traiter la requête localement sur le Fog.
- $T_{\text{transmission}}$: c'est le temps nécessaire pour transmettre les données nécessaires à la requête entre les nœuds de Fog.
- T_{attente} : c'est le temps d'attente avant que la requête puisse être traitée, ce qui peut inclure la latence due à la mise en file d'attente des requêtes.

3.3.2 La phase de réplication

Nous utilisons l'apprentissage automatique pour concevoir une réplication dynamique. Cette approche intelligente optimise la gestion des ressources et des données distribuées. En utilisant des algorithmes sophistiqués, elle s'adapte en continu aux besoins changeants, améliorant ainsi l'efficacité des ressources et réduisant les temps de latence [76].

Pour cela, nous proposons deux solutions distinctes :

3.3.2.1 Approche Centralisée

Notre stratégie de réplication intègre un processus d'apprentissage reposant sur deux classifications : la une classification locale avec **DBSCAN** et une autre classification globale avec **K-Means**.

L'architecture de cette approche est structurée horizontalement en trois niveaux (**Cloud, Fog, IoT**) et verticalement en plusieurs régions distinctes (**région 1, région 2, etc.**). Les nœuds du Fog sont interconnectés via des liens de latence, chaque région disposant d'un ou plusieurs de ces liens en fonction de sa topologie particulière.

Ci-dessous, l'architecture est schématisée comme suit :

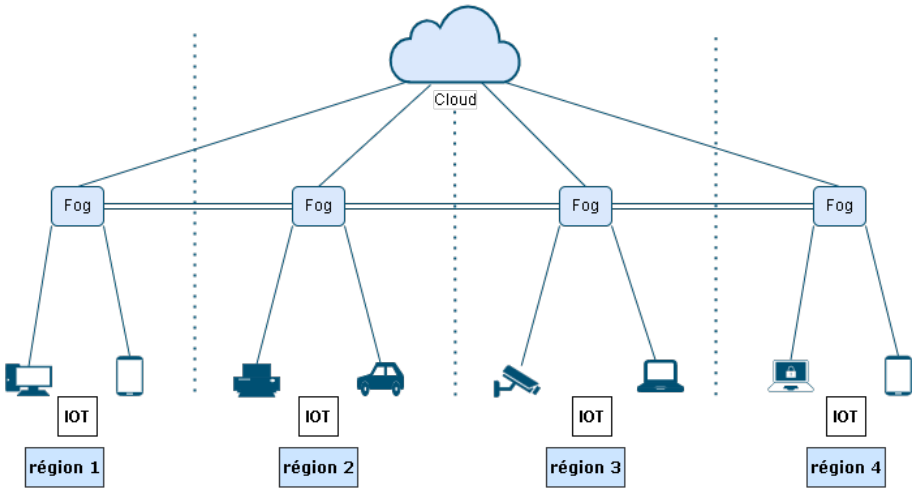

FIGURE 3.2 – Schéma de l'architecture centralisée

I- Classification Locale avec DBSCAN

Dans cette première phase, les nœuds Fog locaux utilisent l'algorithme **DBSCAN** proposé par Martin Ester et al. en 1996. C'est un algorithme de clustering non supervisé très connu basé sur la densité d'applications avec bruit qui fonctionne sur l'hypothèse de regrouper les fichiers en clusters en fonction de leur dépendance.

DBSCAN recherche les objets principaux, c'est-à-dire les objets qui ont des voisinages denses.Il relie les objets centraux et leurs voisins pour former des régions denses appelées clusters. Il regroupe les points de données densément connectés en un seul cluster. Il peut identifier les clusters dans de grands ensembles de données spatiales en examinant la densité locale des points de données. La caractéristique la plus intéressante du clustering **DBSCAN** est qu'il est robuste aux valeurs aberrantes.

L'algorithme **DBSCAN** nécessite deux paramètres : epsilon et minPoints.Epsilon est le rayon du cercle à créer autour de chaque point de données pour vérifier la densité et minPoints est le nombre minimum de points de données requis à l'intérieur de ce cercle pour que ce point de données soit classé comme point central [77].

Les points dans **DBSCAN** sont classifiés en trois catégories : points centraux, points de bordure et points de bruit. Un point central a au moins MinPts voisins dans son voisinage ϵ, ce qui en fait un noyau de clusters. Les points de bordure sont des voisins de points centraux, mais n'ont pas suffisamment de voisins pour être des points centraux eux-mêmes. Les points de bruit ne font partie d'aucun cluster et sont considérés comme des anomalies.

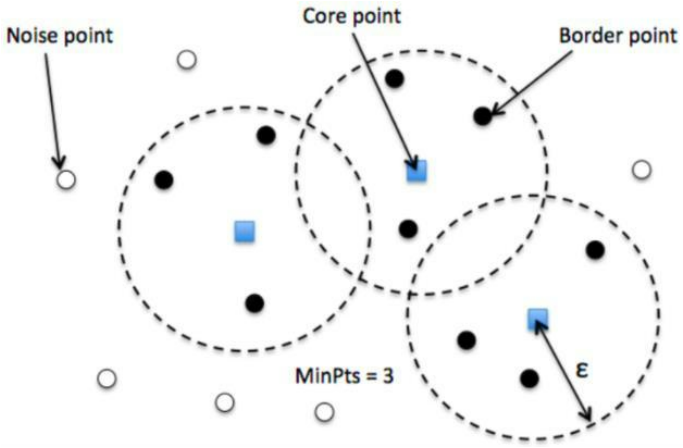

FIGURE 3.3 – Algorithme de DBSCAN

Voici les étapes principales de l'algorithme DBSCAN :

1. **Construction de la Matrice de Connectivité**
   - Nous construisons une matrice de connectivité où chaque ligne représente une requête envoyée à un Fog et chaque colonne représente un fichier susceptible d'être demandé. Chaque élément de la matrice est un booléen indiquant si un fichier est demandé dans une requête spécifique.
   - Cette matrice reflète les schémas de demande des fichiers par les Fogs :

	Fichier 1	Fichier 2	Fichier i
Requête 1	Vrai/faux	Vrai/faux	Vrai/faux
Requête 2	Vrai/faux	Vrai/faux	Vrai/faux
Requête 3	Vrai/faux	Vrai/faux	Vrai/faux
Requête i	Vrai/faux	Vrai/faux	Vrai/faux

TABLE 3.1 – La matrice de connectivité

2. **Application de DBSCAN**

- Nous appliquons ensuite l'algorithme DBSCAN à la matrice de connectivité. Nous identifions les clusters de fichiers qui sont souvent demandés ensemble. Ces clusters révèlent les relations de dépendance entre les fichiers et permettent une meilleure compréhension des besoins de stockage et d'accès aux données.

La figure suivante résume les étapes d'application de **DBSCAN** :

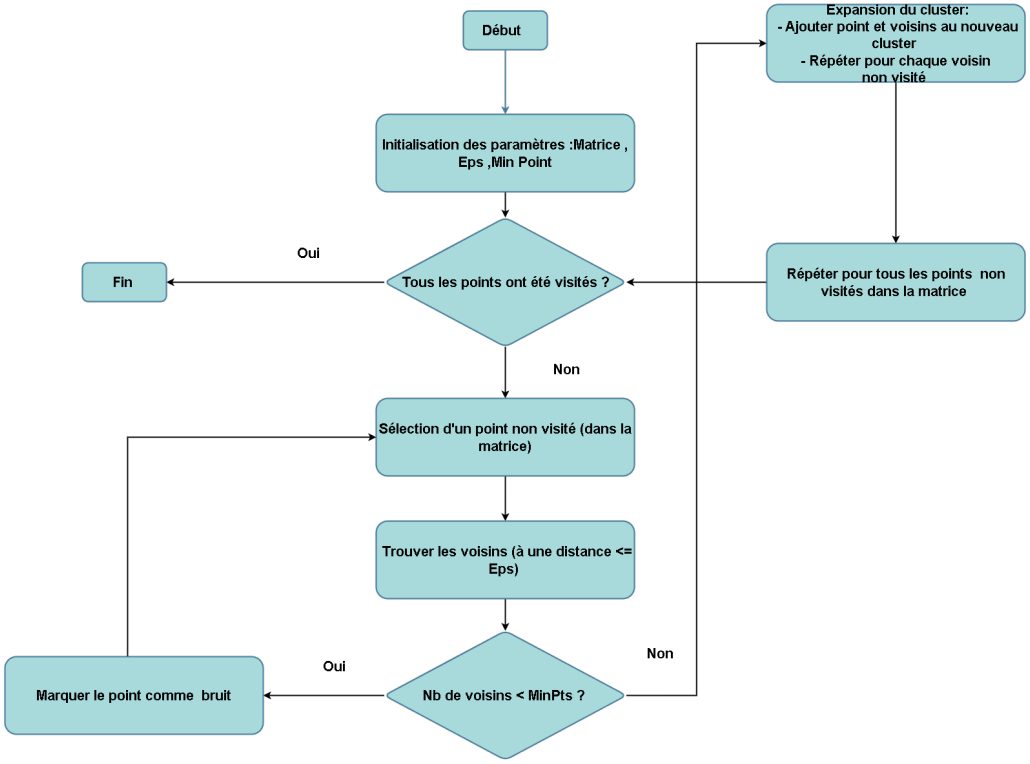

FIGURE 3.4 – Les étapes de l'algorithme DBSCAN

II- Classification Globale avec K-Means

Pour renforcer la précision de la classification et obtenir une vision globale plus détaillé, nous entreprenons une seconde classification sur les clusters formés localement par l'algorithme DBSCAN. Les résultats de **DBSCAN** sont envoyés au Cloud qui est chargé d'effectuer cette seconde classification à l'aide de l'algorithme **K-Means**.

Le clustering K-moyenne ou K-means a été proposé en premier lieu par J. MacQueen [78]. C'est l'un des algorithmes de clustering (de partitionnement) les plus connus et les plus utilisés pour reconnaître automatiquement des groupes d'objets similaires dans la base de données. L'algorithme segmente les données (objets) en K groupes ou clusters prédéfinis, spécifiés par l'utilisateur, de façon à ce que les données similaires soient dans le même groupe.

Cet algorithme vise à réduire la distance entre les points de données et leurs centroïdes au sein d'un cluster. Il représente la forme la plus basique des algorithmes de partitionnement basé sur les centroïdes.

Voici les étapes principales de l'algorithme K-Means :

II-1 Création de la Matrice Globale

Lorsque le cloud reçoit les clusters des différents nœuds Fog, il crée une matrice où chaque ligne représente un fichier et chaque colonne représente les clusters reçus de chaque Fog.

La matrice indique la fréquence d'accès de chaque fichier dans chaque cluster.Cela permet au cloud de visualiser et d'analyser l'activité des fichiers de manière centralisée, facilitant ainsi une gestion optimale de la réplication des données pour garantir leur disponibilité et performance à travers le réseau Fog.

Cette matrice permet au cloud de voir la répartition des fréquences d'accès aux fichiers à travers les clusters des différents nœuds Fog, aidant à prendre des décisions éclairées pour la réplication et la gestion des données dans l'ensemble du réseau.

La matrice créée au niveau du cloud pourrait ressembler à ceci :

	Fog 1			**Fog 2**			**Fog i**		
	Cluster 1	**Cluster 2**	**Cluster i**	**Cluster 1**	**Cluster 2**	**Cluster i**	**Cluster 1**	**Cluster 2**	**Cluster i**
Fichier 1	Fréquence	Fréquence	Fréquence	Fréquence	Fréquence	Fréquence	Fréquence	Fréquence	Fréquence
Fichier 2	Fréquence	Fréquence	Fréquence	Fréquence	Fréquence	Fréquence	Fréquence	Fréquence	Fréquence
Fichier 3	Fréquence	Fréquence	Fréquence	Fréquence	Fréquence	Fréquence	Fréquence	Fréquence	Fréquence
Fichier i	Fréquence	Fréquence	Fréquence	Fréquence	Fréquence	Fréquence	Fréquence	Fréquence	Fréquence

FIGURE 3.5 – Matrice de fréquence d'accès des fichiers au niveau du cloud

II-2 Application de K-means

Nous effectuons une seconde classification en utilisant l'algorithme **K-Means**, ce qui implique l'application de cet algorithme à la matrice représentant la distribution des fichiers dans les clusters provenant de différents nœuds Fog.

Les clusters obtenus à partir de cette classification révèlent les relations de dépendance entre les fichiers provenant de différents nœuds Fog, fournissant ainsi une vue consolidée de la structure des données.

La figure suivante résume les étapes d'application de l'algorithme de **K-means** :

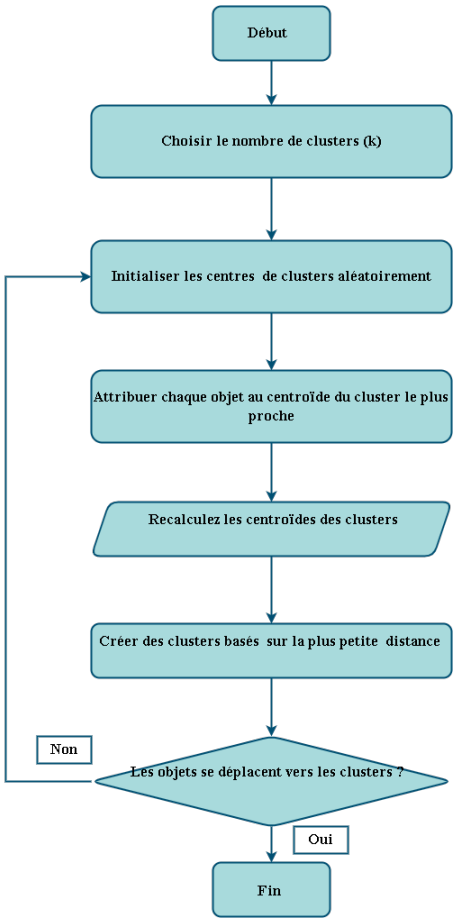

FIGURE 3.6 – les étapes d'application de l'algorithme de k-means

Après avoir obtenu ces clusters, nous calculons la fréquence totale de chaque fichier dans chaque cluster. Cette étape nous permet de comprendre l'activité relative de chaque cluster en termes de fréquence d'accès des fichiers. Ensuite, nous classifions les clusters en trois catégories distinctes : *"hot"*, *"warm"* et *"cold"*, en fonction de leur fréquence totale.

Les règles d'association pour ces catégories pourraient être définies de la manière suivante :

- **Hot** : Les clusters avec une fréquence d'accès élevée par rapport aux autres clusters sont classés comme *"hot"*. Ces clusters ont une activité très élevée par rapport à d'autres clusters.
- **Warm** : Les clusters avec une fréquence d'accès modérée par rapport aux autres clusters sont classés comme *"warm"*. Ils ne sont pas aussi actifs que les clusters *"hot"* mais sont plus actifs que les clusters *"cold"*.
- **Cold** : Les clusters avec une fréquence d'accès relativement faible par rapport aux autres clusters sont classés comme *"cold"*. Ces clusters ont une activité très faible par rapport aux autres clusters.

Pour calculer le ratio de fréquence et classifier les clusters comme *"hot"*, *"warm"* ou *"cold"* pour chaque fog, nous utilisons la formule suivante :

$$frequency\ ratio = \frac{clusterFrequency - minFrequency}{maxFrequency - minFrequency} \tag{3.2}$$

La figure suivante montre les étapes de classifications des clusters selon leur fréquence dans le cloud :

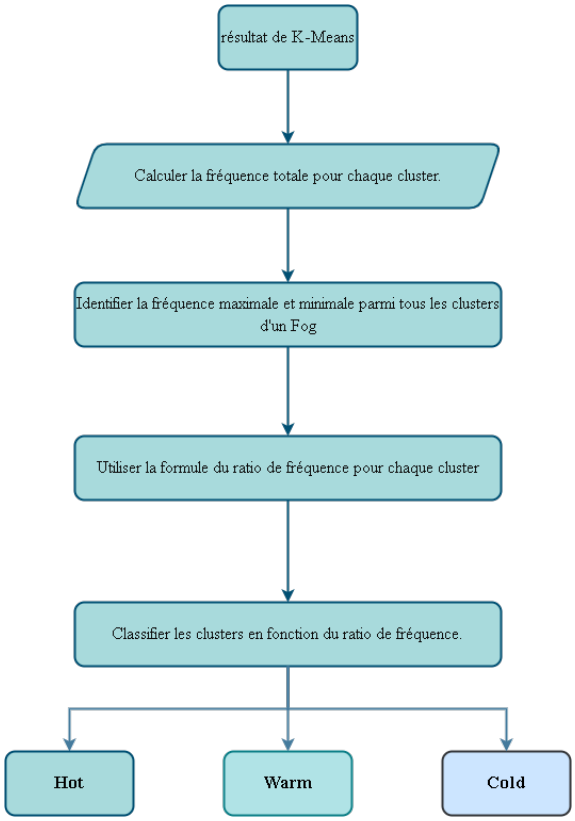

FIGURE 3.7 – les étapes de classifications des clusters selon la fréquence

Une fois les clusters classés, les résultats sont envoyés aux nœuds de Fog pour déclencher le processus de réplication, où les fichiers des clusters "hot" seront répliqués plus fréquemment que ceux des clusters "warm" ou "cold", garantissant ainsi une disponibilité et des performances optimales des données à travers le réseau Fog.

L'approche centralisée offre une gestion unifiée des données dans le réseau Fog en consolidant les résultats locaux au niveau du cloud. Cela permet une vue d'ensemble intégrée des données, facilitant ainsi une gestion efficace de la réplication des données et des analyses globales. L'avantage principal de cette approche réside dans l'utilisation de la puissance de calcul centralisée du cloud pour réaliser des analyses sophistiquées et des décisions informées, conduisant ainsi à une optimisation globale du réseau Fog.

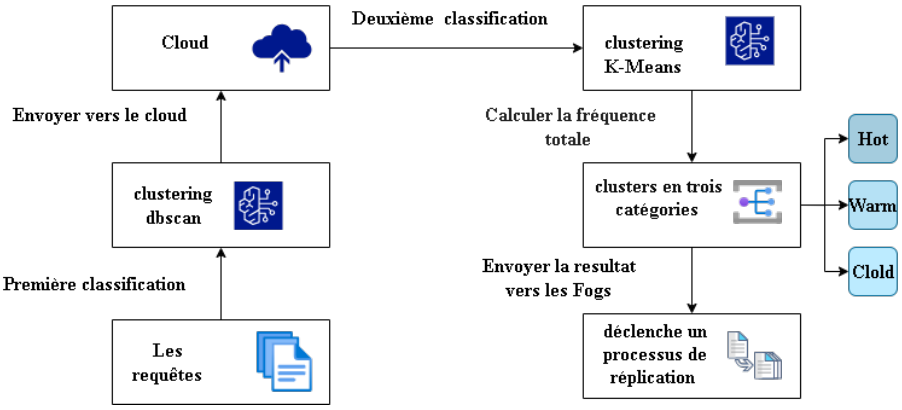

FIGURE 3.8 – Schéma général de classification centralisée

- La figure **3.8** présente de manière détaillée notre première approche centralisée, incluant l'utilisation des algorithmes ***DBSCAN*** et ***K-means*** pour le clustering.

3.3.2.2 Approche Distribuée

Le Cloud Computing implique une latence plus élevée car les données doivent être envoyées vers des centres de données distants, ce qui peut entraîner des retards dans le traitement de données en temps réel. En revanche, si nous rapprochons le traitement des données du bord du réseau, cela permet de réduire la latence et d'améliorer la réactivité des application critiques en temps réel.

Dans la deuxième proposition qui est une solution totalement distribuée, nous suivons un processus similaire en utilisant les mêmes algorithmes de classification de l'approche centralisés. Les fog de traitement sont divisés en deux niveaux :le niveau inférieur utilise le ***DBSCAN*** et le niveau supérieur utilise ***K-means***.

L'architecture de cette approche est organisée en trois niveaux distincts : le niveau **IoT**, le niveau Fog inférieur et le niveau Fog supérieur.

Cette architecture est délibérément dépourvue d'une intégration au Cloud Computing.

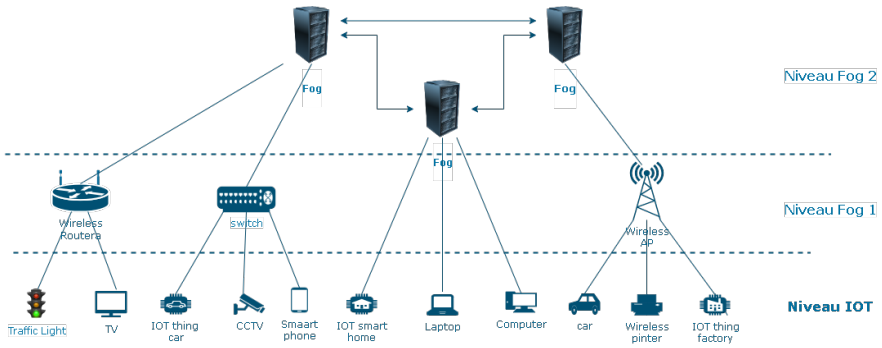

FIGURE 3.9 – intégration au Cloud Computing

I-Classification des Données dans le Niveau Inférieur

Au niveau inférieur du réseau Fog, chaque nœud de niveau inférieur construit une matrice de connectivité. Nous utilisons ensuite l'algorithme **DBSCAN** pour identifier des clusters à partir de cette matrice. Ces clusters locaux sont ensuite transmis aux nœuds de niveau supérieur pour une deuxième classification, permettant une analyse plus approfondie des données agrégées à partir de différents nœuds Fog.

Pour faciliter le transfert des informations et des clusters entre les différents nœuds Fog, nous utilisons l'algorithme d'Arbre basé sur les Algorithmes de Vagues qui fonctionne avec **N** fogs, chaque fog ne connaissant que l'identifiant de ses voisins. Les liens entre les fogs sont bidirectionnels, et l'algorithme est à la fois symétrique et non centralisé ni décentralisé.

Son principe de fonctionnement est le suivant : un fog qui a reçu un message de tous ses voisins, sauf un, envoie un message à ce dernier. Étant donné qu'un fog ne possédant qu'un voisin est une feuille de l'arbre, ces feuilles sont les fog initiateurs. Il est impératif que toutes les feuilles, à l'exception possible d'une, soient des initiateurs. Lorsqu'un fog reçoit un message de tous ses voisins, il prend une décision en conséquence. Cet algorithme est utilisé pour assurer un flux efficace et sécurisé des clusters de chaque nœud Fog vers le niveau supérieur.

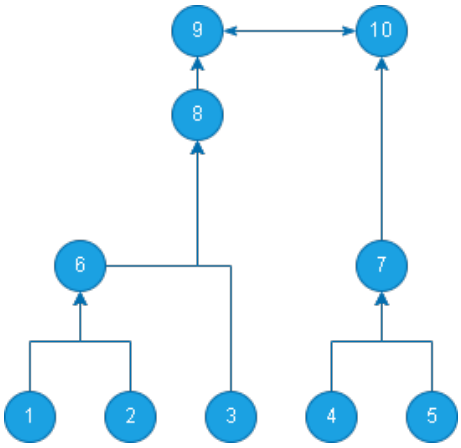

FIGURE 3.10 – Algorithme d'arbre

Le processus se déroule par étapes, où les clusters formés dans chaque Fog se déplacent de manière séquentielle d'un Fog à l'autre jusqu'à atteindre le deuxième niveau de Fog, c'est-à-dire le niveau supérieur.Cette approche garantit une communication efficace et coordonnée entre les différents nœuds Fog, tout en assurant une gestion optimale des données et une répartition équilibrée des charges de travail dans l'ensemble du réseau Fog.

II-Classification des Données dans le Niveau Supérieur

Au niveau supérieur du réseau Fog, les clusters de données provenant des niveaux inférieurs sont réceptionnés pour une classification globale. Une synchronisation efficace entre les différents niveaux de nœuds est cruciale pour la gestion de ces données. Pour garantir cette synchronisation, nous utilisons un algorithme d'arbre basé sur les algorithmes de vagues.

Lorsqu'un fog reçoit tous les clusters de données provenant des niveaux inférieurs de tous ses voisins, il crée une matrice globale. Cette matrice indique la fréquence d'accès de chaque fichier dans chaque cluster, permettant au fog de visualiser et d'analyser l'activité des fichiers de manière centralisée.

Sur cette matrice globale, nous appliquons l'algorithme K-Means pour regrouper les fichiers selon leur fréquence d'accès. Après avoir obtenu ces clusters, nous calculons la fréquence totale de chaque fichier dans chaque cluster. Cette étape permet de comprendre l'activité relative de chaque cluster en termes de fréquence d'accès des fichiers.

Nous classons ensuite les clusters en trois catégories distinctes :

- **Hot** : Clusters avec une fréquence d'accès élevée.
- **Warm** : Clusters avec une fréquence d'accès moyenne.
- **Cold** : Clusters avec une fréquence d'accès faible.

Une fois que les résultats de la classification sont obtenus, ils sont partagés avec les nœuds de Fog voisins au niveau supérieur du réseau. De là, ces résultats sont propagés vers les niveaux inférieurs du réseau, permettant une diffusion graduelle des informations de classification à travers toute l'infrastructure.

Cette méthode de partage assure que chaque nœud de Fog, quelle que soit sa position dans la hiérarchie du réseau, dispose des informations mises à jour sur la classification des clusters. Cette synchronisation ascendante et descendante des résultats garantit une gestion uniforme de la réplication des données et des performances du réseau Fog, favorisant ainsi une efficacité globale accrue et une utilisation optimale des ressources. L'approche distribuée se concentre sur la réduction de la latence et la réduction de la dépendance au cloud en favorisant le traitement local des données au niveau des nœuds de Fog.

Cette approche offre une réduction significative de la latence en évitant de transmettre directement les résultats au cloud, ce qui est crucial pour les applications nécessitant des réponses rapides. De plus, en réduisant la dépendance au cloud, elle offre une meilleure résilience et une plus grande flexibilité dans des environnements où la connectivité au cloud est limitée ou coûteuse.

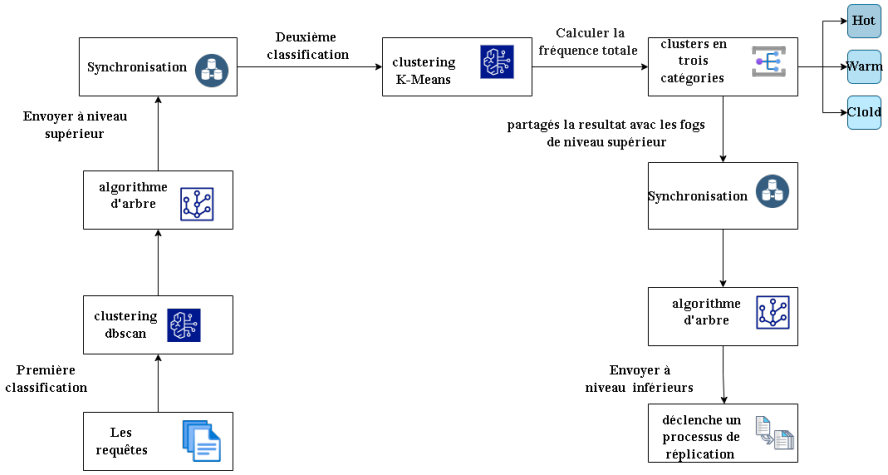

FIGURE 3.11 – Schéma général de classification distribuée

Figure **3.11** offre une représentation détaillée de la deuxième approche distribuée, mettant en lumière l'utilisation des algorithmes ***DBSCAN*** et ***K-means*** pour le clustering, ainsi que le processus de synchronisation facilité par un algorithme d'arbre.

3.4 Les nombres et le placement des répliques

Notre stratégie de réplication de données répond à des questions essentielles telles que :

3.4.1 Les nombres des répliques

Déterminer le nombre de répliques pour chaque donnée est un aspect intéressant de la réplication de données. Trop de répliques peuvent entraîner un gaspillage de ressources, tandis que trop peu de répliques ne suffisent pas à satisfaire le niveau de fiabilité attendu. Notre stratégie calcule le nombre de répliques nécessaire pour assurer une certaine disponibilité de données et augmente la fiabilité du Cloud. L'idée générale est de créer de nouvelles répliques à l'arrivée des requêtes. Cette création vérifie certaines conditions comme le budget et le coût de réplication.

Le nombre de répliques (NR) nécessaire est calculé comme suit :

$$\boldsymbol{NR} = \sqrt{\text{Fréquence d'accès au fichier dans le cluster}} \tag{3.3}$$

3.4.2 Le placement des répliques :

Une fois les répliques générées en suivant les deux approches : centralisée et distribuée, leur placement dans les environnements de fog est crucial et se fait selon des critères spécifiques. Tout d'abord, il faut identifier les nœuds disposant de suffisamment d'espace de stockage pour accueillir la nouvelle réplique. Ensuite, les nœuds qui ne possédent pas des répliqués sont sélectionnés. Enfin, la réplique est positionnée dans le nœud où la demande pour ces données est la plus fréquente, en prenant en compte la fréquence d'accès de chaque cluster.

3.5 La phase de suppression

Pour optimiser l'utilisation de l'espace disque et préserver la cohérence du système, les données expirées sont d'abord identifiées puis supprimées. Suite à cela, une notification de suppression est envoyée aux nœuds Fog concernés, qui doivent ensuite mettre à jour leurs données en conséquence.

Ce processus est détaillé dans le diagramme d'activités ci-dessous, illustrant les étapes clés de suppression et de mise à jour des données au sein du réseau.

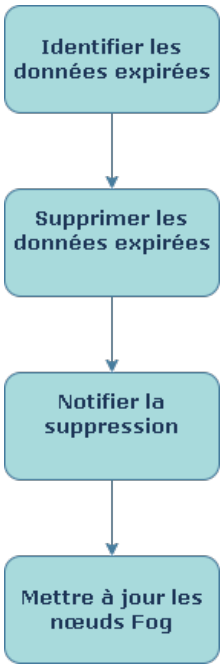

FIGURE 3.12 – Diagramme d'activité de suppression des données

3.6 Conclusion

Dans ce chapitre, nous avons décrit le fonctionnement de notre stratégie de réplication de données dans l'environnement de Cloud et de Fog computing. Notre stratégie de réplication dynamique intègre des techniques d'apprentissage automatique, utilisant deux algorithmes de classification : DBSCAN pour une classification locale et K-means pour une classification globale. Nous avons présenté deux solutions : une centralisée basée sur le Cloud et l'autre totalement distribuée.

Cette approche présente plusieurs avantages. L'utilisation de l'apprentissage automatique optimise la précision et l'efficacité de la réplication en s'adaptant aux variations de la charge de travail. La classification locale avec DBSCAN améliore la réactivité au niveau des nœuds individuels, tandis que la classification globale avec K-means assure une optimisation à grande échelle.

Chapitre 4

IMPLEMENTATION

4.1 Introduction

Dans le chapitre précédent, nous avons présenté notre stratégie de réplication de données basée sur l'apprentissage automatique dans l'environnement Fog computing et nous avons détaillé son fonctionnement.

Ce chapitre est consacré à la phase de mise en œuvre de cette stratégie dans le but d'évaluer les performances et de valider la stratégie proposée.

Nous commencerons par décrire l'environnement dans lequel nous avons construit notre simulateur IfogSim, puis nous discuterons et analyserons les résultats que nous avons obtenus.

4.2 L'environnement de développement

Afin d'évaluer notre stratégie de réplication de données, fondée sur l'apprentissage automatique dans les environnements de Fog, nous avons conçu le simulateur IFogSim. Ce simulateur a été spécialement adapté pour faciliter l'expérimentation et la validation de notre approche. Nous avons utilisé l'environnement de travail suivant pour cette implémentation :

- **Caractéristiques matérielles et logicielles de l'ordinateur utilisé :** Nous avons développé notre application sur une machine avec un processeur Intel(R) Core (TM) i3-6100U CPU @ 2.30GHz et d'une capacité mémoire de 8,00 Go. Le simulateur est sous Windows 10 64 bits.
- **Simulateur utilisé :** iFogSim.
- **Langage utilisé :** Java.
- **IDE utilisée :** Eclipse.

4.3 Langage de programmation Java

Java signifie café en slang (**argot américain**). Le langage Java est un langage de programmation informatique orienté objet créé par James Gosling et Patrick Naughton, employés de Sun Microsystems, avec le soutien de Bill Joy (cofondateur de Sun Microsystems en **1982**), présenté officiellement le **23** mai **1995** au SunWorld.

Le but était d'avoir un langage de développement simple et portable sur plusieurs systèmes d'exploitation tels que UNIX, Windows, Mac OS ou GNU/Linux, avec ou sans modifications, afin de programmer tous les processeurs (ordinateurs ou appareils électroménagers, ...) tout en veillant à la robustesse, la compatibilité, la petite taille du runtime ou des codes générés et aussi facilité de programmation. Java reprend la syntaxe de **C++** tout en le simplifiant et offre aussi un ensemble de classes pour développer des applications de types très variés (réseau, interface graphique, multi-tâches, etc.).

Java comprend bien d'autres aspects (programmation graphique, applets, programmation réseau, multi-tâches). Le langage Java a connu plusieurs évolutions depuis le JDK (Java Development Kit) **1.0**. À partir de la version Java **1.2**, on parle de Java 2 avec 2 grandes innovations : l'API graphique Swing est intégrée , et la machine virtuelle Java de Sun inclut un compilateur "Juste à temps" (Just in Time)[79].

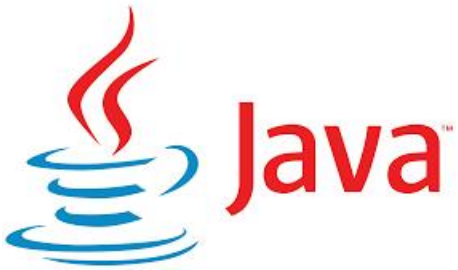

FIGURE 4.1 – Langage Java

4.4 Environnement de développement

4.4.1 Eclipse

Eclipse est un environnement de développement intégré (**EDI**), placé en Open Source par Sun en novembre **2001**. En plus de Java, Eclipse permet également de supporter différents autres langages, comme **Python**, **C**, **C++**, **JavaScript**, **XML**, **Ruby**, **PHP** et **HTML** téléchargeable du site : `https://Eclipse.org/downloads`. Il comprend toutes les caractéristiques d'un IDE moderne (éditeur en couleur, projets multi-langage, refactoring, éditeur graphique d'interfaces et de pages Web).

Conçu en **Java**, **Eclipse** est disponible sous Windows, Linux, Solaris, MacOs ou sous une version indépendante des systèmes d'exploitation (requérant une machine virtuelle Java). De plus, **Eclipse** est écrit en Open Source, téléchargeable directement du site `http://java.sun.com`. Il est puissant et compatible avec toutes les nouvelles technologies **Java** (les technologies **Java EE**, les bases de données, **UML**, **XML**, ...)[80].

FIGURE 4.2 – Eclipse

4.4.2 Simulateur iFogSim

iFogSim est un outil de simulation open source basé sur Java pour simuler des scénarios de calcul de fog computing. Il est développé par Harshit Gupta et l'équipe du Cloud Computing and Distributed Systems (**CLOUDS**) Lab de l'Université de Melbourne en Australie [81].

iFogSim offre des extensions permettant de simuler un environnement Fog computing personnalisé avec un grand nombre de nœuds Fog et d'appareils IoT (par exemple, des capteurs et des actionneurs). Les classes **iFogSim** sont annotées de telle manière que les utilisateurs, n'ayant pas de connaissance préalable de

CloudSim, puissent facilement définir les politiques d'infrastructure, de placement des services et d'allocation des ressources pour le Fog computing. **iFogSim** applique le modèle Sense-Process-Actuate ainsi que le flux de données distribué pour simuler n'importe quel scénario d'application dans l'environnement Fog.

4.4.2.1 Les composants de iFogSim

iFogSim est composé de 3 composants de base :

I-Composants physiques ce sont les modèles des équipements physiques trouvés dans une infrastructure de Fog (par exemple serveurs, capteurs, actionneurs, ...) :

- **FogDevice** : cette classe spécifie les caractéristiques matérielles des appareils Fog et leurs connexions à d'autres dispositifs Fog, capteurs et actionneurs. Ayant été réalisés par extension à partir de la classe `PowerDatacenter` dans `CloudSim`, les principaux attributs de la classe `FogDevice` sont mémoire accessible, processeur, taille de stockage, bandes passantes de liaison montante et descendante (définissant la capacité de communication des appareils Fog). Les méthodes de cette classe définissent comment les ressources d'un périphérique Fog sont planifiées entre les modules d'application qui s'exécutent dessus et comment les modules y sont déployés et mis hors service. Remplacer ces méthodes permet aux développeurs de politiques personnalisées du plug-in pour les fonctions mentionnées ci-dessus [82].
- **Capteur** : les instances de la classe `Sensor` sont des entités qui agissent comme des capteurs IoT décrits dans l'architecture. La classe contient des attributs représentant les caractéristiques d'un capteur, allant de sa connectivité aux attributs de sortie. La classe contient un attribut de référence au dispositif de passerelle Fog auquel le capteur est connecté et la latence de connexion entre eux. Plus important encore, il définit les caractéristiques de sortie du capteur et la distribution d'inter-transmission de tuples qui identifie le taux d'arrivée de tuples à la passerelle. En définissant les valeurs appropriées de ces attributs, des appareils tels que des caméras intelligentes, des voitures connectées, etc. peuvent être simulés [82].
- **Actuator** : cette classe modélise un actionneur en définissant l'effet d'actionnement et ses propriétés de connexion réseau. La classe définit une méthode pour effectuer une action à l'arrivée d'un tuple à partir d'un module d'application, qui peut être remplacée pour implémenter des effets d'actionnement personnalisés. Un attribut de la classe fait référence à la passerelle à laquelle l'actionneur est connecté et la latence de cette connexion.

4.4.2.2 Composants Logique

Les modules d'application (**AppModules**) et les bords d'application (**AppEdges**) sont les composants logiques d'iFogSim. Dans iFogSim, les applications sont considérées comme une collection d'AppModules interdépendants qui promeut par conséquent le concept d'application distribuée [83].

La dépendance entre deux modules est définie par les caractéristiques de AppEdges. Dans le domaine du cloud computing, les AppModules peuvent être mappés avec des machines virtuelles (**VM**) et les AppEdges sont le flux de données logique entre deux machines virtuelles. Dans iFogSim, chaque AppModule (**VM**) traite un type particulier de tuples (tâches) provenant du prédécesseur AppModule (**VM**) du flux de données.

La transmission de tuple entre deux AppModules peut être périodique et à la réception d'un tuple d'un type particulier, le fait qu'un module déclenche un

autre tuple (type différent) vers le module suivant est déterminé par le modèle de sélectivité fractionnaire [83].

4.4.2.3 Composants de Gestion

Le composant de gestion d'iFogSim est constitué d'objets de mappage de contrôleur et de module [81]. L'objet *Module Mapping*, selon les exigences des *AppModules*, identifie les ressources disponibles dans les appareils *Fog* et les place à l'intérieur. Par défaut, iFogSim prend en charge le placement hiérarchique des modules. Si un appareil *Fog* n'est pas en mesure de répondre aux exigences d'un module, le module est envoyé à l'appareil *Fog* de niveau supérieur. L'objet *Controller* lance les *AppModules* sur leurs appareils *Fog* affectés en suivant les informations de placement fournies par l'objet *Module Mapping* et gère périodiquement les ressources des appareils *Fog* [81].

Une fois la simulation terminée, l'objet *Controller* collecte les résultats des coûts, de l'utilisation du réseau et de la consommation d'énergie pendant la période de simulation à partir des appareils *Fog*.

4.5 Interface principale

La version de **iFogSim** n'a pas d'interface graphique, son exécution se fait sur console, donc nous avons créé une interface qui facilite l'accès à la configuration de simulateur.

4.5.1 Configuration des paramètres de simulation

Pour lancer la simulation d'un Cloud avec notre version étendue du simulateur iFogSim, nous devons configurer les paramètres de simulation dans un premier temps. Les Figures au-dessous montrent les différentes fenêtres de configuration de simulation qui contient **3** parties principales :

1-Configuration devices

Cette étape concerne la configuration des paramètres nécessaires pour la topologie du réseau, y compris la spécification d'un nom unique pour chaque dispositif, la puissance de traitement en MIPS, la quantité de **RAM**, la latence réseau, la bande passante montante et descendante en **kbps**, et le coût de traitement par **MIPS**.Elle inclut également l'indication du nombre de fichiers que chaque dispositif doit gérer, l'assignation des fichiers en utilisant "Set Files", l'ajout de dispositifs configurés à la liste via "Add Device", le retrait de dispositifs via "Remove Device", et l'affichage des détails de chaque dispositif en cliquant sur "View Details".

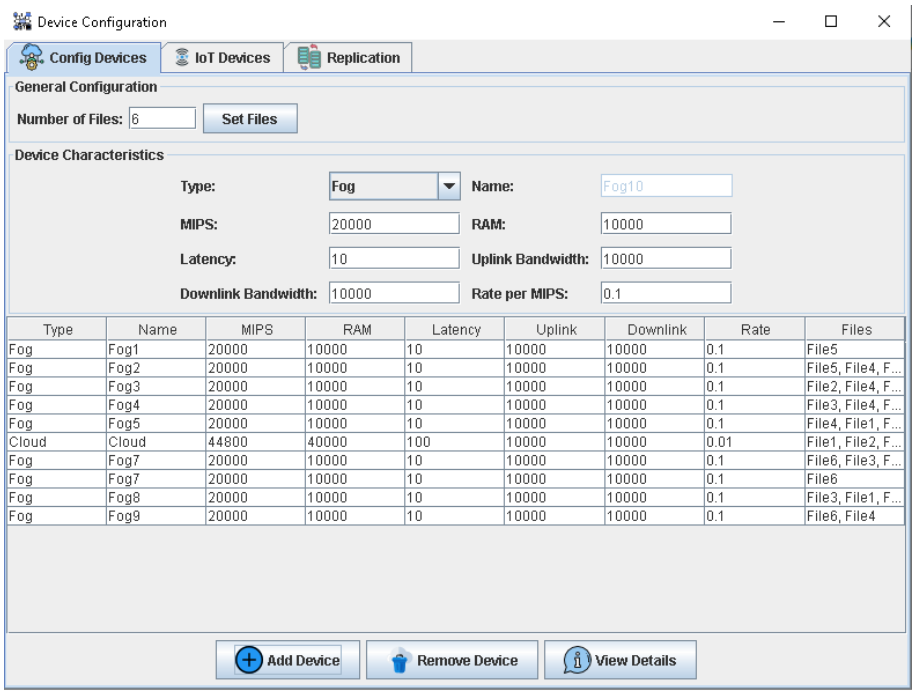

FIGURE 4.3 – Interface de CONFIG DEVICES

2-IOT devices

Cette étape consiste à configurer les dispositifs **IoT**. Commencez par l'entrée du nom du dispositif, la sélection du dispositif Fog auquel il se connectera via le menu déroulant *"Connected Fog Device"*, et l'ajout des requêtes de fichiers nécessaires. L'interface permet d'ajouter ou de supprimer des dispositifs **IoT**, facilitant ainsi la gestion des connexions entre les dispositifs **IoT** et **Fog**.

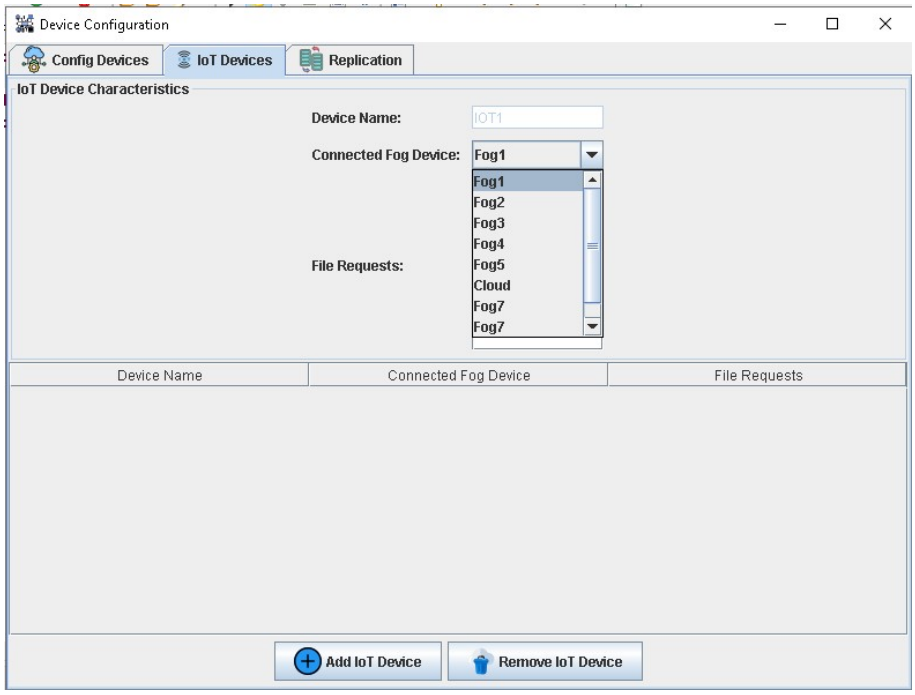

FIGURE 4.4 – Connected fog devices(interface de IOT devices)

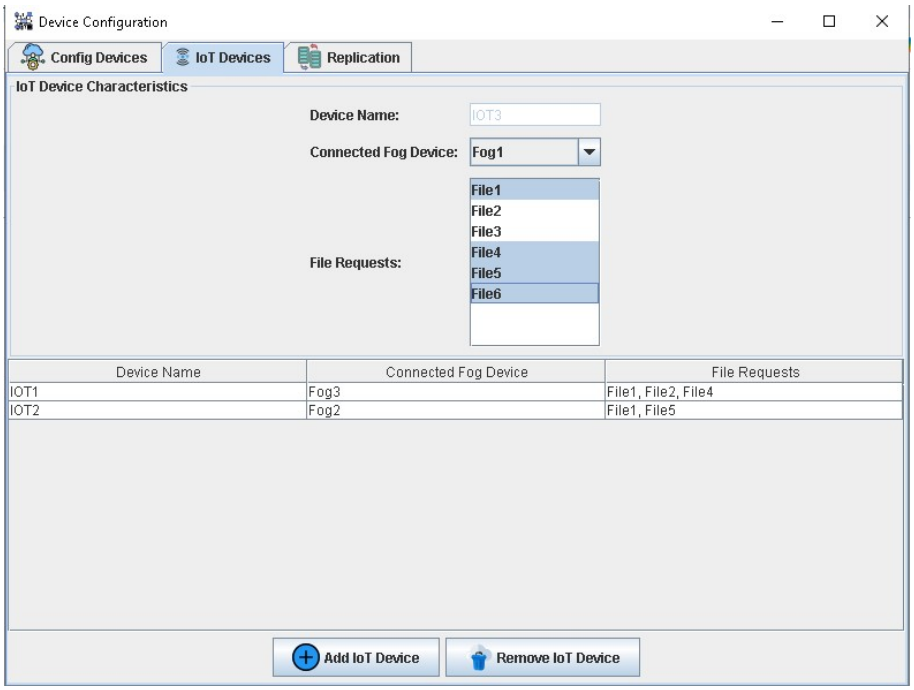

FIGURE 4.5 – File requests (interface des dispositifs IoT)

3-RÉPLICATION

Après la personnalisation des paramètres de simulation, dans l'onglet *Replication*, une sélection entre *Centralized* et *Distributed*, suivie d'un clic sur *Execute*. Les résultats s'affichent dans la console.

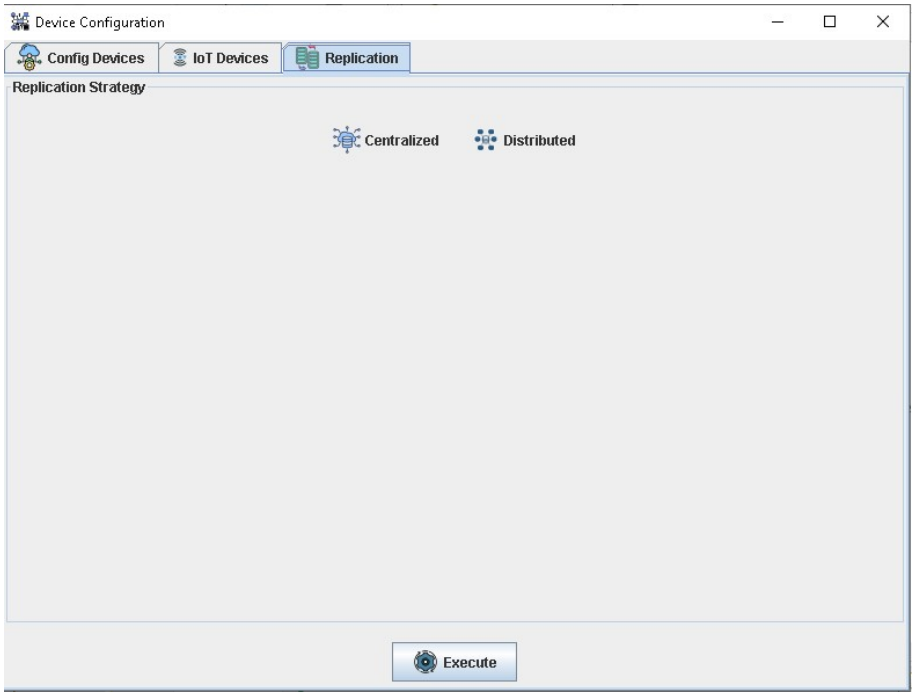

FIGURE 4.6 – Interface de RÉPLICATION

4.6 Résultats expérimentaux

Nous présentons dans cette partie quelques résultats des différentes expérimentations que nous avons obtenues. Plusieurs séries de simulations ont été effectuées pour comparer les deux approches suivantes :

1. **Approche centralisée :** Il s'agit de notre première stratégie qui repose sur l'utilisation du Cloud pour la classification et la distribution des données, en exploitant une vue d'ensemble pour optimiser la répartition des fichiers.
2. **Approche distribuée :** Il s'agit de notre deuxième stratégie qui se concentre sur la décentralisation de la classification et de la distribution des données, en s'appuyant principalement sur les dispositifs Fog pour le traitement et le stockage des fichiers.

4.7 Expérience 1 : l'impact du nombre de mobiles

Dans cette première simulation, nous avons étudié l'effet de la variation du nombre de mobiles sûr le temps de réponse moyen, la consommation de l'énergie par le Cloud et les Fogs, et l'utilisation de la bande passante, pour les deux approches.

Cette simulation a été réalisée avec les paramètres de simulation suivants : **2** Fog, **6** fichiers, le nombre de mobiles varie entre **10** et **60**.

1. Effet sur le temps de réponse moyen :

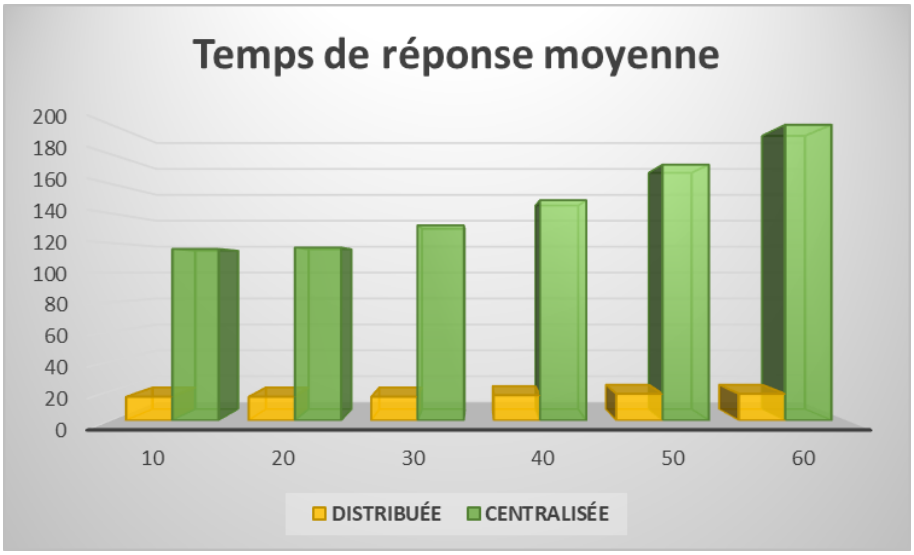

FIGURE 4.7 – Graphique à barres pour l'effet du nombre de mobiles sur le temps de réponse

Nous avons étudié l'effet de la variation du nombre de mobiles sur le temps de réponse dans les deux approches.

Les résultats de simulation présentés dans la figure **4.7** nous ont permis de constater que, dans la première approche centralisée, le temps de réponse moyen augmente à chaque fois que le nombre de mobiles augmente, car la latence entre le Cloud et les Fogs est élevée. Par contre, dans la deuxième approche distribuée, le temps de réponse moyen reste constant et beaucoup plus faible.

2. Effets sur la consommation d'énergie :

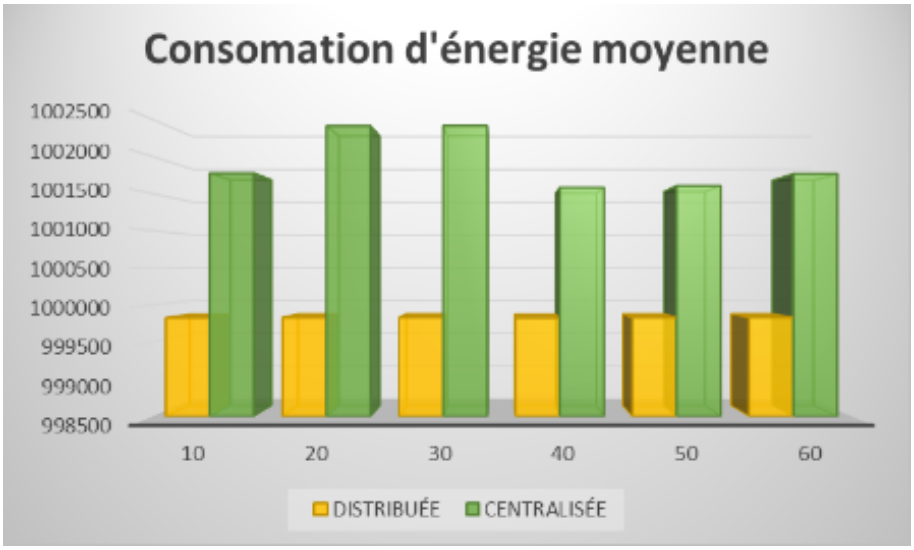

FIGURE 4.8 – Graphique à barres pour l'impact du nombre de mobiles sur la consommation d'énergie moyen

Dans cette série de simulations, nous avons étudié l'effet de la variation du nombre de mobiles et son impact sur la consommation d'énergie moyenne.

Nous constatons que dans la première approche centralisée, la consommation d'énergie moyenne augmente significativement avec le nombre de mobiles, allant de **10** à **40**.Cependant, à partir de **40** mobiles, la consommation d'énergie commence à diminuer légèrement. En contraste, dans l'approche distribuée, nous remarquons que la consommation d'énergie reste relativement stable et plus faible, même avec un nombre croissant de mobiles. Cela démontre l'efficacité énergétique de la stratégie distribuée par rapport à la stratégie centralisée, particulièrement en présence d'un grand nombre de dispositifs mobiles.

3. Effet sur l'utilisation de la bande passante :

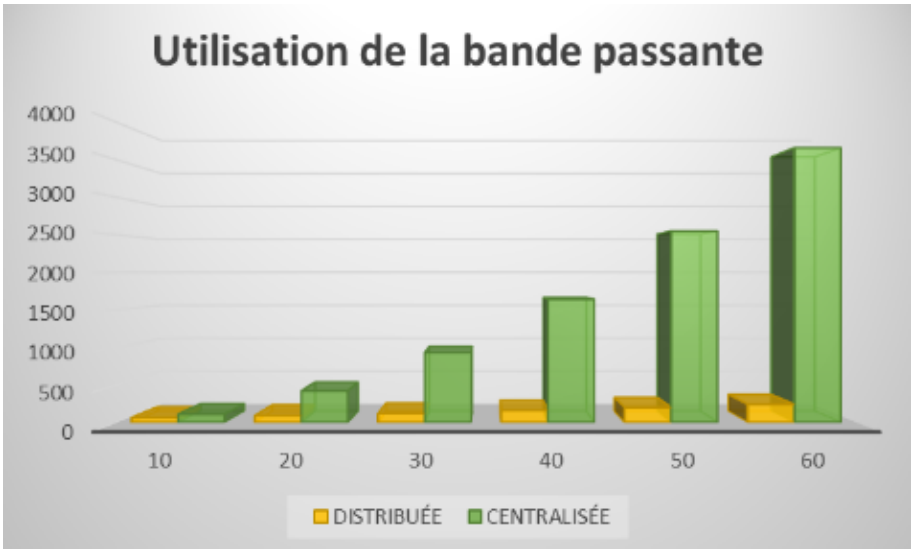

FIGURE 4.9 – Graphique à barres pour l'impact du nombre de mobiles sur l'utilisation du la bande passante

Dans cette simulation, nous avons calculé l'impact du nombre de mobiles sur l'utilisation totale du réseau ,la Figure **4.9** montre de manière évidente, l'infériorité de l'approche centralisée qui entraîne une augmentation significative de l'utilisation du réseau à mesure que le nombre de mobiles augmente. Par rapport à l'approche distribuée nous remarquons qu'elle maintient une utilisation du réseau beaucoup plus faible. Cela s'explique par le fait que l'approche centralisée sollicite fortement le cloud, tandis que l'approche distribuée limite le trafic réseau en utilisant principalement les dispositifs Fog pour le traitement et le stockage des données.

4.8 Expérience 2 : l'impact du nombre de fichiers

Le but de cette série de simulations est d'étudier l'impact du nombre de fichiers sur les trois métriques utilisés dans la première simulation : temps de réponse moyen, la consommation d'énergie par le cloud et le fog, ainsi que l'utilisation totale du réseau pour les deux propositions. Cette simulation a été réalisée avec les paramètres de simulation suivants : **2** fogs, **10** mobiles pour chaque fog, et le nombre de fichiers variant entre **5** et **30**.

1. Effet sur le temps de réponse moyen :

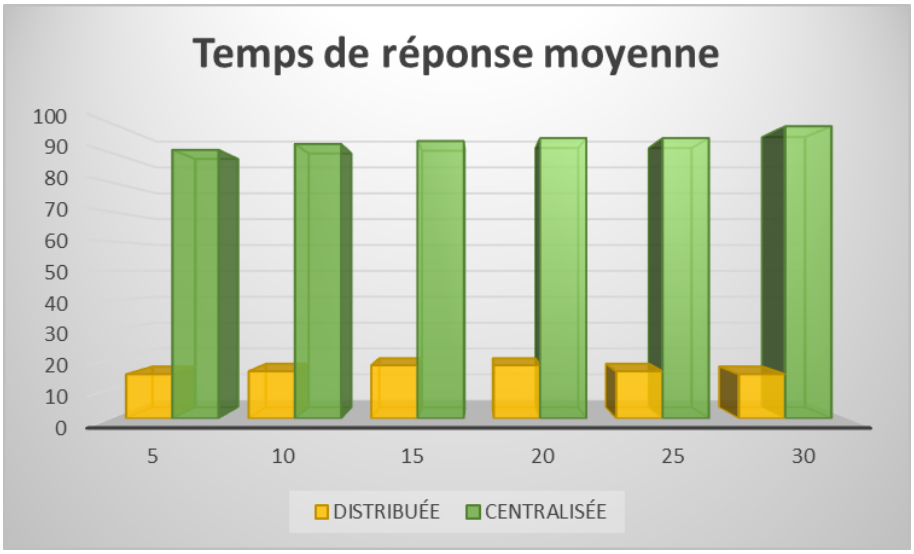

FIGURE 4.10 – Graphique à barres pour Impact du nombre de fichier sur le temps de réponse moyen

Dans cette série de simulation, nous avons étudié l'impact de la variation du nombre de fichiers sur le temps de réponse moyen.

Les résultats de simulation présentés dans la figure **4.10** montrent que dans l'approche centralisée, le temps de réponse moyen augmente considérablement à mesure que le nombre de fichiers augmente, ce qui s'explique par la sollicitation accrue du Cloud, entraînant des congestions réseau et des temps de traitement plus longs. A la différence de la première approche centralisée , nous remarquons dans la deuxième approche qu'elle maintient un temps de réponse moyen constant et significativement plus faible. Cela s'explique par l'utilisation principale des dispositifs Fog pour le traitement et le stockage des données, réduisant ainsi la charge sur le réseau et améliorant l'efficacité globale . L'approche distribuée offre des meilleures performances en termes de temps de réponse moyen, en particulier avec un nombre croissant de fichiers.

2. Effets sur la consommation d'énergie :

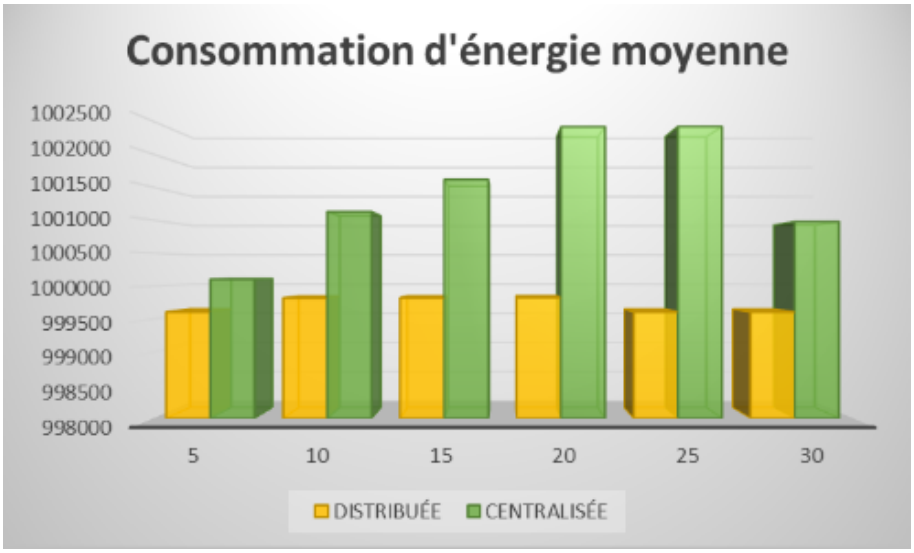

FIGURE 4.11 – Graphique à barres pour Impact du nombre de fichier sur la consommation d'énergie moyen

Nous avons analysé la consommation d'énergie moyenne en relation avec le nombre de fichiers pour deux méthodes distinctes, comme illustré dans la Figure **4.11**. L'approche centralisée, utilisant le cloud, révèle une augmentation progressive de la consommation d'énergie, indiquant une possible saturation des capacités de traitement des serveurs à mesure que les demandes s'accroissent. À l'opposé, l'approche distribuée, qui s'appuie sur le fog computing, maintient une consommation d'énergie modérée et constante. Cette méthode bénéficie de la proximité du traitement des données, alléguant la pression sur les infrastructures centralisées et optimisant l'utilisation énergétique. Ce contraste met en évidence le rôle crucial de la sélection de l'architecture appropriée pour atteindre une gestion énergétique efficace et durable.

3. Effet sur l'utilisation de la bande passante :

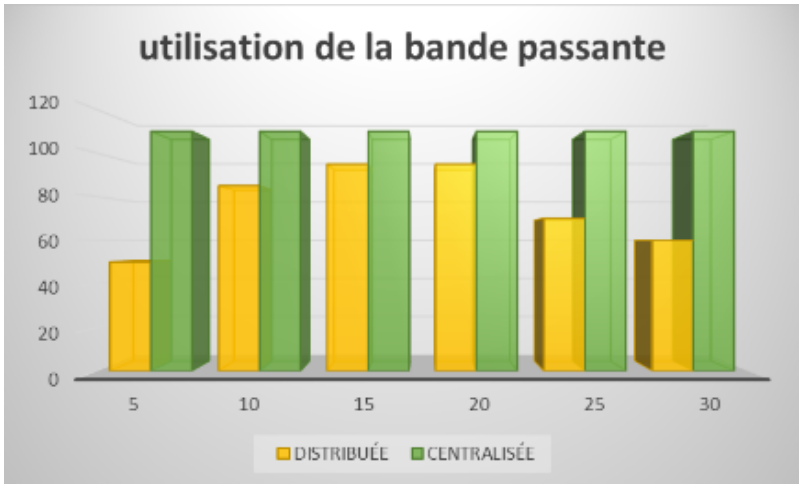

FIGURE 4.12 – Graphique à barres pour l'impact du nombre de mobiles sur l'utilisation du la bande passante

Afin d'étudier l'effet de la variation du nombre de fichiers sur l'utilisation totale du réseau dans les deux approches, nous avons lancé plusieurs simulations.

Les résultats de ces séries de simulations nous ont permis de constater que dans l'approche centralisée, l'utilisation du réseau augmente fortement avec l'augmentation du nombre de fichiers en raison de la dépendance au cloud pour le transfert de données.En revanche, l'approche distribuée maintient une utilisation du réseau plus stable et inférieure, car elle utilise principalement les dispositifs Fog pour le traitement et le stockage des données.Nous remarquons qu'à partir de **25** fichiers, l'utilisation du réseau dans l'approche centralisée commence à diminuer, tandis que l'approche distribuée reste stable avec une utilisation du réseau oscillant autour de **40** à **100**, et redescend à **40**.Cela montre que l'approche distribuée est plus efficace pour gérer l'utilisation du réseau lorsque le nombre de fichiers augmente.

4.9 Expérience 3 : l'impact du nombre de Fog

Le but de cette série de simulations est d'étudier l'impact du nombre de Fog sur le temps de réponse moyen, la consommation d'énergie par le cloud et les Fog, ainsi que l'utilisation totale du réseau pour les deux approches. Cette simulation a été réalisée avec les paramètres de simulation suivants : **6** fichiers, **10** mobiles pour chaque Fog, et un nombre de Fogs variant entre **2** et **12**.

1. Effet sur le temps de réponse moyen :

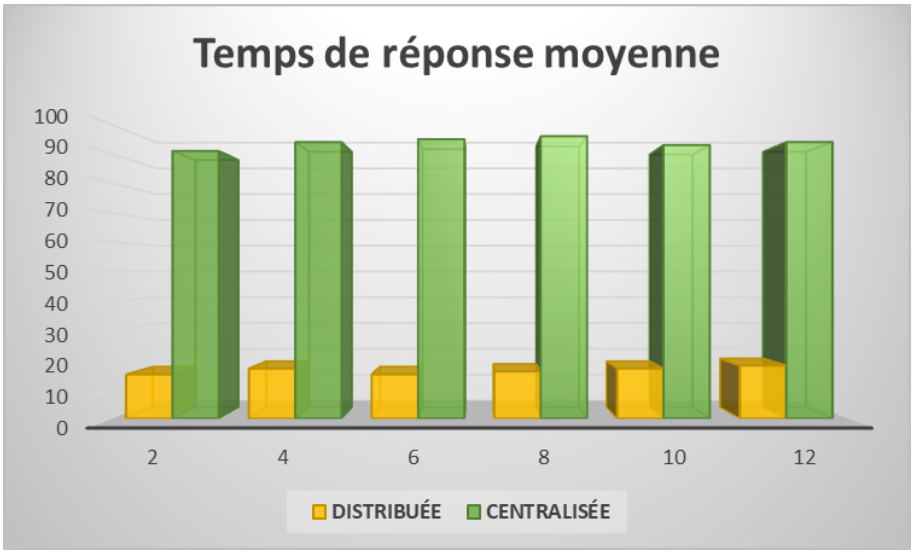

FIGURE 4.13 – Graphique à barres pour l'impact du nombre de Fog sur le temps de réponse moyen

Dans cette série de simulations, nous avons étudié l'impact du nombre de Fog sur le temps de réponse moyen. Significativement à la variation du nombre de Fog, la figure **4.13** montre les résultats obtenus.

Nous remarquons que le temps de réponse moyen reste élevé et stable malgré l'augmentation du nombre de dispositifs Fog dans l'approche centralisée. Cela est dû à la dépendance de cette approche aux traitements effectués par le cloud, entraînant des délais supplémentaires liés aux transferts de données vers et depuis le cloud.

En revanche, dans l'approche distribuée, nous remarquons que le temps de ré-

ponse moyen est nettement inférieur et reste constant même avec un nombre croissant de dispositifs Fog.

Cette amélioration est attribuée à l'utilisation des dispositifs Fog pour le traitement local des données, réduisant ainsi les délais de transfert et permettant un traitement plus rapide des demandes.

2. Effets sur la consommation d'énergie :

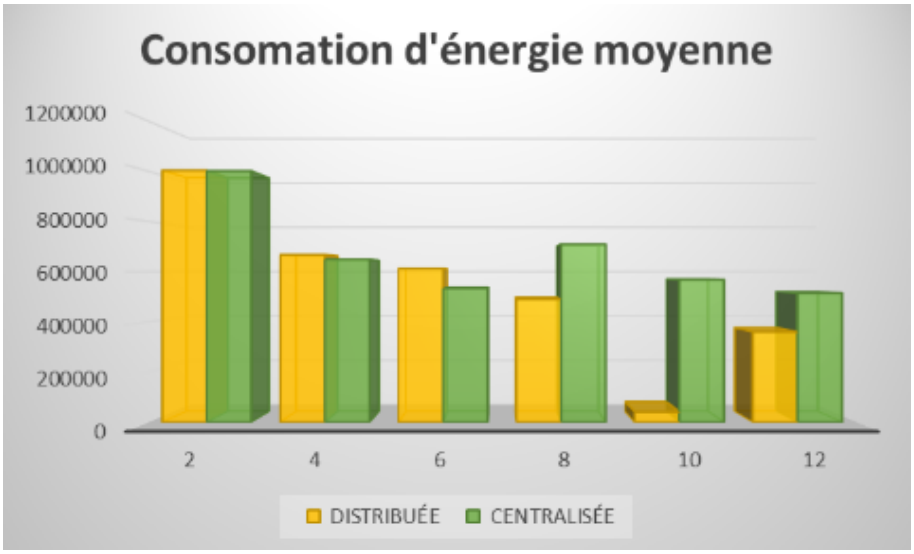

FIGURE 4.14 – Graphique à barres pour l'impact du nombre de Fog sur la consommation d'énergie moyen

Le graphique **4.14** montre l'impact du nombre de dispositifs Fog sur la consommation d'énergie moyen pour les approches centralisée et distribuée. Nous remarquons que dans l'approche centralisée, la consommation de l'énergie moyen est élevé, mais diminue progressivement à mesure que le nombre de dispositifs Fog augmente. Cela s'explique par une meilleure répartition de la charge de traitement entre le cloud et les dispositifs Fog, réduisant ainsi l'énergie consommée par le cloud.

Cependant, nous remarquons que dans l'approche distribuée, le temps d'énergie moyen est initialement plus faible et continue de diminuer de manière plus significative à mesure que le nombre de dispositifs Fog augmente.

Cette diminution est due à l'efficacité accrue des dispositifs Fog pour le traitement local des données, réduisant ainsi la consommation d'énergie liée aux transferts de données vers le cloud.

3. Effet sur l'utilisation de la bande passante :

La figure **4.15** représente les résultats de simulation de la variation du nombre de dispositifs Fog sur l'utilisation totale du réseau pour les deux approches.

Nous constatons que dans la première approche, l'utilisation du réseau augmente de manière significative avec l'augmentation du nombre de dispositifs Fog. Cette augmentation est due au fait que le cloud centralisé doit gérer de plus en plus de connexions et de transferts de données, ce qui augmente la charge sur le réseau.Par contre, dans la deuxième approche, l'utilisation du réseau reste relativement stable et beaucoup plus faible, même avec un plus grand nombre de dispositifs Fog. Cette stabilité est due à la capacité des dispositifs Fog à traiter et stocker localement les données, réduisant ainsi la dépendance au cloud et les

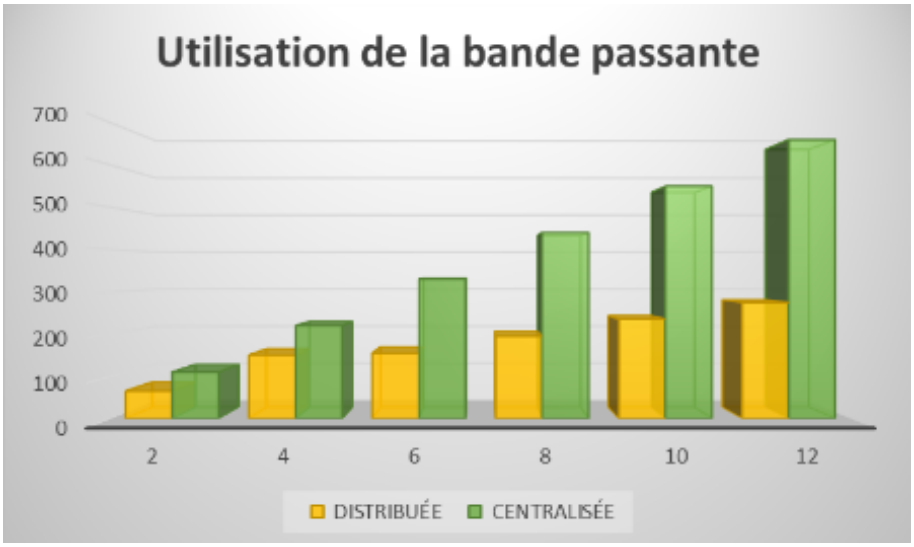

FIGURE 4.15 – Graphique à barres pour l'impact du nombre de Fog sur l'utilisation du la bande passante

transferts de données à travers le réseau.

4.10 Conclusion

Dans ce chapitre, nous avons présenté notre environnement de travail et nous avons détaillé l'outil de simulation **iFogSim** que nous avions étendu afin de simuler notre stratégie ainsi que les différentes approches .

Nous avons réalisé plusieurs simulations en jouant sur différents paramètres tels que le nombre de mobiles, le nombre de fichiers et le nombre de dispositifs Fog. Les résultats obtenus montrent que l'approche distribuée parvient à atteindre les objectifs souhaités en maintenant une consommation d'énergie et une utilisation du réseau stables et relativement faibles, même avec un nombre croissant de dispositifs.

Conclusion générale

Les applications gourmandes en données bénéficient traditionnellement de solutions Cloud centralisées, offrant des ressources quasi illimitées et un bon rapport coût-efficacité. Toutefois, l'utilisation du Cloud peut entraîner des latences élevées et des risques de sécurité accrus en raison du trafic réseau intense. En contraste, le Fog Computing, proche des objets connectés, optimise le stockage et l'analyse des données, réduisant ainsi la latence et améliorant la sécurité.

Dans le travail présent, Notre recherche s'est concentrée sur la réplication des données dans le Fog Computing,soulignant son rôle crucial dans l'amélioration de la disponibilité et de l'accès rapide aux données localement, tout en renforçant la résilience face aux pannes grâce à des copies multiples. Nous avons intégré l'apprentissage automatique dans notre stratégie pour optimiser les performances de réplication, exploitant sa capacité à adapter dynamiquement le processus de réplication aux conditions changeantes du réseau et de l'utilisation des données.

La stratégie que nous avons proposé utilise deux algorithme de classification, l'algorithme DBSCAN pour une classification locale et l'algorithme K-means pour une classification globale, nous avons présenté deux solutions pour la gestion des données dans le Fog Computing : une approche centralisée qui utilise le cloud pour une classification globale, une solution distribuée qui allège la dépendance au cloud en répartissant les tâches entre les nœuds de Fog. Cette stratégie améliore l'utilisation des ressources, réduit la latence, et s'adapte dynamiquement aux variations de charge. L'intégration de l'apprentissage automatique a permis une gestion précise et réactive des processus de réplication, améliorant ainsi les performances globales du système.

Pour mettre en évidence notre approche, nous avons étendu le simulateur iFogSim, et réalisé une série de simulations sous différents scénarios pour afin d'évaluer les performances. Nous avons mesuré le temps de réponse, la consommation d'énergie moyenne, et l'utilisation de la bande passante. Les résultats de simulations montrent une amélioration considérable sur ces trois métriques, validant ainsi l'efficacité de notre stratégie proposée.

PERSPECTIVES :

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

1. Traiter le coût de paiement.
2. Intégrer et tester notre approche dans un environnement Fog réel pour évaluer son efficacité dans des conditions pratiques.
3. Continuer à améliorer la stratégie de gestion des données pour optimiser les performances et l'efficacité.

Bibliographie

[1] Hadjer Ghezal, Karima Belghit, and Foad Bekkari. Une approche de sécurité basée sur authentification dans cloud computing.

[2] Google Cloud. What is cloud computing. https://cloud.google.com/learn/what-is-cloud-computing. Accessed June 8, 2024.

[3] Yehia I. Alzoubi, Valmira H. Osmanj, Ashraf Jaradat, et al. Fog computing security and privacy for the internet of thing applications : State-of-the-art. *Security and Privacy*, 4(2) :e145, 2021.

[4] Rachid Mecheri and Mourd Ben Abdelbasset. *Réplication des données dans le Cloud Computing*. PhD thesis, UNIVERSITE KASDI MERBAH OUARGLA.

[5] Mohammed Taha Hamdi Pacha. Vers une coordination transparente entre le cloud computing et le fog computing. Master's thesis, Universite laarbi tebessi tebessa, 2020.

[6] Ailam Melisa and Bafdel Mellisa. Ordonnancement de tâches dans le cloud. Master's thesis, Université Mouloud Mammeri, 2019.

[7] Achla Gupta and S. Thakur. Cloud computing : its characteristics, security issues and challenges. *Rev. Comput. Eng. Stud*, 4(2) :76–81, 2017.

[8] Hibatullah Alazhrani. A brief survey of cloud computing. *Global Journal of Computer Science and Technology*, 16(3) :7, 2016.

[9] Anca Apostu, Florina Puican, Geanina Ularu, et al. Study on advantages and disadvantages of cloud computing–the advantages of telemetry applications in the cloud. *Recent advances in applied computer science and digital services*, 2103, 2013.

[10] Peshraw Ahmed Abdalla and Asaf Varol. Advantages to disadvantages of cloud computing for small-sized business. In *2019 7th International Symposium on Digital Forensics and Security (ISDFS)*, pages 1–6. IEEE, 2019.

[11] Luis M. Vaquero and Luis Rodero-Merino. Finding your way in the fog : Towards a comprehensive definition of fog computing. *ACM SIGCOMM computer communication Review*, 44(5) :27–32, 2014.

[12] Satyakam Rahul and Rajni Aron. Fog computing architecture, application and resource allocation : A review. *CEUR Workshops*, 4638, 2021.

[13] Fog Computing. the internet of things : Extend the cloud to where the things are. 2015. Cisco White Paper (2019).

[14] Yuan AI, Mugen Peng, and Kecheng Zhang. Edge computing technologies for internet of things : a primer. *Digital Communications and Networks*, 4(2) :77–86, 2018.

[15] Shanhe YI, Zijiang HAO, Zhengrui QIN, et al. Fog computing : Platform and applications. In *2015 Third IEEE workshop on hot topics in web systems and technologies (HotWeb)*, pages 73–78. IEEE, 2015.

[16] Rahul Neware and Urmila Shrawankar. Fog computing architecture, applications and security issues. *International Journal of Fog Computing (IJFC)*, 3(1) :75–105, 2020.

[17] Yehia I. Alzoubi, Valmira H. Osmanaj, Ashraf Jaradat, et al. Fog computing security and privacy for the internet of thing applications : State-of-the-art. *Security and Privacy*, 4(2) :e145, 2021.

[18] S.M. Alzaharni. Fog computing and cloud computing : Architecture, key technologies, applications and security challenges. *International Journal of Advanced Research in Engineering and Technology (IJARET)*, 11 :1131–1138, November 2020.

[19] Madiha Syed, Eduardo B. Fernandez, and Mohammad Ilyas. A pattern for fog computing. In *ACM Proc. 10th Travelling Conference on Patterns Language of Programming (VikingPLoP)*, page 10, April 2016.

[20] Michaela Iorga et al. Fog computing conceptual model. Technical Report Special Publication 500-325, National Institute of Standards and Technology, March 2018.

[21] Moonmoon Chakraborty. Fog computing vs. cloud computing. *arXiv preprint*, 2019.

[22] Mohamed Firdhous, Osman Ghazali, and Suhaidi Hassan. Fog computing : Will it be the future of cloud computing ? 2014.

[23] Vishal Kumar, Asif Ali Laghari, Shahid Karim, et al. Comparison of fog computing & cloud computing. *Int. J. Math. Sci. Comput*, 1 :31–41, 2019.

[24] Paul Avery and Ian Foster. The griphyn project : Towards petascale virtual data grids. Technical report, Technical Report GriPhyN-2001-15, 2001.

[25] Ming Lei and Susan V. Vrbsky. A data replication strategy to increase data availability in data grids. In *GCA*, pages 221–227, 2006.

[26] Eddy Meylan. Bases de données : Réplication des données. Support de cours, Février 2005. Haute Ecole spécialisée de Suisse Occidentale, Informatique de Gestion.

[27] Matthieu Exbrayat. Bases de données réparties : Concepts et techniques. Notes de cours, Décembre 2007. ULP Strasbourg.

[28] Mr. Djebbara. Placement des répliques dans les grilles de données hiérarchiques, 2012/2013.

[29] Eddy Meylan. Bases de données : Réplication des données. Support de cours, Février 2005. Haute Ecole spécialisée de Suisse Occidentale, Informatique de Gestion.

[30] Ailam Melisa and Bafdel Mellisa. *Ordonnancement de tâches dans le Cloud*. PhD thesis, Université Mouloud Mammeri, 2019.

[31] U. Cibej, B. Slivnik, and B. Robic. The complexity of static data replication in data grids. *Parallel Computing*, 31 :900–912, 2005.

[32] K. Ranganathan and I. Foster. Identifying dynamic replication strategies for a high-performance data grid. In *Proc. of the Second International Workshop on Grid Computing*, 2001.

[33] Qingsong Wei et al. Cdrm : A cost-effective dynamic replication management scheme for cloud storage cluster. In *2010 IEEE International Conference on Cluster Computing*. IEEE, 2010.

[34] William H. Bell, David G. Cameron, Luigi Capozza, et al. Simulation of dynamic grid replication strategies in optorsim. In *Grid Computing—GRID 2002 : Third International Workshop*, pages 46–57. Springer Berlin Heidelberg, 2002.

[35] W. Bell, D. Cameron, R. Carvajal-Schiaffino, A. Millar, K. Stockinger, and F. Zini. Evaluation of an economy-based file replication strategy in data-

grids. In *Third International Symposium on Cluster Computing and the Grid (CCGRID)*, 2003.

[36] B. Alami Milani and N. Jafari Navimipour. A comprehensive review of the data replication techniques in the cloud environments : Major trends and future directions. *Journal of Network and Computer Applications*, 64 :229–238, 2016.

[37] Ashish Vulimiri et al. Global analytics in the face of bandwidth and regulatory constraints. In *12th USENIX Symposium on Networked Systems Design and Implementation (NSDI 15)*, 2015.

[38] Flávio RC Sousa and Javam C. Machado. Towards elastic multi-tenant database replication with quality of service. In *2012 IEEE Fifth International Conference on Utility and Cloud Computing*. IEEE, 2012.

[39] Guthemberg Silvestre et al. Aren : a popularity aware replication scheme for cloud storage. In *2012 IEEE 18th International Conference on Parallel and Distributed Systems*. IEEE, 2012.

[40] Yousri Kouki, Thomas Ledoux, and Remi Sharrock. Cross-layer sla selection for cloud services. In *2011 First International Symposium on Network Cloud Computing and Applications*. IEEE, 2011.

[41] Q Wei et al. Cdrm : A cost-effective dynamic replication management scheme for cloud storage cluster. In *IEEE International Conference on Cluster Computing*, pages 188–196. IEEE, Sep 2010.

[42] Nicolas Bonvin, Thanasis G. Papaioannou, and Karl Aberer. A self-organized, fault-tolerant and scalable replication scheme for cloud storage. In *Proceedings of the 1st ACM symposium on Cloud computing*, 2010.

[43] X. Yuan, Y. Xu, and L. Yang. A matrix-based k-means clustering approach for data placement in cloud systems. *Journal of Cloud Computing*, 3(2) :123–130, 2010.

[44] E. J. McCormick, J. W. DeLuca, and J. W. Price. An application of clustering techniques to cloud data systems. *Journal of Data Science*, 12(3) :45–52, 1972.

[45] Qingsong Wei et al. Cdrm : A cost-effective dynamic replication management scheme for cloud storage cluster. In *2010 IEEE international conference on cluster computing*. IEEE, 2010.

[46] Nicolas Bonvin, Thanasis G. Papaioannou, and Karl Aberer. A self-organized, fault-tolerant and scalable replication scheme for cloud storage. *Proceedings of the 1st ACM symposium on Cloud computing*, pages 205–216, 2010.

[47] Nicolas Bonvin, Thanasis G. Papaioannou, and Karl Aberer. A self-organized, fault-tolerant and scalable replication scheme for cloud storage. In *Proceedings of the 1st ACM symposium on Cloud computing*, 2010.

[48] Ismaeel Al Ridhawi, Nour Mostafa, and Wassim Masri. Location-aware data replication in cloud computing systems. In *2015 IEEE 11th international conference on wireless and mobile computing, networking and communications (WiMob)*. IEEE, 2015.

[49] Yanzhen Qu and Naixue Xiong. Rfh : A resilient, fault-tolerant and high-efficient replication algorithm for distributed cloud storage. In *2012 41st International Conference on Parallel Processing*. IEEE, 2012.

[50] Mengxing et al. A strategy of dynamic replica creation in cloud storage. In *Proceedings of the 1st International Workshop on Cloud Computing and Information Security*, pages 389–392, Paris, France, 2013. Atlantis Press.

[51] J. Sousa and J. Machado. Replic : Elastic replication for multi-tenant databases. *Proceedings of the 2012 ACM Symposium on Cloud Computing*, pages 123–134, 2012.

[52] M. Ripeanu and I. Foster. A decentralized, adaptive replica location mechanism. In *HPDC-11 '02*, pages 24–34, Los Alamitos, CA, USA, July 2002. IEEE Computer Society.

[53] P. Jayalakshmi and B. Rashmi. Efficient data replication strategies in cloud computing. *International Journal of Computer Applications*, 123(4) :23–29, 2015.

[54] X. Bai and et al. Rtrm : A real-time replication management strategy for cloud storage system. *Journal of Cloud Computing*, 2(16) :1–15, 2013.

[55] Anupam Gupta, Rajeev Sharma, and Sushil Jain. Proactive data replication in fog computing : A comprehensive review. *Journal of Cloud Computing : Advances, Systems and Applications*, 9(1) :1–15, 2020.

[56] et al. Wang. Optimizing reactive replication in resource-constrained fog environments. *Journal of Fog Computing*, 2019.

[57] Y. Wang, X. Li, and Z. Zhang. Reactive replication strategy for fault-tolerant fog computing systems. *IEEE Transactions on Parallel and Distributed Systems*, 30(8) :1784–1796, 2019.

[58] et al. Li. Hybrid replication strategies in fog computing : Performance optimization and cost reduction. *International Journal of Fog Computing*, 2021.

[59] J. Li, Y. Zhou, and Q. Wu. Hybrid replication strategy for data consistency in fog computing. *ACM Transactions on Internet Technology*, 21(2) :1–19, 2021.

[60] Auteur Chen. Titre de l'article. *Nom du Journal*, Volume(Numéro) :Pages, 2022.

[61] H. Chen, W. Liu, and C. Yang. Policy-based data replication in fog computing environments. *International Journal of Distributed Sensor Networks*, 18(1) :1–12, 2022.

[62] Auteur Lee. Titre de l'article. *Nom du Journal*, Volume(Numéro) :Pages, 2020.

[63] J. Li, Y. Zhou, and Q. Wu. Hybrid replication strategy for data consistency in fog computing. *ACM Transactions on Internet Technology*, 21(2) :1–19, 2021.

[64] Y. Saito and M. Shapiro. Optimistic replication. *ACM Computing Surveys*, 37(1) :42–81, 2005.

[65] S. Drapeau. *RS2.7 : Un Canevas Adaptable de Services de Duplication*. PhD thesis, Institut National Polytechnique de Grenoble, France, Juin 2003.

[66] A. Imine. *Conception Formelle d'Algorithmes de Réplication Optimiste Vers l'Edition Collaborative dans les Reseaux Pair-a-Pair*. PhD thesis, Doctorat de l'universite Henri Poincare Nancy 1, France, Novembre 2006.

[67] K. Ranganathan and I. Foster. Identifying dynamic replication strategies for a high performance data grid. In *Grid : Second International Workshop*, volume 2242, pages 75–86. Springer Berlin, Novembre 2001.

[68] S. Vazhkudai, S. Tuecke, and I. Foster. Replica selection in the globus data grid. In *CCGRID'01 : Proceedings of the 1st International Symposium on Cluster Computing and the Grid*, pages 106–113. IEEE Computer Society, 2001.

[69] M. Karlsson, C. Karamanolis, and M. Mahalingam. A framework for evaluating replica placement algorithms. Technical report, HP Labs, 2002.

[70] Mengxing Huang et al. A strategy of dynamic replica creation in cloud storage. In *1st International Workshop on Cloud Computing and Information Security*. Atlantis Press, 2013.

[71] J. Lee, Jaehwa Chung, and Daewon Lee. Efficient data replication scheme based on hadoop distributed file system. *International Journal of Software Engineering and Its Applications*, 9(12) :177–186, 2015.

[72] Jianliang Xu, Bo Li, and Dik Lun Lee. Placement problems for transparent data replication proxy services. *IEEE Journal on Selected Areas in Communications*, 20(7) :1383–1398, 2002.

[73] H. Lamehamedi, Z. Shentu, B. Szymanski, and E. Deelman. Simulation of dynamic data replication strategies in data grids. In *12th Heterogeneous Computing Workshop (HCW'03)*, Nice, France, April 2003. IEEE Computer Science Press.

[74] Yahya Slimani, Faiza Najjar, and Najla Mami. An adaptive cost model for distributed query optimization on the grid. In *On the Move to Meaningful Internet Systems 2004 : OTM 2004 Workshops*, pages 79–87. Springer Berlin Heidelberg, 2004.

[75] G. Belalem. *Contribution à la gestion de la cohérence de répliques de fichiers dans les systèmes à large échelle*. PhD thesis, Département d'informatique, Faculté des sciences, Université d'Oran, Algérie, Novembre 2007.

[76] Ahmed Yahi. *Clustering des données de puces à ADN*. PhD thesis, UNIVERSITE MOHAMED BOUDIAF-M'SILA, 2019.

[77] Neil Scicluna and Christos-Savvas Bouganis. Arc 2014 : a multidimensional fpga-based parallel dbscan architecture. *ACM Transactions on Reconfigurable Technology and Systems (TRETS)*, 9(1) :1–15, 2015.

[78] James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, pages 281–297, Oakland, CA, USA, 1967.

[79] Langage Java. langage java. https://www.java.com/en/download/.

[80] Eclipse. Eclipse. https://www.eclipse.org/downloads/.

[81] et al. H. Gupta. ifogsim : A toolkit for modeling and simulation of resource management techniques in the internet of things, edge and fog computing environments,", 2017.

[82] Harshit Gupta et al. ifogsim : A toolkit for modeling and simulation of resource management techniques in the internet of things, edge and fog computing environments. *Software : Practice and Experience*, 47(9) :1275–1296, 2017.

[83] Azzaze Nacira Hanane and Boukhecha Ikram. Une stratégie d'ordonnancement et de réplication de données dans les environnements de fog/cloud computing. Mémoire de master en informatique, Université DR MOULAY TAHAR – SAIDA, Septembre 2021. Option : Réseaux Informatique et Systèmes Répartis.