REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEURE ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE ABOU-BEKR BELKAID – TLEMCEN

THÈSE LMD

Présentée à :

FACULTE DES SCIENCES – DEPARTEMENT D'INFORMATIQUE
Laboratoire de Recherche en Informatique de Tlemcen (LRIT)

Pour l'obtention du diplôme de :

DOCTORAT LMD

Spécialité : *Informatique*
Option : *Réseaux et Services*
Par :

SETTOUTI Ahmed Khalid Yassine

Sur le thème

Indexation et sélection des services Cloud Computing IaaS pour des réseaux de capteurs

---

Soutenue publiquement le 15 Mai 2018 à Tlemcen devant le jury composé de :

Pr CHIKH Azeddine	Professeur	Université de Tlemcen	Président
Dr LAHFA Fedoua	Maître de Conférences	Université de Tlemcen	Directeur de thèse
Dr HADDAD Mohamed	Maître de Conférences	Université de Lyon 1	Co-Directeur de thèse
Pr BELALEM Ghalem	Professeur	Université d'Oran 1	Examinateur
Dr KHALFI Mohammed Fethi	Maître de Conférences	Université de Sidi Bel Abbes	Examinateur
Dr BENMAMMAR Badr Eddine	Maître de Conférences	Université de Tlemcen	Examinateur

Indexation et sélection des services Cloud Computing IaaS pour des réseaux de capteurs

SETTOUTI Ahmed Khalid Yassine

15 Mai 2018

Citations

"Amat Victoria Curam"
"La victoire aime la préparation"

Proverbe latin

Je dédie ce travail à mon défunt grand-père BENYELLES Bachir qui a semer en moi ses valeurs et surtout l'amour du savoir.

Résumé

Les réseaux de capteurs sans fil sont de plus en plus utilisés dans le monde actuel. Ils sont aussi bien utiles pour les professionnels, les amateurs, les militaires, les civils, les académiques et les non académiques. Dû à l'excès d'utilisation des nœuds sans fil, les données générées sont de plus en plus importantes. Puisque les capteurs sans fil sont limités en termes de ressources (stockage, traitement et communication), il est utile de les intégrer au cloud computing.
Le nombre des fournisseurs de cloud computing ne cessant de croître, il est nécessaire d'encadrer leurs utilisateurs (propriétaires de réseaux de capteurs sans fil) dans leurs choix du meilleur, afin de traiter et stocker leurs données. Dans cette thèse, nous traitons justement le problème de la sélection du meilleur fournisseur cloud computing en un temps réduit. Pour cela, nous optimisons une approche déjà proposée afin de traiter des clients divers, des services hétérogènes, des réseaux de capteurs sans fil variés... etc. Nous pensons avoir proposé une approche réduisant la durée de la recherche des services et retournant des services plus pertinents dès le début de la recherche.
En d'autres termes, la recherche proposée retourne de meilleures instances cloud computing dès le début, tout en réduisant le temps global pour les chercher. Avec une recherche plus efficace et plus performante, les utilisateurs pourront mieux sélectionner leurs services, économiser leur argent, réduire les ressources pour le faire et explorer plus d'alternatives... etc.
Mots clés: Cloud Computing ; Sélection de services ; Préférences des utilisateurs ; Mesures de qualité ; IaaS publique ; Réseaux de capteurs sans fil ; RCSF ; Évaluation des services ; Indexation ; B_{cloud}Tree.

Remerciements

Je remercie spécialement Mr. HADDAD, pour m'avoir co-encadré d'un côté. Mais aussi pour ses conseils en méthodologie. Nous ne pouvions pas avancer et encore moins à la vitesse exercée sans son aide.
Je remercie spécialement Mme. LAHFA, pour m'avoir encadré d'une part. Avoir ouvert une formation doctorale d'une autre part qui a permis cette recherche.
Je remercie Mr. MERZOUG pour sa motivation et ses encouragements continus durant la finalisation de notre thèse.
Je remercie l'ensemble de ma famille, et surtout ma mère ainsi que ma grande-mère de m'avoir soutenu afin de continuer mes études.
Je remercie Mr. CHIKH d'avoir présidé notre soutenance, ainsi que l'ensemble des examinateurs pour leurs remarques, leurs perspectives et leurs suggestions.

Table des matières

Table des figures

Liste des tableaux

Glossaire

$/GB Dollars par GigaByte. 56

$/h Dollars par Heure. 56

Analytical Hierarchical Process En français, processus analytique hiérarchique. C'est une technique d'analyse et d'organisation de décisions complexes. 30

ANP Analytic Network Process. 34

arbre binaire de recherche semi-équilibré Plus communément appelé arbre AVL, c'est un arbre dont: chaque nœud a au plus deux fils, à chaque nœud le fils droit a une valeur supérieure, à chaque nœud le fils gauche a une valeur inférieure, et à chaque nœud la différence de profondeur des deux sous-arbres gauche et droite est au maximum de 1. 49

AVL Adelson-Velsky et Landis. v, 56

B^{+Cloud}-tree Proposée par Lin & al. [81], la structure B^{+Cloud}-tree est une mise à jour de B^{Cloud}-tree. Des services similaires sont plus susceptibles d'être proches dans une B^{+Cloud}-tree que dans une B^{Cloud}-tree. Un tel changement donne des résultats plus pertinents, en de meilleurs délais. 43

Baseline Une méthode vérifiant tous les cas possibles afin de résoudre un problème mathématique. À l'inverse de toute méthode d'optimisation, cette approche trouve toujours la solution optimale, mais consomme énormément de ressources de calcul et de mémoire (Ceci n'est pas une définition formelle, mais simplement une définition que beaucoup de chercheurs ont associé au terme *Baseline*.). 28

B^{Cloud}-tree En Anglais *Binary Cloud Tree*, qui veut dire arbre binaire de nuage. C'est une structure en arbre pour indexer les services cloud computing, elle a été utilisé par Sundareswaran & al. [142] pour indexer les services cloud computing IaaS. 43

BLE Ce qui veut dire *Bluetooth Low Energy* en Anglais. Et comme son nom l'indique, c'est une version qui consomme le minimum possible d'énergie du protocole de communication Bluetooth. 4

°C degré celsius. 4

CC Cloud Computing. 1

cloud industriel Proposé par le professeur Bo Hu Li et le professeur Lin Zhang, en Chine, en 2009. Le cloud industriel nécessite l'utilisation du cloud computing dans un environnement industriel afin de transformer l'usine en un ensemble d'éléments intelligents, interactifs et organisés en fonction d'un but commun qui est de produire les articles avec une efficacité accrue. 1

CSIF Cloud Service Infrastructure Framework. 20

Ctrix Xen Ctrix Xen est hyperviseur qui offre le déroulement de plusieurs machines virtuelles concurrentielles sur la même machine matérielle. 17

Dash7 Alliance Protocol Protocole de communication basé sur la norme ISO/IEC 18000-7. Il a été conçu spécialement pour les nœuds sans fil. 4

distance de Minkowski Considérée par la majorité des mathématiciens comme étant la formaule mère de la distance euclidienne, la distance de Minkwski est une métrique d'espace vectoriel normée. Elle se formule par: $\sum_{i=0}^{n} |x_i - y_i|$, alors que les x_i et les y_i sont les coordonnées de 2 points dans $\mathbb{R}^n$. 36

FIS Fuzzy Inference System. 27

Fonction à courbe Z Présentée par Guy MacDonald Morton en 1966, la fonction à courbe Z (d'ordre Z, de Morton... etc.) est très utilisée dans les bases de données commerciales. Son principe est de relier un espace multi-dimensions à un autre d'une seule dimension tout en gardant la localité des points de données. Ceci veut dire que si 2 éléments de l'espace de départ sont proches l'un de l'autres, ils se verront attribuer 2 valeurs proches aussi dans l'esapce d'arrivée. 49

fonction ordre Z La fonction ordre Z, la courbe de Lebesgue, l'ordre de Morton ou la courbe de Morton est une fonction qui a comme ensemble de départ un espace multidimensionnel, mais comme ensemble d'arrivée un espace à une seule dimension. Le tout en gardant la localité des points de données. 43

GB GigaByte. 56

GHz Giga-Hertz. 56

Go Giga-octet. 7

Google Drive Un produit de Google permettant de stocker, gérer et partager des fichiers en ligne [5]. 7

Graph Edit Distance Proposée par Alberto Sanfeliu and King-Sun Fu en 1983, la Graph Edit Distance est une mesure de similitude ou de dissimilitude entre 2 graphes. Elle se formule par: $GED(g_1, g_2) = \min_{(e_1,\ldots,e_k)\in P(g_1,g_2)} \sum_{i=1}^{k} c(e_i)$, alors que c est une fonction de coût et $\mathcal{P}(g_1, g_2)$ l'ensemble des possibilités transformant g_1 en g_2. 41

Graph Matching Le *graph matching* est un problème d'optimisation essayant de trouver une similarité entre les graphes. Pour ce faire, les chercheurs utilisent en général la graph edit distance pour calculer le coût des possibilités. Il faut savoir que le passage d'un graphe à un autre n'est pas toujours exacte. Dans ce cas, on parle de *inexact graph matching*. 41

H Heure. 63

Heroku Un service cloud computing donnant accès à un système de gestion de bases de données. Il est principalement mesuré au nombre de lignes dans toutes les tables de la base de données réunies. Il est par exemple gratuit en dessous de 10000 lignes, mais payant pour plus [8]. 7

IaaS Infrastructure as a Service. 76

IEEE 802.15.4 Un standard de communication radio largement utilisé par les protocoles actuels comme ZigBee. 5

Indexation Un processus consistant à filtrer, organiser et représenter certaines données et caractéristiques à propos de quelques objets afin de faciliter leurs recherches. i

Informatique en cluster Un ensemble de machines regroupées, connectées et travaillant ensemble. À la différence des grilles informatisées, les nœuds des clusters exécutent souvent les mêmes tâches. Ces dernières sont contrôlées par un logiciel superviseur. 14

Informatique en grille Un ensemble de machines réparties, connectées et travaillant ensemble. À la différence des clusters informatisés, les nœuds des grilles exécutent souvent des tâches différentes. 14

intervalles à ensemble neutrosophique Les ensembles neutrosophiques ont été développés afin de représenter des informations incomplètes, incertaines, imprécises et inconsistentes existantes réellement dans notre monde. De leurs côté, les intervalles d'ensembles neutrosophiques ont justement été développé pour représenter une plage de ces données. 36

ISM Interpretive Structure Modeling. 34

Ko/s/user Kilo-Octet par seconde par utilisateur. 63

Mark Weiser Né dans les années 50 en Illinois, Mark Weiser est considéré par la communauté informatique comme le père de l'informatique ubiquitaire. D'ailleurs, il a lui même choisi le terme en 1988. 12

MEMSIC Entreprise spécialisée dans le commerce des produits tels que les tableaux de capteurs et les nœuds à capteurs. 5

Micro Controller Un processeur tellement miniaturisé qu'il n'ait plus besoin de ventilation ou d'un quelconque système de refroidissement afin de fonctionner correctement. 5

mm millimètre. 4

mV milli Volt. 4

RCSF Réseau de Capteurs Sans Fil. 5

RCSFs Réseaux de Capteurs Sans Fil. 1

Rough Set Theory Proposé par Zdzislaw I. Pawlak en 1991, le rough set est une théorie en informatique donnant une approximation de bornes à un ensemble d'objets. Ceci se fait si et seulement si nous avons un ensemble d'objets (à borner), un ensemble d'attributs et une fonction donnant la valeur de l'attribut pour un objet donné. 34

Salesforce Un service de cloud computing d'un niveau PaaS proposant des outils de gestion de base de données et des outils de programmtion respectivement comme Visualforce et APEX. 12

Service Level Agreement Un contrat entre l'utilisateur et le fournisseur Cloud Computing. Il décrit un ensemble de contraintes que les deux parties devront respecter. 9

Service Measurement Index Une technique permettant de mesurer un service, en donnant une valeur relative à sa qualité, perçue par un ou plusieurs clients. 30

Skyline Computation Une technique utilisée afin de filtrer les résultats. Son principe consiste à garder les résultats dominants uniquement. Un résultat est dit dominant s'il n'est pas pire que les autres dans le lot de données. 28

SRB Service Ressources Broker. 20

Syntec numérique Syntec Numérique est le syndicat professionnel des entreprises de services du numérique (ESN), des éditeurs de logiciels et des sociétés de conseil en technologies. 11

Texas Instruments Fondée en 1941, et basée à Dallas. La compagnie Texas Instruments fabrique et conçoit des semi-conducteurs et des circuits intégrés pour des fournisseurs d'électronique en général. vi, 4

To Téra-octet. 7

TRSC Trust-based Recommandation System in service-oriented Cloud computing. 26

Virtual PC Windows Virtual PC, est un programme de virtualisation de Microsoft pour leur système d'exploitation Windows. Il a été initialement lancé le 19 Septembre 2009 et arrêté le 14 Février 2011. 17

VMWare vSphere 4 VMWare vSphere (formellement VMware Infrastructure 4) est une plateforme de virtualisation pour un environnement cloud computing de la compagnie VMWare. 17

ZigBee Protocole de communication basé sur la norme IEEE 802.15.4. Il est utilisé très souvent par les périphériques mobiles contraints à des batteries faibles ou relativement faibles. 4

Introduction générale

De nos jours, de plus en plus de gens utilisent les réseaux de capteurs sans fil. D'un côté, ils sont pratiques, mobiles, compacts et économiques et de l'autre côté, ils ont des contraintes de batterie et de ressources limitées comme celles du calcul, du stockage et de la communication. C'est justement l'une des raisons pour lesquelles, le cloud computing a été intégré aux Réseaux de Capteurs Sans Fil (RCSFs). Les nœuds sans fil peuvent générer des données qu'ils ne peuvent même pas traiter et stocker, il est donc souhaitable d'envoyer ces derniers à des services cloud computing distants certes, mais toujours disponibles grâce au réseau Internet.
Beaucoup de travaux se sont intéressés à l'intégration des services Cloud Computing (CC) aux RCSFs tels que :

- La planification des processus dans un environnement en nuage [59].
- La prédiction de performance des services Cloud Computing [63, 62, 101, 36].
- La coopération entre fournisseurs de Cloud Computing [71].
- La sélection des services de virtualisation pour un environnement en nuage [48].
- Le cloud industriel [49, 111, 154].
- La classification des attributs (caractéristiques) des services Cloud selon leurs importances [105].
- La génération d'un modèle de facturation suite à une sélection de métriques de qualité selon les préférences clients [53, 24, 69].
- Le choix du fournisseur le plus confiant au lieu du fournisseur ayant la meilleure qualité ou le moindre coût [106, 149, 132, 145].
- Un modèle de service Cloud Computing communautaire afin de gérer les données transmises par les réseaux de type BSN des patients d'hôpitaux [114].

Le nombre de fournisseurs cloud computing ne cessant de croître, avec le nombre des centres de données ainsi que les services en nuage, il devient nécessaire d'accompagner les administrateurs de RCSFs dans leurs processus de choix du meilleur fournisseur ou le meilleur service, au meilleur prix. Et dans cette problématique, nous avons 2 catégories :

- ceux qui se positionnent du côté du fournisseur [39]. Dans ce cas, les chercheurs essayent d'optimiser la manière d'allouer les ressources [66, 34, 122], de placer les services [97], de coopérer entre plusieurs fournisseurs [71, 117, 92], de fédérer entre les services cloud [123], d'équilibrer les charges [44], d'estimer l'utilisation des ressources [165], d'évaluer l'utilisation des ressources [28], de gérer les factu-rations [137], et même de planifier [45] ou organiser [90] l'exécution des tâches dans le nuage ;
- ceux qui se positionnent du côté client [142]. Dans ce cas, les chercheurs essayent d'optimiser la sélection des services ou de types de services [32], la découverte de services [139], la collaboration entre les utilisateurs [130], le plan d'exécution de plusieurs services [20] et la composition de services [98, 102] ;

Dans notre approche, nous nous sommes positionnés du côté client pour sélectionner des services cloud computing de type IaaS publique, pouvant traiter les données générées par des nœuds de capteurs sans fil tout en réduisant le temps de la recherche et en maximisant la pertinence des résultats dès le début de la recherche.
Un point très important pour se faire est l'indexation, car si elle est mal faite, notre approche ne peut pas retourner des résultats aussi pertinents. Donc, notre deuxième contribution s'est faite sur la problématique de l'indexation, consistant en l'ordre des services selon les caractéristiques si et seulement s'ils représentent un coût dont dispose le client.
Après avoir passé en revue les solutions de la communauté scientifique, 2 travaux nous ont intéressés. Ceux de Sundareswaran & al. [142] et Lin & al. [81] qui ont le même objectif que le notre : maximiser la pertinence des résultats en un temps raisonnable. Notre contribution intervient dans l'ajout d'une méthode principale afin de valider les services, tout en prenant en compte leur hétérogénéité, leurs qualités et les préférences des clients. Après l'évaluation de nos résultats, nous les avons comparé à ceux des 2 méthodes précédentes, notre approche fournit de bien meilleurs résultats dès le début, et achève sa recherche plus tôt.
Notre approche peut d'un côté faciliter pour les clients de cloud computing la sélection de meilleurs fournisseurs, et de l'autre côté, économiser le temps nécessaire à la recherche.
Cette thèse est organisée comme suit :

- le chapitre 1 énonce quelques concepts théoriques nécessaires à la compréhension du domaine, de la problématique, des approches de référence, ainsi que les travaux similaires. Nous présentons par exemple la définition du Cloud Computing, ainsi que celle des capteurs et les réseaux sans fil ;
- le chapitre 2 présente un état de l'art regroupant les différents travaux dans le domaine, ayant le même objectif, la même démarche ou la même problématique ;
- le chapitre 3 décrit notre approche en détail ainsi que notre apport par rapport aux deux approches que nous avons étendues [142, 81]. Nous donnons par exemple les différences (validation des services et calcul du score), ainsi que les points forts de notre contribution par rapport aux deux travaux de référence [142, 81] (prise en compte de la diversité des services, préférences de clients. . . etc.) ;
- le chapitre 4 propose de dérouler un exemple (cas d'étude) de la contribution proposée dans cette thèse. Pour des raisons de lisibilité et de compréhensibilité, nous avons préféré utiliser un jeu de données réduit pour l'exemple ;
- le chapitre 5 décrit, illustre et interprète les résultats de la simulation, comparant le travail proposé à ceux de Sundareswaran & al. [142] et Lin & al. [81]. Ceci regroupe généralement les résultats (services sélectionnés), leurs scores (leurs qualités) et le temps nécessaire pour les retrouver.
- nous finissons la thèse proposée par une conclusion et des perspectives ;

Chapitre 1

Généralités

1.1 Introduction

Les réseaux de capteurs sans fil envahissent de plus en plus la vie quotidienne de ses utilisateurs, ils sont utilisés par les médecins (capture de battements cardiaques), les agriculteurs (détection de l'étanchéité du sol), les météorologues (surveillance de la température et de la vitesse du vent) et les gardes forestiers (surveillance de feux de forêts). Ceci dit, la liste précédemment présentée n'est pas exhaustive.
Puisque les nœuds de capteurs sans fil n'étaient pas destinés à la base pour une telle charge de travail et que les données qu'ils génèrent ne cessent de croitre, le cloud computing a été intégré aux réseaux en question.
Dans ce chapitre, nous allons passer en revue des concepts nécessaires à la compréhension de notre contribution. Ça comprend les réseaux de capteurs sans fil et l'informatique en nuage.

1.2 Généralités sur les réseaux de capteurs sans fil

Dans cette section, nous allons passer en revue les définitions et concepts pour les réseaux de capteurs sans fil. Nous nous intéressons généralement aux objectifs, composants et procédés des RCSFs.

1.2.1 Capteur

Un capteur est un dispositif informatique qui détecte un changement dans l'environnement dans lequel il évolue. Il peut surveiller la chaleur, l'humidité, l'accélération, la luminosité... etc [134]. Après production d'une modification dans le milieu physique, le capteur génère des données, devant être transmises sous la forme d'un signal électrique vers une station de base [134].

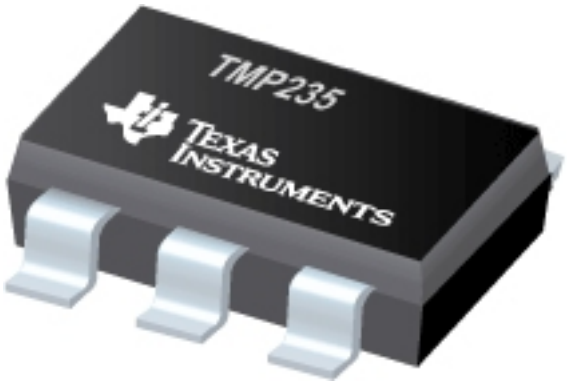


FIGURE 1.1 – Capteur de température TMP235 de Texas Instruments [2]

La figure 1.1 représente un capteur de température à sortie analogique[1], via un signal électrique. Une augmentation en température de 1C fait hausser le signal de 10mV et vice versa [2]. D'un autre côté, le capteur en tout et pour tout ne mesure pas plus que 3mm [2] ne donnant ainsi qu'un signal électrique et laissant le reste des tâches comme l'envoi des données dans le réseau ou le traitement à d'autres composants du nœud sans fil.

1.2.2 Nœud à capteur

Un nœud à capteur(s) est un élément informatique compact, léger dont le but est de donner au(x) capteur(s) les ressources minimales en termes de communication, calcul et stockage respectivement afin de transmettre, traiter et stocker les données générées par le dispositif de capture lui-même [31].

1.2.3 Nœud à capteur sans fil

Un nœud à capteur sans-fil est un nœud à capteur dont le dispositif de communication est sans fil [133] (généralement une antenne radio plate [30]). Ce dernier utilise généralement des protocoles de communication radio tels que le ZigBee, Dash7 Alliance Protocol, Wifi et BLE afin de transmettre et recevoir les paquets des données générées.

1. Il existe des capteurs à sortie digitale et d'autres à sortie analogique. Ceux possédant une sortie analogique donnent un signal électrique analogique, alors que les autres produisent un signal sous forme de séquence de valeurs discrètes.

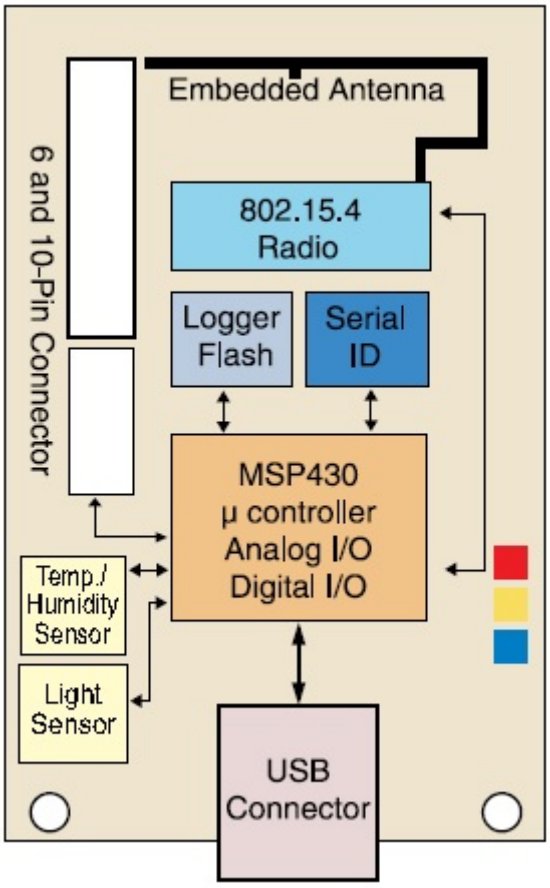


FIGURE 1.2 – Diagramme de blocs d'un nœud à capteurs []

La figure 1.2 présente un nœud sans fil très utilisé dans notre laboratoire et anciennement la propriété de MEMSIC. Comme nous le voyons, le capteur passe le signal au Micro Controller. Ce dernier créera ensuite des paquets en temps voulu et l'enverra au module radio. Enfin, le composant de transmission radio envoie le tout par le biais de l'antenne embarquée sous la norme IEEE 802.15.4.

1.2.4 Réseau de capteurs sans fil

Un Réseau de Capteurs Sans Fil (RCSF) est un ensemble organisé de nœuds sans fil munis d'un ou plusieurs capteurs [9]. L'objectif du réseau est d'acheminer les données collectées par les dispositifs de capture à une station de base, pour un traitement ultérieur, une analyse approfondie ou un simple stockage [31].

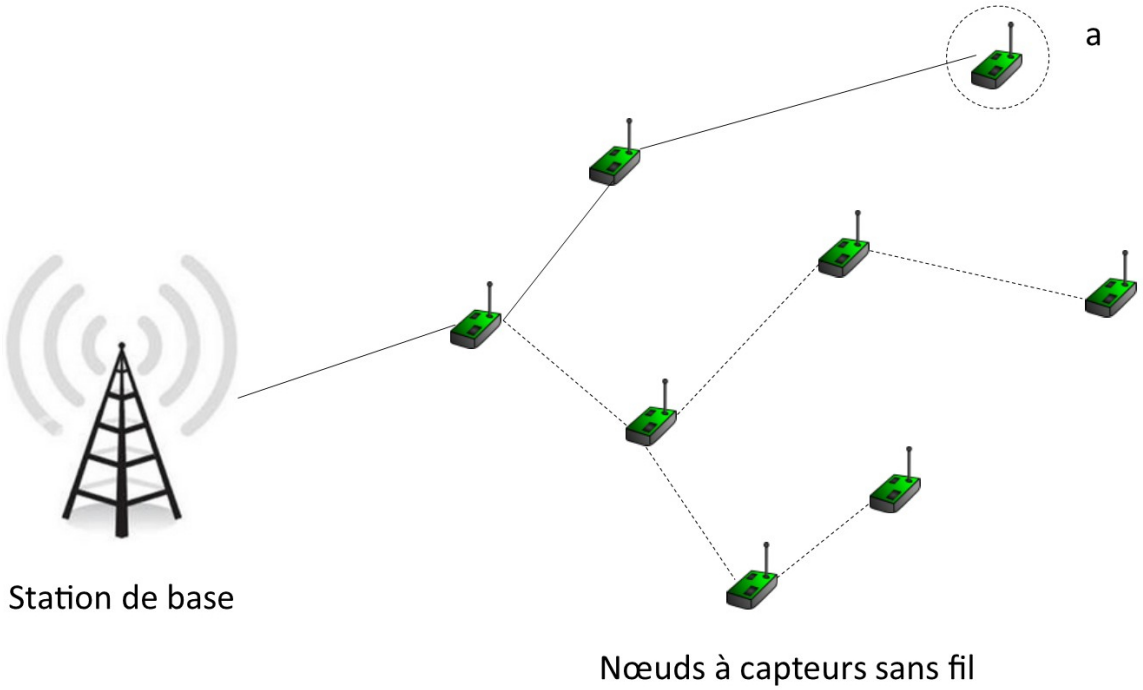


FIGURE 1.3 – Exemple d'un réseau de capteurs sans fil

La figure 1.3 nous montre la collaboration entre les nœuds sans fil afin de rallier les paquets de données générés par les capteurs à la station de base. Plus précisément, le paquet du nœud à l'emplacement **a** (coin haut droit dans la figure 1.3) doit passer par ces voisins afin d'atteindre la station de base.

1.3 Généralités sur le Cloud Computing

Dans cette section, nous allons présenter brièvement les concepts du Cloud Computing. Nous ne nous contentons pas de définir seulement, mais de présenter aussi ses niveaux, ses modèles de déploiement, ainsi que ses caractéristiques. Nous allons clôturer la section par une comparaison entre le cloud computing avec 2 concepts qui lui sont similaires : l'informatique à la demande et l'informatique ubiquitaire.

1.3.1 Définition

Le Cloud Computing (ou informatique en nuage) est un modèle permettant un accès à des ressources (physiques ou virtuelles) configurables à distance [94].
L'accès doit d'un côté être ubiquitaire et à la demande des clients. Et de l'autre côté, il doit nécessiter un minimum d'effort de la part du client comme ne doit en aucun cas solliciter l'interaction du fournisseur [95].
Le nuage informatisé doit assurer une certaine qualité pour ces clients tel qu'un taux de disponibilité minimum (généralement de 99,95% [2]) [94, 51].

2. Pour un tel pourcentage, le fournisseur aura à peine 4 heures et demi par an afin de gérer ses pannes.

1.3.2 Caractéristiques du Cloud Computing

Bien que la définition précédente (voir la sous-section 1.3.1) liste un ensemble de caractéristiques permettant de distinguer le modèle du Cloud Computing des autres modèles et architectures (comme la virtualisation, l'informatique à la demande et l'informatique ubiquitaire), il est important de décrire chaque critère à lui seul. Dans ce qui va suivre, nous allons justement détailler les propriétés du Cloud Computing.

1.3.2.1 Service à la demande

Le client peut s'allouer autant de ressources que nécessaire, et ceci à n'importe quel moment sans intervention avec les services du fournisseur mais toujours à sa demande.

1.3.2.2 Accès via Internet

Toutes les ressources du nuage informatique sont accessibles par le biais du réseau Internet. Ces moyens sont à disposition de tout type d'internautes qu'ils soient lourds ou légers.

1.3.2.3 Groupement de ressources

Les différentes ressources sont groupées pour un ensemble de clients. Selon la charge des processus et leur nombre, les ressources sont allouées et libérées.
Les consommateurs n'ont aucune connaissance sur la position exacte de leurs traitements. Ceci dit, ils peuvent choisir dans quel centre de données par exemple dérouler leurs services, mais pas dans quel serveur exactement.

1.3.2.4 Flexibilité rapide

Les ressources doivent être allouées, supprimées, étendues ou réduites de la manière la plus facile et rapide possible.
Pour les clients, ces dernières paraissent comme illimitées, accessibles et modifiables à n'importe quel moment et à partir d'un quelconque emplacement connecté, comme si elles étaient locales.

1.3.2.5 Service mesuré

Les utilisateurs payent ce qu'ils utilisent uniquement. Le service est donc mesuré par son utilisation. Ceci peut varier selon le type de service, mais le principe reste toujours le même. Par exemple :

- un service de stockage peut mesurer l'espace alloué [3], sans pour autant facturer les autres ressources comme le processeur et la bande passante ;
- un service de base de données peut mesurer la taille des tables [4], sans pour autant inclure le coût de leur création et leur traitement ;

3. Google Drive est certes gratuit pour 10Giga-octet (Go), mais payant pour 100Go, 1 Téra-octet (To) et 10To [5].

4. Heroku est certes gratuit pour une petite base de données, mais payant pour les plus importantes [8].

1.3.3 Modèles de déploiement Cloud Computing

Déployer un nuage informatisé peut se faire selon quatre modèles différents. Le principal critère les différentiant est le choix de l'audience.

1.3.3.1 Cloud Computing publique

Bien qu'il soit géré par une entreprise particulière [5] proposant des services de Cloud Computing, le modèle publique du Cloud Computing reste un modèle accessible par tout internaute connecté. Par conséquent, il est le moins sécurisé, mais pas pour autant le moins utilisé.

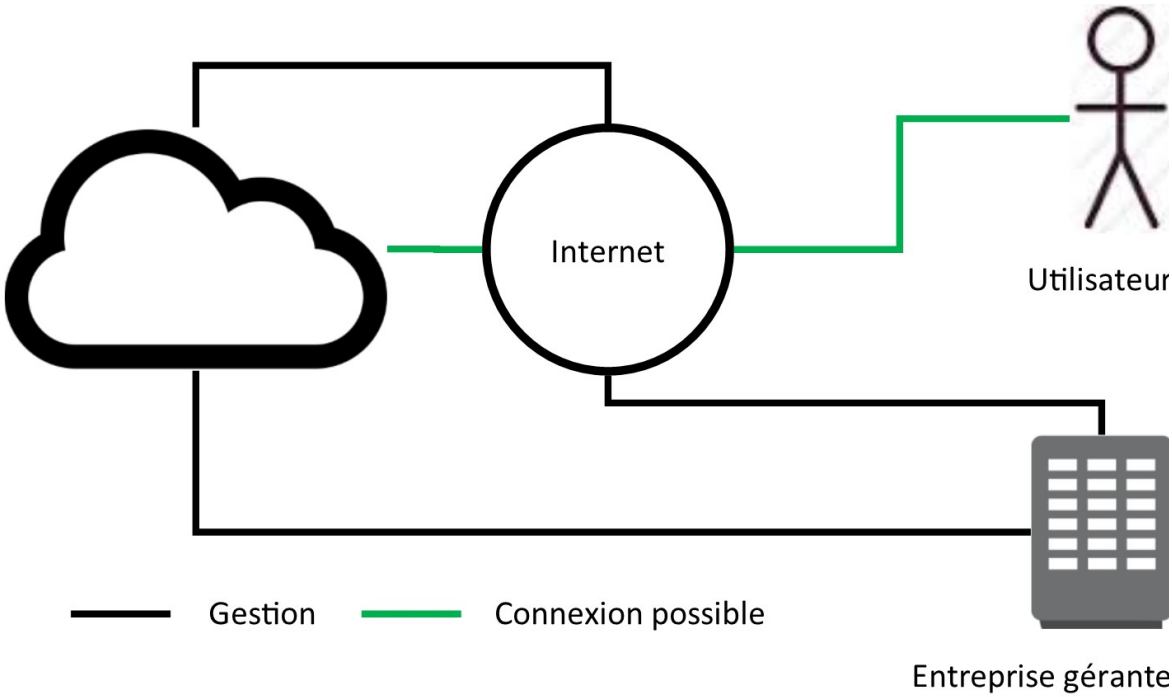


FIGURE 1.4 – Modèle de déploiement publique du cloud computing

La figure 1.4 illustre un cas d'exemple pour le modèle publique du cloud computing. Un service publique en nuage reste toujours accessible au grand publique. La seule condition est qu'il doit être connecté à Internet. Comme tout matériel, il doit être géré par un organisme, une entreprise... etc. Mais ceci pourrait être fait d'une manière locale ou via Internet.

1.3.3.2 Cloud Computing privé

L'infrastructure du nuage ou d'une partie du nuage est dédiée à une entreprise spécifique. Un internaute non concerné par l'entreprise ne pourra pas accéder aux installations privées. Ceci dit, l'infrastructure pourrait être gérée par l'entreprise en question (cliente) ou une tierce (gérante).

5. Comme Azure par Microsoft et Elastic cloud par Amazone.

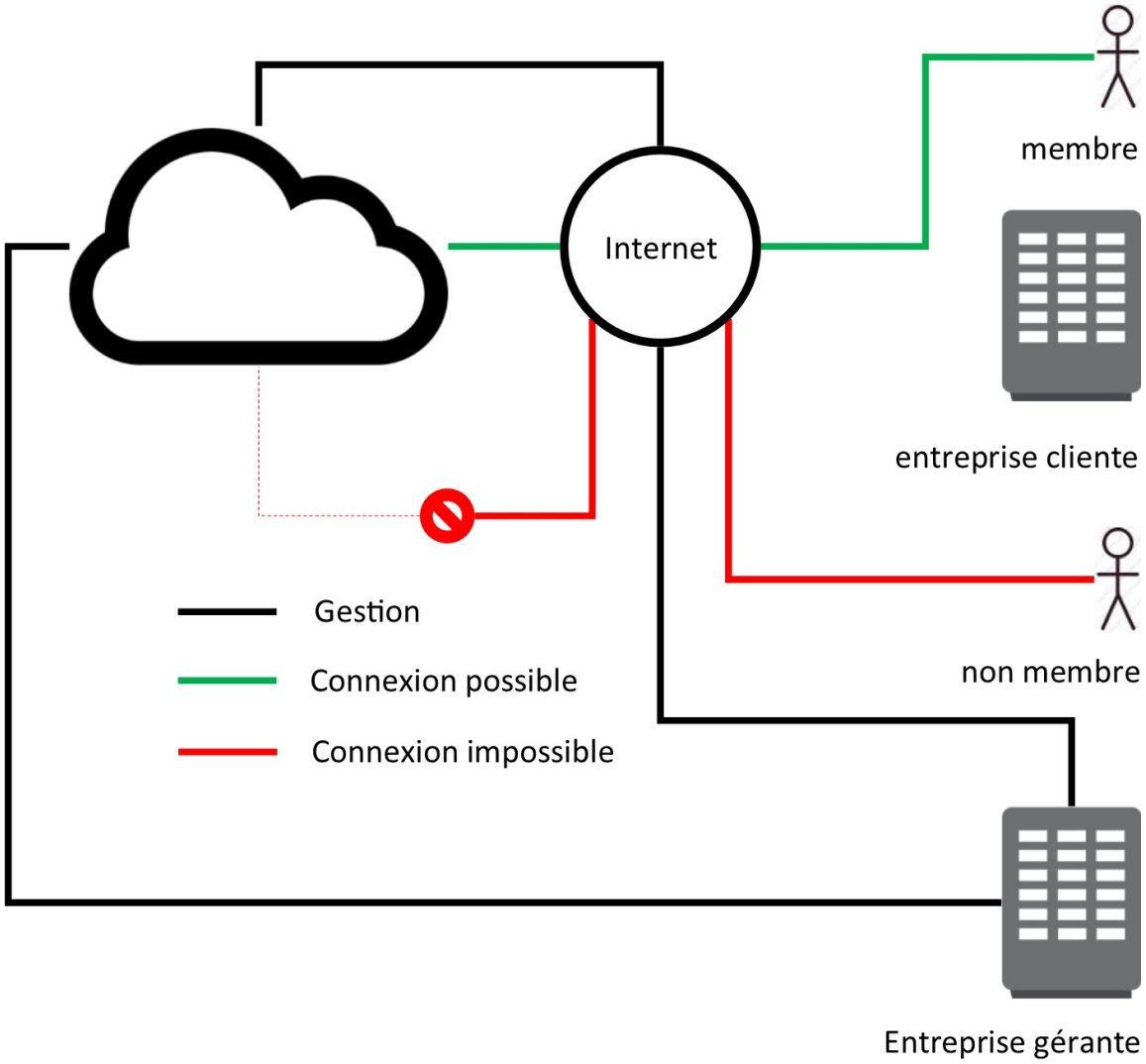


FIGURE 1.5 – Modèle de déploiement privé du cloud computing

La figure 1.5 illustre un exemple d'un modèle de déploiement privé du cloud computing. Un membre de l'entreprise cliente peut accéder aux installations qui lui sont dédiées alors qu'un autre ne pourra pas. D'un autre côté, le matériel peut être géré par un organisme tiers (entreprise gérante) ou l'entreprise elle-même (cliente).

1.3.3.3 Cloud Computing communautaire

L'infrastructure du nuage ou une partie du nuage est dédiée à une communauté d'entreprises partageant les mêmes requis et préférences spécifiées dans leurs contrats de type Service Level Agreement. L'infrastructure en question peut être gérée par la communauté en totalité, une partie de cette dernière, une entreprise membre ou un organisme tiers.

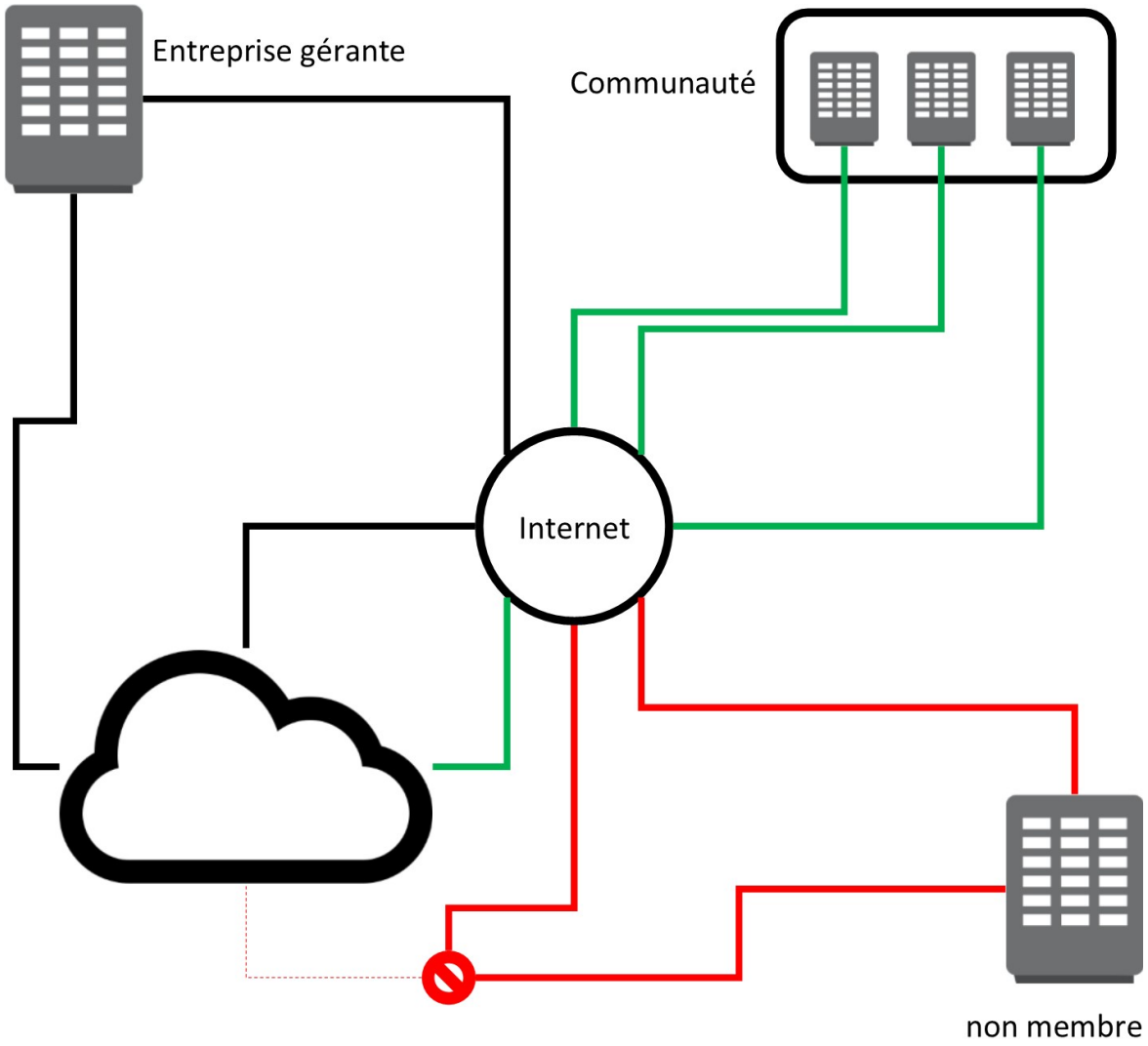


FIGURE 1.6 – Modèle de déploiement communautaire du cloud computing

La figure 1.6 illustre à titre d'exemple un modèle de déploiement communautaire. La gestion de l'infrastructure peut être exercée par l'entreprise en dehors de la communauté (entreprise gérante). Mais, peut être aussi faite par la communauté elle-même ou par une partie de la communauté. Ceci dit, un groupe d'entreprises doit avoir au moins un minimum de requis et de principes en commun afin de bénéficier et d'en tirer au maximum des bénéfices d'un tel modèle.

1.3.3.4 Cloud Computing hybride

L'infrastructure est une composition d'au moins 2 modèles précédemment cités.

1.3.4 Niveaux de services Cloud Computing

1.3.4.1 Infrastructure-as-a-Service (IaaS)

On donne aux utilisateurs des ressources accessibles et partagées, se présentant sous forme d'unités de calcul, de supports de stockage, de moyens de communication (réseau)... etc. Avec ces moyens là, les consommateurs pourront dérouler leurs propres

logiciels et outils à distance. Ceci veut dire que le client a le contrôle sur le système d'exploitation, l'aspect applicatif et le stockage. Mais il n'a pas la gestion des couches basses comme l'infrastructure et le réseau matériel. Ceci dit, il peut parfois avoir la gestion logicielle et partielle du réseau (si le fournisseur autorise cela) comme le choix d'un pare-feu.

1.3.4.2 Software-as-a-Service (SaaS)

Utiliser des applications web s'exécutant sur les serveurs du nuage. Les logiciels sont accessibles par des clients différents, répartis sur le globe. Ces derniers n'ont de contrôle que sur les paramètres laissés ouverts explicitement par le fournisseur du service. Mais n'ont aucun contrôle sur la gestion de l'infrastructure, le système d'exploitation, le réseau ou de l'application elle-même...etc.

1.3.4.3 Platform-as-a-Service (PaaS)

Un niveau intermédiaire entre les 2 niveaux précédemment cités est le PaaS. Le client peut dérouler des applications de sa création sur les outils offerts ou proposés par le fournisseur. Ces derniers peuvent regrouper les bases de données et les langages de programmation. Le consommateur ne gère pas les couches basses comme les serveurs et les systèmes d'exploitation. Mais il a tout de même le contrôle sur les applications déployées ainsi qu'une possibilité de la configuration de l'environnement les hébergeant.

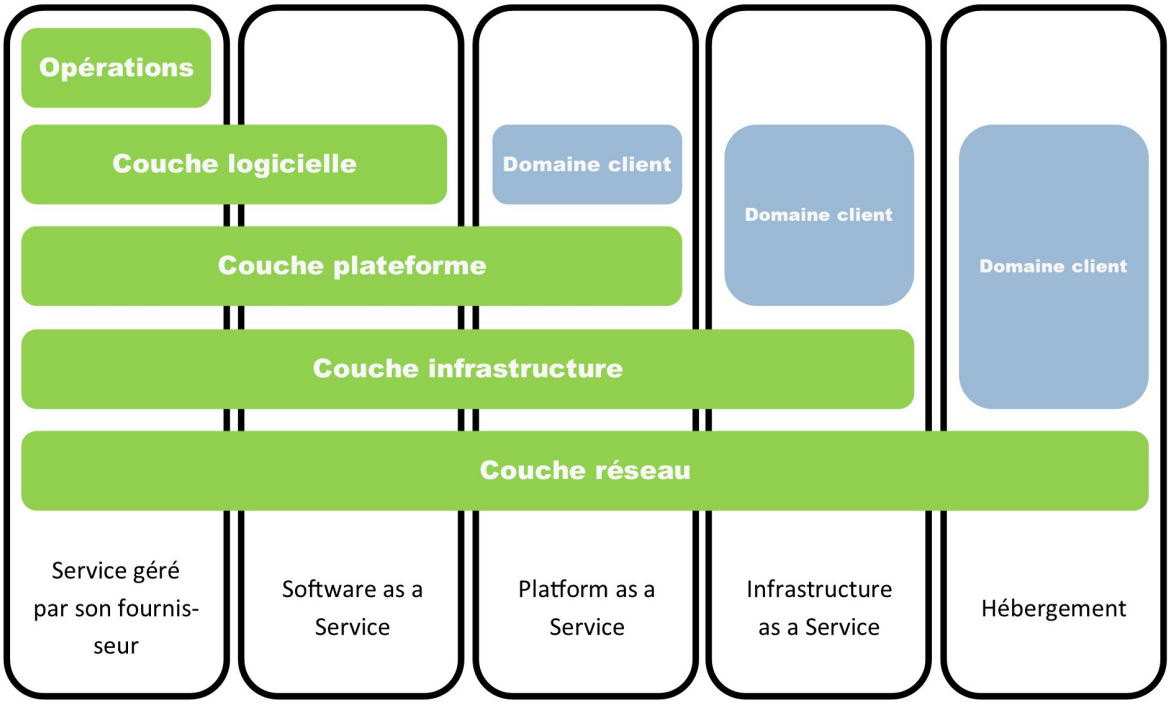


FIGURE 1.7 – Répartition des gestions client/fournisseur dans un environnement en nuage selon le niveau [46]

La figure 1.7 nous montre la façon dont Syntec numérique fait la différence entre les différents niveaux de cloud computing. Nous pouvons citer Facebook et la plupart

des produits Google comme représentation des plus célèbres du modèle SaaS.
Le PaaS est un niveau assez difficile à comprendre vu l'ambiguïté dans le départage du contrôle entre le client et le fournisseur. Par exemple, le client peut avoir uniquement le contrôle sur des données logiques comme le service de Heroku [6]. Ou il peut avoir le contrôle sur un SGBD réel mais dans un environnement cloud comme MySQL Cloud [1]. Et pour finir, le client peut avoir le contrôle sur tout un environnement de développement intégré comme Salesforce [6].
Le niveau IaaS donne le contrôle au client à une machine virtuelle à partir de son système d'exploitation. Par conséquent, le client a un ordinateur distant qu'il ne possède pas, mais qui est toujours disponible et moins cher.

1.3.5 Différences par rapport aux concepts similaires

Bien qu'il existe une définition claire, formelle et très bien référencée [94] à propos du cloud computing, les utilisateurs font toujours des amalgames entre l'informatique en nuage et d'autres concepts comme l'informatique ubiquitaire et l'informatique à la demande.
Dans ce qui va suivre, nous allons lister quelques notions similaires au cloud computing, les définir et les comparer.

1.3.5.1 Cloud Computing VS Virtualisation

De nos jours, la virtualisation est la clé de base du Cloud Computing. Mais, ce n'est pas obligatoire qu'un nuage informatisé soit virtualisé. Il suffit (comme nous l'avons dit dans la sous section 1.3.1 et la sous-section 1.3.2) que les ressources soient partagées via Internet, qu'elles soient accessibles de partout à n'importe quel moment...etc. Il existe plusieurs conditions mais la virtualisation n'y figure pas. Ceci dit, il est en pratique difficile d'assurer la disponibilité de ressources partagées, extensibles et à la demande, sans oublier le fait que tout choix du client doit s'exécuter avec un effort minimal et ne doit pas nécessiter l'intervention du fournisseur. Avec la demande incessante et toujours croissante du Cloud Computing, la virtualisation est devenue une facilité dont les fournisseurs ne se privent plus. D'ailleurs, la virtualisation est tellement intégrée dans les plateformes en nuage actuelles qu'il est difficile de ne pas mentionner le terme "virtuel" dans la définition du Cloud Computing.

1.3.5.2 Cloud Computing VS Ubiquitous Computing

Quand Mark Weiser donna naissance au concept de l'informatique ubiquitaire [153], elle était désignée aussi par le terme de *Calm Technology*. L'idée initiale était de mettre plusieurs serveurs pour un client au lieu du contraire dans l'architecture client-serveur [152].
La première caractéristique pour ces services ubiquitaires est qu'« ils sont invisibles » [152]. Mark Weiser prend l'exemple des lunettes de vue [152], effectivement, une personne portant des lunettes n'est pas obligée de se concentrer sur son outil pour voir. Elle peut tout simplement voir le monde sans se soucier des détails d'utilisation.
Le cloud computing est exactement pareil, car nous donnons aux utilisateurs un accès

6. L'entreprise Salesforce propose des services en nuage pour la gestion des relations clients, elle offre la possibilité de lancer un site de vente (juste pour exemple) le plus rapidement possible sans installer quoi que ce soit [3].

à des ressources informatiques. Puisque ces dernières semblent illimitées, accessibles de partout et utilisables comme s'ils étaient proches. Nous pouvons dire qu'ils sont invisibles [151].
En d'autres termes, nous pouvons dire qu'un service cloud computing est un service ubiquitaire.

1.3.5.3 Cloud Computing VS On-demand Computing

On dit *Informatique à la demande* ou *Informatique utilitaire*, et c'est un modèle de fourniture de ressources informatiques aux clients qui les facture selon leurs utilisations [38].
Les gens ont rapidement abandonné cette idée pour aller vers l'informatique en nuage [41]. Bien que tout comme les nuage informatisés, l'informatique utilitaire est un :

— service mesuré et à la demande ;
— groupe de ressources ;

Mais contrairement aux nuages informatisés, l'utility computing n'est pas forcément un service :

— accessible par Internet ;
— facile et/ou rapide à étendre ;
— utilisable en un minimum d'effort par rapport au client ;
— ne nécessitant aucune intervention de la part de son fournisseur ;
— accessible à la fois par des clients légers et lourds en même temps ;
— dont les client ignorent où s'exécutent exactement leurs tâches ;

Puisque les caractéristiques de l'informatique utilitaire font toutes partie des propriétés de celle en nuage. Nous pouvons dire qu'un service cloud computing est un service utility computing.

1.3.5.4 Comparaison globale

Critère	Utility	Ubiquitous	Cloud
Virtualisation	Non	Non	Pratique
Service à la demande	Oui	Non	Oui
Effort minimal du client	Non	Oui	Oui
Aucune intervention du fournisseur	Non	Non	Oui
Accès via Internet	Non	Oui	Oui
Adapté aux clients lourds et légers en même temps	Non	Non	Oui
Groupement de ressources	Oui	Non	Oui
Contrôle des ressources par le client	Non	Non	Non
Connaissance de l'emplacement des ressources par le client	Non	Non	Non
Flexibilité rapide	Non	Oui	Oui
Service mesuré	Oui	Non	Oui

TABLE 1.1 – Tableau comparatif entre concepts similaires au cloud computing

On constate à partir de la table 1.1 que le cloud computing, en plus d'avoir ses propres caractéristiques, a pu combiner les avantages des deux visions. L'informatique en nuage permet d'un coté aux utilisateurs d'utiliser des ressources toujours présentes d'une manière invisible. Et d'un autre coté, elle offre à ces clients des services facilement utilisables à la demande. Afin de garantir ces deux critères légèrement opposés, la notion de cloud computing avait besoin de conditions supplémentaires comme l'adaptation des clients lourds et légers en même temps.
D'autres concepts se rapprochent du cloud computing comme l'Informatique en grille et l'Informatique en cluster. Mais, ce sont plus des notions de collections de ressources qui font des traitements parallèles que des services à la demande. En d'autres termes, ce sont plus des notions orientées vers le traitement que l'accès ou les clients.
On peut citer aussi L'informatique en jungle (Jungle Computing) [88]. Ça se manifeste par le groupement de l'informatique en nuage, en grille et en cluster [125].

1.4 Intégration des réseaux de capteurs sans fil au Cloud Computing

Dans cette section, nous allons passer en revue l'utilité de l'intégration des RCSFs aux services CC, les problèmes liés à une telle intégration et quelques solutions proposées.

1.4.1 Objectifs à atteindre

D'un côté, nous avons des capteurs avec des ressources informatiques portables et une alimentation limitée, et de l'autre, nous avons des ressources quasi-illimitées en termes de communication, stockage et traitement accessibles de partout, à la demande et à un coût relativement faible.
La raison de l'intégration des réseaux de capteurs sans fil au Cloud Computing n'a pas été faite parce qu'ils se ressemblent, mais parce qu'ils se complètent. Prenons un exemple simple, supposons un réseau de 500 nœuds munis chacun de 9 capteurs générant chacun 1 octet par seconde. Nous avons donc 370 Mo par jour, ce qui donne à peu près 10 Go par mois. En commençant avec un disque dur de 80 Go dans la station de base, nous pouvons facilement tomber en pénurie d'espace de stockage en à peine 8 mois. Alors que si nous passons la tâche du stockage à des services en nuage, nous pouvons facilement étendre la capacité si besoin est, à la demande, sans efforts, et sans interactions avec le fournisseur.

1.4.2 Problèmes liés à l'intégration

Associer deux technologies complémentaires est certes une idée prometteuse, mais pas toujours facile et surtout pas sans risques.
Un administrateur d'un RCSF peut posséder les nœuds de capteurs tandis qu'il loue le service en nuage. Ce qui veut dire qu'il peut contrôler ce qui se passe dans son RCSF, mais n'a aucun contrôle sur ses données une fois sorties sur Internet.
Par exemple, lors de l'utilisation du service loué, le fournisseur peut déclarer faillite ou changer de mode de facturation, dans ce cas, l'administrateur RCSF devra choisir un autre fournisseur cloud, mais aussi subventionner le transfert des données d'un point

à un autre.
On va donner dans ce qui va suivre une liste non exhaustive de problèmes rencontrés en intégrants les RCSFs aux services CC.

1.4.2.1 La sécurité

Après les multiples attaques et tentatives d'attaque sur les grands services hébergés en cloud ou ceux des fournisseurs eux-mêmes, la sécurité dans les nuages est devenue un souci constant et une priorité absolue. Il faut noter que plus le nuage est accessible par plus de monde, plus il est vulnérable aux attaques. Dans un modèle de déploiement publique du cloud computing, il est accessibles à ses clients comme il l'est à ses attaquants. Même s'il existe des contre mesures afin d'assurer un certain niveau de sécurité, le modèle privé reste jusqu'à nos jours le modèle le plus sécurisé dans le cloud computing.
De l'autre côté, les nœuds de capteurs sans fil sont trop pauvres en ressource informatiques afin de prévenir une attaque cybernétique. En plus de cela, les RCSFs sont les réseaux les plus attaqués physiquement : quand un nœud envoie des données au nuage, nous ne pouvons pas être sûrs qu'il est sécurisé. Une personne tierce pouvant facilement changer un nœud ou 2 dans le réseau afin d'espionner le trafic du réseau par exemple.

1.4.2.2 Le manque de contrôle

Si nous comparons un serveur traditionnel avec un service en nuage d'un niveau IaaS, alors le manque de contrôle du client sur le matériel physique fait la différence. Si nous sommes sûrs d'une éventuelle attaque dont nous ignorons sa contre action, alors éteindre un serveur peut s'avérer être une solution radicale mais sûre (et possible). Alors qu'on ne peut pas le faire sur un service distant d'un côté, et partiellement contrôlable d'un autre.

1.4.2.3 La dépendance d'un tiers organisme

Le manque de contrôle n'est jamais le seul problème quand nous utilisons un service en nuage. Car nous dépendons toujours du fournisseur, puisqu'il peut faire des changements sur son matériel ainsi que son modèle de facturation, sans demander l'autorisation ou consulter ses clients (si le contrat ne l'interdit pas).

1.4.2.4 La sélection

Sur le marché, il existe plusieurs services CC. Pour un administrateur de RCSF, lequel va prendre en charge au mieux ses besoins.
Effectivement, rien que les fournisseurs de services cloud computing offrant un niveau IaaS se comptent par centaines facilement. Alors que pouvons-nous dire de tous les fournisseurs confondus, tous leurs services mélangés ainsi que toutes leurs locations mises dans le même lot. Nous aboutirons à un nombre qui ne facilitera pas la sélection aux fournisseurs des RCSFs voulant alléger la charge des tâches sur leurs nœuds sans fil et ainsi passer vers un traitement distant mais toujours accessible.
D'ou notre intérêt pour cette thématique à laquelle nous allons apporter notre contribution.

1.4.3 Solutions proposées

Juste après l'explosion de l'utilisation du cloud computing, J. Brodkin [19] a présenté les sept plus grands risques en utilisant de tels services. Bien que les problèmes mentionnés ne sont pas aussi généraux que la dépendance d'une tierce entreprise ou le manque de contrôle, J. Brodkin [19] avait toujours comme objectif la prise de contrôle de la sécurité par le client. C'est d'ailleurs la solution aux difficultés citées dans les sous-sections 1.4.2.2 et 1.4.2.3.
En ce qui concerne la confidentialité et l'intégrité, plusieurs solutions ont été apportées. Lounis & al. [84] ont proposé une architecture de service cloud computing intégrant les réseaux de capteurs sans fil pour un usage médical. Plus spécialement, ils ont présenté un système afin de collecter les données générées par les capteurs médicaux et un mécanisme de sécurité afin de transmettre ces les données en un minimum de risques. La solution de cryptage proposée a été comparée avec la méthode ABE, et s'est avérée être plus performante en termes de temps d'exécution.
Plus tard, les mêmes chercheurs [85] ont proposé le même système, mais il a été testé selon différents critères. Pour cette fois, Lounis & al. [85] ont préféré la scalabilité et la surcharge au temps d'exécution, vu que le système a été orienté vers les situations d'urgence. Plus clairement, les chercheurs ont comparé leur approche avec celle de ABE selon le temps d'exécution par rapport au nombre d'attributs des données. Après cela, ils ont testé leur contribution selon les requêtes en attente par rapport au temps. Leurs résultats ont pu prouver la faisabilité et la performance de la méthode employée.
En 2017, Kurdi & al. [70] ont proposé un système centralisé dans un nuage regroupant toutes les bases de données hospitalières du royaume de l'arabie saoudite. D'un côté, c'est un travail intégrant les réseaux de capteurs corporels à un service cloud computing. D'un autre côté, c'est un papier plein de promesses, qui ne sont pas prouvées que ce soit par une évaluation ou par une comparaison. Les auteurs ont proposé par exemple un système sécurisé, et ont présenté tout un mécanisme d'authentification, mais ils n'ont pas testé leur système en termes de sécurité (comme la résistance aux attaques... etc).
Concernant les travaux de sélection de services, nous avons préféré les citer dans le chapitre 2 suivant.

1.5 Conclusion

Dans ce chapitre, nous avons proposé des notions de base liées à notre thématique qui sont le Cloud Computing avec ses types de déploiement, ses niveaux et ses caractéristiques, puis le RCSF avec ses contraintes et ses limites. Nous avons ensuite exposé quelques problèmes dûs à l'association des services CC aux RCSFs dont la sécurité, le manque de contrôle... etc.
Les défis sont nombreux et trop longs à détailler, pour cela, on s'est focalisé sur 2 problématiques qui sont la sélection et l'indexation. Pour ce faire, nous allons passer en revue tous les travaux récents sur ces 2 aspects en particulier dans le chapitre suivant.

Chapitre 2

État de l'art

2.1 Introduction

Sélectionner un service ou un ensemble de services pour un RCSF selon des qualités, des préférences et des exigences définies au préalable n'est pas chose aisée. Il faut par exemple filtrer les éléments ne répondant pas aux exigences des clients du nuage. Après cela, il faut tout de même ordonner les résultats selon les préférences des utilisateurs. Durant tout ce processus, il faut toujours mesurer la qualité des services sélectionnés. Vu la difficulté du choix, beaucoup de travaux de recherche ont été proposés.
Dans ce chapitre, nous allons passer en revue brièvement certains travaux touchant de près ou de loin la sélection des services cloud computing, qu'on a essayé de hiérarchiser selon leurs objectifs, problématiques (fournisseurs ou clients). Le but de cet état de l'art est d'approfondir nos connaissances sur les difficultés en utilisant l'informatique en nuage et d'essayer de proposer des solutions à 1 ou 2 aspects. Dans ce chapitre, nous avons aussi essayé d'apporter une critique sur chaque solution apportée dans le temps.

2.2 Historique

Les définitions formelles et référencées jusqu'à nos jours à propos du Cloud Computing [93, 94, 51] étaient publiques à partir de 2008. Il a fallu un temps pour la communauté scientifique afin d'assimiler le concept complexe qu'est l'informatique en nuage. Par conséquent, la sélection de services de Cloud Computing a commencé sous la forme de comparaison entre les différents outils de virtualisation [48].
Par exemple, Han & al. [48] comparèrent en 2009 entre Virtual PC, Ctrix Xen et VMWare vSphere 4 afin de sélectionner le meilleur outil de virtualisation pour les fournisseurs de services CC. Pour ce faire, ils ont d'abord testé les performances des 3 logiciels dans le même environnement. Après, ils ont proposé une formule pour calculer un score (S-Rank), sachant que le calcul est basé sur les statistiques des performances précédemment mesurées. Puis, ils ont supposé pour leurs tests d'évaluation, un ensemble de 10 fournisseurs IaaS et 100 fournisseurs SaaS. Enfin, les chercheurs ont pu démontré l'efficacité de leur approche en termes de temps d'évaluation, par rapport à 2 méthodes (RR et une méthode à sélection aléatoire). Ceci dit, la proposition de Han [48] reste un simple calcul mathématique basé sur des résultats de tests préalablement sélectionnés. Mais dans la réalité, nous ne pouvons pas prédire les traitements que vont

faire les clients des services cloud computing.
Parmi les premiers travaux sélectionnant les services cloud computing, nous pouvons citer Zeng & al. [162]. Ils [162] proposèrent un algorithme en 2 étapes, la première retourne les services disponibles, et la seconde maximise le gain tout en réduisant le coût des services. Les chercheurs ont présenté un formalisme mathématique et un pseudo-code, afin de calculer le gain et le coût des services retournés. Mais, les auteurs n'ont pas jugé nécessaire d'ajouter une évaluation ou une comparaison. En plus de cela, les calculs mathématiques utilisés sont trop simplistes pour argumenter un papier scientifique.
Ce genre de défauts sont généralement présents dans les premiers travaux de ce domaine. Par exemple, les auteurs pensent avoir maximiser le gain sans pour autant décrire ce qu'est réellement le gain. Les chercheurs ont aussi parlé de retourner les services disponibles, alors que le cloud computing est par définition toujours disponible (les auteurs ont fait allusion à la découverte de services).
Jusqu'à l'année 2010, les travaux s'étaient plus placés côté fournisseur à cause de l'actualité scientifique à ce moment là. Mais, des travaux comme Anandasivam & al. [12] et Ke & al. [65] se sont peu à peu penchés vers les besoins des clients et la qualité des services vue par les consommateurs.
Anandasivam & al. [12] se sont spécialement intéressés aux besoins des clients des services cloud computing d'un niveau IaaS. Pour ce faire, les chercheurs ont d'abord classé les consommateurs en 5 catégories qui sont les développeurs, les entreprises, les PME, les scientifiques et les utilisateurs finaux. Après, ils donnent des statistiques de comportement et de préférences selon les catégories fixées précédemment. Pour le comportement, ils présentent les raisons fréquentes d'un changement du fournisseur par classe de consommateurs, ainsi que le temps nécessaire pour prédire les demandes des utilisateurs selon leurs catégories. Pour les préférences, les chercheurs présentent par exemple l'importance des caractéristiques comme le système d'exploitation ou le niveau du support par rapport aux différents groupes de consommateurs. En plus de cela, ils présentant la capacité financière selon les catégories d'utilisateurs fixées au préalable.
Ke & al. [65] présentèrent (ELECTRE) une méthode de décision à choix multiples pour calculer la satisfaction client, concernant les services en nuage, en utilisant une fonction utilité. Pour ce faire, les chercheurs ont d'abord formalisé un SLA pour les services en nuage informatisé. Ils ont ensuite modélisé une fonction utilité, qui différencie entre une solution optimale d'une autre qui l'est moins. Et enfin, les auteurs ont présenté une méthode multi-critères à décisions multiples afin de donner un ordre à la découverte des services. Malgré l'intérêt des scientifiques à une explication rigoureuse, les auteurs n'ont tout de même pas jugé intéressant d'y inclure une évaluation, une comparaison ou un exemple.
Sun & al. [141] proposèrent une méthode à critères multiples, aidant à la sélection des services cloud computing. Pour ce faire, les chercheurs ont d'abord considéré les préférences des clients comme des valeurs qualitatives ou semi-qualitatives. Après cela, ils ont utilisé un processus d'analyse hiérarchique (AHP) afin de transformer les préférences des utilisateurs en poids numériques, pour sélectionner les services en nuage. Les auteurs ont pu évaluer leur approche, mais ils ne se sont pas comparés avec des travaux similaires.

Travaux	Prob.	Méthodes	Objectifs	Critiques
Han & al. [48]	La recommandation d'outils de virtualisation pour un environnement cloud	Calcul de la mesure (S-Rank)	Réduction du temps moyen nécessaire à l'évaluation	Calcul mathématique trop simple, tests pré-sélectionnés
Zeng & al. [162]	Sélection des services cloud computing	Calcul pondéré	Minimiser le coût et maximiser le gain	Aucune évaluation, aucune comparaison
Anandasivam & al. [12]	Extraction des besoins de clients cloud	Data mining	Extraire le prix moyen, la durée utilisation moyenne ... etc	Aucune preuve
Ke & al. [65]	Sélection de services CC	MDCM, fonction utilité	Maximiser l'utilité des services sélectionnés pour des utilisateurs multiples	Aucune évaluation, aucune comparaison et aucun exemple
Sun & al. [141]	Sélection des services CC	AHP, MCDM	Maximiser la sensibilité des valeurs obtenues	Aucune comparaison

TABLE 2.1 – Tableau comparatif des contributions historiques

La table 2.1 montre les différents travaux cités dans la section 2.2, elle regroupe les contributions ayant inspiré la majorité des travaux dans la sélection, recommandation et compositions de services. Par exemple, Anandasivam [12] n'a certes proposé qu'une simple analyse et extraction de données. Mais, il a aussi mis le point sur une direction à prendre, il a donné quels sont les besoins et les préférences des clients, sur lesquels doivent investir les fournisseurs des services cloud computing.
Ce sont certainement des travaux non liés entre eux, mais qui nous ont beaucoup inspirés dans notre travail. Les scénarios de tests pré-sélectionnés [65] nous ont aidés à mieux modéliser les fonctions des applications surveillant les RCSFs.
D'autre part, les besoins extraits du travail de Anandasivam [12] nous ont beaucoup inspirés en prenant en compte les mesures de qualité pour les services cloud computing. Le travail de Sun [141] nous a par exemple aidé dans le calcul de la fonction du score (voir la sous-section 3.4.3).

2.3 État de l'art

Les travaux de survey sont d'excellents points de départ pour tout chercheur débutant dans un domaine scientifique. Dans cette section, nous allons passer en revue les travaux de ce type dans le domaine de sélection des services cloud computing.
Rehman Ur Zia & al. [119] passèrent en revue les méthodes multi-critères à décisions multiples proposées pour une sélection des services cloud computing. Parmi les propo-

sitions, nous trouvons ELECTRE [65], PROMETHEE [25], AHP, min-max, max-min, TOPSIS... etc. Pour faire une comparaison entre les différentes approches, les chercheurs ont proposé un exemple de 13 services. L'évaluation a compris le nombre d'alternatives sélectionnées et le score de l'évaluation. Malgré la présence de comparaison dans un tel papier (ce qui est relativement rare pour un travail d'état de l'art), les chercheurs ne voient pas l'intérêt du temps nécessaire afin d'arriver à de tels résultats (compraison selon le temps d'exécution).
Mandal & al. [91] proposèrent un mécanisme afin de découvrir, évaluer et sélectionner les services Cloud Computing. Pour cela, ils présentèrent une architecture d'abord, et un algorithme ensuite. À la fin, les auteurs évaluèrent leur approche en termes de coût du service, ainsi que le temps nécessaire pour la sélection. C'est vrai que les chercheurs ont comparé leur contribution à celle de Sundareswaran & al. [142], mais ils ont négligé les autres travaux pris en compte dans leur état de l'art. En plus de cela, les auteurs n'ont proposé aucune comparaison en ce qui concerne le coût du service sélectionné.
Récemment, Yang & al. [157] proposèrent une surveillance des travaux de sélection des services cloud computing se basant sur les préférences des clients. Vu que l'objectif initial était la motivation et la sensibilisation, les auteurs ont proposé une analyse des travaux en actualité et une discussion sur les futurs orientations de la communauté scientifique. Contrairement aux autres travaux cités dans cette section, les chercheurs ne proposent aucune comparaison entre les approches discutées.
Pour la suite de l'état de l'art, nous allons voir 2 grandes catégories : ceux qui se positionnent côté fournisseur et ceux qui se positionnent côté client.

2.4 Travaux se positionnant côté fournisseur

Les fournisseurs et les clients ne voient pas le service de la même façon. Les fournisseurs essayent par exemple de minimiser les ressources consommées pour une qualité optimale, tandis que les clients cherchent les meilleurs services au plus bas prix. Dans cette section, nous allons passer en revue les travaux se positionnant côté fournisseur dans le domaine de sélection des services CC. Les autres contributions seront abordées dans la section suivante 2.5.
Yang & al. [158] proposèrent un outil pour découvrir et sélectionner les ressources des nuages informatisés, selon les exigences et besoins de leurs clients. Pour ce faire, les chercheurs ont d'abord présenté le Cloud Service Infrastructure Framework (CSIF) et le Service Ressources Broker (SRB). Après, ils ont décrit l'ensemble du formalisme mathématique, proposé un pseudo-code afin de découvrir, sélectionner, réserver et assigner les ressources des nuages aux utilisateurs. Enfin, ils ont testé leur approche par rapport à une méthode de sélection aléatoire, selon le nombre de requêtes réussies, ainsi que le taux d'utilisation du réseau. Plus précisément, les chercheurs ont réussi à prouver que leurs approche est plus efficace en utilisant moins les supports de communication du nuage. Bien que le travail soit intéressant, les chercheurs n'expliquent pas comment ils jugent une requête réussie (car il n'y a que les clients qui peuvent le faire réellement).
Li & al. ont présenté CloudCMP [74, 76, 75] et CloudProphet [78, 77]. Ils ont comparé entre des fournisseurs publiques de Cloud Computing. Bien qu'ils sélectionnent les services, ils font plus de la prédiction de performance. Les travaux sont bien présentés, expliqués et démontrés, mais la prédiction des performances diffère de la sélection ou

de l'indexation des services.
Zhao & al. présentèrent une plateforme CARE [164] qui relie les services en nuage avec leurs services de stockage de base de données associés comme Microsoft Azure, Google's App Engine et Amazon EC2/Simple DB/S3. Pour ce faire, les chercheurs ont d'abord choisi des scénarios de tests bien spécifiques, afin de mesurer la performance de chaque fournisseur de nuage informatisé. Après, ils ont réussi à sélectionner le meilleur en se basant sur les statistiques de performance précédemment obtenues, sans aucun calcul mathématique. Bien que le travail soit argumenté par un exemple, il n'a été ni comparé, ni évalué. En plus de cela, les chercheurs prennent en compte l'aspect temps de réponse des applications en nuage, sans pour autant prendre en compte la position du fournisseur ainsi que le client dans le réseau.
Bao & al. [14] proposèrent un algorithme pouvant fragmenter un processus, pour optimiser le coût en temps de son exécution. Pour ce faire, les chercheurs divisaient la tâche entrante, en un nombre fixé au préalable de processus. Après cela, ils ont essayé de trouver pour chaque processus atomique, le meilleur emplacement dans le nuage à exécuter. Enfin, les chercheurs ont testé la performance de leur approche, qui se résume à la réduction du coût de l'exécution de la tâche initiale. Malgré la présence d'évaluation et de comparaison, le travail manque de formalisme mathématique et de code. En plus de cela, les auteurs supposent dans leur travail que les emplacements demeurent dans le même centre de données, car dans le cas contraire, le coût de l'exécution aurait augmenté au lieu de diminuer (plus de détails sont présentés dans la sous-section 2.5.1).
Yu [160] proposèrent un outil afin de sélectionner des fournisseurs de cloud computing, selon leurs méthodes de virtualisation. Pour ce faire, les chercheurs ont d'abord proposé un modèle pour la virtualisation dans les nuages informatisés. Après, ils ont supposé l'existence de plusieurs algorithmes de sélection de services comme clients, pour à la fin noter quel fournisseur ont-ils choisi réellement. Les auteurs ont pu démontrer que l'algorithme de sélection propre à eux converge plus rapidement, vers le fournisseur optimal, comparé aux autres algorithmes existants. Malgré la pauvreté en explication mathématique, les auteurs ont proposé un pseudo-code et une comparaison. Mais, les chercheurs se basent sur une notion qui n'est pas forcément publique. De nos jours, les fournisseurs des nuages informatisés ne donnent pas toujours accès aux détails relatifs à leurs outils de virtualisation.
Cavalcante & al. [22] proposèrent une sélection de la composition optimale d'un ensemble de services IaaS. Pour ce faire, les chercheurs ont d'abord spécifié 18 plans d'exécution, qu'ils utiliseront plus tard dans l'évaluation. Ensuite, les auteurs mesuraient le temps nécessaire pour chaque scénario (plan d'exécution), afin de prouver que leur approche est efficace, par rapport à une ancienne proposition. Malgré l'importance de la contribution, les chercheurs n'ont pas pris en considération le coût pour la totalité des services sélectionnés (pour la composition des services en CC, voir la sous-section 2.5.1).
Qian & al. [109] proposèrent le projet CSS qui permet de sélectionner la meilleure infrastructure dans un nuage afin de maximiser l'utilisation des serveurs, centres de données, clusters...etc. Les chercheurs n'ont pas pris comme critère le coût du déploiement seulement, mais aussi l'emplacement du service et ses composants dans le nuage. Selon les tests déroulés, la plateforme proposée reste efficace malgré le grand nombre de composants applicatifs. En plus de cela, le framework expliqué dans l'article [109] reste très bien illustré.

Zaremba & al. [161] proposèrent une plateforme pour lier les besoins des clients aux meilleurs fournisseurs IaaS. Ils ont utilisé les données liées (RDF et SPARQL) afin de faire la projection des caractéristiques des services sur les besoins des clients. Malgré l'effort des auteurs à prouver la performance des outils proposés, les chercheurs n'ont pas donné accès aux graphes de données de liaison (qui est le plus important).
Nizamani & al. [103] proposèrent QaCompPSS, qui a permis de sélectionner le meilleur fournisseur Cloud Computing en termes de qualité. Cependant, les chercheurs ont pris d'un côté en compte les prix et modèles de facturation, et de l'autre côté, le nombre de machines virtuelles et la charge sur la machine physique. Ceci dit, l'objectif principal était de réduire le coût pour les fournisseurs. Malgré l'intérêt porté à la qualité et au prix, les chercheurs n'ont pas jugé nécessaire d'argumenter leur travail avec un formalisme mathématique, ni un code. En plus de cela, l'évaluation et la comparaison de la contribution par rapport aux autres dans le papier, manque cruellement de rigueur et de richesse.
Kang & al. [59] proposèrent une approche pour sélectionner une machine virtuelle dans un environnement en nuage afin de planifier et répartir les tâches des processus utilisateurs. L'objectif était de maximiser la prédiction de la meilleure machine virtuelle ainsi que ses performances, avant de voir le prix ou la qualité, tout en faisant un apprentissage sur un ensemble d'exemples. Malgré l'intérêt de la contribution, un apprentissage nécessite un temps afin d'arriver à un stade utile, ce côté (temps nécessaire à l'apprentissage) a été d'ailleurs négligé de la part des auteurs.
En 2014, Raed & al. [64] intégrèrent la composition de services à leur approche précédente [61]. Ça veut dire qu'ils découpent le processus client en sous-processus atomiques. Et que chaque sous-processus peut être affilié à un service en nuage à part. L'objectif initial était de réduire le temps nécessaire à la composition des services. En plus de cela, le calcul mathématique donnait plus d'importance à leur travail. Mais, leur choix se base sur une ontologie, afin de lier les préférences des applications clientes aux caractéristiques des fournisseurs. D'une part, les chercheurs n'ont pas donné accès à l'ontologie, et d'une autre part, la comparaison est inexistante, et l'évaluation est pauvre.
Ho & al. [52] proposèrent un modèle de facturation dynamique afin de virtualiser les réseaux sans fil. Pour faire cela, il a mis au point une théorie de jeu en se basant sur l'approche Stackelberg. Malgré la présence d'une explication mathématique et d'une évaluation. Cette approche s'est positionné du côté fournisseur, puisque les auteurs ont préféré maximiser le revenu des administrateurs de centre de données.
Bermbach & al. [18] proposèrent une étude comparative avec des expérimentations sur les outils stockage en nuage. Malgré les explications et détails présentés dans le papier, les chercheurs se sont basé sur des mesures jugées à notre goût trop matérielles. Un fournisseur de nuage publique peut généralement changer son matériel sans consulter l'avis, ni l'opinion de ces clients.

Travaux	Prob.	Méthodes	Objectifs	Critiques
Yang & al. [158]	Découverte et sélection des ressources dans un environnement en nuage	CSIF, SRB	Réduire l'utilisation du réseau et maximiser le taux de requêtes satisfaites	Les chercheurs n'expliquent pas comment une requête est jugée satisfaite
Li & al. [74, 76, 75, 78, 77]	Prédiction de performance des services en nuage	Tests réels selon des scénarios pré-sélectionnés	Maximiser la pertinence de la prédiction	Prédire la performance n'est pas toujours liée à la sélection des services
Zhao & al. [164]	Sélection de services de bases de données en cloud pour des application déjà en nuage	Tests réels selon des scénarios relatifs aux traitement de bases de données	Maximiser la précision dans la prédiction des meilleurs applications CC	Déduction sans calcul mathématique, ni traitement algorithmique. Prise en compte du temps de réponse des applications en négligeant la position dans le réseau.
Bao & al. [14]	Sélection d'emplacements optimals pour l'exécution de sous-processus	Fragmenter le processus entrant et exécuter les sous-processus séparément	Réduire le coût et le temps d'exécution	Manque de formalisme mathématique
Yu & al. [160]	Sélection du meilleur fournisseur CC selon la méthode de virtualisation	Gestion de réputation	Réduire le temps de convergence à la solution idéale	Les fournisseurs de CC ne donnent généralement pas leurs méthodes de virtualisation

Cavalcante & al. [22]	Sélection de la composition optimale d'un ensemble de services CC	Mesurer les performances des services atomiques selon 18 scénarios pré-définis	Réduction du temps nécessaire à la recherche	Ignorance par rapport au coût de la totalité des services composés
Qian & al. [109]	Classification et sélection de la meilleure infrastructure	Résultats de tests de performance des serveurs	Maximiser la scalabilité	Scénario de tests pré-sélectionnés
Zaremba & al. [161]	Sélectionner les services IaaS selon les besoins des clients	RDF et SPARQL pour calculer la similarité	Maximiser le nombre d'offres satisfaisantes et la scalabilité	Ontologie inaccessible
Nizamani & al. [103]	Sélection du fournisseur CC selon la qualité	Ontologie, OWL 2	Réduire le coût pour les fournisseurs	La comparaison manque cruellement de rigueur. Aucun code, ni formalisme mathématique
Kang & al. [59]	Sélection de machine virtuelle dans un environnement cloud	Un apprentissage à partir des exemples passés	Améliorer la prédiction de la meilleure machine virtuelle	L'apprentissage nécessite un temps de convergence vers une solution plus optimale
Raed & al. [64]	Composition de service et sélection de services IaaS pour des SaaS	Découpage de processus et répartition des traitements dans le nuage	Réduire le temps nécessaire à la composition	Aucune comparaison, aucun accès à l'ontologie
Ho & al. [52]	Modèle de facturation dynamique pour virtualiser des RCSFs dans un nuage	Théorie de jeu basée sur Stackelberg	Maximiser le revenu du fournisseur	Position côté fournisseur

Bermbach & al. [18]	Outil de comparaison entre fournisseurs de services de stockage CC	Dérouler plusieurs tests en même temps sur les nuages de stockage	Évaluer la performance des services de stockage en nuage	Négligence du coût des services, ainsi que le temps nécessaire à l'outil pour sélectionner

TABLE 2.2: Tableau comparatif des contributions se positionnant côté fournisseur

La table 2.2 nous présente les tendances côté fournisseur de la sélection des services CC. Ça nous a permis d'avoir une idée claire et précise sur les paramètres d'évaluation particulièrement comme le temps nécessaire à la recherche des services [22], les techniques utilisées comme La théorie des jeux basée sur Stackelberg [52]. Cependant, chacun présente un point faible comme l'utilisation d'une ontologie inaccessible [161] ou le manque de comparaison.
Positionner un travail du côté fournisseur dans le cloud computing comporte des contraintes en plus. Puisque les ressources d'un nuage sont toujours accessibles, facilement extensibles et ne nécessitant aucune intervention de la part du fournisseur, les services cloud computing semblent pour leurs clients infinis (illimités). Si nous nous positionnons côté fournisseur, la première contrainte à prendre en compte est la limite des ressources matérielles. Même si les services semblent illimités, les ressources sur lesquelles s'exécutent ces derniers ne le sont pas. Ce n'est qu'une différence parmi tant d'autres entre la position côté client et celle du fournisseur. Et c'est pour cette raison principalement qu'on a opté pour les clients en premier lieu.

2.5 Travaux se positionnant côté client

Vu l'intérêt que porte la communauté scientifique aux besoins des consommateurs dans les nuages informatisés, le concept des préférences et besoins des clients a petit à petit pris de l'importance dans les travaux sélectionnant des services cloud computing. Dans cette section, nous allons passer en revue les travaux se positionnant côté client dans le domaine de composition, recommandation ou sélection des services CC... etc.

2.5.1 Composition des services en nuage

La composition est un cas plus général de la sélection des services. Un processus client peut par exemple être trop complexe, le diviser est donc une idée intéressante afin d'arriver à le cerner. Dans cette sous-section, nous nous intéressons aux travaux sur la composition des services cloud computing.
Jalloh & al. [56] proposèrent une méthode afin de sélectionner la composition optimale de services cloud computing en se basant sur le calcul matriciel. Pour ce faire, les chercheurs ont d'abord pris en compte des services fonctionnellement équivalents. À l'aide des méthodes décisionnelles à choix multiples, les auteurs ont ordonné les préférences, pour leur sélection. À la fin, la composition sélectionnée était la plus proche de la

solution idéale (généralement virtuelle). Même si les chercheurs ont parlé d'évaluation dans leur papier, nous n'en trouvons aucune, mais seulement un cas d'étude. Ceci dit, nous aurions aimé avoir une évaluation sur la précision ou le temps d'exécution.
Xu & al. proposèrent un projet intitulé S-ABC [155] sélectionnant et composant les services selon leur domaine. Même si les chercheurs ont proposé une formalisation mathématique, un code, une évaluation et une comparaison, les approches nécessitent un apprentissage et un temps d'adaptation afin d'arriver à un déroulement optimal.

Travaux	Méthodes	Objectifs	Critiques
Jalloh & al. [56]	Calcul matriciel	Minimiser la distance entre la solution idéale (virtuelle) et la solution choisie	Formalisme mathématique pauvre, aucun algorithme, aucun programme, aucune évaluation et aucune comparaison.
Xu & al. [155]	Algorithme de colonies d'abeilles artificiel	Réduire le temps nécessaire à l'apprentissage	Nécessité d'un temps pour arriver à un comportement optimal

TABLE 2.3 – Tableau récapitulatif des travaux sur la composition des services CC

La table 2.3 présente les contributions orientées vers la composition des services ou la sélection des compositions optimales de services en nuage. Nous ne pouvons pas comparer un travail de composition (même s'il sélectionne des services aussi [21, 79]) avec une approche comme la notre pour la simple raison que diviser un service en deux dans un nuage peut générer des coûts supplémentaires. Si par exemple, nous divisons un service complexe A en deux services atomiques B et C, alors nous devons payer pour les deux services séparément, la communication entre eux et la communication entre les clients et les deux services B et C. Puisque nous avons inclus le prix et le reste du budget comme métrique de qualité. Nous ne pouvons pas par conséquent se permettre de diviser un service client en plusieurs parties. D'ailleurs, Deal & al. [33] et Barlaskar & al. [15] étudient très bien la question d'optimisation du transfert d'un service cloud d'un fournisseur X à un autre fournisseur différent Y.

2.5.2 Recommandation des services cloud computing

Le nombre de services en nuage à sélectionner peut devenir important. Dans un tel cas de figure, une recommandation peut s'avérer utile afin d'économiser du temps. Elle se résume en la réduction de l'ensemble des services cloud computing de départ pour minimiser la durée du processus de sélection. Dans cette sous-section, nous nous intéressons aux travaux de recommandation de services en nuage.
Kong & al. [68] proposèrent une plateforme de recommandation des services en nuage, selon la confiance. Pour cela, les chercheurs présentèrent d'abord l'architecture de leur système (Trust-based Recommandation System in service-oriented Cloud computing (TRSC)), qui mesure à quel point nous pouvons avoir confiance aux services en nuage. Ceci dit, la confiance est exprimée avec deux valeurs différentes, la première représente la confiance directe, et la seconde celle de la confiance recommandée. Plus précisément,

la confiance directe est calculée en se basant sur l'interaction d'un utilisateur A avec le service cloud computing C. Tandis que la confiance recommandée se base sur l'interaction des utilisateurs qui sont déclarés comme confiants par l'utilisateur A concernant le service cloud computing C. L'objectif d'un tel projet était d'optimiser la précision de la recommandation. Malgré l'intérêt des auteurs porté sur l'explication concise et la comparaison adéquate, les chercheurs n'expliquent pas comment ils jugent une approche pertinente, et ils se contentent d'exposer les valeurs de recommandation pour chaque service dans un exemple.

Bedi & al. [17] proposèrent un système sélectionnant les services CC les plus confiants, en utilisant la logique floue. Le système en question est coopératif entre utilisateurs, se basant sur les recommandations faites par les autres consommateurs de confiance. Afin de gérer l'incertitude des données relatives aux recommandations des utilisateurs, les chercheurs ont proposé un Fuzzy Inference System (FIS). Puisque l'objectif était de prouver la faisabilité de l'approche d'une part, et confirmer que l'estimation de la confiance des services peut aider les utilisateurs à mieux les sélectionner pour préserver leurs données, d'une autre part. Les auteurs ont fait des tests selon le taux de tâches réalisées avec succès par rapport à la charge de cette dernière. Le travail est intéressant, car les auteurs ont pu expliquer rigoureusement et évaluer spécifiquement leur approche. Mais, les chercheurs n'ont pas jugés nécessaire de comparer leur travail par rapport à d'autres approches similaires.

Redl & al. [118] proposèrent une méthode automatique, afin de relier les contrats de services des clients aux services CC les plus adaptés. Malgré l'hétérogénéité des besoins, préférences et requis des clients CC, les chercheurs ont pu trouver les ressemblances sémantiques entre les différents contrats d'utilisateurs, grâce aux algorithmes d'apprentissage. De ce fait, les auteurs ont pu sélectionner les services CC les plus adaptés pour leurs clients. Vu que les chercheurs voulaient choisir le service optimal en utilisant l'apprentissage, ils ont comparé leur approche avec une plus basique et une autre sans recommandation. Le résultat prouvait que le travail proposé pouvait réduire le coût des services sélectionnés d'une part, et pouvait le faire encore plus si le nombre d'exemple d'apprentissage venait à augmenter. C'est un papier fait dans les règles de l'art, à une exception, qui est de ne pas juger l'optimalité des services par leur coût uniquement. Par exemple, un utilisateur peut s'intéresser à la qualité plus que le prix ou le temps de recherche.

Rabbani & al. [115] proposèrent une méthode de sélection des services cloud computing, en faisant appel à des agents intelligents. Les chercheurs ont remarqué que si nous entraînions les agents suffisamment, ils pouvaient à la fin remarquer les similarités et différences entre les services pour pouvoir les sélectionner. Vu que l'objectif était de répondre au mieux aux requis des clients, il serait intéressant de voir l'évaluation d'une telle approche selon le temps d'apprentissage, la pertinence des résultats... etc. Mais, les auteurs n'ont intégré ni évaluation, ni comparaison à leur travail.

Lin & al. [82] proposèrent PCSS qui sélectionne les meilleurs fournisseurs en termes de capacité de protection de la vie privée. Pour ce faire, les chercheurs ont construit d'abord une hiérarchie à 2 niveaux, afin de quantifier la protection de la vie privée. Après, les auteurs ont utilisé la logique floue, afin de calculer la valeur déterminant la protection de la vie privée du service CC. Ensuite, les chercheurs ont réussi à classer les attributs dans la hiérarchie afin de recommander et trier les services pour les utilisateurs. Puisque l'objectif était de réduire le temps nécessaire à la recommandation et la sélection, les auteurs ont proposé des tests montrant une réduction en temps

d'exécution par rapport à une méthode qu'ils ont appelé Baseline. Ceci dit, ça aurait été intéressant de comparer l'approche proposée avec une autre du même type, et ne pas se limiter à une méthode trop gourmande en temps d'exécution et en espace.
Soltani & al. proposèrent QuARAM Recommander [135] permettant de recommander les services CC IaaS en se basant sur l'historique des actions et situations précédemment rencontrées. Puisque l'objectif était de prouver que le raisonnement par cas serait une bonne méthode, une évaluation a été proposée, montrant les statistiques obtenues. On peut dire par exemple, que l'outil atteint un taux de succès supérieur à 50%. Mais, ça aurait été intéressant de comparer l'approche proposée avec d'autres procédés dans la recommandation des services CC utilisant un autre type d'apprentissage.
Jiang & al. [58] proposèrent une méthode pour recommander les services Cloud Computing. Pour cela, les chercheurs ont en premier lieu présenté le score à domination partielle. Après, ils ont combiné l'opérateur proposé avec le calcul à l'horizon (Skyline Computation), afin de redéfinir l'objectif de l'optimisation de la recommandation des services. Enfin, ils ont proposé un algorithme pouvant rechercher efficacement et retourner les K meilleurs services ayant les qualités optimales. Malgré la présence d'un pseudo-code, les chercheurs n'ont pas présenté d'évaluation, ni de comparaison de leur contribution avec d'autres approches similaires.
Jayapriya & al. [57] proposèrent une approche afin de recommander les services cloud computing, selon leur qualité. Pour ce faire, les chercheurs ont d'abord prédit la qualité de service, puis évalué et choisi les services les plus prometteurs. Le principe était simple : si un utilisateur (client dit actif dans l'approche) lance une requête, l'algorithme va d'abord restaurer les anciens utilisateurs similaires, selon les critères de qualité choisis par le consommateur. Après, il présente au client actif la liste des services préalablement choisis par les autres. C'est une approche intéressante pour une problématique prometteuse. D'un autre côté, le travail a été bien expliqué, rigoureusement argumenté, évalué et comparé avec d'autres approches similaires. Ceci dit, le travail nécessite un temps, des ressources et des efforts de la part des utilisateurs afin de se préparer. Effectivement, le système ne pourra jamais retrouver les utilisateurs similaires au client actif, s'il n'a aucune connaissance des utilisateurs actuels.

Travaux	Méthodes	Objectifs	Critiques
Kong & al. [68]	Confiance directe et confiance recommandée	Une meilleure précision dans la recommandation	Une prise en compte des actions passées des clients nécessite un temps pour que le système proposé converge
Bedi & al. [17]	Logique floue	Maximiser le nombre de résultats confiants	Les auteurs s'intéressent à la quantité, sans pour autant s'intéresser à la qualité, aucune comparaison
Redl & al. [118]	Selon le SLA des clients	Réduction du coût des services recommandés	Nécessité d'un apprentissage, comparaison selon le coût uniquement
Rabbani & al. [115]	Un apprentissage sur le feedback des clients	Réduire le temps pour la sélection avec l'apprentissage	Nécessité d'un apprentissage, aucune évaluation, aucune comparaison
Lin & al. [82]	Calcul matriciel, logique floue	Réduire le temps de la recommandation	Une comparaison par rapport à la méthode gourmande, selon le temps d'exécution
Soltani & al. [135]	Raisonnement par cas	Maximiser le taux de résultats à succès	Absence de comparaison
Jiang & al. [58]	Calcul à l'horizon	Maximiser la qualité des services sélectionnés	Aucune évaluation et aucune comparaison.
Jayapriya & al. [57]	Similarités entre besoins et préférences des utilisateurs	Maximiser la précision de l'évaluation des services et la prédiction des performances	Nécessité d'un apprentissage et d'une prédiction des performances

TABLE 2.4 – Tableau récapitulatif des travaux de recommandation de services CC

La table 2.4 présente les contributions orientées vers la recommandation des services cloud computing. Ceci dit, recommander des services exigent un traitement en continu dans le temps, contrairement à la sélection qui se déclenche uniquement par le client. Les travaux cités dans cette sous-section suivent tous le même schéma : des utilisateurs font des choix et ils apprennent de leurs choix pour mieux recommander les services ; ou des utilisateurs recommandent eux même des services à d'autres clients. Ce genre de procédés exigent un temps pour atteindre un état stable [1]. La table met le point sur un ensemble de techniques utilisées dont le travail de [68] qui semble très pertinente, nous remarquons aussi que la majorité des travaux ont pour objectif de réduire le temps.
Nous ne pouvons pas nous comparer par conséquent à ce genre de travaux, puisque notre approche ne prend pas en compte les choix des clients et ne recommande pas les services ; elle ne fait que la sélection en indexant les instances et en les recherchant à la demande du client.
Ceci dit, la recommandation n'est pas inintéressante pour autant. Prendre en compte les recommandations des utilisateurs, leurs choix et l'historiques des qualité peut réduire grandement le temps nécessaire à la sélection : en réduisant la taille de l'ensemble initial des services à sélectionner. Pour cela, on espère étudier ces techniques et travaux dans un travail futur et dédié à cela.

2.5.3 Évaluation des services en nuage

Beaucoup de travaux choisissent de sélectionner des services cloud computing tout en les évaluant : donner un score par exemple aux services faciliterai leur sélection. Dans cette sous-section,, nous nous intéressant aux travaux proposant une évaluation pour les services en nuage, sans pour autant proposer un processus pour la sélection.
Garg & al. [42] modélisèrent les préférences des utilisateurs sous une forme hiérarchique (Ce qu'on appelle communément Service Measurement Index) ainsi que les capacités des services nuage. Ils ont ensuite proposé un algorithme de type Analytical Hierarchical Process afin de mesurer la qualité des services, et leur donner une priorité. Le travail est intéressant, vu que les auteurs expliquent mathématiquement, décrivent formellement, et proposent même un cas d'étude (exemple). Mais, les chercheurs n'ont pas jugé intéressant d'évaluer leur approche, ou la comparer à d'autres travaux similaires.
Chhetri & al. [27] proposèrent un framework pour les clients Cloud afin de sélectionner les meilleurs services IaaS. Les objectifs étaient de minimiser le prix des instances, maximiser la disponibilité et réduire le temps de réponse. Mais prendre en compte des métriques de réseau comme mesure de qualité, exige qu'on prenne en compte la position du client, comme celle du fournisseur dans le réseau. Ceci dit, les auteurs n'ont pas pris en compte les positions des centres de données, et ont négligé aussi les localisations des clients.
Zhang & al. [163] proposèrent un outil pour surveiller et évaluer les services de Cloud computing, selon la qualité de l'expérience. Pour ce faire, les chercheurs ont pris en compte l'évaluation personnelle de l'utilisateur, et grâce à un modèle de prédiction basé sur les chaînes de Markov, ils ont pu évaluer les services CC pour une sélection plus facile. Mis à part le fait que les chercheurs ont pris en compte les évaluations personnelles des clients (pouvant être incertaines), le papier ne présente ni algorithme,

1. Un état dans lequel un système donne des résultats fiables à la majorité des entrées.

ni pseudo-code, ni programme, ni évaluation, et ni comparaison avec d'autres travaux similaires.

Qu & al. [110] proposèrent une approche évaluant les services CC, selon les exigence incertaines de leurs utilisateurs, ainsi que leurs performances dynamiques. Pour ce faire, les chercheurs ont proposé une approche bien à eux, afin de calculer le degré de satisfaction des clients, selon les demandes des consommateurs, et en se basant sur les performances passées des produits CC. Les auteurs ont employé la logique floue, afin de capturer les préférences et requis des utilisateurs. En se basant sur les performances passées des services, les chercheurs ont pu sélectionner les services qui vérifient le plus les requis et les préférences des consommateurs. Vu que l'objectif était l'efficacité, les tests du papier ont démontré une complexité linéaire par rapport au nombre de services. Mais les auteurs n'ont pas vu l'intérêt de comparer leur approche à d'autres travaux similaires.

Chahal & al. [23] proposèrent une méthode évaluant les services cloud computing pour aider les clients CC à la sélection. Pour faire cela, les chercheurs ont d'abord proposé un modèle hiérarchique afin de modéliser les besoins et exigences des clients CC. Ils ont ensuite présenté un algorithme AHP qui donne un score à chaque service dans le lot. Malgré la rigueur dans l'explication de la contribution, ainsi que l'exemple déroulé dans le papier, les auteurs n'ont pas vu l'intérêt d'une évaluation, ainsi qu'une comparaison. En plus de cela, les chercheurs ont considéré les caractéristiques des services CC comme étant des attributs qualitatifs ou semi-quantitatifs.

Somu & al. [136] proposèrent un modèle de calcul pour un score aux services cloud computing. Les chercheurs ont d'abord rappelé les notions d'hyper-graphes, d'hyper-lien... etc. Ensuite, les auteurs ont utilisé la distance euclidienne, ainsi que la distance de bloque pour représenter la similarité entre les différents services. À la fin, les résultats choisis sont ceux dont les nœuds sont liés par un hyper-lien. Malgré la proposition formelle, d'une modélisation mathématique, d'un cas d'étude, ainsi que d'un pseudo-code, l'approche proposée [136] reste non comparée et non évaluée.

Balasubramanian & al. [13] proposèrent un schéma pour évaluer la sécurité des données pour les services cloud computing, en se basant sur la logique floue. Les chercheurs ont supposé dans leur scénario, qu'après génération de valeur évaluant la sécurité d'un service donné, ils cryptaient les données pour un transfert et les stockaient dans un serveur cloud. Mais, les chercheurs n'ont pas défini clairement les paramètres initialement pris en compte afin de calculer la valeur décrivant le taux de sécurité du fournisseur. En d'autres termes, la contribution se résume au schéma seulement, sans vraiment parler d'algorithme.

Jagli & al. [54] proposèrent une méthode afin d'évaluer la capacité qu'un service SaaS soit personnalisable. Puisque l'objectif de l'approche était différent de ses travaux similaires, les auteurs étaient obligés de proposer un modèle de qualité pour les services cloud computing de niveau SaaS. Même si l'algorithme proposé était simple, bien expliqué et facilement compréhensible, les métriques utilisées étaient de nature semi-quantitative. En plus de cela, les chercheurs n'ont proposé ni des évaluations, ni des comparaisons.

Travaux	Méthodes	Objectifs	Critiques
Garg & al. [42]	AHP, SMI	Maximiser la qualité des services sélectionnés	Aucune évaluation, aucune comparaison
Chhetri & al. [27]	Maximiser la précision de l'évaluation	Réduire le coût des instances, maximiser la disponibilité des services et réduire le temps de réponse	Négligence de la position des 2 points dans le réseau
Zhang & al. [163]	Tests répétitifs et permanents des services CC	Améliorer la précision du choix	Inutile si le fournisseur décide de changer de matériel
Qu & al. [110]	Calcul mathématique	Prouver la faisabilité de leur approche	Calcul mathématique trop simpliste et aucune comparaison.
Chahal & al. [23]	Calcul matriciel, AHP	Sélectionner les services conformes aux besoins des clients	Aucune évaluation, aucune comparaison, aucune métrique quantitative
Somu & al. [136]	Modèle de calcul basé sur les hypergraphes	Maximiser la pertinence des résultats	Aucune évaluation et aucune comparaison.
Balasubrmanian & al. [13]	La logique floue	Maximiser la confiance en les services sélectionnés	Manque de clarté dans le choix des paramètres pour calculer la réputation du fournisseur
Jagli & al. [54]	Arbres de décision	Maximiser la capacité qu'un service a à être personnalisable des services sélectionnés	Aucune évaluation et aucune comparaison.

TABLE 2.5 – Tableau récapitulatif des travaux sur l'évaluation des services CC

La table 2.5 présente les contributions orientées vers l'évaluation des services en nuage sans pour autant proposer un processus pour la sélection (ni pour la composition ou la recommandation d'ailleurs). Nous remarquons que les techniques d'évaluation les plus utilisées sont la logique floue, AHP. Nous remarquons aussi que les objectifs les plus utilisés sont la qualité de service et la disponibilité.
Les travaux exposés restent des approches intéressantes, mais ils sont difficilement comparables à des contributions portant sur la sélection aussi. Notre approche est un exemple d'une contribution indexant, recherchant et sélectionnant des services en nuage. Nous évaluons les instances à un moment donné, mais les performances et les résultats seraient différents si nous isolons l'évaluation du reste du système.

2.5.4 Sélection des services cloud computing

Tout comme l'approche proposée, certains travaux sont orientés vers la sélection. Dans cette section, nous nous intéressons aux contributions de ce genre, en exposant leurs objectifs, leurs méthodologies et nos critiques.
Ruiz-Alvares & al. [120] proposèrent une méthode automatique, afin de sélectionner le service de stockage en nuage le plus adapté à une base de données d'une application. Pour cela, les chercheurs ont d'abord calculé les performances et les limites des services un par un. Ils ont ensuite relié chaque base de données à son fournisseur de stockage en nuage le plus adapté, selon les capacités du service. Les auteurs ont expliqué les mesures de qualité prises en compte, mais n'ont pas jugé nécessaire d'expliquer comment procéder à un tel calcul. En plus de cela, les chercheurs ont évalué leur approche en temps, mais n'ont pas comparé pas leur travail avec d'autres approches similaires.
Après le travail d'état de l'art, Rehman Ur Zia & al. [148] proposèrent une approche afin de sélectionner des services Cloud IaaS, et en se basant sur l'historique de la qualité de service. Pour faire cela, les chercheurs ont évalué chaque service sur différents moments. Selon les préférences des clients, les auteurs ont pu regrouper à la fin les mesures prises comme historiques, afin de calculer le score global de chaque service à sélectionner. Malgré la rigueur démontrée dans l'explication de la méthodologie (TOPSIS) et l'objectif (sélectionner le service répondant au plus aux préférences du client), les auteurs n'ont cependant pas jugé nécessaire d'évaluer leur approche, ni de la comparer avec d'autres travaux similaires.
Patiniotakis & al. [105] proposèrent une façon des plus originales pour sélectionner des services en nuage. Avant de donner un score, ils calculèrent l'importance d'un service par rapport à d'autres en utilisant les méthodes de décision à choix multiples, ainsi que la logique floue. Après avoir donné un score à chaque service dans la matrice des fournisseurs, la décision alla au nuage avec le score le plus important. Ce travail peut gérer des attributs ou des caractéristiques imprécis (comme : haut, bas et moyen) en plus des caractéristiques numériques, mais les auteurs n'ont inclus ni évaluation, ni comparaison pour leur approche.
Sahri & al. proposèrent le projet DBaaS-Expert [121] permettant de sélectionner le meilleur fournisseur Cloud Computing de service de base de données. Pour ce faire, les chercheurs avaient besoin de deux outils qui sont l'ontologie permettant de modéliser le mécanisme de gestion des données des fournisseurs, ainsi que l'évaluateur qui donne un score aux services CC selon des métriques bien spécifiques. Pour cela, les chercheurs ont proposé une liste de dimensions, ainsi que l'ontologie, après avoir fait la liste des fournisseurs disponibles. Ensuite, les auteurs ont évalué les services selon les

dimensions prises en compte, et les exigences des utilisateurs. Même si les chercheurs présentèrent un formalisme mathématique, ils se sont justifié l'absence de comparaison avec d'autres approches, par le manque de ces dernières. Ceci dit, les auteurs auraient pu au moins dérouler leur cas d'étude avec plus de rigueur.
Yongwen & al. [83] étudièrent la possibilité d'intégrer la Rough Set Theory dans la sélection des services CC. Vu que la théorie des ensembles approximatifs se base sur des notions comme *objets*, *attributs* et *valeurs*. Alors, les chercheurs ont considéré les services CC comme des objets, les paramètres cruciaux au choix comme des attributs, et les données collectées comme des valeurs d'attributs. À la fin, l'étude a pu démontré par la preuve mathématique et un exemple, qu'il est effectivement possible d'intégrer la théorie des ensembles approximatifs dans la sélection des services CC. D'une autre part, les chercheurs ont abouti à une restriction de la liste initialement prise en compte des services CC. Si par exemple, nous prenons en considération une liste de 100 services CC, le résultat serait une restriction comme 50 services. En d'autres termes, nous ne faisons pas une sélection, ni une évaluation d'ailleurs. Mais, nous donnons plutôt au service une priorité (service haut, moyen ou bas). Ceci aura comme impact la réduction de la précision de la sélection, car nous ne pouvons trier ou classer les services de la même priorité.
Le & al. [72] proposèrent une approche de décision à critères multiples, basée sur la logique floue, pour sélectionner les meilleurs services Cloud Computing. Pour ce faire, les chercheurs ont proposé une approche basée à la fois sur Interpretive Structure Modeling (ISM), ainsi que sur Analytic Network Process (ANP). Le premier a servi à la modélisation des relations interactives entre les critères d'évaluation, alors que le second a servi à la gestion des données incertaines. Vu que l'objectif était de maximiser les résultats utiles, un résultat optimal a été perçu comme un service maximisant la fonction utilité. Ceci dit, les auteurs ont négligé le temps nécessaire afin d'arriver à de tels résultats.
Raed & al.[62] proposèrent une méthode pour calculer la qualité de service point à point, dans le cas d'une collaboration entre plusieurs services. Pour ce faire, les chercheurs ont d'abord modélisé l'architecture multi-couches du cloud computing. Elle représente les différents niveaux de l'informatique en nuage (IaaS, PaaS et SaaS), ainsi que les moyens dont les services disposent afin de collaborer avec d'autres de niveaux plus bas. Les auteurs ont ensuite utilisé les invocations passées des clients, afin de calculer la qualité de services. Puisque l'objectif du papier était de proposer une approche précise en termes de prédiction de performances, les chercheurs ont alors testé la précision de la prédiction, par rapport à d'autres travaux similaires. Mais, les approches étendues par les chercheurs sont toutes de nature mono-couche. Ceci veut dire, qu'elles ne prennent pas en compte la collaboration entre différents services de différents niveaux du cloud computing. En d'autres termes, les auteurs ont comparé l'architecture multi-couches aux différentes architectures mono-couche existantes. Les chercheurs ont dans ce cas ignoré la méthode calculant la qualité dans les tests de performances.
Souidi & al. [138] par contre virent le même contexte du choix du meilleur fournisseur de Cloud Computing d'un autre point de vue. Avant de choisir, et avant même de collecter les caractéristiques des services, ils ont testé les performances du nuage sur une période étendue. Selon le moment dans lequel le client final veut choisir, l'outil proposé prend en compte les variations du score de chaque service afin d'obtenir le meilleur dans les quelques secondes qui vont suivre. C'est une approche centrée sur

le *temps réel*, qui perd automatiquement toute crédibilité si l'utilisation du service va durer longtemps. Nous supposons qu'un client lance l'algorithme proposé [138] à un instant t, et que l'outil retourne un service A. Plus le temps de l'utilisation sera important, plus il serait probable que le résultat à l'instant t', serait différent. Dans ce cas, nous pouvons dire que le framework proposé [138] soit adapté à des utilisations réduites en termes de durée.
Omerovic [104] proposa une approche aidant les clients CC à mieux sélectionner leurs services. L'objectif était de maximiser le bénéfice, tout en réduisant le risque pour les consommateurs. Le chercheur a proposé une analyse, ainsi qu'un ensemble de règles bien simples permettant de sélectionner les services. Malgré l'originalité de l'approche, la contribution demeure à notre goût trop simpliste. En plus de cela, l'auteur n'a pas vu l'intérêt d'une évaluation, ainsi qu'une comparaison. Vu qu'il n'a pas mentionné la simplicité comme point fort à l'approche, nous nous attendions à une comparaison en termes de complexité en mémoire par exemple.
Gad ElRab & al. [40] proposèrent une solution adaptée à des besoins dynamiques des utilisateurs mobiles afin de sélectionner le meilleur serveur Cloud Computing pour transférer leurs données. Le problème dans un tel cas de figure est la limite en batterie des clients ainsi que la bande passante. Du coup, les chercheurs ont eu l'idée de comparer leur travail aux autres approches similaires, en termes de coût économique (prix total), délai d'attente, et énergie consommée durant le transfert. Malgré la performance démontrée dans les tests établis, la contribution reste trop reliée à son contexte, et ne peut être facilement adaptée à un autre, comme l'intégration des réseaux de capteurs sans fil.
Lee & al. [73] présentèrent des méthodes permettant la sélection de services en se basant sur la méthode AHP floue, une stratégie d'évaluation balancée et la méthode de communication Delphi floue. Leur but était de sélectionner les meilleurs services IaaS pour des entreprises. La stratégie d'évaluation balancée a été utilisée pour donner un modèle pour les facteurs de décision (finance, client, apprentissage, extensibilité... etc). La méthode Delphi floue a été choisie pour sélectionner les facteurs les plus importants selon l'opinion des décideurs pour chaque perspective. La méthode AHP floue a été déroulée pour comparer les critères de décision et sélectionner les plus importants. Pour un travail aussi compliqué, nous trouvons qu'il n'est pas assez expliqué. Pour une approche intégrant au minimum 3 algorithmes différents, nous pensons que la présence d'un pseudo-code, une évaluation et une comparaison expérimentale sont requises.
Sun & al. proposèrent le projet de Cloud-Fuser [140]. Dans cette approche, les chercheurs ont proposé une ontologie afin de gérer les similitudes entre les concepts en général dans une base de données et les ont liés aux notions de services cloud computing. Ensuite, ils ont présenté une analyse hiérarchique pour calculer la similarité sémantique entre les concepts. Enfin, ils ont proposé une méthode de décision à choix multiples pour sélectionner le meilleur service Cloud computing. Mais, les auteurs n'ont pas donné accès à l'ontologie, qui est la clé même du travail.
Do & al. [35] proposèrent une approche collaborative entre utilisateurs afin de sélectionner les meilleurs fournisseurs Cloud Computing. Premièrement, les chercheurs classaient les utilisateurs en groupes (aléatoirement), sachant que les clients du même groupe peuvent communiquer les prix des services entre eux, alors que les autres ne le peuvent pas (de différents groupes). Les utilisateurs prenaient ensuite un fournisseur au hasard, et ils communiquaient les prix de leurs services. Chaque client ayant un service plus cher se voyait automatiquement changer de fournisseur. Après un certain temps

d'adaptation du système global, les clients finissaient par trouver le meilleur fournisseur possible. Ce travail est très intéressant, et innovateur, mais il nécessite un temps d'adaptation, qui n'est pas négligeable, pour que les clients aboutissent au service voulu.

Ma & al. [87] proposèrent une approche en se basant sur les intervalles à ensemble neutrosophique, pour sélectionner le meilleur service de Cloud Computing, en termes de confiance. Dans cette approche [87], les chercheurs essayaient de trouver un équilibre entre prise de risques, qualité et coût. Pour ce faire, les auteurs ont établi leurs opérateurs et règles de calcul, afin de formuler d'un côté leur problématique sous forme de problème MCDM, et d'un autre côté prouver mathématiquement leur approche. Mais malgré, l'importance donnée par les chercheurs au formalisme mathématique, les auteurs n'ont pas proposé d'évaluation ou de comparaison. D'un autre côté, l'algorithme proposé [87] résulte des valeurs semi-quantitatives, ce qui peut altérer la précision du processus.

Lianyong & al. [108] proposèrent une méthode afin d'optimiser le coût d'un service cloud computing pour ses utilisateurs. Pour cela, les chercheurs ont enquêté d'abord sur les facteurs influençant sur les prix effectifs des services en nuage (phase 1). Après, ils ont proposé leurs méthodes en se basant sur les paramètres trouvés dans la phase 1 d'une part, et la notion du temps comme facteur d'autre part. À la fin, Lianyong & al. [108] testent leur approche par rapport à d'autres travaux similaires, et prouvent en fin de compte que leur contribution est supérieure en termes de prédiction de coût de déploiement par rapport à l'horaire d'utilisation et à la taille de la tâche du client. On cite ce travail parce qu'il fait partie des rares contributions dans notre état de l'art à traiter les modèles de facturation des services en nuage tout en se positionnant du côté du client cloud, et la prise en compte de la taille de la tâche cliente. Les chercheurs considèrent l'horaire dans la journée à laquelle le client lance sa requête dans leur framework, ce qui est important, mais seulement pour les tâches de courte durée. Si la tâche perdure, le service optimale à choisir risque de changer, puisque l'horaire lui-même a changé.

Ruadulescu & al. [116] proposèrent une méthode étendue de TOPSIS afin de sélectionner des fournisseurs cloud computing. L'extension en question se résuma par l'utilisation de la distance de Minkowski. Après la proposition d'un index de mesures pour les services, les chercheurs ont présenté un cas d'étude et une évaluation. Mais la comparaison s'est limitée à l'approche TOPSIS d'origine. En d'autres termes, les chercheurs ont pu prouver que la contribution est faisable, mais n'ont pu prouver sa performance que par rapport à la contribution d'origine.

Totiya & al. [146] proposèrent un outil pour sélectionner les services cloud computing. En se basant sur le modèle SMI, les chercheurs ont aidé le client CC à choisir le service adéquat par un questionnaire. Ensuite, le client pouvait créer un vecteur de besoin, ainsi que choisir la mesure de similarité adéquate. Le travail traita une intéressante problématique, mais ne proposa ni évaluation, ni comparaison. En plus de cela, les auteurs ont proposé un questionnaire trop long, qui pourrait démotiver les clients à l'utiliser.

Travaux.	Méthodes	Objectifs	Critiques
Ruiz-Alvares & al. [120]	Tests de performance, liaison des capacités des services aux bases de données clientes	Réduire le temps nécessaire à la sélection	Tests pré-sélectionnés, aucune comparaison
Rehman Ur Zia & al. [148]	TOPSIS, MCDM, historique de la qualité des services CC	Maximiser la précision des services sélectionnés	Nécessité d'un retour de qualité des services CC IaaS
Patiniotakis & al. [105]	Mesures de similarité, MCDM, logique floue	Maximiser la qualité des services sélectionnés	Caractéristiques imprécis, aucune évaluation, aucune comparaison
Sahri & al. [121]	Calcul matricielle, logique floue	Prouver la faisabilité de leur approche	Aucune évaluation, aucune comparaison
Yongwen & al. [83]	Rough Set Theory	Vérifier la possibilité d'intégrer le concept de Rough Set Theory dans la sélection des services CC	Pas de sélection, mais uniquement la réduction d'un ensemble de services de départ
Le & al. [72]	Logique floue, ISM, ANP	Maximiser l'utilité des services sélectionnés	Aucun algorithme, aucune évaluation et aucune comparaison.
Raed & al. [62]	Prédication des performances et apprentissage	Maximiser la prédiction des performances	L'apprentissage nécessite un temps afin d'arriver à une convergence acceptable.
Souidi & al. [138]	Analyse des performance de manière itérative et continue	Assurer une sélection selon le temps de la sélection	Si le fournisseur change de matériel, les tests devraient être à refaire
Omerovic [104]	Calcul mathématique simple	Réduction des risques	Aucune évaluation et aucune comparaison.
Gad El Rab & al. [40]	Une solution adaptative aux besoins dynamiques des utilisateurs	Réduction du coût, délai d'attente et énergie consommée	Prise en compte de métriques de qualité fortement liées au matériel physique des nuages

Lee & al. [73]	La communication delphi floue, AHP floue, stratégie d'évaluation balancée	Prouver la faisabilité de leur approche	Pauvre formalisme mathématique pour une approche aussi complexe, aucun algorithme, aucun programme, aucune évaluation et aucune comparaison.
Sun & al. [140]	AHP, ontologie, similarité sémantique, MCDM	Maximiser la précision de la sélection	Aucun accès à l'ontologie nécessaire au processus
Do & al. [35]	Une collaboration entre utilisateurs, théorie des jeux	Réduire le coût du fournisseur pour les clients	Nécessité d'un temps au système afin de trouver un équilibre (équilibre de Nash)
Ma & al. [87]	Ensemble d'intervalles neutrosophiques	Trouver un compromis entre prise de risques, qualité et coût	Choix statiques peuvent fausser les résultats.
Lianyong & al. [108]	Prédiction du coût d'un service CC à partir de son modèle de facturation, la durée d'utilisation des clients et la charge désirée	La réduction du coût des services CC pour les clients	Prendre en compte l'horaire du lancement de la requête
Ruadulescu & al. [116]	TOPSIS, distance de Minkowski, SMI	Prouver la performance de l'approche E-TOPSIS par rapport à TOPSIS	Comparaison faible
Totiya & al. [146]	SMI, questionnaire	Précision de la sélection	Aucune évaluation, aucune comparaison, questionnaire trop long

TABLE 2.6: Tableau récapitulatif des travaux dans la sélection des services CC

La table 2.6 présente les contributions orientées vers la sélection des services sans pour autant faire une recommandation ou une composition. Nous remarquons que les techniques les plus utilisées sont les méthodes décisionnelles à choix multiples comme TOPSIS, mais leur objectif étant de maximiser la qualité ou la précision ne nous convient pas. Car les objectifs proposés par la majorité des travaux de l'état de l'art est la réduction du temps qui à notre avis est bien plus importante que la qualité.

2.5.5 Évaluation et sélection des services en nuage

Tout comme notre approche, certains proposent une sélection des services en nuage en les évaluant. Dans cette sous-section, nous nous intéressons aux contributions qui proposent une évaluation et une sélection en même temps.
Garg & al. [43] proposèrent une plateforme pouvant prioritiser les services en nuage, par le biais d'un modèle de qualité (SMI). Pour ce faire, les chercheurs ont d'abord modélisé les préférences des utilisateurs sous forme hiérarchique (SMI). Après cela, ils ont calculé une mesure de qualité pour chaque service, afin de les prioritiser, selon les préférences des clients. Le travail est bien expliqué, démontré et évalué, mais en aucun cas comparé avec des travaux similaires. Vu que l'objectif de l'approche était de maximiser la qualité, les tests déroulés sont le temps d'exécution par rapport au nombre de services au départ, ainsi que la qualité des instances par rapport à leur coût. Ceci dit, ils ne prennent pas en considération la position du client et du fournisseur dans le réseau, malgré le fait ils aient pris en considération le temps de réponse des services comme mesure de qualité.
Tajvidi & al. [143] proposèrent un outil pour sélectionner les services CC, selon les critères individuels des clients, à propos de la qualité de service. En utilisant la logique floue, l'outil proposé collecta les mesures individuelles des consommateurs, pour une évaluation et une sélection selon la qualité des services. Malgré la présence d'un formalisme mathématique et d'un exemple, les chercheurs n'ont pas jugé nécessaire d'évaluer, ni même de comparer leur travail avec d'autres approches similaires.
Ma & al. [86] proposèrent un modèle afin de sélectionner et évaluer les services les plus fiables et dignes de confiance. D'abord, les chercheurs ont représenté la confiance que peut avoir le client en son service sous formes d'intervalles de nombres. Ainsi, les services les moins performants sont écartés en utilisant des déviations statistiques. Ensuite, les auteurs ont proposé 2 formules afin de comparer des valeurs de confiance de tous les services. La première formule est de nature probabiliste, mais la seconde est de nature géométrique. À la fin, les chercheurs ont présenté une comparaison avec d'autres travaux similaires en termes de précision par rapport au taux d'apprentissage et des valeurs initiales de confiance (sur lesquelles ils ont appliqué les déviations statistiques). Malgré l'absence de code, d'algorithme et de pseudo-code, le travail reste intéressant, sauf qu'il nécessite un apprentissage, afin de converger vers un comportement optimal.
Upadhyay [147] proposa une architecture et une méthode afin d'évaluer et de prioritiser les services cloud computing. Pour cela, l'auteur a d'abord présenté les mesures de qualité prises en compte dans son papier. Ensuite, le chercheur a éliminé les services ne répondant pas aux exigences des utilisateurs. Après, il a utilisé le calcul matriciel afin de trier les services restants du meilleur au pire. Puisque l'auteur a voulu optimiser la recherche des services, nous nous attendions à des tests de performance, mais le chercheur a négligé l'évaluation, la comparaison avec des approches similaires et l'argumentation par un code ou un pseudo-code.

Travaux	Méthodes	Objectifs	Critiques
Garg & al. [43]	SMI, MCDM	Réduction du temps d'exécution, Augmenter la qualité des services CC	Négligence du coût et la qualité par rapport au temps dans les tests, négligence des positions des clients et des fournisseurs
Tajvidi & al. [143]	Logique floue	Maximiser la qualité des services sélectionnés	Aucun évaluation, ni comparaison
Ma & al. [86]	Degré de déviation, degré d'approximation, analyse de zone de probabilité, analyse géométrique	Maximiser la précision de la sélection des services CC	Nécessité d'un apprentissage, négligence du temps d'exécution, du coût et de la qualité
Upadhyay & al. [147]	Calcul matriciel, recherche par élimination	Maximiser la précision de l'évaluation	Aucune évaluation, aucune comparaison

TABLE 2.7 – Tableau récapitulatif des travaux proposant une évaluation et une sélection pour les services CC

La table 2.7 présente les contributions proposant 2 approches : une pour l'évaluation des services CC, et une autre pour leur sélection. Nous remarquons dans la table que la majorité des travaux avaient pour objectif la maximisation de la précision [86] de l'évaluation en utilisant différentes comme le calcul matriciel [147].

2.5.6 Travaux hors catégorie

Malgré les classes variées proposées dans cette thèse, il reste des contributions qui ne font pas nécessairement partie de l'une d'elle. Pour cela, nous avons créé une classe spéciale : elle peut contenir des architecture complète par exemple permettant la découverte, l'évaluation, ainsi que la sélection des services CC.
Pulikal & al. [107] proposèrent une méthode originale afin de sélectionner des services CC SaaS, en les reliant aux profils des clients. Pour faire cela, les chercheurs ont d'abord proposé une définition, ainsi qu'une spécification pour les profils des consommateurs. Ensuite, ils ont présenté l'algorithme utilisé afin de trouver les services SaaS, qui répondent au mieux aux exigences et préférences des clients. Ceci dit, même si les auteurs ont rigoureusement expliqué leur approche, et proposé un exemple, la contribution reste non évaluée, et non comparée à d'autres travaux similaires.
Bassiliades & al. [16] proposèrent un outil afin de relier les développeurs d'application web, aux services les plus adéquats de cloud computing d'un niveau PaaS. Par "service adéquat", nous voulons dire un service qui remplit l'ensemble des besoins des programmeurs, développeurs et concepteurs de l'application. Malgré la proposition de modèle mathématique, de code et d'évaluation, ce travail ne peut fonctionner sans son ontologie, et les chercheurs n'y ont pas donné accès.

Hajlaoui & al. [47] proposèrent un outil pour découvrir et sélectionner des services cloud computing d'un niveau IaaS. Pour cela, les chercheurs ont utilisé d'abord les modèles de variabilité afin de gérer les similarités entre les services en nuage. Ensuite, Hajlaoui a réduit le problème traité en un problème de Graph Matching. La raison pour cela était l'existence d'algorithmes adaptés au contexte comme les Graph Edit Distance. Le travail proposé reste une contribution intéressante, et bien expliquée, mais les auteurs ont évalué leur approche selon la précision seulement, et ont négligé par conséquent le temps d'exécution et le coût des services.
Hasan & al. [50] proposèrent une architecture complète, afin de découvrir et sélectionner des services cloud computing. Les chercheurs ont présenté un algorithme indépendant XNSim, afin de relier les services en nuage à ses utilisateurs. À la fin, l'algorithme est comparé avec ces prédécesseurs comme MNSim et SNSim. C'est certes un travail intéressant mais qui nécessite les retours des utilisateurs, ainsi qu'un temps pour converger vers un comportement optimal.
Yadav & al. [156] proposèrent une méthode afin d'évaluer et de sélectionner le meilleur service CC, pour une requête de client cloud entrante. Pour cela, les chercheurs ont d'abord considéré le problème d'évaluation, ainsi que la sélection comme étant un problème multi-choix à décisions multiples. Ensuite, ils ont proposé un processus AHP, afin de résoudre le problème. L'objectif étant depuis le début la réduction du temps nécessaire pour un tel processus, les chercheurs ont eu l'idée d'évaluer leur approche selon la qualité et le temps d'exécution. Mais, ils n'ont en aucun cas comparé leur contribution à d'autres travaux similaires, ou pris en compte le coût.

Travail	Prob.	Méthode	Objectifs	Critiques
Pulikal & al. [107]	Découverte et sélection des services CC SaaS	Relier les profils utilisateurs aux applications SaaS optimales	Maximiser la pertinence des résultats sélectionnés	Aucun algorithme, aucun programme, aucune évaluation et aucune comparaison.
Bassiliades & al. [16]	Relai des préférences de développeurs aux services PaaS adéquats	Calcul de similarité par le biais d'une ontologie	Optimiser la pertinence des résultats	Explication réduite sur l'ontologie utilisée
Hajlaoui & al. [47]	Découverte et sélection des services CC IaaS	Réduction du problème en un de graphe matching	Maximiser la précision des services sélectionnés	Aucun code
Hasan & al. [50]	Découverte et sélection des services CC	XNSim	Réduire le temps nécessaire à l'évaluation	Nécessite du feedback
Yadav & al. [156]	Evaluation et sélection du meilleur service CC pour une requête client entrante	MCDM, AHP	Réduction du temps de d'exécution	Aucun comparaison

TABLE 2.8 – Tableau comparatif des contributions hors catégorie dans la sélection de services cloud computing

La table 2.8 nous présente l'ensemble des travaux spéciaux dans le domaine de sélection, composition ou recommandation des services cloud computing. Ces travaux proposent par exemple :

- une découverte en plus de la sélection des services [107, 47, 50] ;
- une liaison entre des services cloud computing et des requêtes sémantiques (en plus de la sélection) comme :
  - les profils des utilisateurs [107] ;
  - les préférences des développeurs [16] ;
  - les requêtes des machines virtuelles des clients [156] ;

Puisque ces travaux sont généralement proposée sous forme d'une architecture complète, isoler le module de recherche ou celui de l'indexation du reste du système peut nuire à la performance du composant. De ce fait, nous avons préféré garder ces travaux pour une comparaison dans un futur proche et dans un papier dédié à cela.

2.5.7 Travaux de Référence

Le premier papier traitant le problème d'indexation de services pour le Cloud Computing d'une façon aussi directe est certainement le travail de Sundareswaran & al. [142]. Les chercheurs ordonnent les services dans un arbre B^{Cloud}-tree selon une fonction ordre Z. Les chercheurs ont pu démontré l'efficacité de l'approche par rapport à l'approche exhaustive (gourmande) traitant tous les cas. Pour cela, une analyse du temps d'exécution et de la scalabilité ont été affiché selon le nombre de services dans le eu de données. Nous nous sommes basés en premier lieu sur ce travail pour concevoir notre approche permettant de retrouver un maximum de services pertinents en un temps réduit. La différence majeure entre notre approche et celle de Sundareswaran & al. [142]. C'est qu'ils supposent au début que tous les résultats dans l'arbre quelque soient leurs positions sont acceptables. Alors que dans la vie réelle ainsi que dans notre approche, un service de Cloud Computing peut ne pas convenir aux pré-requis de l'utilisateur ou de ceux du réseau de capteurs sans fil. Par conséquent, nous disons que le service est incompatible et ne doit pas figurer parmi les résultats.
Lin & al. [81] reprend le travail de Sundareswaran & al. [142] afin d'optimiser le positionnement des services dans l'arbre et l'appeler B^{+Cloud}-tree. L'objectif était de réduire le temps nécessaire pour la recherche d'un service. Une évaluation est même proposée prenant en compte les approches [81], [142] et l'approche Baseline.
Le travail de Lin & al. [81] en plus de Sundareswaran & al. [142] nous paraissent intéressants, et donc, nous nous proposons de les comparer à notre contribution. Nous pensons que ces 2 travaux sont des travaux de référence pour nous car :

- ils proposent des approche de sélection ;
- ils proposent comme démarche une indexation ;
- ils essayent de mettre plus en avant les résultats pertinents ;
- ils ont évalué et comparé leurs approches avec d'autres travaux similaires selon le temps nécessaire à la recherche ;

L'objectif de notre thèse étant clair, nous allons passer maintenant à l'implémentation, validation et comparaison.

2.6 Conclusion

Dans ce chapitre, nous avons passé en revue de manière brève les solutions apportées dans l'état de l'art concernant la sélection des services cloud computing. Pour cela, nous avons collecté toutes les solutions se positionnant du côté fournisseur ensemble, et les solutions se positionnant du côté client ensemble en essayant de donner un avis critique. Nous avons ensuite comparé les techniques pour clients pour essayer de trouver les approches les plus intéressantes qu'on s'est proposé d'étendre.
Les 2 techniques [142, 81] utilisent l'indexation dans des arbres binaires par le biais des fonctions de Morton, et une recherche en profondeur dans leur structure. Nous avons remarqué qu'on pouvait améliorer encore plus les performances de ces 2 techniques, tout en considérant l'hétérogénéité des clients et la diversité des services.
Nous avons présentés une liste non exhaustive de travaux dans le domaine de sélection, recommandation, évaluation et composition des services cloud computing. Étant donné le nombre important de travaux cités, nous avons préféré catégoriser les travaux en famille. Pour chaque classe de contributions, nous avons souligner ses différences par rapport à notre approche proposée dans la thèse.

Dans les chapitres suivants, nous allons modéliser, tester comparer et valider la solution proposée à ces 2 techniques.

Chapitre 3

Contributions

3.1 Introduction

Sélectionner des services ne peut pas se faire sans fond mathématique solide bien décrit et des preuves scientifiques irréfutables. Dans les chapitres précédents, nous avons cité notre approche sans vraiment entrer dans les détails. Nous avons dit avoir recours à l'indexation et à la recherche des services cloud computing, sans vraiment dire comment, ni pourquoi.
Dans ce chapitre, nous allons d'abord formaliser mathématiquement notre contribution ainsi que les approches que nous avons étendues ([142, 81]). Nous allons ensuite présenter le modèle des RCSFs, des services CC, des clients... etc. Ceci veut dire l'ensemble des objets interagissant dans notre approche. À la fin, nous allons décrire explicitement le processus indexant et recherchant les services : à partir de la requête du client, jusqu'à la restauration des résultats.

3.2 Formalisme mathématique

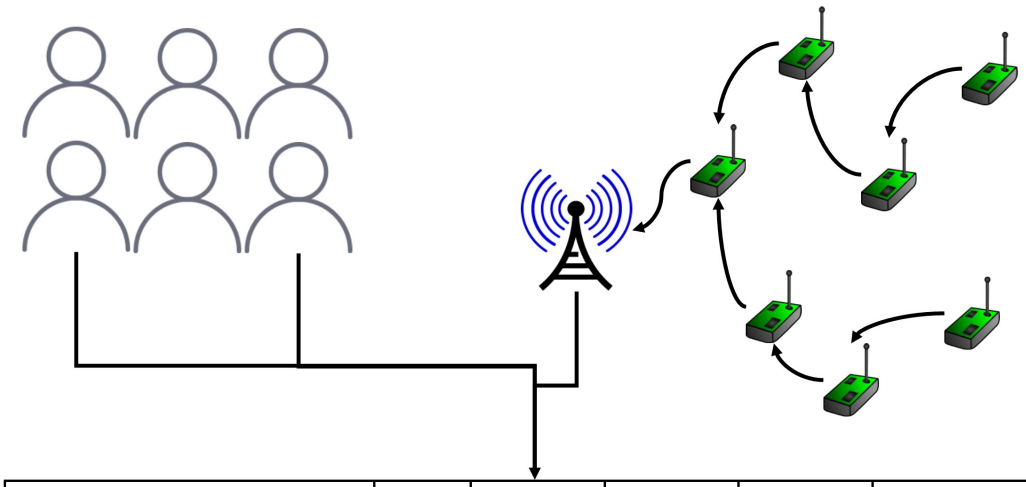
Dans cette section, nous proposons un modèle mathématique afin de traiter la problématique de la sélection de service en nuage pour des clients administrateurs des RCSFs.
Soit une personne possédant un réseau de capteurs de N_{No} nœuds. Cette première veut gérer les données surveillées dans un service en nuage pendant la période de mission T sans pour autant dépasser un certain budget B initialement réservé à la location du service cloud.
Parmi les méthodes existantes dans le domaine de sélection de service, nous allons tenter d'optimiser la pertinence des résultats retournés tout en minimisant le temps de la recherche.
Nous allons supposer que les clients peuvent varier d'un simple amateur avec son capteur thermique à la maison, à un administrateur d'un grand réseau de surveillance forestière au niveau mondial. Pour cela, nous allons :

1. choisir des mesures de qualité les plus générales possibles (voir la sous-section 3.2.2) ;
2. supposer que les préférences des clients sont hétérogènes (voir sous-section 3.2.2) ;

Pour des soucis d'implémentation, notre approche a besoin d'être la plus générique possible afin de garantir une future comparaison avec d'autres approches. Pour cela, nous allons introduire notre innovation sous forme de principe aux deux approches initialement étudiées [142, 81] (voir la section 3.3).

3.2.1 Réseaux de capteurs sans fil

Avant de parler du réseau lui-même, il faut savoir qu'un RCSF a une mission à achever. Vu que cette dernière va lui coûter du temps et de l'argent, nous notons T la durée totale de la mission du réseau et B le budget total alloué à la tâche globale du RCSF.
Un réseau de capteurs sans fil est en fait un ensemble de nœuds à capteurs sans fil organisés en fonction d'un but commun qui est l'acheminement des données générées par les capteurs à un emplacement donné. Dans notre cas, la location en question est justement un service de Cloud Computing IaaS publique.



Description	Type	ECU (GB)	RAM (GB)	HDD (GB)	Débit (GB/s)
Contrôle d'accès	Client	0.0977	0.0010	0	0.0002
Encodage	Node	0.1172	0.0010	0	0.0000
Cryptage/Décryptage	Node	0.1270	0.0024	0	0.0000
Analyse de données	Client	0.2930	0.2441	0	0.0146
Alerte des utilisateurs	Client	0.0005	0.0020	0	0.0000

FIGURE 3.1 – Représentation d'un réseau de capteurs sans fil

La figure 3.1 démontre comment un réseau de capteurs sans fil est considéré dans notre étude. En effet, il est considéré comme un ensemble de services, un ensemble de clients [1] et un ensemble de nœuds pouvant incorporer plus d'un capteur.
Comme le montre la figure 3.1, un ensemble de clients réseau veulent se connecter à un réseau de capteurs sans fil tout en ayant un certain nombre de services. Pour

1. Il faut distinguer entre un *client réseau* et un *client cloud*. Le client cloud est le propriétaire du RCSF, alors que le client réseau est le client du client cloud.

sélectionner des services CC, nous n'avons pas besoin de savoir la consommation d'énergie, le type de phénomène surveillé ou encore la topologie. Nous nous devons seulement de savoir combien génère les nœuds et en quelle période.

$$((d_1, t_1), (d_2, t_2), \ldots, (d_{N_{No}}, t_{N_{No}}))$$

Le vecteur précédant illustre un réseau de capteur sans fil à N_{No} nœuds sachant que d_i et t_i notent respectivement la taille de données en octets ainsi que leur période de transmission en secondes.
A partir d'un vecteur pareil, nous pouvons estimer la taille du disque dur et la bande passante à louer dans le service IaaS par le formule dans l'équation suivante :

$$HDD_{No} = BW_{No} = \sum_{i=0}^{N_{No}} d_i * (T/t_i)$$

Après estimation des ressources consommées par les nœuds à capteurs sans fil, nous pouvons dorénavant estimer les capacités à louer pour les services. La figure 3.1 inclut un tableau illustrant les services attendus par les clients du réseau. Nous pouvons remarquer les ressources requises pour chaque service dans le tableau ainsi que le type de chacun.
Si un service réseau [2] est représenté par le vecteur suivant :

$$S_i = (type_i, ecu_i, ram_i, hdd_i, db_i)$$

tel que *type*, *ecu*, *ram*, *hdd* et *db* dénotent respectivement le type du service, la fréquence d'horloge, la taille de la mémoire vive, la taille du disque dur ainsi le débit nécessaires tous au service réseau, alors, nous pouvons calculer les ressources du service CC nécessaires à tous les services du RCSF par les formules suivants :

$$\begin{cases} ECU_S &= \sum_{i=0}^{N_S} ecu_i * f(S_i) \\ RAM_S &= \sum_{i=0}^{N_S} ram_i * f(S_i) \\ HDD_S &= \sum_{i=0}^{N_S} hdd_i * f(S_i) \\ BW_S &= \sum_{i=0}^{N_S} bw_i * f(S_i) \end{cases}$$

Sachant que ECU_S, RAM_S, HDD_S et BW_S sont respectivement la fréquence du processeur, la taille de la RAM, la taille du disque dur et la bande passante nécessaires à l'instance cloud computing pour faire fonctionner tous les services réseau. D'un autre côté, la fonction f est définie comme suit :

$$f(S_i) = \begin{cases} N_{No} & \iff type_i = Node \\ N_{client} & \iff type_i = Client \end{cases}$$

tel que N_{client} est le nombre total de clients réseau (qu'ils soient en ligne ou hors ligne). Pour le nombre maximal d'utilisateurs connectés, nous noterons N_{user}.
Enfin, nous pouvons extraire les requis globaux du réseau de capteurs sans fil en utilisant le système suivant :

$$\begin{cases} ECU_{WSN} &= ECU_S \\ RAM_{WSN} &= RAM_S \\ HDD_{WSN} &= HDD_S + HDD_{No} \\ BW_{WSN} &= BW_S + BW_{No} \end{cases}$$

2. Il faut différencier un service réseau tel que le contrôle d'accès du service cloud computing tel qu'une instance large de Amazon EC2. Nous appellerons le premier *service réseau* et le deuxième *service cloud computing*.

Puisqu'on a choisi des métriques de qualité tels que le temps de réponse et la disponibilité, nous aurons besoin de la position du RCSF afin de déduire la qualité de la communication entre le réseau à capteurs sans fil et le service cloud computing. Par conséquent, le vecteur définissant un RCSF quelconque est le suivant :

$$(ECU_{WSN}, RAM_{WSN}, HDD_{WSN}, BW_{WSN}, Location) \tag{3.1}$$

3.2.2 Clients

Afin de gérer la diversité des clients (contrairement à [142, 81]), nous allons considérer un consommateur final par les préférences fixées par ce dernier. Le terme choix, goût ou préférence des clients désigne dans notre étude surtout une métrique de qualité. Les critères les plus pris en compte par les consommateurs de manière générale selon [10, 67] sont respectivement la disponibilité et le temps de réponse. Pour différencier une solution chère d'une autre moins coûteuse, nous avons ajouté le reste du budget alloué aux mesures.

$$(i_{Av} = 15, i_{RT} = 15, i_{SM} = 70) \tag{3.2}$$

Du coup, l'équation 3.2 représente un utilisateur préférant économiser de l'argent au lieu d'avoir un service CC plus disponible ou mieux répondant. Ainsi, il donne une préférence de 15% à chacune des mesures de qualité *Temps de réponse (Response Time (RT))* et *Disponibilité (Availability (Av))*, et 70% pour le critère de qualité *Budget restant (Saved Money (SM))*.
Bien qu'on n'ait pris que trois mesures de qualité, l'approche peut facilement supporter plus de mesures ou différents critères. Par conséquent, nous pouvons orienter l'algorithme de sélection de service vers le choix du fournisseur le plus sécurisé, le plus performant, le moins cher... etc.

3.2.3 Service de cloud computing IaaS publique

Puisque nous allons sélectionner des services de cloud computing à un niveau IaaS, nous allons en d'autres termes choisir des machines virtuelles [3] distantes hébergées par le fournisseur cloud computing lui même. Ces premières sont généralement caractérisées par leurs capacités : capacités de calcul, de stockage et de communication. A part ça, une même instance peut être offerte par le fournisseur dans 2 centres de données différents. Par exemple, le fournisseur CloudSigma [4] possède des centres de données répartis en 10 différentes locations. En plus de ça, chacune des locations présente toutes les instances offertes par le fournisseur lui même. D'une autre part, un centre de données localisé à Zurich aura fort probablement un temps de réponse et une disponibilité plus favorables qu'une autre aux états-unis pour un client localisé en France.
En plus de ça, chaque instance possède un prix pour sa machine virtuelle et un autre pour la bande passante. Par conséquent, la représentation d'un service cloud computing de type IaaS et publique est la suivante :

$$(ECU, RAM, HDD, BW, P_H, P_{BW}, Location) \tag{3.3}$$

Tels que ECU, RAM, HDD et BW sont les capacités en calcul, stockage et communication de l'instance ; P_H et P_{BW} sont respectivement le prix horaire de l'instance et

3. communément appelés instances

le prix de la bande passante du centre de données.
Après avoir modéliser mathématiquement tous les objets interagissants dans notre système, nous pouvons passer à la phase de l'indexation. Elle transforme l'ensemble des données entrantes de notre algorithme de recherche, en une structure bien formelle ordonnant spécialement les services CC. Dans notre approche ainsi que les travaux [142, 81], nous indexons les services dans un arbre AVL.

3.3 Indexation des services

Dans cette section, nous allons aborder l'ultime différence entre l'approche proposée et celles qu'on a étendues [142, 81]. La contribution réside essentiellement dans l'indexation des services cloud computing publiques IaaS dans un arbre binaire de recherche semi-équilibré. La différence entre notre approche et les autres réside dans la Fonction à courbe Z [99] utilisée pour encoder les services puis l'insérer dans l'arbre.

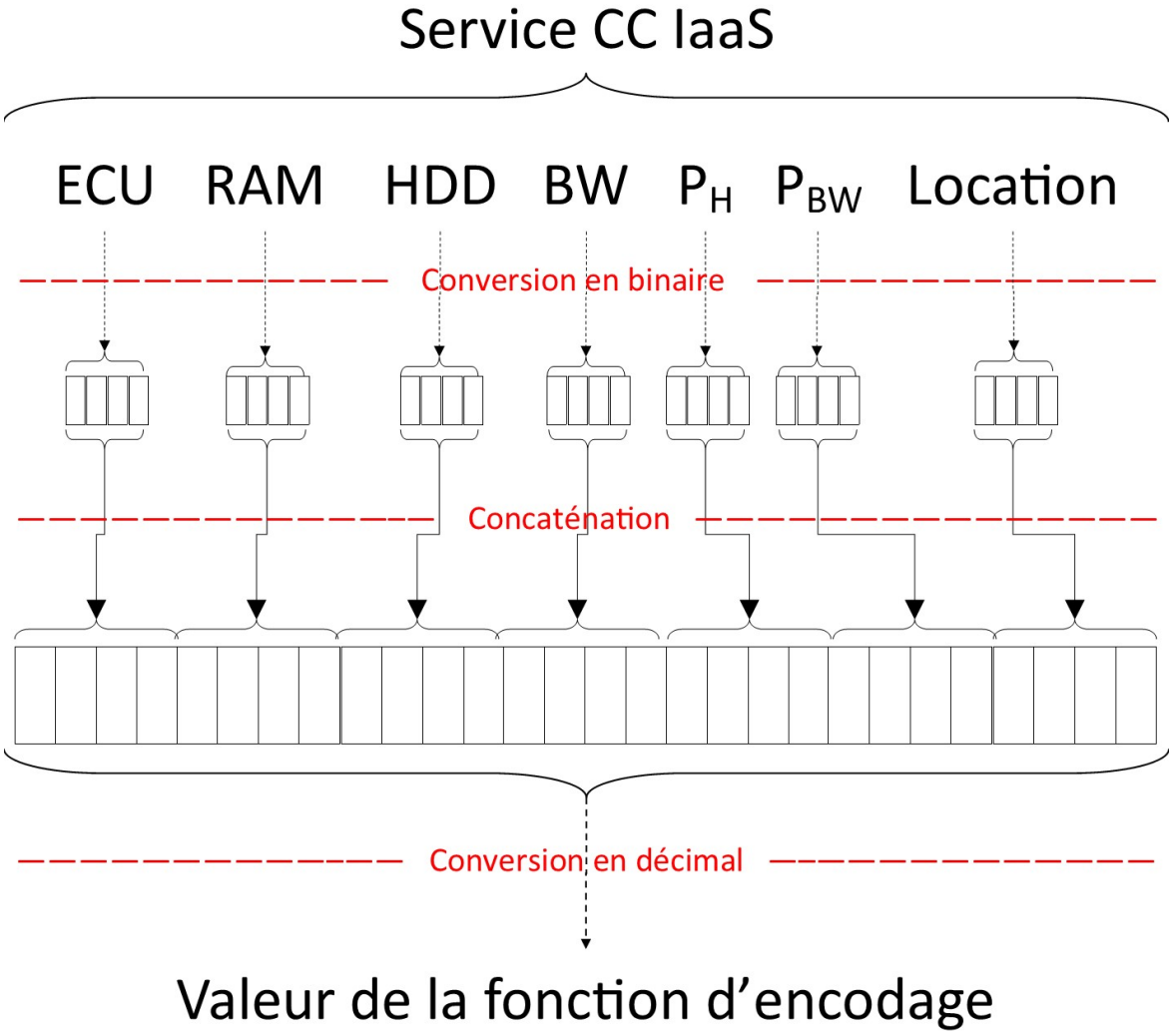


FIGURE 3.2 – Calcul des valeurs de la fonction à courbe Z pour l'approche de Sundareswaran & al. [142]

La figure 3.2 illustre comment calculer les valeurs de la fonction à courbe Z selon

l'approche de Sundareswaran & al. [142]. D'abord, nous convertissons les valeurs des attributs du service en binaire. Sur la figure 3.2, les quantités sont codés sur 4 bits chacune. Mais nous pouvons choisir plus ou moins d'espace pour faire la transformation. Ensuite, nous concaténons les résultats du changement de base de numération. Et enfin, nous reconvertissons le tout en décimal afin d'avoir la valeur de la fonction en courbe Z.

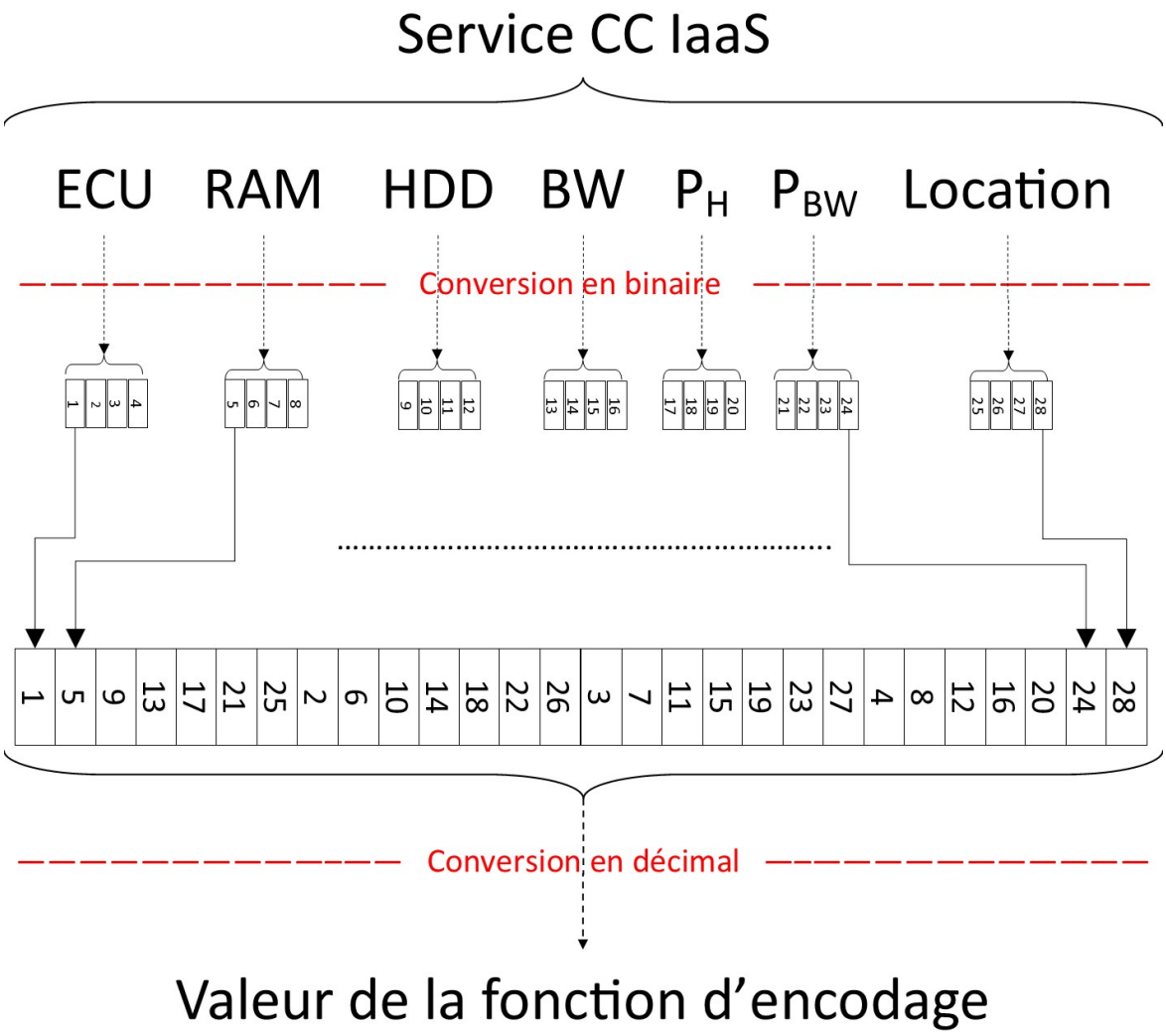


FIGURE 3.3 – Calcul des valeurs de la fonction à courbe Z pour l'approche de Lin & al. [81]

La figure 3.3 représente comment est calculée la valeur de la fonction de courbe de Morton selon l'approche de Lin & al. [81]. Le processus est le même que celui de Sundareswaran & al. [142] à l'exception de l'étape de la concaténation. Au lieu de prendre les blocs des représentations binaires des attributs des services tels qu'ils sont, l'approche de Lin & al. [81] alterne les bits des représentations binaires. Le but était de rapprocher plus les instances cloud computing similaires dans l'arbre AVL.
Dans notre approche, nous supposons une diversité des services, une hétérogénéité des clients du nuage et généricité de la solution. La diversité des services veut dire que les instances IaaS prises en compte ne sont pas forcément similaires. Du coup, elles peuvent

varier en termes de location, de fournisseur, de modèle de facturation et surtout en termes de capacités (stockage, calcul et communication). Pour cela, nous proposons avec notre approche le concept de validité d'une solution (voir la sous-section 3.4.2). Ce dernier vérifie si un service peut effectivement assurer la tâche demandée par le fournisseur de RCSF.
D'un autre côté, nous supposons une hétérogénéité en clients de nuage. Ce qui veut dire qu'ils peuvent avoir des préférences différentes vis à vis des mesures de qualité prises en compte dans la modélisation (voir la sous-section 3.2.2). Pour cela, nous proposons avec notre approche le concept d'optimalité. Ce dernier caractérise tout simplement une solution plus intéressante au client cloud par rapport aux autres.
En dernier lieu et afin de respecter une certaine généricité de l'approche, nous supposons un principe qui est de *ne pas perdre de temps à chercher des solutions qui sont déjà non valides*. Pour cela, nous allons d'abord catégoriser les attributs des services cloud en deux groupes distincts. Ces classes de caractéristiques représentent la capacité et la qualité. Mais, un attribut donné peut changer de catégorie si nous choisissons de pencher vers la perception du client au lieu du fournisseur et vice versa.

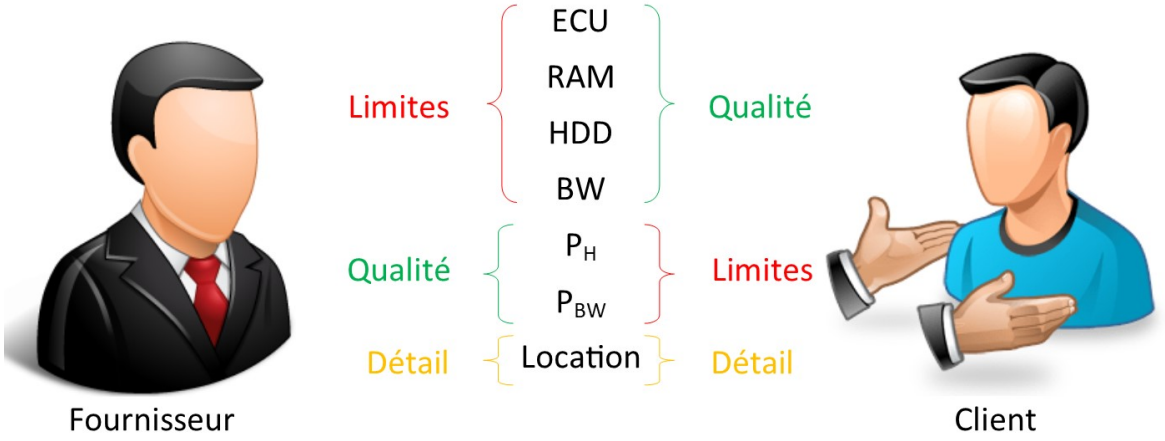


FIGURE 3.4 – La différence de perception fournisseur/client quant aux attributs de services cloud computing

La figure 3.4 montre la divergence de la valeur des caractéristiques d'un service pour un client par rapport à son fournisseur. Par exemple, la taille de la mémoire vive (RAM) est une limite ou une capacité pour le fournisseur du service alors que ça représente une qualité pour le client. Car avec une plus importante taille de mémoire vive, le client peut faire ses calculs plus rapidement. Alors que le fournisseur doit allouer plus d'espace et faire payer les clients d'une autre manière afin de leur garantir ce privilège.
D'autre part, les différents prix des instances sont une capacité pour le client, alors que c'est plus une qualité pour le fournisseur. Plus exactement, un client ne peut pas choisir une instance plus chère que son budget réservé à la mission de son réseau de capteurs sans fil. Par conséquent, les prix des services cloud computing représentent une limite pour le client. Par ailleurs, les fournisseurs de tels services se partagent un marché dans lequel ils sont concurrents. Donc, les prix représentent plus une qualité (qui est de coûter moins) qu'autre chose.
Afin d'être sûr qu'on ne va pas rechercher dans les services qui ne sont déjà pas valides pour un client donné dans l'arbre AVL, nous indexons les services en nuage selon les

attributs de la catégorie "limite" selon le client cloud. Ceci aura pour avantage de :

1. ordonner les services dans un arbre AVL selon les capacités économiques des clients. Ces derniers pourront ainsi dérouler leurs recherches sur la partie concernée de l'arbre et non la totalité de la structure. Par conséquent, l'algorithme de recherche prendra moins de temps d'un côté et retournera des résultats plus pertinents d'un autre ;
2. réduire le temps réservé à l'indexation des services et à la création de l'arbre binaire ;
3. réduire la taille de l'index en utilisant moins d'attributs ;

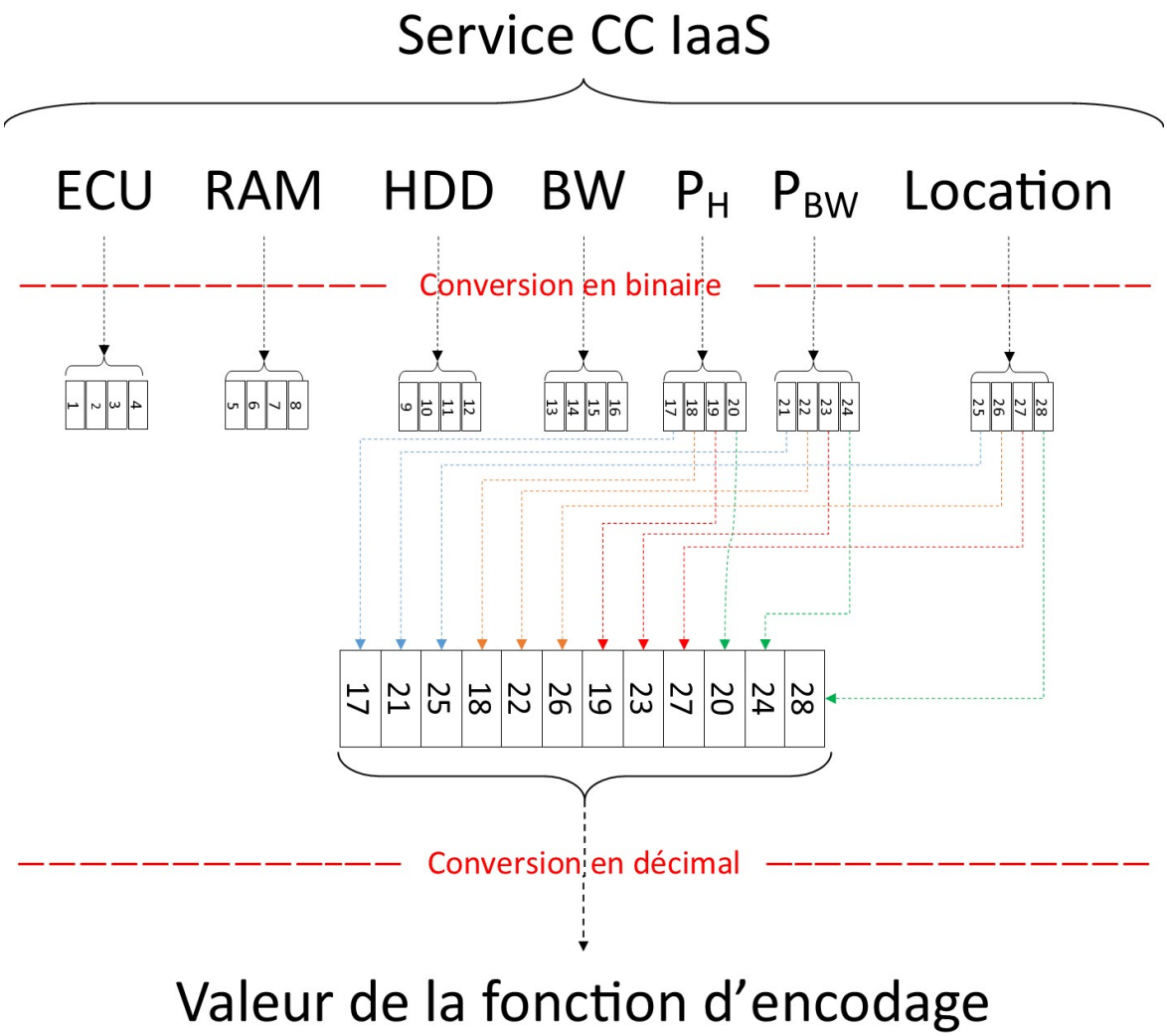


FIGURE 3.5 – Calcul des valeurs de la fonction d'encodage pour notre approche

La figure 3.5 montre la façon avec laquelle nous calculons la fonction à courbe Z. Comme nous l'avons dit précédemment, nous prenons en compte les attributs représentant une limite pour les clients et nous ne prenons pas en compte les attributs représentant la qualité. Mais pour des soucis pratiques, nous avons inclus la location des services au processus d'indexation. Pour la simple raison qu'un fournisseur peut proposer le même service avec les mêmes prix dans deux endroits différents. Si nous ne prenons pas en compte la location, nous aurons une seule valeur pour la fonction

de Morton pour deux services différents puisque le temps de réponse et la disponibilité ne seront pas les mêmes. Avoir une valeur dupliquée dans un arbre AVL n'est pas toujours pratique voir même impossible vu que c'est un arbre binaire ordonné.

3.3.1 Conversion en binaire des attributs

La conversion en binaire varie d'un attribut à un autre ; plus précisément, d'une classe d'attributs à une autre. À titre d'exemple, nous ne pouvons pas convertir la capacité de calcul d'une instance IaaS ainsi que sa location de la même sorte. La capacité de calcul est un entier ou un réel. Il peut donc être convertit par la concaténation de restes de divisions successives. Mais, la location est une chaîne de caractères. Elle n'a aucune valeur numérique, et ne peut donc être convertie aussi facilement.
Pour cela, Sundareswaran & al. [142] propose 4 classes d'attributs. Même si nous en avons utilisé que 2 dans l'évaluation, nous avons changé légèrement la façon de convertir pour toutes les 4 classes. Dans le reste de la sous-section, nous allons expliquer comment convertir les attributs des groupes concernés.

3.3.1.1 Attribut réel

Comme *ECU*, *RAM*, et *HDD*, les attributs réels sont des attributs qui peuvent tout simplement prendre des valeurs réelles. Au contraire des attributs entiers qui ne le peuvent pas. Afin de convertir une telle classe d'attributs, nous avons besoin de découper l'ensemble de l'intervalle des valeurs possibles en sous-intervalles. Par exemple, l'attribut *ECU* peut prendre des valeurs de 0 jusqu'à 44. Pour convertir un tel attribut, nous découpons l'intervalle [0,44] en un nombre fixe de sous-intervalles [126].

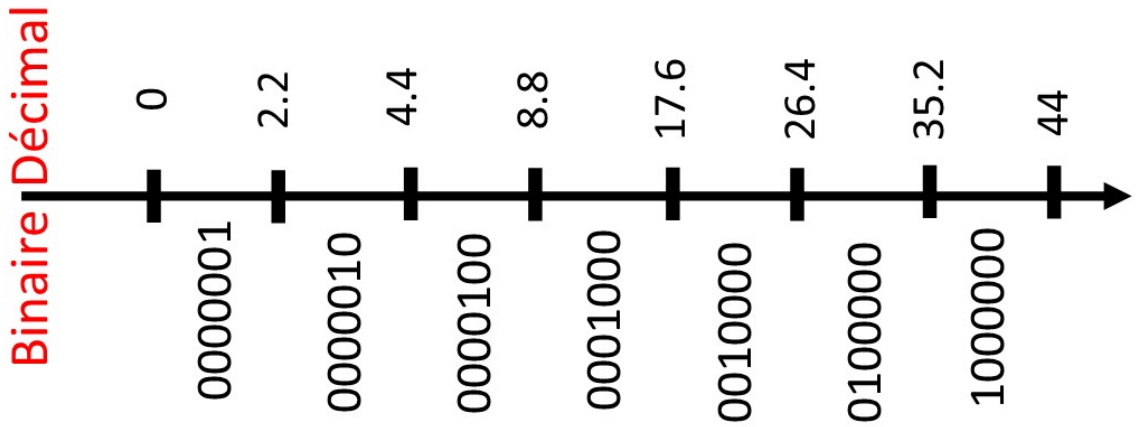


FIGURE 3.6 – Conversion en binaire des valeurs de l'attribut *ECU*

La figure 3.6 montre de quelle manière une valeur de l'attribut *ECU* d'un service va se convertir en binaire sur une longueur de 7 bits. D'abord, nous découpons les sous-intervalles. Nous donnons le nombre 0000001 comme représentation pour le premier. Plus nous passons au sous-intervalle suivant, nous décalons le 1 vers la gauche. Ce qui donne une représentation unique pour chaque sous-intervalle. D'un autre coté, deux valeurs différentes appartenant au même intervalle se verront donner

la même représentation en binaire. Mais, nous aurons économisé en nombre de bits pour représenter le changement de base de numération.

3.3.1.2 Attribut à catégories

Nous pouvons citer la location, l'identifiant du fournisseur ainsi que les références des instances [4]. Ce sont des attributs de chaînes de caractères. N'ayant pas de valeurs numériques, la conversion en binaire ne se fait pas directement. Afin d'y arriver, nous trions les chaînes selon un ordre qui peut être aléatoire ou spécifique. De même que les attribut précédents, nous donnons 0000001 à la première chaîne. Et au fur et à mesure qu'on avance dans la liste. Nous décalons le 1 d'une case vers la gauche. À la fin, nous aurons une représentation binaire sur un nombre de bits égal au nombre de valeurs possibles que peut avoir l'attribut en question [127].

Chaîne de caractères	Valeur en binaire
Frankfurt-Germany	000000001
Honolulu-USA	000000010
Manila-Philippines	000000100
Miami-USA	000001000
Perth-Australia	000010000
San Jose-USA	000100000
Warsaw-Poland	001000000
Washington DC-USA	010000000
Zurich-Switzerland	100000000

TABLE 3.1 – Conversion en binaire des valeurs de l'attrribut *Location*

La table 3.1 montre de quelle manière est calculée la conversion en binaire d'une valeur de l'attribut *Location*. Puisque nous avons 9 valeurs possibles pour le critère, la conversion se fait donc sur 9 bits. La ville allemande de Frankfurt est la première valeur possible. Elle vaut donc une valeur binaire de 1. Plus nous descendons dans le tableau, plus le 1 se décale d'une position vers la gauche. À la fin, nous arrivons à la ville suisse qui est Zurich. Puisque c'est la dernière et nous codons sur 9 bits, elle vaut la valeur de 100000000 en binaire.

3.4 Recherche des services

Nous avons choisi de suivre le processus métier décrit dans la figure 3.7. Pour des raisons de conformité et d'équité, nous avons adapté les 2 approches précédentes et la notre à ce processus métier.

4. Nous voulons dire par références d'instances les noms courts donnés aux services IaaS pour désigner leurs performances comme *Small-1* ou *Large-2*.

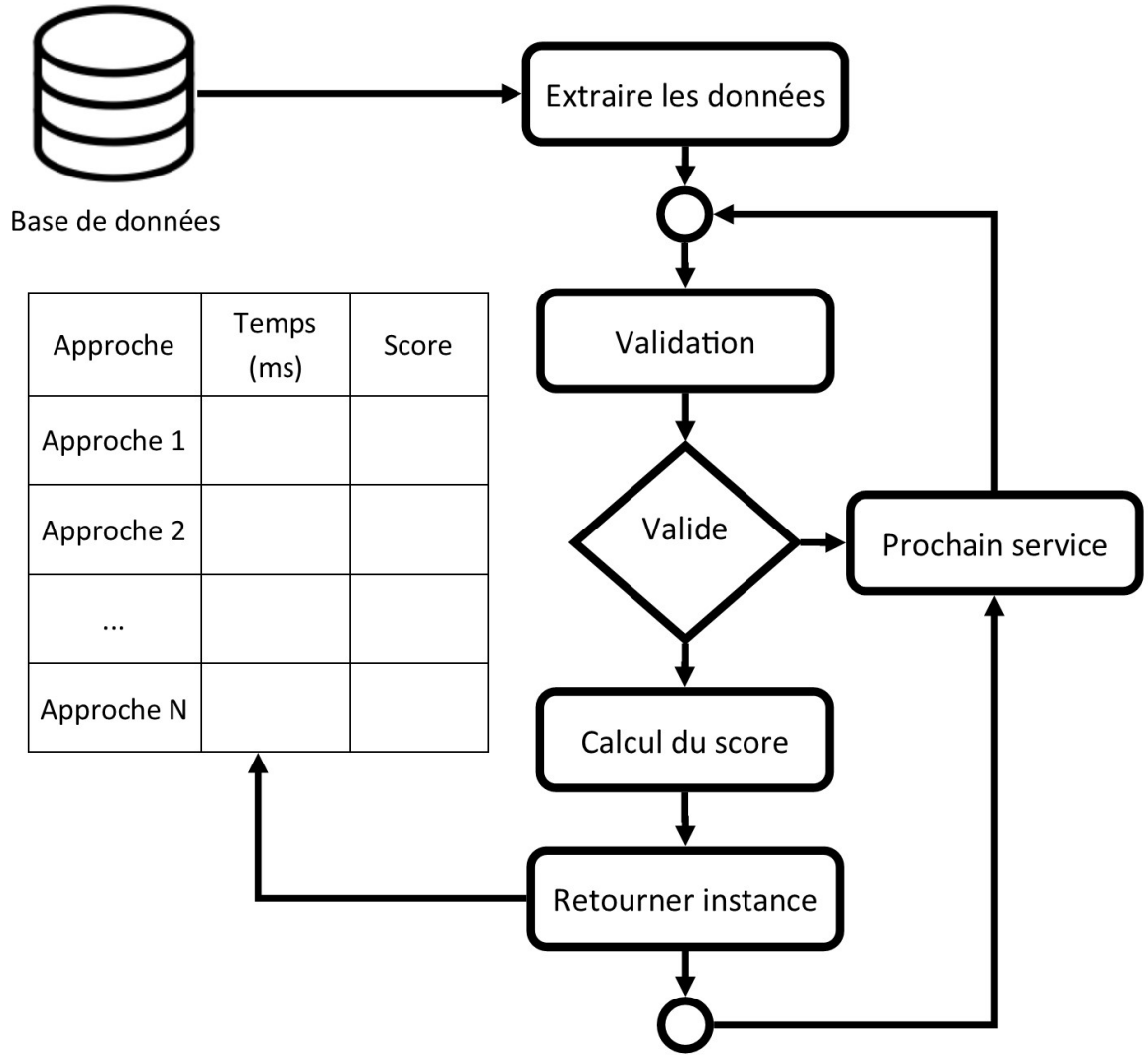


FIGURE 3.7 – Processus de recherche des services Cloud Computing

Comme présenté dans la figure 3.7, l'algorithme de recherche utilisé dans la partie évaluation repose sur le schéma suivant :

1. Collecter les données à propos des services en nuage que ce soit d'une base de données, d'un fichier ou d'une application tierce.
2. Parcourir la partie de l'arbre concernée (comme démontré dans la sous-section 3.4.1).
3. Pour chaque nœud parcouru, nous vérifions s'il est valide ou pas (comme expliqué dans la sous-section 3.4.2).
4. Si un résultat est valide, nous calculons son score (comme expliqué dans la sous-section 3.4.3).
5. Les services retournés sont retenus munis de leurs scores respectifs ainsi que le moment auquel ils ont été retourné.

Le but d'une telle manœuvre est de donner des chances égales à toute approche étudiée et de bénéficier d'un maximum de potentiel pour une comparaison la plus juste possible.

3.4.1 Recherche dans l'arbre AVL

Le principe de base de notre approche est de ne pas parcourir les services qui sont déjà invalides. Puisque l'arbre dans lequel les services en nuage sont indexés est ordonné, nous n'allons pas parcourir les nœuds de services jugés trop faibles pour les requis du client ou ceux des instances jugées trop chères à la capacité du client. Pour cela, nous calculons des bornes pour la recherche.

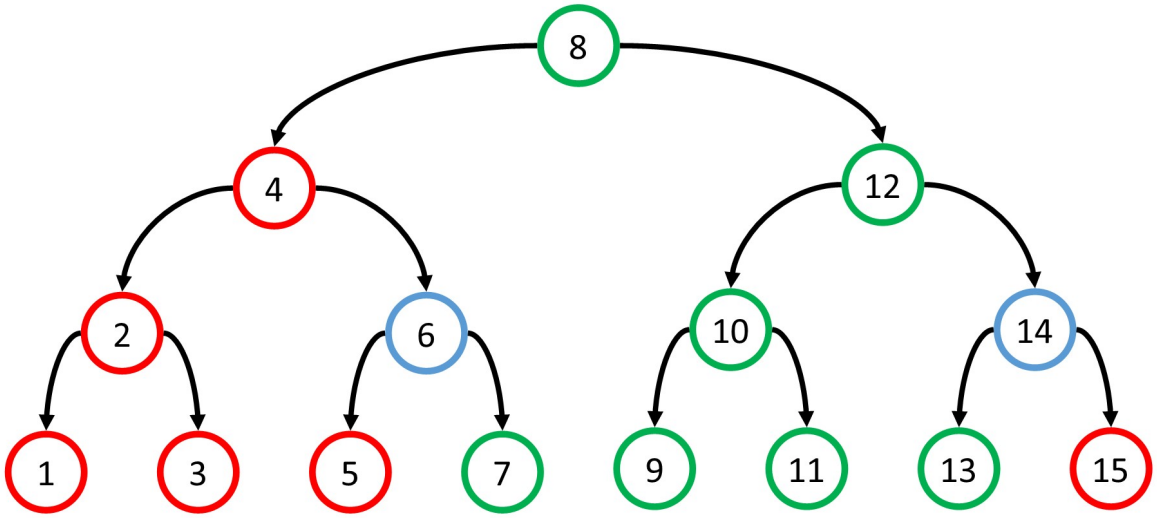


FIGURE 3.8 – Bornes de la recherche dans l'arbre AVL

La figure 3.8 montre l'importance des bornes afin de réduire les nœuds à parcourir dans l'arbre, et minimiser par conséquence le temps nécessaire pour la recherche des services. Dans cet exemple, les bornes obtenues sont représentées par les éléments 6 et 14 (en bleu). Donc, la recherche ne concerne pas des nœuds en dehors de l'intervalle [6,14] (en rouge) et se consacre uniquement aux éléments de l'intérieur (en vert).
Afin de calculer des bornes pour la recherche dans l'arbre d'indexation des services, nous avons besoin d' :

1. un réseau de capteurs sans fil tel exprimé dans l'équation 3.1 ;
2. un budget global réservé à l'allocation des ressources en nuage B ;
3. une durée totale de la mission du(des) réseau(x) de capteurs sans fil T ;
4. un jeu de données concernant les services cloud computing IaaS publiques ;

Attribut de service CC	Borne inférieure	Borne supérieure
ECU [GHz]	ECU_{WSN}	Max_{ECU}
RAM [GB]	RAM_{WSN}	Max_{RAM}
HDD [GB]	HDD_{WSN}	Max_{HDD}
BW [GB]	0	Max_{BW}
P_H [$/h]	0	B/T
B_{BW} [$/GB]	0	B/BW_{WSN}
Location	Frankfurt-Germany	Zurich-Switzerland

TABLE 3.2 – Calcul des bornes pour la recherche dans l'arbre AVL

La table 3.2 illustre la manière avec laquelle nous calculons les bornes pour rechercher dans l'arbre AVL. Sachant que ECU_{WSN}, RAM_{WSN}, HDD_{WSN} et BW_{WSN} sont respectivement les ressources en calcul, en mémoire vive, en mémoire morte et en bande passante nécessaires au RCSF afin d'achever sa mission en totalité (comme mentionné dans l'équation 3.1). Max_{ECU}, Max_{RAM}, Max_{HDD} et Max_{BW} sont respectivement la valeur maximale des ressources ECU, RAM, HDD et BW dans le jeu de données [5].
Afin de chercher dans l'arbre AVL, nous convertissons d'abord les bornes en binaire pour calculer leurs valeurs de la fonction de Morton. Ensuite, nous obtenons les nœuds limitant la plage à rechercher qui sont simplement le nœud ayant la plus petite valeur de Morton supérieure à celle de la borne inférieure et le nœud ayant la plus grande valeur de la fonction à courbe Z inférieure à celle de la borne supérieure. Par conséquence, l'intervalle d'éléments à parcourir sont les nœuds bornés par les deux précédemment calculés.

3.4.2 Validation d'un service

Dans notre approche, nous gérons des services et des clients cloud computing hétérogènes. Du coup, il est nécessaire de filtrer les services parcourus durant la recherche. Le but d'une telle procédure est de vérifier si le service en cours satisfait l'ensemble des requis du réseau à capteurs sans fil client. Par conséquent, nous disons qu'un service cloud computing publique de type IaaS tel défini dans l'équation 3.3 est valide pour le réseau à capteurs sans fil tel défini dans l'équation 3.1 si et seulement si le système suivant est satisfaisant :

$$\begin{cases} ECU_{WSN} & <= & ECU \\ RAM_{WSN} & <= & RAM \\ HDD_{WSN} & <= & HDD \\ B & >= & P_H * T + B_{WSN} * P_{BW} - BW \end{cases}$$

3.4.3 Calcul du score d'un résultat

Sur un ensemble de résultats valides, quelques uns sont forcément mieux que les autres. Pour cette raison, nous calculons un score pour donner un rang à chaque service. Si un service possède un score plus important, alors c'est une meilleure instance. Pour calculer un tel rang, nous nous basons sur la formule suivante :

$$Av * i_{Av} + RT * i_{RT} + (B - (P_H * T + B_{WSN} * P_{BW} - BW)) * i_{SM} \tag{3.4}$$

Tels que Av et RT sont respectivement la disponibilité et le temps de réponse du service par rapport au client cloud, qui sont calculés à partir des deux locations client et fournisseur.

5. Nous remarquons l'indépendance de la génération de l'arbre AVL, le calcul de la fonction de Morton ainsi que le calcul des bornes des caractéristiques des clients du nuage précédemment mentionnés dans l'équation 3.2. Effectivement, nous ne pouvons pas se permettre de générer un nouvel arbre pour chaque client. Pour cette raison, nous avons fait en sorte que les étapes précédemment citées soient les moins relatives possibles aux préférences des utilisateurs cloud telles définies dans l'équation 3.2

3.5 Conclusion

Dans ce chapitre, nous avons décrit avec toute la rigueur possible notre contribution qui se décline en 2 sous-contributions : l'une portant sur l'indexation, et l'autre sur la sélection. Pour faire cela, nous avons d'abord introduit mathématiquement les objets interagissant dans notre système. Ensuite, nous avons décrit le processus de sélection des services cloud computing, qui a été divisé en 2 phases : une première étape d'indexation, et une seconde pour la recherche. Ceci dit, un outil bien expliqué ne veut pas dire pour autant qu'il est faisable, fiable ou performant.
Pour prouver la faisabilité de notre contribution, nous allons dérouler un cas d'étude (exemple) dans le chapitre suivant.
Afin de prouver la fiabilité de notre approche, nous avons partagé publiquement le code source de notre projet [127, 126, 128, 129] sur la plateforme d'hébergement de code source *GitHub*.
Pour prouver la performance de notre contribution par rapport aux 2 approches étendues [142, 81], nous allons dérouler une série de tests et analyser les résultats dans les chapitres à suivre.

Chapitre 4

Étude de cas

L'exemple n'est pas le meilleur moyen de convaincre, c'est le seul.
(Ghandi)

4.1 Introduction

Dans le chapitre précédent, nous avons développé nos contributions avec son modèle mathématique, son mode de fonctionnement et les paramètres optimisés. Après , un lecteur peut trouver des difficultés à imaginer le processus complet, ou à trouver la motivation d'un fragment de code dans le projet [127, 126, 128, 129]. Puisque nous avons évité la comparaison avec des approches ne présentant pas de code, d'exemple, d'évaluation ou de comparaison (voir la chapitre 2), nous nous devions de les présenter pour notre contribution.
Expliquer un concept est une chose, mais le mettre en pratique en est une autre. Dans ce chapitre, nous allons dérouler un exemple complet de notre approche. Tout d'abord, nous commençons par la restauration des données concernant les services cloud computing. Ensuite, nous indexons ces derniers dans un arbre AVL. Après, nous extrayons les données concernant les réseaux de capteurs sans fil ainsi que les utilisateurs. Enfin, nous exécutons les requêtes des clients en trouvant les bornes, validant les services et calculant les scores des résultats.

4.2 Indexation

4.2.1 Instances IaaS publiques

Considérons la table 4.1 contenant l'ensemble des services cloud computing publiques d'un niveau IaaS pour l'exemple qui va suivre.

Service	ECU GHz	RAM GB	HDD GB	BW GB	P_H \$/h	P_{BW} \$/GB	Location
1	2.20	1	50	80	0.01950	0.0443	Zurich
2	2.20	2	50	160	0.02000	0.04000	Washington
3	4.40	2	50	320	0.03200	0.04000	San Jose
4	8.80	8	80	640	0.10900	0.04000	Miami
5	17.60	16	160	1280	0.24400	0.07150	Honolulu
6	26.40	32	320	2560	1.64800	0.01118	Perth
7	35.20	48	480	3840	1.71700	0.01708	Manila
8	44.00	64	640	5120	0.83400	0.04180	Frankfurt

TABLE 4.1 – Exemple de services CC IaaS

4.2.2 Conversion en binaire des attributs

Comme mentionné dans la section 3.3, il faut calculer la valeur de la fonction de Morton pour chaque service afin de l'indexer dans l'arbre AVL. Et pour cela, il faut d'abord encoder les attributs des services selon la méthode décrite dans la figure 3.5.

Service	ECU	RAM	HDD	BW	P_H	P_{BW}	Location
1	00000001	00000001	00000001	00000001	00000001	00100000	00000001
2	00000001	00000010	00000001	00000010	00000010	00000100	00000010
3	00000010	00000010	00000001	00000100	00000100	00000100	00000100
4	00000100	00001000	00000010	00001000	00001000	00000100	00001000
5	00001000	00010000	00000100	00010000	00010000	01000000	00010000
6	00010000	00100000	00001000	00100000	01000000	00000001	00100000
7	00100000	00110000	00010000	01000000	10000000	00000010	01000000
8	01000000	01000000	00100000	10000000	00100000	00001000	10000000

TABLE 4.2 – Conversion binaire des attributs des services dans la table 4.1

La table 4.2 illustre de quelle façon nous allons convertir les attributs des services IaaS pris en compte dans la table 4.1. Pour cela, nous considérons les attributs ECU, HDD, Bw, P_H et P_{BW} comme étant des attributs flottants. D'un autre côté, nous considérons l'attribut *Location* comme étant un attribut à catégories. Et enfin, nous considérons l'attribut restant (RAM) comme étant un attribut entier [1].

4.2.3 Calcul de la valeur de la courbe Z

Afin d'indexer les services, nous devons concaténer les bits des représentations binaires en alternant les digits. La valeur de la dite fonction de Morton sera simplement la représentation en décimale du nombre binaire résultant de la concaténation.

1. Nous convertissons un attribut entier simplement par concaténation de restes de division successives comme démontré dans les travaux [142, 81] et implémenté dans la classe JAVA [128].

Service	Représentation binaire	Valeur de courbe Z
1	000000010000000000000101	65 540
2	000000000000000010101000	168
3	000000000000000111000000	448
4	000000000000101010000000	2688
5	000010000101000000000000	544768
6	000100001000000000000010	1081344
7	100001000000000000010000	8650752
8	001000100000010000000000	2229248

TABLE 4.3 – Valeur de la courbe de Morton pour les services dans la table 4.1

La table 4.3 représente les valeurs de la courbe de Morton relatives aux services décrits dans la table 4.1. Afin d'indexer les instances cloud computing, il faut simplement les insérer dans un arbre AVL selon leurs valeurs de courbe Z. Nous rappelons que la valeur de la courbe de Morton n'est que la conversion en décimal de la représentation binaire dans la table 4.3.

4.2.4 Construction de l'arbre AVL

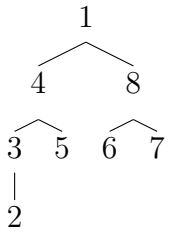


FIGURE 4.1 – Arbre AVL des services de la table 4.1

La figure 4.1 illustre l'arbre résultant de l'indexation des services de la table 4.1 selon la méthode présentée dans la figure 3.5. La génération complète de l'arbre AVL marque la fin de la phase d'indexation des instances publiques cloud computing.

4.3 Recherche

Après la construction d'un arbre AVL contenant l'ensemble des services publiques en nuage d'un niveau IaaS, nous allons consulter la source de données afin d'extraire les préférences des clients et les pré-requis des réseaux sans fil.

4.3.1 Réseaux de capteurs sans fil

RCSF	ECU GHz	RAM GB	HDD GB	Débit Ko/s/user	T [H]	B [$]	N_{user}	N_{client}
1	1.0	1.0	20	$3 * 10^{-5}$	720	601	1	1
2	0.2	0.1	0.4	$2 * 10^{-6}$	8640	942	10	100
3	0.175	0.150	0.0015	$8 * 10^{-7}$	4320	3603	100	100000
4	1.0	1.0	1.0	1.0	720	14	1000	10^8

TABLE 4.4 – Réseaux de capteurs sans fil pour l'exemple

La table 4.4 présente l'ensemble des réseaux de capteurs sans fil considérés pour l'exemple en cours. Il faut noter que les unités et les symboles utilisés ont tous été définis dans la sous-section 3.2.1. D'un autre côté, nous supposons pour des raisons de simplicité que tous les réseaux cités dans la table 4.4 sont localisés à Tlemcen.

4.3.2 Validation

Après avoir pris connaissance des réseaux de capteurs sans fil existants, nous avons besoin de savoir si les instances CC existantes sont effectivement compatibles avec les capacités des clients (des RCSFs). Pour cela, chaque réseau de capteurs sans fil a besoin de faire passer à tous les services IaaS un test de validité comme expliqué dans la sous-section 3.4.2.

Service	1	2	3	4	5	6	7	8
1	Oui	Oui	Oui	Oui	Oui	Oui	Oui	Oui
2	Oui	Oui	Oui	Oui	Non	Non	Non	Non
3	Non	Non	Non	Non	Oui	Oui	Oui	Oui
4	Non	Non	Non	Non	Non	Non	Non	Non

TABLE 4.5 – Validation des services dans la table 4.1 par rapport aux réseaux de capteurs sans fil dans la table 4.4

La table 4.5 présente les résultats du processus de validation de chaque service pris en compte pour l'exemple dans la table 4.1 par rapport à chaque RCSF considéré pour l'exemple dans la table 4.4. Un tel procédé est précédemment expliqué dans la sous-section 3.4.2 et implémenté dans la classe JAVA [129].

4.3.3 Recherche dans l'arbre AVL

Après avoir vérifié la validité des services CC IaaS par rapport aux réseaux de capteurs sans fil, nous pouvons rechercher les services dans l'index qui est sous forme d'arbre AVL. Avant cela, nous allons d'abord calculer les bornes pour chaque RCSF.

4.3.3.1 Calcul des bornes

On rappelle qu'on calcule les bornes afin de ne pas chercher dans tout l'arbre et ne s'occuper que d'une partie. Ceci a pour but de réduire encore le temps nécessaire à la

recherche comme mentionné dans la sous-section 3.4.1.

Service	ECU GHz	RAM GB	HDD GB	BW GB	P_H \$/h	P_{BW} \$/GB	Location
1	0001.00	0001	20	0	0	0	Zurich
2	0002.00	0001	40	0	0	0	Zurich
3	0017.50	0015	150	0	0	0	Zurich
4	1000.00	1000	100000000	0	0	0	Zurich

TABLE 4.6 – Borne inférieure pour les RCSFs de la table

Service	ECU GHz	RAM GB	HDD GB	BW GB	P_H \$/h	P_{BW} \$/GB	Location
1	44	64	640	5120	0.835	7.73	Frankfurt
2	44	64	640	5120	0.11	15.15	Frankfurt
3	44	64	640	5120	0.835	289.60	Frankfurt
4	44	64	640	5120	0.02	$5.42 * 10^{-6}$	Frankfurt

TABLE 4.7 – Borne supérieure pour les RCSFs de la table

Les deux tables 4.6 et 4.7 montrent les résultats du calcul des bornes inférieures et supérieures pour chaque réseau à capteur sans fil dans la table 4.4. Pour cela, nous appliquons le procédé expliqué dans la table 3.2.

Conversion en binaire des attributs des bornes Après avoir calculé les bornes, il faut calculer leurs valeurs de courbe de Morton afin de les situer dans l'arbre. Pour cela, nous appliquons le procédé démontré dans la figure 3.5 (exactement de la même sorte que le procédé utilisé pour les attributs des services CC).

Réseau	Borne inférieure			Borne supérieure		
	P_H \$/h	P_{BW} \$/GB	Location	P_H \$/h	P_{BW} \$/GB	Location
1	1	1	1	10000000	10000000	10000000
2	1	1	1	00010000	10000000	10000000
3	1	1	1	10000000	10000000	10000000
4	1	1	1	00000010	00000001	10000000

TABLE 4.8 – Conversion en binaire des attributs des bornes des 2 tables 4.6 et 4.7 nécessaires au calcul de la fonction de courbe Z

La table 4.8 représente la conversion en binaire des deux bornes précédemment calculées dans les 2 tables 4.6 et 4.7. Il n'est pas nécessaire de convertir tous les attributs, puisque nous ne prenons en considération que les paramètres représentant une limite au client cloud (voir la section 3.3).

Calcul de la valeur de la courbe de Morton pour les bornes Après conversion en binaire des attributs des bornes, il reste à calculer la valeur de la fonction de Morton. Ceci pour pouvoir situer les bornes calculées dans l'arbre AVL ; et réduire ainsi le nombre de nœuds à parcourir pendant la recherche.

Réseau	Borne inférieure		Borne supérieure	
	Binaire	Courbe Z	Binaire	Courbe Z
1	111	7	111000000000000000000000	14680064
2	111	7	011000000100000000000000	6307840
3	111	7	111000000000000000000000	14680064
4	111	7	001000000000000000100010	2097186

TABLE 4.9 – Calcul de la courbe Z pour les 2 bornes pour chaque réseau

La table 4.9 représente les valeurs de Morton relatives aux bornes calculées précédemment dans les tables 4.6 et 4.7. À titre d'exemple, l'algorithme de recherche doit impérativement parcourir tous les nœuds pour le premier réseau. Alors que pour le second, il ne doit pas parcourir le dernier service dans l'arbre AVL qui est le service 7.

Recherche dans l'arbre Comme mentionné dans les travaux [142, 81], nous parcourons l'arbre en profondeur d'abord sans pour autant dépasser les bornes calculées pour chaque réseau à capteurs sans fil.

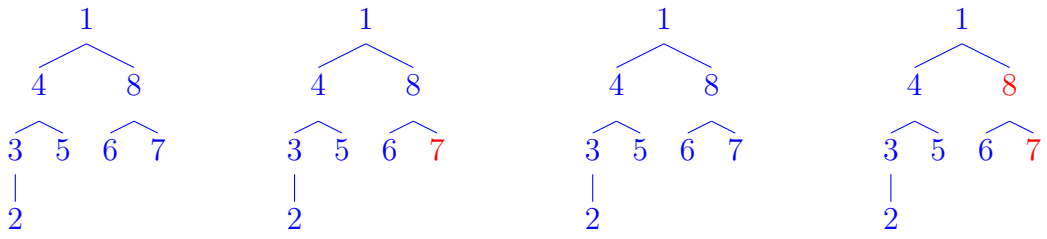


FIGURE 4.2 – Nœuds autorisés dans l'arbres AVL pour chaque réseau

La figure 4.2 démontre clairement l'importance du calcul des bornes inférieure et supérieure précédemment calculées dans les 2 tables 4.6 et 4.7. Comme nous pouvons le constater, les nœuds qui vont être parcourus sont en bleu alors que les autres ne le seront jamais. Réduire un arbre contenant 8 nœuds de 25% n'est certes pas si intéressant que ça. Mais réduire un arbre d'une centaine ou d'un millier de services du même pourcentage est au contraire des plus intéressants.

4.3.4 Administrateurs des RCSFs

Afin de différencier une solution simplement valide d'une autre qui est aussi optimale, nous avons besoin d'un processus de calcul pour un score relatif au service trouvé et jugé valide (voir la sous-section 3.4.3). C'est à ce stade qu'on a besoin des préférences hétérogènes des clients cloud.

Utilisateur	$i_{Av}[\%]$	$i_{RT}[\%]$	$i_{SM}[\%]$
1	15	15	70
2	15	70	15
3	70	15	15

TABLE 4.10 – Préférences des clients cloud pour l'exemple

La table 4.10 représente l'ensemble des consommateurs cloud computing pris en compte dans notre exemple. Tel mentionné dans la sous-section 3.2.2, nous considérons les utilisateurs cloud comme un ensemble de valeurs attribuées à des métriques de qualité. Dans notre cas (que ce soit dans l'exemple ou la formalisation mathématique), nous considérons 3 métriques de qualité distinctes définies dans la sous-section 3.2.2. Par conséquent, le 2^{eme} client cloud par exemple préfère un service répondant rapidement qu'un autre plus disponible ou moins cher.

4.3.4.1 Performances du réseau Internet

L'équation 3.4 (voir la sous-section 3.4.3) ne nécessite pas que les préférences des utilisateurs. Elle requière aussi les performances réseaux pris en compte comme mesures de qualité.

Ville	Temps de réponse [ms]	Disponibilité [%]
Zurich	120	99.950
Washington DC	132	99.990
San Jose	100	99.940
Miami	83	90.000
Honolulu	42	89.000
Perth	56	95.890
Manila	189	80.000
Frankfurt	23	97.990

TABLE 4.11 – Performances Internet pour un client à Tlemcen [7]

La table 4.11 présente l'ensemble des performances réseau telles que le temps de réponse et la disponibilité des services Cloud Computing pour un client installé à Tlemcen. Les valeurs ne vont pas changer pour les réseaux car ils sont tous dans la même location.

4.3.4.2 Résultats de la recherche

Après la validation, le parcours de l'arbre et le calcul du score, nous pouvons afficher les résultats pour un futur traitement.

Utilisateur	Réseau	Services							
		1	2	3	4	5	6	7	8
1	1	3295.432	3475.78	2988.982	2546.074	1848.034	1453.768	3175.642	1400.524
	2	591.2664008	586.17674	504.31324	26.118	non valide	non valide	non valide	non parcouru
	3	non valide	non valide	non valide	non valide	1803,894	-2438.6685	-2629.758	18.2325
	4	non valide	non valide	non valide	non valide	non valide	non valide	non parcouru	non parcouru

TABLE 4.12 – Score des services retournés dans la recherche pour l'utilisateur 1

Utilisateur	Réseau	Services							
		1	2	3	4	5	6	7	8
2	1	3295.432	3475.78	2988.982	2546.074	1848.034	1453.768	3175.642	1400.524
	2	591.2664008	586.17674	504.31324	26.118	non valide	non valide	non valide	non parcouru
	3	non valide	non valide	non valide	non valide	1803,894	-2438.6685	-2629.758	18.2325
	4	non valide	non valide	non valide	non valide	non valide	non valide	non parcouru	non parcouru

TABLE 4.13 – Score des services retournés dans la recherche pour l'utilisateur 2

Utilisateur	Réseau	Services							
		1	2	3	4	5	6	7	8
3	1	3295.432	3475.78	2988.982	2546.074	1848.034	1453.768	3175.642	1400.524
	2	591.2664008	586.17674	504.31324	26.118	non valide	non valide	non valide	non parcouru
	3	non valide	non valide	non valide	non valide	1803,894	-2438.6685	-2629.758	18.2325
	4	non valide	non valide	non valide	non valide	non valide	non valide	non parcouru	non parcouru

TABLE 4.14 – Score des services retournés dans la recherche pour l'utilisateur 3

Les tables 4.12, 4.13 et 4.14 présentent les résultats retournés durant la recherche dans l'arbre AVL pour chacun des RCSFs présents dans la table 4.4. Nous remarquons qu'un service est soit non parcouru à la base, soit non valide, soit compatible et ayant un score. Ce sont les instances du dernier cas qui sont retournées aux utilisateurs cloud.

4.4 Conclusion

Dans ce chapitre, nous avons déroulé un exemple complet pour notre approche. Ceci avait pour but d'approfondir la compréhension du lecteur concernant le processus d'indexation et de recherche avant de passer à l'évaluation et la comparaison de notre contribution à celles que nous avons étendues (ce qui sera fait dans le chapitre suivant).

Chapitre 5

Évaluation

5.1 Introduction

Afin de prouver la performance de notre approche par rapport aux travaux similaires ([142, 81]), nous passons la même batterie de tests sur les 4 travaux dans un scénario unique et préalablement établi. Dans ce chapitre, nous allons d'abord présenter l'environnement dans lequel nous avons dérouler les tests. Ensuite, nous allons décrire dans quelles circonstances nous avons dérouler les tests. Enfin, nous allons présenter les résultats des tests, ainsi que notre interprétation de ces derniers.
Nous avons utilisé un ordinateur de type PC d'un processeur core2duo cadencé à 2.4GHz et d'une mémoire vive de 4Go. La simulation a été déroulé sur Java 64 bits, version 8, mise à jour 25. Le tout s'exécutait sur Windows 10 64 bits.
Dans l'évaluation de nos approches, nous avons considéré 72 services IaaS publiques, 3 métriques de qualité, 3 clients, 4 RCSFs... etc. La table 5.1 contient le reste des valeurs relatives aux paramètres des tests pour la comparaison.

Paramètre	Valeur
Nombre de services IaaS publiques	100
Locations de services IaaS	10
Nombre de métriques de qualité	3
Nombre de clients	10
Nombre de réseaux de capteurs sans fil	10
Nombre de services des réseaux de capteurs sans fil	10
Nombre d'itérations de raffinement	10

TABLE 5.1 – Valeurs des paramètres prises en compte dans l'évaluation

Les valeurs peuvent sembler réduites, mais les algorithmes testés vont devoir s'exécuter à 100 reprises pour faire une seule itération. Puisque nous avons 10 clients et 10 RCSFs, et que les algorithmes testés s'exécutent pour chaque client et pour chaque RCSF, l'évaluation demanderai alors l'exécution des approches testées à $100(10 * 10)$ reprises. Sans parler qu'il faut exécuter le même scénario (avec ses clients, RCSFs et services) à plusieurs itérations afin d'éviter les piques anormaux dans les graphes : nous avons préféré appelé ce paramètre *nombre d'itérations de raffinement*.
Quatre approches sont concernées par l'évaluation et la comparaison, qui sont :

1. l'approche de Sundareswaran & al. [142] (BCloudTree) ;

2. la contribution de Lin & al. [81] (BPlusCloudTree) ;
3. une recherche dans un tableau sans ordre, ni indexation pour les services (algorithme générique) ;
4. notre approche qui est proposée dans cette thèse ;

Afin d'assurer une certaine homogénéité dans les tests, nous avons voulu fixer quelques règles qui sont les suivantes :

— toutes les approches auront les mêmes données concernant les clients (leurs préférences), les services CC IaaS (leurs caractéristiques) et les RCSFs (nœuds et services demandés) ;
— nous intégrons la même fonction de validation à toutes les approches, pour avoir ainsi les mêmes résultats ;
— nous intégrons la même fonction pour calculer le même score à toutes les approches ;

Les mesures prises en compte (citées précédemment) ont pour but de donner le même pied d'égalité aux différentes approches évaluées, afin de se concentrer sur l'ordre donné aux services dans l'arbre d'un côté, et dans les résultats d'un autre côté.

5.2 Cumul des scores en fonction de la durée du test

Durant l'évaluation de notre approche, ainsi que celles de Sundareswaran & al. [142] et Lin & al. [81], nous calculons la somme des scores de tous les services retournés (jugés valides), du début de la simulation jusqu'au moment t. Puis, nous réitérons l'opération jusqu'à la fin de la simulation.

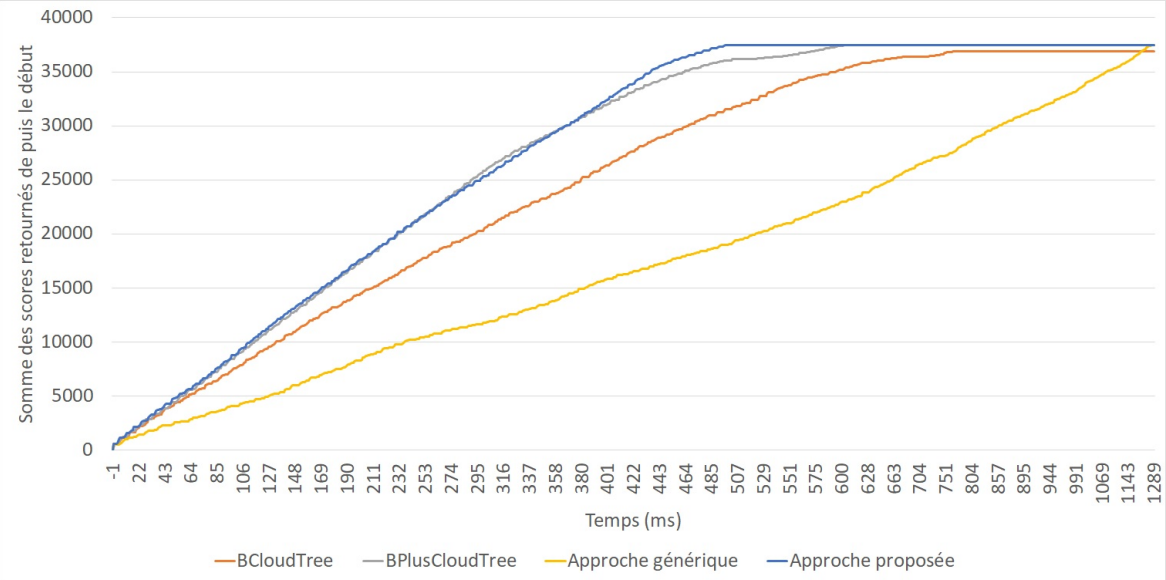


FIGURE 5.1 – Score cumulé selon le temps

La figure 5.1 représente la somme des scores en fonction du temps. Elle comporte quatre courbes de forme logarithmique dans un même repère. Nous remarquons les points suivants :

— les courbes commencent toutes à partir du même point ;
— les 3 courbes bleue, grise et orange dépassent largement la courbe jaune dès le début ;
— les 3 courbes bleue, grise et orange sont relativement proches entre elles par rapport à la courbe jaune ;
— la courbe bleue dépasse les autres vers $380ms$;
— les courbes atteignent toutes la même limite ;
— la courbe bleue atteint sa limite la première vers les $500ms$, elle est suivi par la courbe grise, puis l'orange, et enfin la jaune ;
— les courbes finissent toutes au même point ;

Au tout début de la simulation, aucune approche n'aurait retourné des résultats : c'est pour cela que toutes les approches commencent par le point $(-1, 0)$ [1].
Au cours du déroulement de l'expérience les approches retournent des résultats : ce qui explique les courbes croissantes. Mais les contributions de Sundareswaran & al. [142], de Lin & al. [81], ainsi que la notre retournent les résultats les plus pertinents dès le début, alors que l'approche générique ne fait rien de cela. Ceci se voit par la nature des graphes dans la figure : les trois approches optimisées ont des graphes logarithmiques, alors que l'approche générique prend une apparence linéaire plutôt.
Vu que des mesures ont été prises afin de garantir les mêmes entrées et les mêmes sorties avec les mêmes scores pour les 4 méthodes testées, nous pouvons en déduire que si le un graphe atteint sa limite (qui est la même que les autres), cela voudra dire que son approche a achevé son exécution. De ce fait, on peut conclure que l'approche proposée termine plus tôt que les autres, elle est suivie par celle de Lin & al. [81], puis celle de Sundareswaran & al. [142], et enfin l'approche générique.
L'approche proposée marque sa performance à partir des $380ms$: ceci s'explique par le dépassement de son graphe des autres. En plus de cela, la contribution proposée termine sa recherche pour la totalité des services, clients et RCSFs considérés vers les $500ms$: ceci s'explique par l'arrivée du graphe bleu à sa limite avant les autres.
Puisque les analyses du cumul d'une mesure ne sont pas toujours facilement compréhensibles, nous avons ajouté 3 autres tests aux différentes approches dans les sections à venir.

5.3 Score cumulé par étape

Précédemment, nous avons calculé le cumul des scores des résultats retournés à partir du début de la simulation, jusqu'à l'instant en abscisse. Dans cette section, nous allons nous intéresser au score retourné dans l'étape en cours. En d'autres termes, l'évaluation propose de dérouler le même algorithme de recherche pour chaque client et pour chaque RCSF ; ce qui fait un maximum de 100 services par étape. Dans cette section, nous allons calculer le cumul des scores de ces 100 services, au lieu de faire la somme de tous les scores des services retournés jusqu'à l'instant.

1. Nous n'avons pas commencé par 0 en abscisses, parce que certains travaux évalués retournent des résultats avant le $1ms$, et Java considère le service comme retourné à $0ms$. Nous avons supposé alors que l'expérience débute à l'instant $-1ms$.

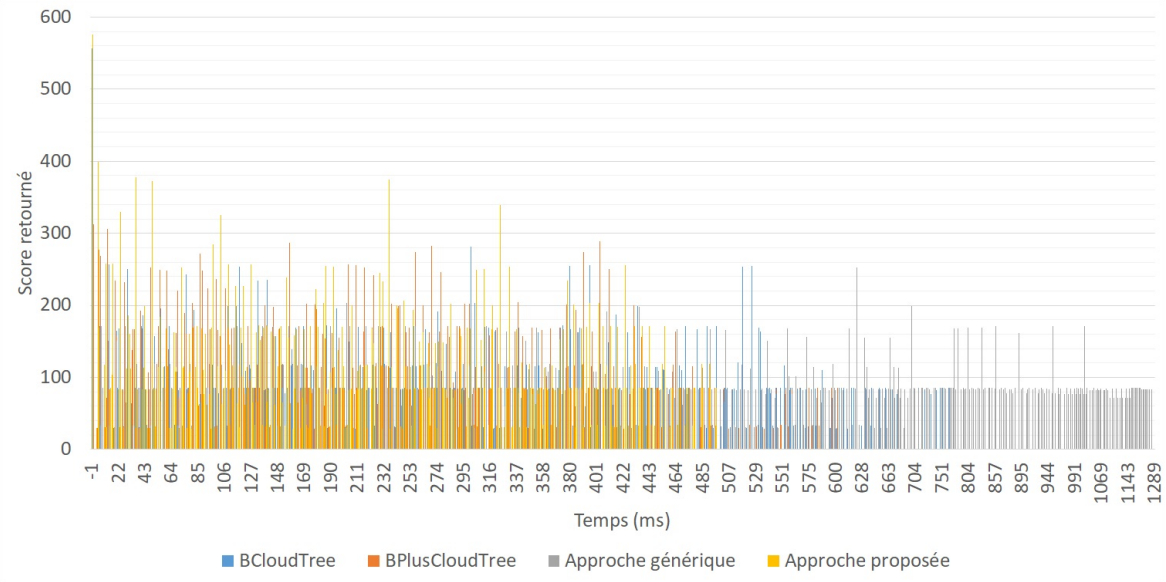


FIGURE 5.2 – Cumul de score par étape en terme de temps

La figure 5.2 présente le cumul des scores des résultats, pour chaque instant en abscisse. Nous remarquons que les graphes (de nature histogramme) sont une succession de pics. Nous notons que chaque hausse représente des résultats intéressants (pertinents, optimaux. . . etc.). Plus le pic est haut, plus les services retournés à l'instant en abscisse sont favorables. On remarque les points suivants :

— un pic jaune important par rapport aux autres dès le début ;
— pour la plupart des cas, le graphe jaune présente des pics plus importants que les autres ;
— le graphe gris présente des pics relativement faibles, et continue à ce rythme jusqu'à la fin ;
— les graphes jaune, bleu et orange présentent des pics importants au début, mais n'en manifestent aucun vers la fin de l'expérience ;

L'approche proposée retourne des résultats intéressants (pertinents) dès le début et continue de le faire jusqu'à l'instant $316ms$: ceci s'explique d'une part par l'importance des pics jaunes par rapport aux autres de différentes couleurs (jusqu'à l'instant $316ms$). D'une autre part, il y'a le pic initiale qui dépasse largement tous ses semblables (quelque le moment en abscisse et quelque soit l'approche).
Concernant l'approche générique, elle ne retourne pas des résultats intéressants, mais garde le même rythme jusqu'à la fin de simulation. Ceci s'explique par la faiblesse des pics gris et leur présence jusqu'à la fin (c'est à dire $1289ms$). Tout cela est dû à l'absence d'ordre des services et d'indexation dans la contribution générique. L'ordre des résultats est donc exactement l'ordre des services en entrée.
Les autres approches par contre ordonnent et indexent les services de façon à ce que les plus importants soient retournés et testés les premiers. D'où l'importance des pics jaune, bleu et orange par rapport à ceux de l'approche générique (pics gris).

5.4 Temps minimal par score

Afin de compléter le test principal (cité en premier), nous proposons dans cette section de calculer le temps minimal par score. Ceci veut dire qu'on va essayer de trou-

ver pour chaque score retourné, le temps minimal nécessaire à chacune des approches, afin de restaurer les premiers résultats (services) ayant le cumul de scores en question.

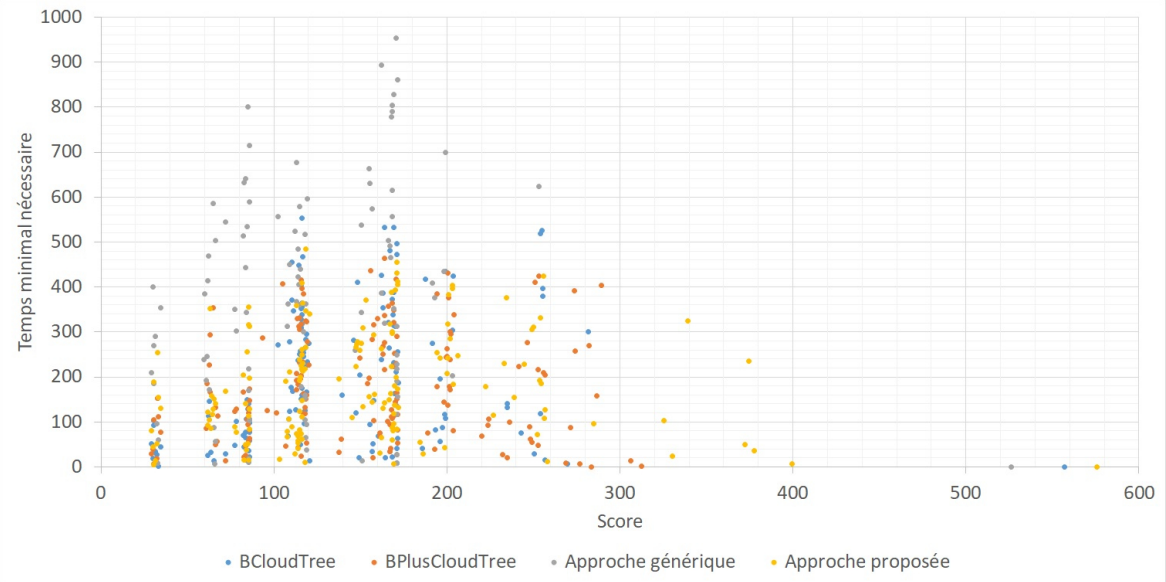


FIGURE 5.3 – Temps minimal par rapport au score

La figure 5.3 représente le temps minimal (en ordonnée) nécessaire pour chaque contribution, afin de restaurer pour la première fois le score indiqué en abscisse. Le graphe est sous forme de nuage de points, et plus les points se dispersent vers le coin bas droit, plus leur approche est meilleure. Nous remarquons les points suivants :

— le point le plus éloigné de l'origine en abscisses est un point jaune ;
— le point le plus haut en ordonnée est un point gris ;
— les points oranges et bleus sont plus centrés que les autres (gris et jaunes) ;
— les points gris se dispersent plus vers le haut ;
— les points jaunes se dispersent plus vers la droite ;

L'algorithme générique prend plus de temps et retourne des résultats moins pertinents par rapport aux autres approches : ceci s'explique par le fait que les points les plus haut sont gris, et que les points gris sont justement pas centrés vers l'origine, mais tendent plutôt vers le haut.
Les approches optimisées (notre contribution, Sundareswaran & al. [142] et Lin & al. [81]) indexent les services et essayent de retourner les résultats les plus pertinents en premier dès le début de la simulation. Ceci s'explique par le fait que les points jaunes, bleus et oranges prennent moins de hauteur que les points gris.
L'approche proposée restent plus performante, puisqu'elle nécessite moins de temps afin de retourner des services ayant des scores très importants. Ceci s'explique par le fait que points jaunes tendent plus vers la droite dans le graphe par rapport aux points oranges et bleus.

5.5 Score maximal par plage de temps

Afin de compléter le test principal (cité en premier), nous proposons dans cette section le score maximal par plage de temps. Ceci n'est qu'une version plus présentable

du graphe 5.2 dans la section 5.3. En d'autres termes, nous allons calculer le score maximal retourné pour chaque plage de temps.

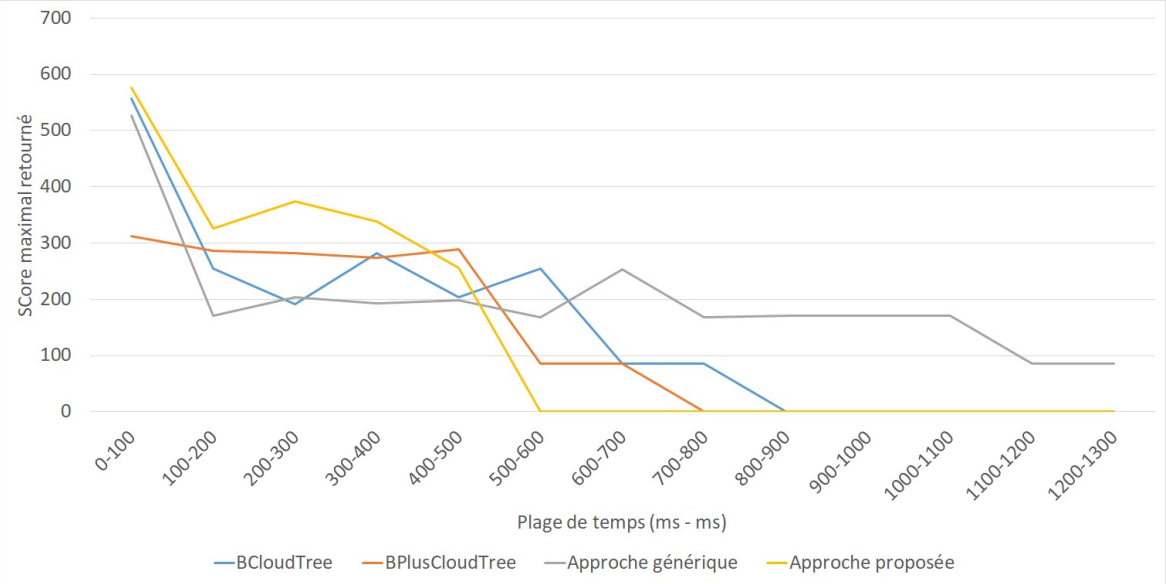


FIGURE 5.4 – Score maximal par plage de temps

La figure 5.4 présente le score cumulé maximal par étape sur des plages d'une période de $100ms$. Par exemple, entre le début de la simulation et l'instant $100ms$, les approches BCloudTree, BPlusCloudTree, l'algorithme générique et contribution proposée ont marqué respectivement un cumul maximal de $556,8448333$; $312,6998333$; $526,2575000$ et $575,9333333$. Par ailleurs, nous remarquons les points suivants :

- les courbes sont toutes descendantes, à l'exception de la courbe grise ;
- durant les 400 premières microsecondes, la courbe jaune surpasse les autres ;
- après les $400ms$, la courbe jaune commence à chuter plus vite que les autres ;
- vers les $600ms$, la courbe jaune touche sa borne inférieure ;
- la courbe orange la suit vers les $800ms$;
- la courbe bleue touche à son tour l'axe des abscisses vers les $900ms$;
- la courbe grise ne touche jamais l'axe des abscisses ;

L'algorithme générique continue sa recherche jusqu'à la fin, alors que les autres approches (optimisées) s'arrêtent entre $500ms$ et $900ms$. Tout ceci s'explique par l'atteinte des courbes jaune, bleue et orange l'axe des abscisses entre les $500ms$ et $900ms$, tandis que la courbe grise ne descend pas en dessous de l'axe ($y = 80$). Ceci est dû au fait que les contributions optimisées retournent les meilleurs résultats dès le début, tandis que l'algorithme générique ne le fait pas.

L'approche proposée retournent des résultats plus pertinents que ceux des approches Sundareswaran & al. [142] et Lin & al. [81]. Ceci s'explique par la position du graphe jaune en dessus des graphes bleu et orange dès le début jusqu'à $400ms$.

Après les $400ms$ jusqu'à $600ms$, la contribution proposée perd de son score pour ses résultats, puisque ses pertinents derniers ont été épuisés. Ceci s'explique par la position du graphe jaune en dessous des autres bleu et orange.

Il faut noter que les services retournés par notre approche dès le début et qui n'ont pas encore été retourné par les autres contributions optimisées seront retournés durant cette période.

5.6 Conclusion

Suite aux différents graphes et courbes explicités précédemment, nous voyons une amélioration notable par rapport aux techniques auxquelles nous nous sommes comparés (particulièrement la méthode de Lin & al. [81], en termes de temps de recherche et de la pertinence des résultats).
Dans ce chapitre, nous avons présenté une comparaison de notre approche proposée dans cette thèse avec les travaux étendues (Sundareswaran & al. [142] et Lin & al. [81]). Les tests se sont portés particulièrement sur le temps nécessaire à la recherche et la pertinence des services retournés.
Nous avons jugés utile de présenter les graphes des scores cumulés (depuis le début, par étape), le temps minimal par score et le score maximal par plage de temps. Ceci pour mettre en avant la pertinence de notre proposition et nous démarquer des autres solutions.

Conclusion générale et perspectives

Aujourd'hui, l'utilisation des RCSFs a inondé tous les domaines aussi bien de la vie courante que de la recherche scientifique : les capteurs médicaux, les détecteurs véhiculaires, les capteurs dans l'agriculture, les détecteurs environnementaux et domotiques. Mais il y a encore plein de verrous à régler qui sont les ressources restreintes et l'alimentation limitée d'où l'idée d'allier les réseaux avec le cloud computing.
L'informatique en nuage va apporter aux RCSFs justement tout l'espace de stockage, les ressources de calcul ainsi que les supports de communication nécessaires et extensibles aux nœuds. Ces dernières années, nous avons vu l'explosion de l'utilisation du cloud computing avec tous les problèmes que cela peut poser pour l'utilisateur. Problème qui se traduit dans la difficulté du choix des services en nuage. Donc, notre intérêt pour cette problématique était évident, et suite à une recherche approfondie de l'état de l'art, nous nous sommes proposés comme objectif principal la sélection et l'indexation des services cloud computing publiques d'un niveau Infrastructure as a Service (IaaS).
Il faut savoir que le cloud computing peut se décliner sous 3 niveaux (IaaS, PaaS et SaaS) et 4 modèles de déploiement (publique, privé, hybride et communautaire). Les chercheurs peuvent se placer du côté client comme du côté fournisseur. Les problèmes traités du cloud computing peuvent se résumer aux points suivants (liste non exhaustive) : composition de services, découverte de services, allocation de ressources, déploiement de services, ordonnancement et planification des exécutions dans un milieu cloud computing et la sélection ainsi que l'indexation des services. Ces derniers sont les 2 points traités dans cette thèse. Étant donné le nombre important des choix offerts au client et la manière dont ils sont indexés, cela peut être ardu pour un consommateur de faire un choix judicieux en toute connaissance de cause.
Comme solution, notre contribution a touché à l'automatisation de cette opération qui va quand même prendre en compte les besoins et les désirs des clients qui peuvent être très hétérogènes.
La deuxième contribution a touché à l'indexation, en changeant la manière dont les attributs étaient affectés pour mettre plus en avant les résultats les plus pertinents. Dans notre état de l'art, nous nous sommes intéressés aux travaux de Sundareswaran & al. [142] et de Lin & al. [81] qui ont utilisé une fonction de Morton afin d'encoder les services. Après, ils utilisent les codes obtenus pour ordonner les services dans des arbres binaires. Ensuite, ils sélectionnent les services selon les requêtes de leurs clients. Dans nos travaux de recherche, nous nous sommes focalisés sur l'extension de ces 2 travaux afin d'améliorer leurs performances en jouant sur 2 points : la sélection des services CC variés et l'indexation pour des clients hétérogènes.

Conclusion générale

Après validation et test de notre contribution, nous avons obtenu les avantages suivants par rapport aux 2 techniques Sundareswaran & al. [142] et de Lin & al. [81] :

- prise en compte des préférences clients malgré leur diversité ;
- prise en compte de l'hétérogénéité des services cloud computing ;
- prise en compte d'une large variété de réseaux de capteurs sans fil allant d'un simple capteur à lui seul à un réseau de surveillance forestière ;
- prise en compte des métriques de qualité générales et publiques : un client cloud n'a pas besoin d'avoir des connaissances approfondies en informatique pour comprendre le temps de réponse d'un service et le taux de disponibilité d'un autre ;
- minimisation du temps de la sélection des services ;
- maximisation des scores des résultats dès le début de l'algorithme de recherche.

Cependant comme tout travail de recherche, notre contribution présente certaines limites qui sont les suivantes :

- les préférences des utilisateurs sont représentées par des poids numériques sans vérifier l'exactitude des chiffres ;
- les choix effectifs des utilisateurs après avoir eu une liste sélective des services correspondants n'est pas prise en compte : nous ne prenons pas en compte les actions et choix des utilisateurs faits après sélection des services (voir [80]).
- supposer que les préférences clients ainsi que la description des services sont exactes. Alors que dans la réalité, c'est tout le contraire.
- supposer que les caractéristiques des instances Cloud Computing n'évoluent pas dans le temps. Or, ce n'est pas vrai dans la réalité : à un instant T, un service en nuage peut être à 120ms de temps de réponse. Après une heure, le temps de réponse peut augmenter (360ms) comme il peut diminuer (48ms).
- le score donné à la fin de la recherche ne représente que le service en lui même sans prendre en considération les autres services dans la source de données. Alors qu'on devrait calculer le score à la base des autres services aussi.

Perspectives

Ces limites peuvent contenir des perspectives et les points suivants contiennent des références pour d'éventuels travaux de recherche :

- comparer plus d'approches similaires (voir [91]) : nous pouvons commencer par les implémenter. Nous pouvons après ajouter dans chacune des approches notre contribution visant à traiter les caractéristiques représentant la capacité des utilisateurs cloud uniquement. Il serait intéressant par la suite de comparer les approches d'origine entre elles ou avec les travaux modifiés.
- calculer les préférences des utilisateurs au lieu de simplement les supposer exister (voir [112, 110, 113]).
- prendre en compte les actions des utilisateurs ainsi que l'historique des choix afin d'affiner plus la recherche et la rendre par conséquent plus efficace (voir [111, 131, 113]).
- essayer une approche collaborative entre les utilisateurs : chaque utilisateur peut recommander un service à un autre. Comme Fan & al. [37], Zou & al. [166], Yin & al. [159] et Meng & al. [96] le démontrent dans leurs travaux, un service

Cloud Computing peut être recommandé par d'autres utilisateurs directement sans passer par un coursier intermédiaire.

— proposer une approche compétitive [124] : les utilisateurs se disputent afin d'allouer leurs ressources dans les meilleures conditions possibles.
— prendre en compte l'incertitude des besoins clients ainsi que les caractéristiques des services [100].
— proposer une approche à base d'apprentissage (voir [150, 148]).
— proposer une approche sur les méthodes évolutionnaires (voir [26]) ou en utilisant les techniques bayésiennes comme le projet CherryPick [11].
— proposer une approche basée sur des agents intelligents (voir [29, 55, 89]).
— proposer une approche cherchant les métriques de qualité à partir d'un service tierce (voir [60]).
— traiter un problème plus général qu'une simple sélection de services dans un cas d'intégration réseaux de capteurs aux nuages informatisés : nous pourrions tout simplement sélectionner les meilleurs nœuds et le meilleur service cloud pour une mission ayant la durée et le budget donnés.
— traiter des clients cloud plus spécifiques que des fournisseurs de réseaux de capteurs sans fil : par exemple, Tamani [144] traite les réseaux véhiculaires sélectionnant des services cloud computing. Dans un tel cas de figure, il faut considérer le temps de réponse et la disponibilité comme étant des requis et pas des métriques de qualité.

Bibliographie

[1] Mysql cloud service. https://cloud.oracle.com/opc/paas/ebooks/Oracle_MySQL_Cloud_Service.pdf, 2016. dernier accès : 01/03/2018.

[2] Tmp23x low-power, high-accuracy analog output temperature sensors. http://www.ti.com/product/TMP235/datasheet, Décembre 2017.

[3] What is salesforce ? https://www.salesforce.com/products/what-is-salesforce/, 2017. dernier accès : 01/03/2018.

[4] Cloud servers hosting. https://www.cloudsigma.com/us/, Janvier 2018. dernier accès : 27/03/2018.

[5] Guide des tarifs. https://www.google.com/drive/pricing/, 2018. dernier accès : 01/03/2018.

[6] Pricing heroku. https://www.heroku.com/what, 2018. dernier accès : 01/03/2018.

[7] Ripe ncc. https://atlas.ripe.net/, Février 2018. dernier accès : 01/03/2018.

[8] What is heroku. https://www.heroku.com/pricing, 2018. dernier accès : 01/03/2018.

[9] Ian F Akyildiz and Ismail H Kasimoglu. Wireless sensor and actor networks : research challenges. *Ad hoc networks*, 2(4) :351–367, 2004.

[10] Hamzeh Mohammd Alabool and Ahmad Kamil Mahmood. Common trust criteria for iaas cloud evaluation and selection. In *Computer and Information Sciences (ICCOINS), 2014 International Conference on*, pages 1–6. IEEE, 2014.

[11] Omid Alipourfard, Hongqiang Harry Liu, Jianshu Chen, Shivaram Venkataraman, Minlan Yu, and Ming Zhang. Cherrypick : Adaptively unearthing the best cloud configurations for big data analytics. In *NSDI*, pages 469–482, 2017.

[12] Arun Anandasivam, Philipp Best, and Simon See. Customers' preferences for infrastructure cloud services. *Proceedings - 12th IEEE International Conference on Commerce and Enterprise Computing, CEC 2010*, pages 144–149, 2010.

[13] Muthusenthil Balasubramanian and Hyunsung Kim. Trust evaluation scheme for cloud data security using fuzzy based approach. *International Journal of Applied Engineering Research*, 12(13) :3908–3913, 2017.

[14] Huihui Bao and Wanchun Dou. A qos-aware service selection method for cloud service composition. In *Parallel and Distributed Processing Symposium Workshops & PhD Forum (IPDPSW), 2012 IEEE 26th International*, pages 2254–2261. IEEE, 2012.

[15] Esha Barlaskar, Peter Kilpatrick, Ivor Spence, and Dimitrios S Nikolopoulos. Myminder : A user-centric decision making framework for inter-cloud migration.

In *Proceedings of the 7th International Conference on Cloud Computing and Services Science (CLOSER)*, pages 560–567, 2017.

[16] Nick Bassiliades, Moisis Symeonidis, Georgios Meditskos, Efstratios Kontopoulos, Panagiotis Gouvas, and Ioannis Vlahavas. A semantic recommendation algorithm for the paasport platform-as-a-service marketplace. *Expert Systems with Applications*, 67 :203–227, 2017.

[17] Punam Bedi, Harmeet Kaur, and Bhavna Gupta. Trustworthy service provider selection in cloud computing environment. In *Communication Systems and Network Technologies (CSNT), 2012 International Conference on*, pages 714–719. IEEE, 2012.

[18] David Bermbach, Jörn Kuhlenkamp, Akon Dey, Arunmoezhi Ramachandran, Alan Fekete, and Stefan Tai. Benchfoundry : A benchmarking framework for cloud storage services. In *International Conference on Service-Oriented Computing*, pages 314–330. Springer, 2017.

[19] Jon Brodkin. Gartner : Seven cloud-computing security risks. *Infoworld*, 2008 :1–3, 2008.

[20] Benjamin Byholm and Ivan Porres. Optimized deployment plans for platform as a service clouds. In *Companion Proceedings of the10th International Conference on Utility and Cloud Computing*, pages 41–46. ACM, 2017.

[21] Yang Cao, Shilong Wang, Ling Kang, and Yuan Gao. A tqcs-based service selection and scheduling strategy in cloud manufacturing. *The International Journal of Advanced Manufacturing Technology*, 82(1-4) :235–251, 2016.

[22] Everton Cavalcante, Thais Batista, Frederico Lopes, Flavia C Delicato, Paulo F Pires, Noemi Rodriguez, Ana Lúcia de Moura, and Reginaldo Mendes. Optimizing services selection in a cloud multiplatform scenario. In *Cloud Computing and Communications (LATINCLOUD), 2012 IEEE Latin America Conference on*, pages 31–36. IEEE, 2012.

[23] Rajanpreet Kaur Chahal and Sarbjeet Singh. Ahp-based ranking of cloud-service providers. In *Information Systems Design and Intelligent Applications*, pages 491–499. Springer, 2016.

[24] Soumyakanti Chakraborty, Sumanta Basu, and Megha Sharma. Pricing infrastructure-as-a-service for online two-sided platform providers. *Journal of Revenue and Pricing Management*, 13(3) :199–223, 2014.

[25] Chen-Tung Chen and Kuan-Hung Lin. A decision-making method based on interval-valued fuzzy sets for cloud service evaluation. In *New Trends in Information Science and Service Science (NISS), 2010 4th International Conference on*, pages 559–564. IEEE, 2010.

[26] Hsin-Kai Chen, Cheng-Yuan Lin, and Jian-Hung Chen. A multi-objective evolutionary approach for cloud service provider selection problems with dynamic demands. In *European Conference on the Applications of Evolutionary Computation*, pages 841–852. Springer, 2014.

[27] Mohan Baruwal Chhetri, Sergei Chichin, Quoc Bao Vo, and Ryszard Kowalczyk. Smart cloud broker : Finding your home in the clouds. In *Automated Software Engineering (ASE), 2013 IEEE/ACM 28th International Conference on*, pages 698–701. IEEE, 2013.

[28] Mohan Baruwal Chhetri, Sergei Chichin, Quoc Bao Vo, and Ryszard Kowalczyk. Smart cloudbench—a framework for evaluating cloud infrastructure performance. *Information Systems Frontiers*, 18(3) :413–428, 2016.

[29] Sergei Chichin, Mohan Baruwal Chhetri, Quoc Bao Vo, Ryszard Kowalczyk, and Marcin Stepniak. Smart cloud marketplace-agent-based platform for trading cloud services. In *Proceedings of the 2014 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)-Volume 03*, pages 388–395. IEEE Computer Society, 2014.

[30] T Crossbow. Telosb mote platform. `http://www.memsic.com/userfiles/files/Datasheets/WSN/telosb_datasheet.pdf`, Janvier 2014.

[31] Waltenegus Dargie and Christian Poellabauer. *Fundamentals of wireless sensor networks : theory and practice*. John Wiley & Sons, 2010.

[32] Christian Davatz, Christian Inzinger, Joel Scheuner, and Philipp Leitner. An approach and case study of cloud instance type selection for multi-tier web applications. In *Proceedings of the 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, pages 534–543. IEEE Press, 2017.

[33] Galen Patrick Deal. *Smart-Transfer : A Cost-Minimized Inter-Service Data Storage and Transfer Scheme*. PhD thesis, 2017.

[34] José Luis Díaz, Joaquín Entrialgo, Manuel García, Javier García, and Daniel Fernando García. Optimal allocation of virtual machines in multi-cloud environments with reserved and on-demand pricing. *Future Generation Computer Systems*, 71 :129–144, 2017.

[35] Cuong T Do, Nguyen H Tran, Eui-Nam Huh, Choong Seon Hong, Dusit Niyato, and Zhu Han. Dynamics of service selection and provider pricing game in heterogeneous cloud market. *Journal of Network and Computer Applications*, 69 :152–165, 2016.

[36] Yehia Elkhatib, Faiza Samreen, and Gordon S Blair. Same same, but different : A descriptive differentiation of intra-cloud iaas services. *arXiv preprint arXiv :1802.03641*, 2018.

[37] Wen-Juan Fan, Shan-Lin Yang, Harry Perros, and Jun Pei. A multi-dimensional trust-aware cloud service selection mechanism based on evidential reasoning approach. *International Journal of Automation and Computing*, 12(2) :208–219, 2015.

[38] Dan Farber. On-demand computing : What are the odds ? `http://www.zdnet.com/article/on-demand-computing-what-are-the-odds/`, November 2002. dernier accès : 01/03/2018.

[39] Soodeh Farokhi. Towards an sla-based service allocation in multi-cloud environments. In *Cluster, Cloud and Grid Computing (CCGrid), 2014 14th IEEE/ACM International Symposium on*, pages 591–594. IEEE, 2014.

[40] Ahmed AA Gad-ElRab, Eman H Zaky, and Neveen I Ghali. An adaptive data mapping storage selection algorithm in mobile cloud computing. *International Journal of Computer Applications*, 143(12), 2016.

[41] Simson Garfinkel. *Architects of the information society : 35 years of the Laboratory for Computer Science at MIT*. MIT press, 1999.

[42] Saurabh Kumar Garg, Steve Versteeg, and Rajkumar Buyya. Smicloud : A Framework for Comparing and Ranking cCoud Services. *Utility and Cloud Computing (UCC), 2011 Fourth IEEE International Conference on*, (Vm) :210–218, 2011.

[43] Saurabh Kumar Garg, Steve Versteeg, and Rajkumar Buyya. A framework for ranking of cloud computing services. *Future Generation Computer Systems*, 29(4) :1012–1023, 2013.

[44] Ioannis Giannakopoulos, Dimitrios Tsoumakos, and Nectarios Koziris. A decision tree based approach towards adaptive profiling of cloud applications. *arXiv preprint arXiv :1704.02855*, 2017.

[45] Misha Goyal and Mehak Aggarwal. Optimize workflow scheduling using hybrid ant colony optimization (aco) & particle swarm optimization (pso) algorithm in cloud environment. *Int. J. Adv. Res. Ideas Innov. Technol*, 3(2), 2017.

[46] Philippe Grange, Céline Ferreira, and Éric Lerouge. *CLOUD COMPUTING*.

[47] Jalel Eddine Hajlaoui, Mohamed Nazih Omri, and Djamal Benslimane. A qos-aware approach for discovering and selecting configurable iaas cloud services. *Computer Systems Science and Engineering*, 2017.

[48] Seung-Min Han, Mohammad Mehedi Hassan, Chang-Woo Yoon, and Eui-Nam Huh. Efficient service recommendation system for cloud computing market. In *Proceedings of the 2nd international conference on interaction sciences : information technology, culture and human*, pages 839–845. ACM, 2009.

[49] Ronny Hans, David Dahlen, Sebastian Zöller, Dieter Schuller, and Ulrich Lampe. Enabling virtual manufacturing enterprises with cloud computing : An analysis of criteria for the selection of database as a service offers. In *Advances in Sustainable and Competitive Manufacturing Systems*, pages 427–438. Springer, 2013.

[50] Samer Hasan and Vatsavayi Valli Kumari. Numerical similarity algorithms for cloud service discovery and selection system.

[51] Brian Hayes. Cloud computing. *Communications of the ACM*, 51(7) :9–11, 2008.

[52] Tai Manh Ho, Nguyen H Tran, SM Ahsan Kazmi, and Choong Seon Hong. Dynamic pricing for resource allocation in wireless network virtualization : A stackelberg game approach. In *Information Networking (ICOIN), 2017 International Conference on*, pages 429–434. IEEE, 2017.

[53] Hao Hu and Jianlin Zhang. The evaluation system for cloud service quality based on servqual. In *Proceedings of the 2012 International Conference on Information Technology and Software Engineering*, pages 577–584. Springer, 2013.

[54] Dhanamma Jagli, Seema Purohit, and N Subash Chandra. Evaluating service customizability of saas on the cloud computing environment.

[55] Arezoo Jahani, Farnaz Derakhshan, and Leyli Mohammad Khanli. Arank : A multi-agent based approach for ranking of cloud computing services. *Scalable Computing : Practice and Experience*, 18(2) :105–116, 2017.

[56] Minkailu Mohamed Jalloh, Shitong Zhu, Fang Fang, and Jun Huang. On selecting composite network-cloud services : a quality-of-service based approach. In *Proceedings of the 2015 Conference on research in adaptive and convergent systems*, pages 242–246. ACM, 2015.

[57] K Jayapriya, N Ani Brown Mary, and RS Rajesh. Cloud service recommendation based on a correlated qos ranking prediction. *Journal of Network and Systems Management*, 24(4) :916–943, 2016.

[58] Ying Jiang, Xin Zhao, and Zhuo Zhao. An improved k-dominance method for qos based service recommendation in cloud. In *Online Analysis and Computing Science (ICOACS), IEEE International Conference of*, pages 20–23. IEEE, 2016.

[59] Dhanwant S Kang, Hua Liu, Munindar P Singh, and Tong Sun. Adaptive process execution in a service cloud : service selection and scheduling based on machine learning. In *Web Services (ICWS), 2013 IEEE 20th International Conference on*, pages 324–331. IEEE, 2013.

[60] Ahmad Karawash, Hamid Mcheick, and Mohamed Dbouk. Quality-of-service data warehouse for the selection of cloud services : a recent trend. In *Cloud Computing*, pages 257–276. Springer, 2014.

[61] Raed Karim, Chen Ding, and Ali Miri. An end-to-end qos mapping approach for cloud service selection. In *Services (SERVICES), 203 IEEE Ninth World Congress on*, pages 341–348. IEEE, 2013.

[62] Raed Karim, Chen Ding, and Ali Miri. End-to-end performance prediction for selecting cloud services solutions. In *Service-Oriented System Engineering (SOSE), 2015 IEEE Symposium on*, pages 69–77. IEEE, 2015.

[63] Raed Karim, Chen Ding, and Ali Miri. End-to-end qos prediction of vertical service composition in the cloud. In *Cloud Computing (CLOUD), 2015 IEEE 8th International Conference on*, pages 229–236. IEEE, 2015.

[64] Raed Karim, Chen Ding, Ali Miri, and Xumin Liu. End-to-end qos mapping and aggregation for selecting cloud services. In *Collaboration Technologies and Systems (CTS), 2014 International Conference on*, pages 515–522. IEEE, 2014.

[65] Chih-Kun Ke, Zheng-Hua Lin, Mei-Yu Wu, and Shih-Fang Chang. An optimal selection approach for a multi-tenancy service based on a sla utility. In *Computer, Consumer and Control (IS3C), 2012 International Symposium on*, pages 410–413. IEEE, 2012.

[66] Poonam P Khot and SD Satav. A profit maximization scheme for enhancing quality of service (qos) in cloud computing. *International Journal of Computer Applications*, 145(11), 2016.

[67] Ravi Khurana and Rajesh Kumar Bawa. Qos based cloud service selection paradigms. In *Cloud System and Big Data Engineering (Confluence), 2016 6th International Conference*, pages 174–179. IEEE, 2016.

[68] Dehua Kong and Yuqing Zhai. Trust based recommendation system in service-oriented cloud computing. In *Cloud and Service Computing (CSC), 2012 International Conference on*, pages 176–179. IEEE, 2012.

[69] Jörn Kuhlenkamp and Markus Klems. Costradamus : A cost-tracing system for cloud-based software services. In *International Conference on Service-Oriented Computing*, pages 657–672. Springer, 2017.

[70] Rabea Kurdi, Maha Aljehani, A Subasi, and SM Qaisar. Cloud computing based healthcare information systems : A proposal for the kingdom of saudi arabia. In *Electrical and Computing Technologies and Applications (ICECTA), 2017 International Conference on*, pages 1–5. IEEE, 2017.

[71] Stine Labes. Towards successful business models of cloud service providers through cooperation-based solutions. 2017.

[72] Sun Le, Hai Dong, Farookh Khadeer Hussain, Omar Khadeer Hussain, Jiangang Ma, and Yanchun Zhang. Multicriteria decision making with fuzziness and criteria interdependence in cloud service selection. In *Fuzzy Systems (FUZZ-IEEE), 2014 IEEE International Conference on*, pages 1929–1936. IEEE, 2014.

[73] Sangwon Lee and Kwang-Kyu Seo. A hybrid multi-criteria decision-making model for a cloud service selection problem using bsc, fuzzy delphi method and fuzzy ahp. *Wireless Personal Communications*, 86(1) :57–75, 2016.

[74] Ang Li, Xiaowei Yang, Srikanth Kandula, and Ming Zhang. Cloudcmp : comparing public cloud providers. In *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement*, pages 1–14. ACM, 2010.

[75] Ang Li, Xiaowei Yang, Srikanth Kandula, and Ming Zhang. Comparing public-cloud providers. *IEEE Internet Computing*, 15(2) :50–53, 2011.

[76] Ang Li, Xiaowei Yang, Ming Zhang, and S Kandula. Cloudcmp : Shopping for a cloud made easy. *HotCloud*, 10 :1–7, 2010.

[77] Ang Li, Xuanran Zong, Srikanth Kandula, Xiaowei Yang, and Ming Zhang. Cloudprophet : towards application performance prediction in cloud. *ACM SIGCOMM Computer Communication Review*, 41(4) :426–427, 2011.

[78] Ang Li, Xuanran Zong, Ming Zhang, Srikanth Kandula, and Xiaowei Yang. Cloudprophet : predicting web application performance in the cloud. *ACM SIGCOMM Poster*, 2011.

[79] Lianhui Li, Jiucheng Hang, Hongxia Sun, and Li Wang. A conjunctive multiple-criteria decision-making approach for cloud service supplier selection of manufacturing enterprise. *Advances in Mechanical Engineering*, 9(3) :1687814016686264, 2017.

[80] Helan Liang, Yanhua Du, and Sujian Li. An improved genetic algorithm for service selection under temporal constraints in cloud computing. In *International Conference on Web Information Systems Engineering*, pages 309–318. Springer, 2013.

[81] Dan Lin, Anna C Squicciarini, Venkata N Dondapati, and Smitha Sundareswaran. A cloud brokerage architecture for efficient cloud service selection. *IEEE Transactions on Services Computing*, 2016.

[82] Li Lin, Tingting Liu, Jian Hu, and Jianbiao Zhang. A privacy-aware cloud service selection method toward data life-cycle. In *Parallel and Distributed Systems (ICPADS), 2014 20th IEEE International Conference on*, pages 752–759. IEEE, 2014.

[83] Yongwen Liu, Moez Esseghir, and Leila Merghem Boulahia. Cloud service selection based on rough set theory. In *Network of the Future (NOF), 2014 International Conference and Workshop on the*, pages 1–6. IEEE, 2014.

[84] Ahmed Lounis, Abdelkrim Hadjidj, Abdelmadjid Bouabdallah, and Yacine Challal. Secure and scalable cloud-based architecture for e-health wireless sensor networks. In *Computer communications and networks (ICCCN), 2012 21st international conference on*, pages 1–7. IEEE, 2012.

[85] Ahmed Lounis, Abdelkrim Hadjidj, Abdelmadjid Bouabdallah, and Yacine Challal. Healing on the cloud : Secure cloud architecture for medical wireless sensor networks. *Future Generation Computer Systems*, 55 :266–277, 2016.

[86] Hua Ma, Zhigang Hu, and Meiling Cai. Trustworthy service selection integrating cloud model and possibility degree ranking of interval numbers. *Chinese Journal of Electronics*, 26(6) :1177–1183, 2017.

[87] Hua Ma, Zhigang Hu, Keqin Li, and Hongyu Zhang. Toward trustworthy cloud service selection : A time-aware approach using interval neutrosophic set. *Journal of Parallel and Distributed Computing*, 96 :75–94, 2016.

[88] Jason Maassen, Niels Drost, Henri E Bal, and Frank J Seinstra. Towards jungle computing with ibis/constellation. In *Proceedings of the 2011 workshop on Dynamic distributed data-intensive applications, programming abstractions, and systems*, pages 7–18. ACM, 2011.

[89] Abid Mahmood, Umar Shoaib, and M Shahzad Sarfraz. A recommendation system for cloud services selection based on intelligent agents. *Indian Journal of Science and Technology*, 11(10), 2018.

[90] Amjad Mahmood and Salman A Khan. Hard real-time task scheduling in cloud computing using an adaptive genetic algorithm. *Computers*, 6(2) :15, 2017.

[91] Amit Kr Mandal, Suvamoy Changder, and Anirban Sarkar. Selection of services for data-centric cloud applications : A qos based approach. In *Advanced Computing, Networking and Security (ADCONS), 2013 2nd International Conference on*, pages 102–107. IEEE, 2013.

[92] Ms Megha, Rani Raigonda, and Ms Jayalaxmi. Cost Effective Data Storage for MultiCloud Architecture with Time Efficiency. 6(7) :1200–1202, 2017.

[93] Peter Mell and Tim Grance. The nist definition of cloud computing. *Communications of the ACM*, 53(6) :50, 2010.

[94] Peter Mell, Tim Grance, et al. The nist definition of cloud computing. 2011.

[95] Peter M Mell and Timothy Grance. Sp 800-145. the nist definition of cloud computing. 2011.

[96] Shunmei Meng, Zuojian Zhou, Taigui Huang, Duanchao Li, Song Wang, Fan Fei, Wenping Wang, and Wanchun Dou. A temporal-aware hybrid collaborative recommendation method for cloud service. In *Web Services (ICWS), 2016 IEEE International Conference on*, pages 252–259. IEEE, 2016.

[97] Selimi Mennan. On the service placement in community network micro-clouds. 2017.

[98] Abdelhak Merizig, Okba Kazar, and Maite Lopez-Sanchez. A dynamic and adaptable service composition architecture in the cloud based on a multi-agent system. *International Journal of Information Technology and Web Engineering (IJITWE)*, 13(1) :50–68, 2018.

[99] Guy M Morton. *A computer oriented geodetic data base and a new technique in file sequencing*. International Business Machines Company New York, 1966.

[100] Bin Mu, Su Li, and Shijin Yuan. Qos-aware cloud service selection based on uncertain user preference. In *International Conference on Rough Sets and Knowledge Technology*, pages 589–600. Springer, 2014.

[101] Bilkisu Larai Muhammad-Bello and Masayoshi Aritsugi. A transparent approach to performance analysis and comparison of infrastructure as a service providers. *Computers & Electrical Engineering*, 2017.

[102] Afshin Naseri and Nima Jafari Navimipour. A new agent-based method for qos-aware cloud service composition using particle swarm optimization algorithm. *Journal of Ambient Intelligence and Humanized Computing*, pages 1–14, 2018.

[103] Shahzad Nizamani and Amirita Kumari. A quality-aware computational cloud service for computational modellers. In *International Multi Topic Conference*, pages 184–194. Springer, 2013.

[104] Aida Omerovic. Supporting cloud service selection with a risk-driven cost-benefit analysis. In *European Conference on Service-Oriented and Cloud Computing*, pages 166–174. Springer, 2015.

[105] Ioannis Patiniotakis, Stamatia Rizou, Yiannis Verginadis, and Gregoris Mentzas. Managing imprecise criteria in cloud service ranking with a fuzzy multi-criteria decision making method. In *European Conference on Service-Oriented and Cloud Computing*, pages 34–48. Springer, 2013.

[106] Michalis Pavlidis, Haralambos Mouratidis, Christos Kalloniatis, Shareeful Islam, and Stefanos Gritzalis. Trustworthy selection of cloud providers based on security and privacy requirements : Justifying trust assumptions. In *International Conference on Trust, Privacy and Security in Digital Business*, pages 185–198. Springer, 2013.

[107] Prasad Pulikal, Claudio Giovanoli, and Stella Gatziu Grivas. Profile based service selection for cloud brokering systems with a focus on saas. In *Enterprise Systems (ES), 2015 International Conference on*, pages 206–213. IEEE, 2015.

[108] Lianyong Qi, Jiguo Yu, and Zhili Zhou. An Invocation Cost Optimization Method for Web Services in Cloud Environment. 2017, 2017.

[109] Hangwei Qian, Hualong Zu, Chenghua Cao, and Qixin Wang. Css : Facilitate the cloud service selection in iaas platforms. In *Collaboration Technologies and Systems (CTS), 2013 International Conference on*, pages 347–354. IEEE, 2013.

[110] Chenhao Qu and Rajkumar Buyya. A cloud trust evaluation system using hierarchical fuzzy inference system for service selection. In *Advanced information networking and applications (aina), 2014 ieee 28th international conference on*, pages 850–857. IEEE, 2014.

[111] Lie Qu, Yan Wang, and Mehmet A Orgun. Cloud service selection based on the aggregation of user feedback and quantitative performance assessment. In *Services computing (scc), 2013 ieee international conference on*, pages 152–159. IEEE, 2013.

[112] Lie Qu, Yan Wang, Mehmet A Orgun, Ling Liu, and Athman Bouguettaya. Context-aware cloud service selection based on comparison and aggregation of user subjective assessment and objective performance assessment. In *Web Services (ICWS), 2014 IEEE International Conference on*, pages 81–88. IEEE, 2014.

[113] Lie Qu, Yan Wang, Mehmet A Orgun, Ling Liu, Huan Liu, and Athman Bouguettaya. Cccloud : Context-aware and credible cloud service selection based on subjective assessment and objective assessment. *IEEE Transactions on Services Computing*, 8(3) :369–383, 2015.

[114] Muhannad Quwaider and Yaser Jararweh. A cloud supported model for efficient community health awareness. *Pervasive and Mobile Computing*, 28 :35–50, 2016.

[115] Imran Mujaddid Rabbani, Aslam Muhammad, and Martinez Enriquez AM. Intelligent cloud service selection using agents. In *The 9th International Conference on Computing and InformationTechnology (IC2IT2013)*, pages 105–114. Springer, 2013.

[116] Constanţa Zoie RĂDULESCU and Iulia Cristina RĂDULESCU. An extended topsis approach for ranking cloud service providers. *Stud. Inform. Control*, 26 :183–192, 2017.

[117] K Hemant Kumar Reddy, Geetika Mudali, and Diptendu Sinha Roy. A novel coordinated resource provisioning approach for cooperative cloud market. *Journal of Cloud Computing*, 6(1) :8, 2017.

[118] Christoph Redl, Ivan Breskovic, Ivona Brandic, and Schahram Dustdar. Automatic sla matching and provider selection in grid and cloud computing markets. In *Proceedings of the 2012 ACM/IEEE 13th International Conference on Grid Computing*, pages 85–94. IEEE Computer Society, 2012.

[119] Zia Ur Rehman, Omar K. Hussain, and Farookh K. Hussain. Iaas cloud selection using MCDM methods. *Proceedings - 9th IEEE International Conference on E-Business Engineering, ICEBE 2012*, pages 246–251, 2012.

[120] Arkaitz Ruiz-Alvarez and Marty Humphrey. An automated approach to cloud storage service selection. In *Proceedings of the 2nd international workshop on Scientific cloud computing*, pages 39–48. ACM, 2011.

[121] Soror Sahri, Rim Moussa, Darrell DE Long, and Salima Benbernou. Dbaas-expert : A recommender for the selection of the right cloud database. In *International Symposium on Methodologies for Intelligent Systems*, pages 315–324. Springer, 2014.

[122] Shadi Samieifar and Farhad Mardukhi. Dynamic Resource Allocation in Cloud Computing Using a Combination of Meta-heuristic Algorithms. 17(4) :332–344, 2017.

[123] M Satheesh. NEGOTIATOR LEARNING AND GROUPING BASED RANKING MODEL IN FEDERATED. 24(2) :10–12, 2017.

[124] Fateh Seghir, Abdellah Khababa, Jaafer Gaber, Abderrahim Chariete, and Pascal Lorenz. A new discrete imperialist competitive algorithm for qos-aware service composition in cloud computing. In *The International Symposium on Intelligent Systems Technologies and Applications*, pages 339–353. Springer, 2016.

[125] Frank J Seinstra, Jason Maassen, Rob V Van Nieuwpoort, Niels Drost, Timo Van Kessel, Ben Van Werkhoven, Jacopo Urbani, Ceriel Jacobs, Thilo Kielmann, and Henri E Bal. Jungle computing : Distributed supercomputing beyond clusters, grids, and clouds. In *Grids, Clouds and Virtualization*, pages 167–197. Springer, 2011.

[126] Ahmed Khalid Yassine Settouti, Fedoua Didi, and Mohammed Haddad. `https://github.com/akysettouti/CCBench/blob/master/CCBench/src/zCurveFunction/ZCurveFloatAttribute.java`, Mars 2017. dernier accès : 27/03/2018.

[127] Ahmed Khalid Yassine Settouti, Fedoua Didi, and Mohammed Haddad. `https://github.com/akysettouti/CCBench/blob/master/CCBench/src/zCurveFunction/ZCurveCategoryAttribute.java`, Mars 2017. dernier accès : 27/03/2018.

[128] Ahmed Khalid Yassine Settouti, Fedoua Didi, and Mohammed Haddad. `https://github.com/akysettouti/CCBench/blob/master/CCBench/src/zCurveFunction/ZCurveNumberAttribute.java`, Mars 2017. dernier accès : 27/03/2018.

[129] Ahmed Khalid Yassine Settouti, Fedoua Didi, and Mohammed Haddad. `https://github.com/akysettouti/CCBench/blob/master/CCBench/src/essential/CCService.java`, Mars 2017. dernier accès : 27/03/2018.

[130] Guojun Sheng, Ying Cao, Yanxia Lu, and Yao Li. A collaborative filtering method for trustworthy cloud service selection. In *Information Science and Control Engineering (ICISCE), 2016 3rd International Conference on*, pages 13–16. IEEE, 2016.

[131] Shree Krishna Shrestha, Yasuo Kudo, Bishnu Prasad Gautam, and Dipesh Shrestha. Recommendation of a cloud service item based on service utilization patterns in jyaguchi. In *Knowledge and Systems Engineering*, pages 121–133. Springer, 2014.

[132] Sarbjeet Singh and Jagpreet Sidhu. Compliance-based multi-dimensional trust evaluation system for determining trustworthiness of cloud service providers. *Future Generation Computer Systems*, 67 :109–132, 2017.

[133] Kazem Sohraby, Daniel Minoli, and Taieb Znati. *Wireless sensor networks : technology, protocols, and applications*. John Wiley & Sons, 2007.

[134] Ramin Soltani, Boulat Bash, Dennis Goeckel, Saikat Guha, and Don Towsley. Covert single-hop communication in a wireless network with distributed artificial noise generation. In *Communication, Control, and Computing (Allerton), 2014 52nd Annual Allerton Conference on*, pages 1078–1085. IEEE, 2014.

[135] Sima Soltani, Patrick Martin, and Khalid Elgazzar. Quaram recommender : Case-based reasoning for iaas service selection. In *Cloud and Autonomic Computing (ICCAC), 2014 International Conference on*, pages 220–226. IEEE, 2014.

[136] Nivethitha Somu, Kannan Kirthivasan, and Shankar Sriram VS. A computational model for ranking cloud service providers using hypergraph based techniques. *Future Generation Computer Systems*, 68 :14–30, 2017.

[137] Jiayi Song and Roch A Guérin. Pricing and bidding strategies for cloud computing spot instances. 2017.

[138] Mohamed Souidi, Sami Souihi, Said Hoceini, and Abdelhamid Mellouk. An adaptive real time mechanism for iaas cloud provider selection based on qoe aspects. In *Communications (ICC), 2015 IEEE International Conference on*, pages 6809–6814. IEEE, 2015.

[139] Giandomenico Spezzano. Using service clustering and self-adaptive mopso-cd for qos-aware cloud service selection. *Procedia Computer Science*, 83 :512–519, 2016.

[140] Le Sun, Jiangang Ma, Yanchun Zhang, Hai Dong, and Farookh Khadeer Hussain. Cloud-fuser : Fuzzy ontology and mcdm based cloud service selection. *Future Generation Computer Systems*, 57 :42–55, 2016.

[141] Mingrui Sun, Tianyi Zang, Xiaofei Xu, and Rongjie Wang. Consumer-centered cloud services selection using ahp. In *Service Sciences (ICSS), 2013 International Conference on*, pages 1–6. IEEE, 2013.

[142] Smitha Sundareswaran, Anna Squicciarini, and Dan Lin. A brokerage-based approach for cloud service selection. In *Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on*, pages 558–565. IEEE, 2012.

[143] Masoumeh Tajvidi, Rajiv Ranjan, Joanna Kolodziej, and Lizhe Wang. Fuzzy cloud service selection framework. In *Cloud Networking (CloudNet), 2014 IEEE 3rd International Conference on*, pages 443–448. IEEE, 2014.

[144] Nouredine Tamani, Bouziane Brik, Nasreddine Lagraa, and Yacine Ghamri-Doudane. Vehicular cloud service provider selection : A flexible approach. In *GLOBECOM 2017-2017 IEEE Global Communications Conference*, pages 1–6. IEEE, 2017.

[145] Mingdong Tang, Xiaoling Dai, Jianxun Liu, and Jinjun Chen. Towards a trust evaluation middleware for cloud service selection. *Future Generation Computer Systems*, 74 :302–312, 2017.

[146] Songkran Totiya and Twittie Senivongse. Framework to support cloud service selection based on service measurement index. In *Proceedings of the World Congress on Engineering and Computer Science*, volume 1, 2017.

[147] Nitin Upadhyay. Managing cloud service evaluation and selection. *Procedia Computer Science*, 122 :1061–1068, 2017.

[148] Zia ur Rehman, Omar Khadeer Hussain, and Farookh Khadeer Hussain. Multi-criteria iaas service selection based on qos history. In *Advanced Information Networking and Applications (AINA), 2013 IEEE 27th International Conference on*, pages 1129–1135. IEEE, 2013.

[149] Lifeng Wang and Zhengping Wu. A trustworthiness evaluation framework in cloud computing for service selection. In *Cloud Computing Technology and Science (CloudCom), 2014 IEEE 6th International Conference on*, pages 101–106. IEEE, 2014.

[150] Xiaogang Wang, Jian Cao, and Yang Xiang. Dynamic cloud service selection using an adaptive learning mechanism in multi-cloud computing. *Journal of Systems and Software*, 100 :195–210, 2015.

[151] Mark Weiser. Hot topics-ubiquitous computing. *Computer*, 26(10) :71–72, 1993.

[152] Mark Weiser. The world is not a desktop. *interactions*, 1(1) :7–8, 1994.

[153] Mark Weiser. Ubiquitous computing. `http://www.ubiq.com/hypertext/weiser/UbiHome.html`, Mars 1996. dernier accès :.

[154] Haijiang Wu, Dan Ye, Shanshan Liu, Yan Yang, and Lin Bai. A service selection approach in cloud manufacturing for smes. In *Enterprise Interoperability VI*, pages 333–343. Springer, 2014.

[155] Xiaofei Xu, Zhizhong Liu, Zhongjie Wang, Quan Z Sheng, Jian Yu, and Xianzhi Wang. S-abc : A paradigm of service domain-oriented artificial bee colony algorithms for service selection and composition. *Future Generation Computer Systems*, 68 :304–319, 2017.

[156] Neeraj Yadav and Major Singh Goraya. Two-way ranking based service mapping in cloud environment. *Future Generation Computer Systems*, 81 :53–66, 2018.

[157] Letian Yang, Li Liu, and Qi Fan. A survey of user preferences oriented service selection and deployment in multi-cloud environment. In *Parallel and Distributed Computing, Applications and Technologies (PDCAT), 2017 18th International Conference on*, pages 354–359. IEEE, 2017.

[158] Yichao Yang, Yanbo Zhou, Lei Liang, Dan He, and Zhili Sun. A sevice-oriented broker for bulk data transfer in cloud computing. In *Grid and Cooperative Computing (GCC), 2010 9th International Conference on*, pages 264–269. IEEE, 2010.

[159] Yuyu Yin, Yueshen Xu, Wenting Xu, Min Gao, Lifeng Yu, and Yujie Pei. Collaborative service selection via ensemble learning in mixed mobile network environments. *Entropy*, 19(7) :358, 2017.

[160] Ruozhou Yu, Xudong Yang, Jun Huang, Qiang Duan, Yan Ma, and Yoshiaki Tanaka. Qos-aware service selection in virtualization-based cloud computing. In *Network Operations and Management Symposium (APNOMS), 2012 14th Asia-Pacific*, pages 1–8. IEEE, 2012.

[161] Maciej Zaremba, Sami Bhiri, Tomas Vitvar, and Manfred Hauswirth. Matchmaking of iaas cloud computing offers leveraging linked data. In *Proceedings of the 28th Annual ACM Symposium on Applied Computing*, pages 383–388. ACM, 2013.

[162] Wenying Zeng, Yuelong Zhao, and Junwei Zeng. Cloud service and service selection algorithm research. In *Proceedings of the first ACM/SIGEVO Summit on Genetic and Evolutionary Computation*, pages 1045–1048. ACM, 2009.

[163] Yuchao Zhang, Hongfu Liu, Bo Deng, and Fuyang Peng. A reliable qoe-aware framework for cloud service monitoring and ranking. In *Proceedings of the 2013 International Conference on Electrical and Information Technologies for Rail Transportation (EITRT2013)-Volume II*, pages 401–409. Springer, 2014.

[164] Liang Zhao, Anna Liu, and Jacky Keung. Evaluating cloud platform architecture with the care framework. In *Software Engineering Conference (APSEC), 2010 17th Asia Pacific*, pages 60–69. IEEE, 2010.

[165] Qazi Zia Ullah, Shahzad Hassan, and Gul Muhammad Khan. Adaptive resource utilization prediction system for infrastructure as a service cloud. *Computational intelligence and neuroscience*, 2017, 2017.

[166] Guobing Zou, Yanglan Gan, Jianxing Zheng, and Bofeng Zhang. Service composition and user modeling for personalized recommendation in cloud computing. In *Computing, Communication and Networking Technologies (ICCCNT), 2014 International Conference on*, pages 1–7. IEEE, 2014.

Résumé

Les réseaux de capteurs sans fil sont de plus en plus utilisés dans le monde actuel. Ils sont aussi bien utiles pour les professionnels, les amateurs, les militaires, les civils, les académiques et les non académiques. Dû à l'excès d'utilisation des nœuds sans fil, les données générées sont de plus en plus importantes. Puisque les capteurs sans fil sont limités en termes de ressources (stockage, traitement et communication), il est utile de les intégrer au cloud computing.

Le nombre des fournisseurs de cloud computing ne cessant de croître, il est nécessaire d'encadrer leurs utilisateurs (propriétaires de réseaux de capteurs sans fil) dans leurs choix du meilleur, afin de traiter et stocker leurs données. Dans cette thèse, nous traitons justement le problème de la sélection du meilleur fournisseur cloud computing en un temps réduit. Pour cela, nous optimisons une approche déjà proposée afin de traiter des clients divers, des services hétérogènes, des réseaux de capteurs sans fil variés…etc. Nous pensons avoir proposé une approche réduisant la durée de la recherche des services et retournant des services plus pertinents dès le début de la recherche.

En d'autres termes, la recherche proposée retourne de meilleures instances cloud computing dès le début, tout en réduisant le temps global pour les chercher. Avec une recherche plus efficace et plus performante, les utilisateurs pourront mieux sélectionner leurs services, économiser leur argent, réduire les ressources pour le faire et explorer plus d'alternatives…etc.

Mots clés : Cloud Computing ; Sélection de services ; Préférences des utilisateurs ; Mesures de qualité ; IaaS publique ; Réseaux de capteurs sans fil ; RCSF ; Evaluation des services ; Indexation ; B^{Cloud}-tree.

Abstract

Wireless sensor networks are more and more used these last decades. They are also useful for professionals, amateurs, militaries, civilians, academics and non-academics. Due to the excessive use of wireless sensor nodes, the generated data become increasingly important. Since wireless sensors are limited in terms of resources like storage, processing and communication. It is consequently very convenient to integrate cloud computing into these data generating networks.

As the number of cloud providers keeps growing, it is necessary to mentor their users and owners of wireless sensor networks in their task of choice of the best in order to process and store their data. In this thesis, we treat the problem of selecting the best cloud-computing provider in a short time. To do so, we optimize an approach already proposed to deal with various customers, heterogeneous services and various wireless sensor networks. We think have proposed an approach giving the best results in a shorter research time.

In other words, the proposed research tends to return better cloud computing instances, while reducing overall time for searching. With more efficient search, users will be able to select better services, save money, reduce resources to do so and explore more alternatives.

Keywords : cloud computing; service selection; user preference; quality measure; public IaaS; wireless sensor networks; WSN; service ranking; indexing; B^{Cloud}-tree.

ملخص

يتم استخدام شبكات الاستشعار اللاسلكية بشكل متزايد في العالم الحالي. كما أنها مفيدة للمهنيين، الهواة، العسكريين، المدنيين، الأكاديميين والغير أكاديميين. نظرًا للإفراط في استخدام العقد اللاسلكية، أصبحت كمية البيانات التي يتم إنشاؤها مهمة. وبما أن المستشعرات اللاسلكية محدودة من حيث الموارد (التخزين، المعالجة والاتصال)، فمن المفيد دمجها في الحوسبة السحابية.

لا يزال عدد من مقدمي سحابة في النمو، فمن الضروري تأطير مستخدميها (أصحاب شبكة استشعار لاسلكية) في اختيارهم للأفضل، قصد معالجة وتخزين البيانات الخاصة بهم. في هذه الدراسة، نتعامل مع مشكلة اختيار أفضل مزود للحوسبة السحابية في وقت قصير. لهذا، نحن نعمل على تحسين النهج المقترح بالفعل للتعامل مع مختلف العملاء، والخدمات المتباينة، وشبكات الاستشعار اللاسلكية المختلفة ... إلخ. نعتقد أننا قد اقترحنا منهاجًا يقلل من وقت البحث عن الخدمات ويعرض خدمات أكثر ملاءمة من بداية البحث.

بعبارة أخرى، يؤدي البحث المقترح إلى عرض حالات حوسبة سحابية أفضل في وقت مبكر، مع تقليل الوقت الإجمالي للبحث. بحث أكثر كفاءة يمكن المستخدمين اختيار أفضل خدمات، الحد من الموارد للقيام، واستكشاف المزيد من البدائل ... الخ.

كلمات مفتاحية: الحوسبة السحابية؛ اختيار الخدمة؛ تفضيل المستخدم؛ قياس الجودة؛ IaaS العامة؛ شبكات الاستشعار اللاسلكية؛ ترتيب الخدمة الفهرسة.