République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université A. Mira de Béjaia

Faculté des Sciences Exactes

Département d'Informatique

MÉMOIRE DE MASTER RECHERCHE

En Informatique

Option : Système d'Information Avancé

Thème

Découverte de services dans l'Internet des Objets

Présenté par :

ABBAS Rania

Soutenu le 02 juillet 2024 devant le jury composé de :

Présidente :	Mme. Khaled Hayette	U. A/Mira Béjaia.
Examinatrice :	Mme. Ait Hacene Souhila	U. A/Mira Béjaia.
Encadrante :	Mme. GASMI Badrina	U. A/Mira Béjaia.
Co-Encadrante :	Mme. AZIZOU Zahia	U. A/Mira Béjaia.

Béjaia, Juillet 2024.

⁕ Remerciements ⁕

Tout d'abord, je rends grâce à ALLAH, qui m'a donné la force, le courage et la patience nécessaires pour accomplir ce travail modeste.

Je tiens à exprimer ma profonde gratitude à mon encadrante, Mme GASMI Badrina. Je vous remercie pour votre encadrement de haute qualité, votre suivi régulier, votre disponibilité, ainsi que pour vos précieux conseils et critiques constructives. Votre aide a été inestimable et a grandement contribué à la réussite de ce travail.

Ma reconnaissance va également à ma co-encadrante, Mme AZIZOU Zahia. Je vous suis extrêmement reconnaissante pour votre soutien constant, votre accompagnement bienveillant et votre expertise. Vos conseils avisés et votre présence attentive ont été essentiels pour mener à bien ce projet.

Je témoigne également d'une profonde reconnaissance envers tous les membres du jury pour avoir accepté d'évaluer, d'examiner et d'enrichir mon travail avec leurs remarques précieuses.

Je tiens à exprimer un remerciement spécial à mon frère, **Omar**, dont le soutien constant m'a accompagné tout au long de ce parcours. Sa générosité et ses encouragements ont été d'une importance primordiale pour moi. Tu as joué un rôle essentiel dans mon chemin et je te serai toujours reconnaissante. Merci infiniment, mon frère.

Enfin, j'exprime ma gratitude sincère à ma sœur **Nassima**, qui a été à mes côtés toute ma vie. Sans elle, je n'aurais jamais atteint ce jour. Sa présence a été un soutien inestimable, et je lui serai redevable pour toujours.

⁂ *Dédicaces* ⁂

À **mes parents**, qui ont partagé avec moi mes échecs et mes succès, pour leur amour inconditionnel, leur soutien sans faille, et leurs sacrifices innombrables qui ont rendu tout cela possible. Vous êtes ma source d'inspiration et de force.

À **mon frère OMAR**, qui a constamment veillé sur moi, ne me laissant jamais manquer de quoi que ce soit dans ma vie. Cette réalisation est pour toi.

À **ma sœur NASSIMA**, qui a fait de moi ce que je suis aujourd'hui, et à son mari **MERZOUK Omar**, dont le soutien a été constant.

À **ANIA**, la fille de ma sœur Nassima, qui a fait irruption dans nos vies et les a rendues plus joyeuses. Tu es la lumière qui éclaire nos vies. **MERCI, ANIA.**

Enfin, **hommage à ma tante MALIKA**, qui nous a quittés trop tôt. Je dédie ce travail à sa mémoire bien-aimée.

Table des matières

Table des figures

Liste des tableaux

Liste des abréviations

ABCA	Artificial Bee Colony Algorithm
ACA	Ant Colony Algorithm
APF	Artificial Potential Field
CERP-IoT	Cluster of European Research Projects on the Internet of Things
DHT	Distributed Hash Table
GWO	Wolf optimisation
HTTP	Hyper Text Transfer Protocol
IoT	Internet of Things
IoV	Internet of Vehicles
IP	Internet Protocol
ISS	IoT Service Store
JSON	JavaScript Object Notation
MDM	Multiple Dimensional Measuring
mDNS	multicast Domain Name System
mDNS-SD	multicast Domain Name System Service Discovery
M2M	Machine to Machine
MQTT	Message Queuing Telemetry Transport
MQTT-RD	MQTT for Resource Discovery
NFC	Near Field Communication
QoDisco	Quality of Service Aware Distributed Query Processing for Linked Data
QoS	Quality of Service

RDF	Resource Description Framework
REST	Representational State Transfer
RFID	Radio Frequency Identification
SD-ABCA	Service Discovery based on Artificial Bee Colony Algorithm
SIoT	Semantic Internet of Things
SOAP	Simple Object Access Protocol
SOA	Service Oriented Architecture
SPARQL	SPARQL Protocol and RDF Query Language
SPF	Service Provisioning Function
UDDI	Universal Description, Discovery, and Integration
URL	Uniform Resource Locator
URI	Uniform Resource Locator
Wi-Fi	Wireless Fidelity
WoT	Web of Things
WSN	Wireless Sensor Network
XML	Extensible Markup Language

Introduction générale

Durant les décennies passées, les avancées en technologies de communication et de mise en réseau ont permis le développement de nombreuses structures de réseaux d'information. Les dispositifs électroniques, comme les capteurs, les actionneurs et les RFID (Radio Frequency Identification), sont désormais intégrés dans presque toutes les entités physiques, permettant la communication et la collaboration entre elles. Ce phénomène est connu sous le nom de l'Intenet des Objets (Internet of Things (IoT)).

L'IoT a conduit à une augmentation rapide du nombre d'appareils connectés, Avec une estimation de 125 milliards d'objets connectés d'ici 2030 selon les prévisions d'IHS Markit. Chacun de ces objets offrant un ou plusieurs services, appelés services IoT. Un service est une entité qui fournit un ensemble de fonctionnalités telles que l'accès, la recherche, le traitement de l'information et la communication avec d'autres utilisateurs ou applications. Dans un environnement IoT, les utilisateurs sont entourés d'une multitude de services offerts par leurs objets connectés. Cependant, la découverte et la localisation de ces services deviennent de plus en plus complexes en raison du grand nombre d'objets connectés, de leur mobilité, de leur hétérogénéité et de leurs limitations en termes de ressources.

La découverte des services désigne le processus par lequel les utilisateurs ou les consommateurs de services recherchent des prestations qui répondent à leurs besoins. Les techniques de découverte centralisées se révèlent souvent insuffisantes pour répondre aux exigences croissantes des utilisateurs dans des environnements décentralisés, dynamiques et ouverts. Avec la demande croissante de services IoT, il devient essentiel d'avoir une exploration dynamique, automatique et adaptative des services dans un environnement évolutif, hétérogène, décentralisé et ouvert, où les fournisseurs de services peuvent entrer et sortir de manière dynamique.

Un mécanisme de découverte efficace doit répondre à plusieurs exigences. Il doit être capable de retourner les services qui correspondent à la requête de l'utilisateur dans un environnement où le volume de services IoT est important. Il doit également être capable de s'adapter à une mise à l'échelle des services et des objets dans un environnement IoT composé de milliards d'objets connectés. La question qui se pose alors est : comment concevoir un mécanisme de recherche qui soit à la fois efficace et capable de répondre à ces exigences tout en tenant compte des contraintes de l'environnement IoT ?

La découverte de services est l'un des défis majeurs de l'IoT qui est souvent considérée comme un problème d'optimisation. Pour relever ce défi, nous avons choisi d'utiliser des méta-heuristiques, une famille d'algorithmes inspirés de la nature, qui sont particulièrement efficaces pour résoudre des problèmes où les algorithmes d'optimisation classiques ne parviennent pas à produire des résultats satisfaisants.

Ce mémoire est organisé de manière à explorer en profondeur les défis et les solutions relatifs à la découverte et à la localisation des services dans l'Intenet des Objets, en se concentrant sur trois chapitres pour traiter chaque aspect spécifique de cette problématique émergente.

Le premier chapitre examine les concepts fondamentaux de l'Intenet des Objets et des méthodes d'optimisation. Nous discutons des bases de l'IoT, de ses applications variées, et des méthodes utilisées pour résoudre les problèmes d'optimisation. Cette section pose les fondations nécessaires à la compréhension des défis ultérieurs en matière de découverte de services IoT.

Dans le deuxième chapitre, nous nous concentrons sur la découverte de services dans l'Intenet des Objets, en explorant différentes architectures et approches disponibles pour identifier et localiser efficacement les services offerts par les objets connectés. Nous examinons notamment l'Architecture Orientée Services (SOA), les principes de REST (Representational State Transfer), et les divers modèles de découverte de services adaptés aux environnements IoT.

Enfin, le troisième chapitre présente notre contribution à travers l'algorithme d'optimisation par colonie d'abeilles artificielles (Service Discovery based on Artificial Bee Colony Algorithm : SD-ABCA) pour améliorer la découverte de services dans l'IoT. Nous détaillons l'adaptation spécifique de cet algorithme, ses principes et son évaluation comparative avec l'algorithme de flooding. Le choix de comparer SD-ABCA avec le flooding s'explique par le fait que le flooding est une méthode de base couramment utilisée dans les réseaux ad hoc pour la découverte de services, malgré ses limitations connues. Nos simulations ont démontré que l'efficacité de la découverte de services à l'aide de SD-ABCA est comparable à celle de l'algorithme de flooding en termes de taux de succès et de nombre de sauts nécessaires. Cependant, notre approche présente un avantage significatif : elle ne nécessite pas de table de routage, mais seulement la maintenance d'une liste de voisins. Cela se traduit par une réduction du nombre de messages circulant dans le système. En revanche, l'approche de flooding repose sur une stratégie d'inondation, ce qui n'est pas idéal pour les objets dotés de ressources limitées. De plus, notre modèle décentralisé facilite une mise à l'échelle plus aisée vers des environnements plus vastes.

Chapitre 1

Concepts fondamentaux

1.1 Introduction

L'Internet a transformé les échanges en interconnectant individus, machines et objets physiques, donnant ainsi naissance à l'Intenet des Objets, ouvrant ainsi de nouvelles perspectives dans le domaine de la révolution numérique. Ce premier chapitre se consacre à l'introduction des concepts fondamentaux de l'IoT et à l'exploration des méthodes de résolution des problèmes d'optimisation.

Nous commencerons ce chapitre par une analyse approfondie de l'Intenet des Objets, en définissant ce concept, en explorant son fonctionnement ainsi que les trois couches qui le composent. Nous examinerons également ses divers domaines d'application ainsi que les défis associés. Ensuite, nous traiterons des méthodes de résolution des problèmes d'optimisation. Nous aborderons spécifiquement les problèmes d'optimisation combinatoire, en distinguant les méthodes exactes et les méthodes approchées.

1.2 Intenet des Objets

Le domaine de l'Intenet des Objets marque une convergence révolutionnaire entre la technologie numérique et les capacités physiques, où des objets physiques sont intégrés dans des réseaux informatiques afin de collecter, échanger et utiliser des données de manière interactive.

1.2.1 Définition de l'Intenet des Objets

Le CERP-IoT « Cluster des projets européens de recherche sur l'Intenet des Objets » définit l'Intenet des Objets comme : « une infrastructure dynamique d'un réseau global. Ce réseau global a des capacités d'auto-configuration basée sur des standards et des protocoles de communication interopérables. Dans ce réseau, les objets physiques et virtuels ont des identités, des attributs physiques, des personnalités virtuelles et des interfaces intelligentes, et ils sont intégrés au réseau d'une façon transparente » [1].

Cette vision de l'Intenet des Objets introduira une nouvelle dimension aux technologies de l'information et de la communication : en plus des deux dimensions temporelle et spatiale qui permettent aux personnes de se connecter de n'importe où à n'importe quel moment, nous aurons une nouvelle dimension « objet » qui leur permettra de se connecter à n'importe quel objet [2], comme illustré dans la Figure 1.1 [3].

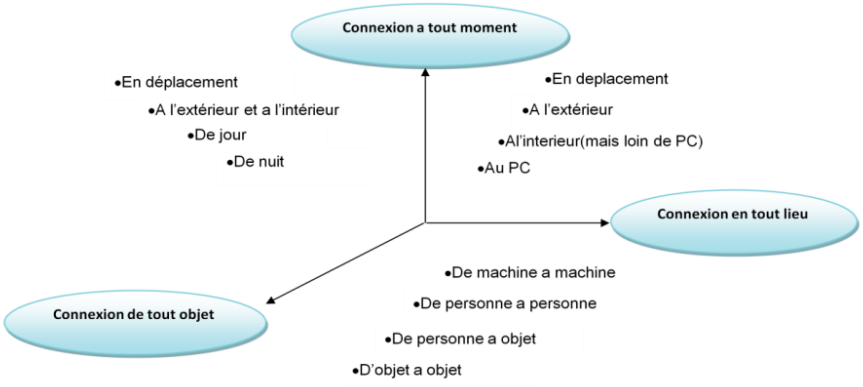

FIGURE 1.1 – Une nouvelle dimension pour l'IoT

1.2.2 Fonctionnement de l'IoT

L'IoT relie les objets à Internet pour partager des données, exploitant des technologies comme la RFID, les réseaux de capteurs sans fil (WSN) et la communication machine à machine (M2M).

RFID (Radio Frequency Identification)

L'identification par radiofréquence est une technologie de communication sans fil qui utilise une fréquence radio pour identifier de manière unique un objet, un animal ou une

personne sans nécessiter de batterie dans les étiquettes. Le système RFID peut être divisé en deux composants principaux : le transpondeur de signal radio (étiquette) et le lecteur d'étiquette [4].

L'étiquette se compose de deux éléments : une puce pour stocker l'identité unique de l'objet et une antenne pour permettre à la puce de communiquer avec le lecteur d'étiquette en utilisant des ondes radio. Ensuite, l'étiquette activée renvoie les données à l'antenne. Ces données déclenchent le contrôleur logique programmable pour effectuer une action particulière. Les étiquettes RFID peuvent être actives (alimentées par une batterie), passives (n'ont pas besoin de batterie) ou semi-passives/actives (qui utilisent l'énergie du circuit intégré lorsque nécessaire) [4].

WSN (Wireless Sensor Network)

C'est un ensemble de nœuds qui communiquent sans fil et qui sont organisés en un réseau coopératif. Chaque nœud possède une capacité de traitement et peut contenir différents types de mémoires, un émetteur-récepteur et une source d'alimentation, comme il peut aussi tenir compte des divers capteurs et des actionneurs. Comme son nom l'indique, le WSN constitue alors un réseau de capteurs sans fil qui peut être une technologie nécessaire au fonctionnement de l'IoT [5].

M2M (Machine to Machine)

C'est « l'association des technologies de l'information et de la communication avec des objets intelligents dans le but de donner à ces derniers les moyens d'interagir sans intervention humaine avec le système d'information d'une organisation ou d'une entreprise » [6].

1.2.3 Couches d'un modèle IoT

L'IoT, bien qu'étudié depuis plus d'une décennie, manque encore d'une architecture standardisée. Cependant, une architecture à trois couches (perception, réseau et application) est communément acceptée, comme illustré dans la Figure 1.2 [7].

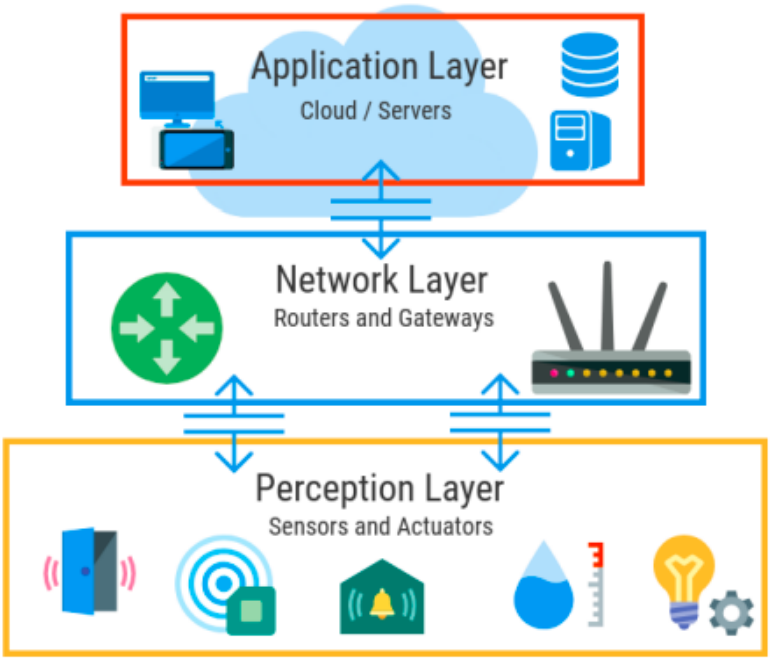

FIGURE 1.2 – Architecture en trois couches

Couche de perception

La fonction principale de la couche de perception consiste à identifier les propriétés physiques telles que la température, l'humidité, le niveau de lumière, la vitesse, etc. À l'aide de divers dispositifs de détection, puis à convertir ces informations en signaux numériques. Les objets de cette couche peuvent posséder des capacités de détection et/ou d'actionnement (Un actionneur est un dispositif capable de recevoir des commandes programmées et d'effectuer des tâches à des moments spécifiques).

Couche réseau

La couche de réseau est chargée de transmettre les données collectées par la couche de perception vers une base de données, un serveur ou un centre de traitement. Les principales technologies utilisées pour cette couche sont les technologies cellulaires 2G/3G/LTE, le Wi-Fi, le Bluetooth, le Zigbee ou l'Ethernet. Grâce à ces différentes technologies, il est possible de traiter les données de plusieurs objets qui seront connectés à l'avenir.

Couche application

La couche d'application reçoit les données transmises depuis la couche réseau et les utilise pour fournir des services ou des opérations. Elle peut offrir un service de stockage pour enregistrer les données reçues dans une base de données, ou un service d'analyse pour évaluer ces données en vue de prédire l'état futur des dispositifs physiques. Plusieurs applications existent dans cette couche, chacune ayant des exigences spécifiques. Par exemple, le transport intelligent et les villes intelligentes sont des applications typiques de cette couche.

1.2.4 Domaines d'application de l'IoT

Les domaines d'application de l'IoT sont très nombreux, et touchent pratiquement tous les axes de la vie quotidienne des individus, parmi ces domaines, nous citons quelques exemples comme illustré dans la Figure 1.3 [8] : santé, domotique, villes intelligentes, énergie et transport.

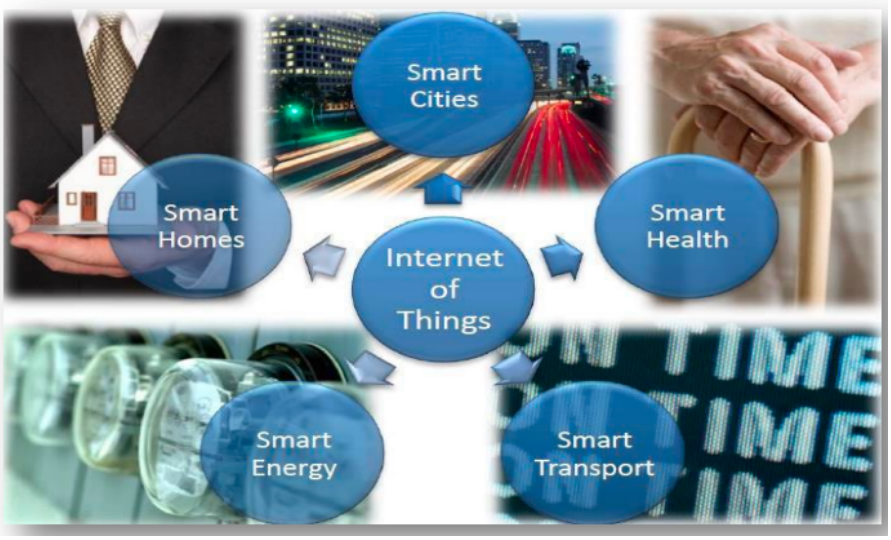

FIGURE 1.3 – Domaines d'application de l'IoT

Santé (Smart Health)

Dans le domaine de la santé, l'Intenet des Objets permettra de déployer des réseaux personnels pour le contrôle et le suivi des signes cliniques, notamment pour les personnes âgées. Les objets connectés peuvent surveiller la tension artérielle, le rythme cardiaque, la qualité de la respiration et la masse graisseuse.

Domotique (Smart Home)

La domotique englobe les techniques utilisées dans une habitation pour centraliser le contrôle des différents systèmes de la maison. Son principe est de rendre la maison intelligente, autonome et capable de prendre des décisions par elle-même. Tout cela est rendu possible grâce à l'Intenet des Objets, qui permet de connecter les appareils domestiques à un réseau et de les contrôler à distance. L'objectif de la domotique est d'améliorer le confort quotidien en automatisant ou en gérant à distance les tâches récurrentes.

Villes intelligentes (Smart Cities)

L'Intenet des Objets améliorera la gestion des réseaux variés qui alimentent nos villes (eau, électricité, gaz, etc.) en permettant un contrôle continu, en temps réel et précis. Des capteurs peuvent être utilisés pour économiser l'eau, améliorer la gestion des parkings et du trafic urbain, réduire les embouteillages et diminuer les émissions de CO2.

Énergie (Smart Grid)

L'Intenet des Objets permet aux nombreux appareils du réseau électrique de partager des informations en temps réel, améliorant ainsi l'efficacité de la distribution et de la gestion de l'énergie.

Transport (Smart Transport)

L'Intenet des Objets rend les véhicules et les infrastructures plus intelligents pour améliorer la sécurité routière et réduire les embouteillages. Les systèmes de transport intelligents optimisent la gestion du trafic, économisant ainsi du temps et de l'énergie. Les véhicules dotés de capteurs avancés améliorent la navigation et renforcent la sécurité, tandis que l'IoT permet une intervention rapide en cas d'accident et améliore la logistique grâce au suivi des flottes.

1.2.5 Défis de l'Intenet des Objets

L'Intenet des Objets présente des défis majeurs tels que l'interopérabilité, la sécurité et la confidentialité, la puissance et la consommation d'énergie, la disponibilité, la mobilité, l'hétérogénéité, la scalabilité et la découverte de services. Aborder ces défis est essentiel pour garantir le succès et l'efficacité des systèmes IoT.

Interopérabilité

C'est l'un des plus grands défis de la mise en œuvre de l'IoT. L'interopérabilité est en fait la coexistence de différents dispositifs, systèmes et mécanismes distincts et la possibilité d'interaction et de collaboration entre ces différents dispositifs et systèmes en toute souplesse, quels que soient les équipements et les logiciels exposés. Pour cela, il existe une diversité de standards, de technologies, de régularisation des systèmes et de protocoles utilisés pour enrichir l'IoT.

Sécurité et confidentialité

La sécurité des personnes, des transmissions, des informations, des services, des réseaux et des appareils reste un problème grave qui pourrait avoir un impact négatif sur les systèmes de l'IoT. Des milliers d'appareils restreints et connectés en permanence à Internet et intégrés dans toutes sortes d'objets de notre vie quotidienne, porteront le risque d'être exposés aux menaces usuelles d'Internet. Il est même possible que de nouvelles conceptions d'attaques surgissent.

Puissance et consommation d'énergie

L'énorme quantité de données générées par les capteurs de surveillance de l'environnement urbain affaiblit également le mouvement du réseau, l'enregistrement des informations et la consommation d'énergie. Les technologies d'accumulation d'énergie doivent répondre aux besoins des systèmes de l'IoT, comme la fourniture de sources d'alimentation pour les petits dispositifs intégrés.

Disponibilité

La disponibilité de l'IoT assure que les services sont utilisables pour les usagers finaux partout et à tout moment grâce à l'utilisation de composants matériels et logiciels. La disponibilité des logiciels garantit la capacité de fournir différents services via différentes applications à tous les utilisateurs finaux qui se trouvent à différents endroits en même temps. En simplifiant la gestion de la redondance des dispositifs et des services, une grande disponibilité des services de l'IoT pourrait être acquise et la disponibilité des services est donc l'un des complications majeures qui devraient être résolues pour conduire convenablement la dynamique des systèmes de l'IoT.

Mobilité

La mobilité est un autre défi remarquable dans les systèmes de l'IoT, car la majorité des services devraient être exposés aux usagers mobiles. Relier en permanence les utilisateurs aux services nécessaires durant leurs mouvements est l'une des hypothèses importantes de l'IoT. Les dispositifs de l'IoT peuvent être bougés pour obtenir des services à toute occasion, ils doivent donc modifier leur adresse IP (Internet Protocol) et leurs réseaux tout en tenant compte de leur positionnement. À cet effet, les protocoles de mobilité actuels des réseaux de détecteurs, des réseaux mobiles ad hoc et des réseaux véhiculaires ad hoc ne peuvent pas gérer divers problèmes d'acheminement en conséquence des capacités de traitement et de puissance limitées de ces capteurs. Pour cela, les protocoles d'acheminement devraient être bien créés pour régler ces difficultés.

Hétérogénéité

Les caractéristiques et les aptitudes des dispositifs diffèrent considérablement, ce qui contribue à faire de l'IoT un écosystème riche et hétérogène également.L'Intenet des Objets est constitué d'objets de différents types avec des aptitudes différentes, dépendant à des réseaux de caractère différent et disposés à plusieurs échelles : Équipements, programmes et protocoles de communication. Avec toutes ces formes d'hétérogénéités physiques et technologiques,

il faudrait mettre en place des mécanismes informés aptes de les cacher et de les gérer. L'hétérogénéité des systèmes peut résulter de différences dans les protocoles et les réseaux concernés, de divergences dans la surface de stockage ou les entités matérielles, ou de divergences dans la dislocation d'entités. Au sein d'un même réseau, des postes de travail munis de capacités informatiques hétérogènes et d'un stockage remarquable peuvent coexister avec d'autres périphériques aux ressources limitées. Cette différence exige que le middleware de l'IoT ait un niveau d'abstraction élevé adéquat afin de dissimuler la diversité des modules et la complication de la communication.

Scalabilité

La scalabilité dans l'Intenet des Objets se réfère à sa capacité à maintenir ses fonctionnalités et performances malgré l'augmentation du nombre d'objets et de services connectés. il est nécessaire d'implémenter des mécanismes robustes pour éviter la dégradation des performances et garantir une qualité de service acceptable. Ces mécanismes sont essentiels pour soutenir les fonctions clés telles que la communication et la découverte de services.

Découverte de services

L'Intenet des Objets se distingue par la multitude d'objets connectés, leur mobilité et leur diversité. Ces caractéristiques compliquent la recherche des services associés à ces objets. Pour améliorer la précision et la flexibilité de la recherche de services, il est essentiel de mettre en œuvre des mécanismes exploitant les technologies du Web Sémantique et des méthodes d'optimisation. Ces approches visent à enrichir les descriptions de services et à faciliter leur découverte au sein de l'IoT.

1.3 Méthodes de résolution de problèmes d'optimisation

L'optimisation, introduite dans un souci d'amélioration des services fournis dans divers domaines d'application, vise à identifier la meilleure solution parmi un ensemble de solutions

possibles. Un problème d'optimisation implique l'utilisation de méthodes spécifiques pour rechercher un optimum.

1.3.1 Problèmes d'optimisation

Un problème d'optimisation est défini par un ensemble de variables, une fonction objectif f et un ensemble de contraintes d'inégalité (ou d'égalité) que les variables doivent satisfaire. L'ensemble des solutions possibles du problème forme l'espace de recherche E, où chaque dimension correspond à une variable. L'espace de recherche E est fini (le décideur précise le domaine de définition des variables, entre autres pour des raisons de temps de calcul). Suivant le problème posé, nous cherchons à minimiser ou maximiser la fonction objectif f . Un problème d'optimisation peut être statique ou dynamique, mono-objectif ou multi-objectif, et avec ou sans contraintes [9].

1.3.2 Problème d'optimisation combinatoire

Un problème d'optimisation combinatoire est défini par l'ensemble S, un ensemble discret et fini, des solutions possibles d'un problème. $X \subseteq S$ est l'ensemble des solutions réalisables. $f : X \to \mathbb{R}$ est une fonction appelée fonction « objectif ». La résolution du problème consiste à minimiser (ou maximiser) la valeur $f(s)$, où $s \in X$ [10].

1.3.3 Classification des méthodes de résolution de problèmes d'optimisation

La résolution de différentes sortes de problèmes rencontrés dans notre vie quotidienne a poussé les chercheurs à proposer des méthodes de résolution et à réaliser de grands efforts pour améliorer leurs performances en termes de temps de calcul nécessaire et/ou de la qualité de la solution proposée. Au fil des années, de nombreuses méthodes de résolution de problèmes de différentes complexités ont été proposées. Ainsi, une grande variété et des différences remarquables au niveau du principe, de la stratégie et des performances ont été discernées. Cette variété et ces différences ont permis de regrouper les différentes méthodes de résolution de différents problèmes en deux classes principales : la classe de méthodes exactes et la classe de

méthodes approchées. L'hybridation des méthodes de ces deux classes a donné naissance à une pseudo-classe qui englobe des méthodes dites hybrides [11] (voir la Figure 1.4 [12]).

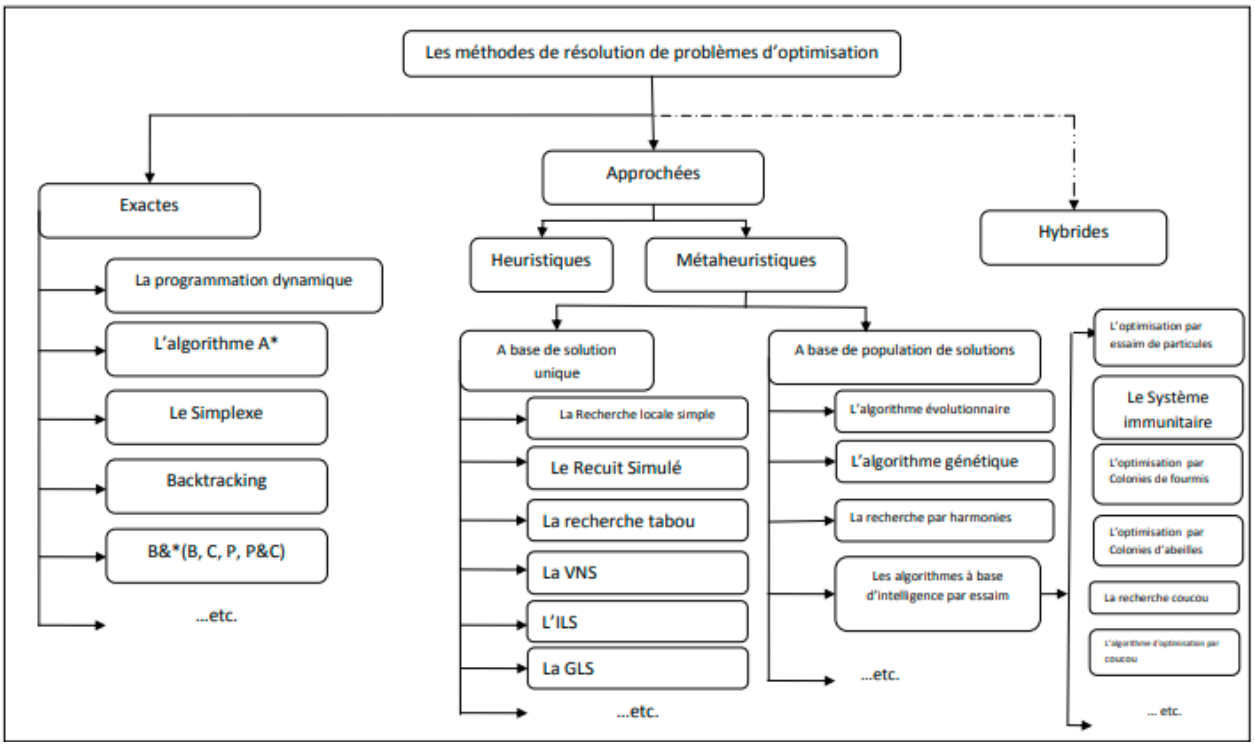

FIGURE 1.4 – Classification de méthodes de résolution de problèmes d'optimisation

1.3.4 Classe des méthodes exactes

Les méthodes exactes sont connues par le fait qu'elles garantissent l'optimalité de la solution. En fait, elles permettent de parcourir la totalité de l'ensemble de l'espace de recherche de manière à assurer l'obtention de toutes les solutions ayant le potentiel d'être meilleures que la solution optimale trouvée au cours de la recherche. Mais elles sont très gourmandes en termes de temps de calcul et de l'espace mémoire nécessaire. C'est la raison pour laquelle, elles sont beaucoup plus utilisées pour la résolution de problèmes faciles. La nécessité de disposer d'une solution de bonne qualité (semi optimale) avec un coût de recherche (temps de calcul et espace mémoire) raisonnable a motivé les chercheurs à proposer un autre type de méthodes de résolution de problèmes, communément connues par les méthodes approchées.

1.3.5 Classe des méthodes approchées

Les méthodes approchées constituent une alternative aux méthodes exactes. Elles permettent de fournir des solutions de très bonne qualité en un temps de calcul raisonnable. De nombreuses méthodes approchées ont été proposées. Elles sont plus pratiques pour la résolution de problèmes difficiles ou de problèmes dont on cherche des solutions en un bref délai. Ces méthodes sont souvent classées en deux catégories : des méthodes heuristiques et des méthodes métaheuristiques.

1. **Méthodes heuristiques**

   Les méthodes heuristiques sont des méthodes spécifiques à un problème particulier. Elles nécessitent des connaissances du domaine du problème traité. En fait, ce sont des règles empiriques qui se basent sur l'expérience et les résultats acquis afin d'améliorer les recherches futures. Plusieurs définitions des heuristiques ont été proposées par plusieurs chercheurs dans la littérature. Par exemple, selon Feigenbaum [13], une heuristique (ou méthode heuristique) est une règle d'estimation, une stratégie, une astuce, une simplification ou tout autre dispositif qui réduit considérablement l'espace de recherche dans des problèmes complexes. Les heuristiques ne garantissent pas des solutions optimales, voire pas de solution du tout. Cependant, elles proposent généralement des solutions suffisamment bonnes dans la majorité des cas.

2. **Méthodes métaheuristiques**

   Les méthodes métaheuristiques sont des méthodes générales, des heuristiques polyvalentes applicables sur une grande gamme de problèmes. Elles peuvent constituer une alternative aux méthodes heuristiques lorsqu'on ne connaît pas l'heuristique spécifique à un problème donné. Selon la définition proposée par Osman et Laportes [14] : Une métaheuristique est un processus itératif qui subordonne et guide une heuristique, en combinant intelligemment plusieurs concepts pour explorer et exploiter tout l'espace de recherche. Des stratégies d'apprentissage sont utilisées pour structurer l'information afin de trouver efficacement des solutions optimales, ou presque optimales.

   La plupart des métaheuristiques s'inspirent de la nature, et les solutions proposées sont généralement classées en deux catégories distinctes :

(a) **Métaheuristiques à base de solution unique**

Ces approches commencent leur exploration avec une unique solution initiale, puis se fondent sur la notion de voisinage pour améliorer progressivement la qualité de cette solution. En effet, la solution de départ subit une série de modifications locales basées sur son voisinage. L'objectif de ces ajustements est d'explorer de manière itérative les alentours de la solution actuelle afin d'en améliorer la qualité. Le voisinage d'une solution comprend toutes les modifications pouvant être apportées à cette solution elle-même. La qualité de la solution finale dépend essentiellement des opérations de modification effectuées par les opérateurs de voisinage.

(b) **Métaheuristiques à base de population de solutions**

Les métaheuristiques basées sur une population de solutions commencent leur exploration avec un ensemble de solutions variées. Elles travaillent sur cette diversité pour extraire la meilleure solution possible (l'optimum global) représentant la résolution du problème en question. Cette approche de l'utilisation d'un ensemble de solutions plutôt qu'une seule renforce la diversité de l'exploration et accroît les chances de découvrir des solutions de qualité.

1.4 Conclusion

L'Intenet des Objets connecte des objets à Internet pour améliorer la qualité de vie et promet des avancées significatives dans divers domaines. Cependant, l'IoT est confronté à des défis importants tels que l'interopérabilité, la sécurité et la confidentialité, la puissance et la consommation d'énergie, la disponibilité, la mobilité, l'hétérogénéité, la scalabilité et la découverte de services. Pour surmonter ces défis, des solutions d'optimisation sont nécessaires, en utilisant des méthodes exactes et approchées, avec une efficacité particulière des métaheuristiques pour fournir des solutions de haute qualité.

Le chapitre suivant se concentrera spécifiquement sur l'un de ces défis majeurs : la découverte de services dans l'IoT.

Chapitre 2

Découverte de services dans l'Intenet des Objets

2.1 Introduction

La découverte des services IoT est essentielle pour exploiter pleinement le potentiel des objets connectés. Elle consiste à identifier et à répertorier les différents appareils et services IoT au sein d'un réseau, et de faciliter leur intégration et leur gestion.

Ce chapitre explore en détail la découverte de services dans l'Intenet des Objets. Il commence par définir les services IoT et analyser leurs caractéristiques, puis aborde en profondeur le problème de la découverte de ces services. Ensuite, il examine les concepts clés du domaine, soulignant l'importance de l'Architecture Orientée Services (SOA) et de l'architecture REST (Representational State Transfer). Les différentes architectures de découverte de services sont également discutées. Enfin, le chapitre explore les principales méthodes de découverte des services IoT, en présentant un aperçu des approches sémantiques, bio-inspirées, basées sur des protocoles, contextuelles et basées sur la qualité de service.

2.2 Service IoT

Un service dans l'environnement IoT peut être n'importe quelle entité logicielle ou matérielle pouvant être invoquée par un utilisateur pour effectuer des tâches spécifiques, comme la lecture, la découverte ou la mesure de données, etc. En outre, les services peuvent être composés ensemble pour fournir des fonctionnalités complexes et améliorer les performances de la qualité de service afin de répondre aux exigences des utilisateurs [15].

2.2.1 Propriétés d'un service

Les propriétés d'un service jouent un rôle essentiel dans la définition de sa fonctionnalité et de ses caractéristiques. Lorsque nous concevons et utilisons des services logiciels, il est important de comprendre ces propriétés, qu'elles soient fonctionnelles ou non fonctionnelles.

Propriétés fonctionnelles : Les propriétés fonctionnelles d'un service sont des caractéristiques qui définissent la fonctionnalité du service. Elles sont généralement définies par les paramètres d'entrée et de sortie, les préconditions et postconditions, ainsi que les effets attendus lors de l'exécution.

Propriétés non fonctionnelles : Les propriétés non fonctionnelles d'un service sont des contraintes qui régissent les propriétés fonctionnelles. Elles se concentrent sur les attributs de qualité de service, tels que les mesures de performance (le temps de réponse, la précision, etc.), les attributs de sécurité (authentification, intégrité, etc.), la fiabilité, la disponibilité, l'évolutivité et la journalisation des accès. Ces propriétés non fonctionnelles définissent les conditions de qualité et de sécurité sous lesquelles le service est fourni et utilisé.

2.3 Problème de découverte de services

La découverte de services est le processus de recherche et de découverte des services en fonction de leurs capacités fonctionnelles qui correspondent aux propriétés fonctionnelles spécifiées dans la demande de l'utilisateur. Dans l'environnement IoT, les choses sont représentées comme un ensemble de services IoT qui sont différents des services traditionnels. Ils sont déployés dans des dispositifs à ressources limitées qui se caractérisent par leur mobilité et leurs capacités limitées en termes d'énergie, de calcul, de communication et de stockage. Par conséquent, les solutions de découverte de services traditionnelles ne sont pas adaptées à la découverte des services IoT, car elles exigent des schémas de communication lourds incorporant des cycles de sommeil pour économiser de l'énergie, et utilisent des formats encombrants. Par conséquent, plusieurs approches [16] [17] [18] [19] [20] [21] ont été proposées pour découvrir les services de l'IoT en tenant compte des contraintes de l'IoT, mais aucune d'entre elles ne répond à toutes ces

exigences en raison de l'absence de normes et d'un service/objet unifié. En effet, contrairement aux services web, les services IoT représentent les caractéristiques des objets et leur description doit donc être légère et enrichie de certains paramètres supplémentaires, notamment : localisation, niveau d'énergie, prise en charge de la QoS et gestion du contexte, etc.

2.4 Concepts fondamentaux de la découverte de services

La découverte de services est un processus complexe qui implique plusieurs étapes et concepts fondamentaux. Ces concepts clés permettent aux utilisateurs de trouver et d'accéder aux services existants dans un réseau.

2.4.1 Description de service

L'objectif de la description de service est de détailler les capacités qu'ils offrent. Elle doit être transparente et précise pour que tout consommateur puisse comprendre les spécificités des services et pour que le protocole de découverte de service puisse les choisir de manière appropriée dans un répertoire. Le format de description des services doit être standardisé pour assurer l'interopérabilité et la compatibilité entre différentes architectures et plateformes réseau.

2.4.2 Annonce de service

L'annonce de service implique de proclamer sur le réseau la présence d'un service nouvellement disponible. Cela peut être réalisé en consignant la description du service dans un répertoire ou en diffusant régulièrement des messages pour indiquer sa disponibilité.

2.4.3 Registre de services

Le registre des services est une collection de descriptions de services déposées par les fournisseurs. Suite à leur dépôt et enregistrement, les utilisateurs parcourent ce registre pour chercher et trouver les services désirés sans nécessiter d'action manuelle. Le registre peut être de nature centralisée ou décentralisée.

2.4.4 Requête des utilisateurs

Les requêtes des utilisateurs sont des sollicitations précises émises par un utilisateur pour un service spécifique à un moment donné. Après avoir émis une requête, le système fournit une liste de services appropriés.

2.4.5 Identification des services

L'identification des services est le processus qui facilite l'échange de services entre les utilisateurs et les fournisseurs au sein d'un réseau. Ce processus commence lorsque le consommateur émet une requête, exprimant ainsi ses critères de sélection pour les services qu'il souhaite acquérir. L'identification des services englobe également l'annonce, la recherche, la sélection et l'invocation des services.

2.4.6 Dynamique du réseau

Dans des environnements comme l'IoT, qui sont hautement dynamiques, de nombreux objets et services interagissent à grande échelle dans un système ouvert et évolutif. La disponibilité des services et des objets change fréquemment, ce qui nécessite une gestion efficace de leur découverte. Cela est réalisé à travers l'utilisation de mécanismes tels que les annonces et les notifications d'événements, permettant d'informer les clients des changements d'état et de disponibilité des services. Ces mécanismes enrichissent les interactions entre les dispositifs en introduisant une dynamique essentielle.

2.5 Architecture Orientée Services (SOA)

L'architecture orientée service (SOA en anglais : Service Oriented Architecture) est une architecture d'application distribuée, de plateforme indépendante et à couplage faible, dans laquelle les différents services communiquent les uns avec les autres par des interfaces à définition simple et précise, et ne tient pas compte de l'interface de programmation sous-jacente et des modèles de communication [22].

2.5.1 Acteurs dans SOA

Comme le montre la Figure 2.1 [23] ci-dessous, le modèle SOA met en interaction trois acteurs : le fournisseur de services, le client de services et l'annuaire de services.

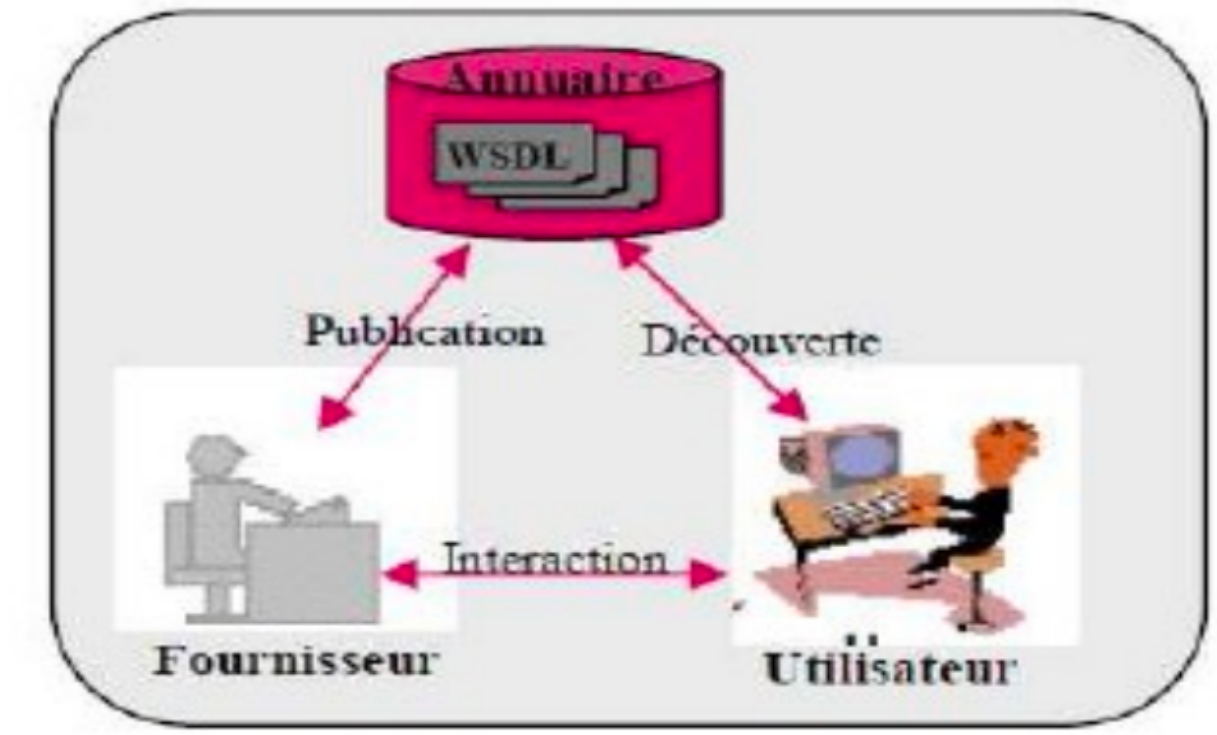

FIGURE 2.1 – Interactions dans l'architecture SOA

Fournisseur de services

Il est propriétaire du service web, son rôle est de publier le service web en utilisant des formats standardisés (XML, WSDL, etc.) et il est également une entité logicielle qui pourrait être adressée en réseau qui crée un ensemble de fonctions en tant que service. Ces fonctionnalités sont représentées dans une description nommée particularisation de service. Cette dernière contient toutes les informations utiles à l'usage du service. Les renseignements pourraient également porter sur des caractéristiques non fonctionnelles du service (c'est-à-dire celles relatives à la qualité du service). Le fournisseur de services admet et effectue les requêtes du client et annonce la particularisation de service dans la liste des services afin que les usagers de services puissent s'apercevoir et arriver au service.

Client de services

Représente l'application cliente qui appelle le service. C'est l'entité qui entreprend l'emplacement du service dans le répertoire, interagit avec le service via un protocole et effectue la fonctionnalité offerte par le service. Le client du service consulte le registre de services pour les services disponibles qui conviennent à ses besoins. La découverte du service se fait grâce à la description du service disponible dans le répertoire.L'utilisateur du service doit comprendre chaque service en ce qui concerne les caractéristiques fonctionnelles et non fonctionnelles dans le but de conquérir au résultat convenable. Après avoir choisi le service qu'il souhaite utiliser, le consommateur du service peut, dans quelques cas, négocier avec le fournisseur les conditions dans lesquelles il peut utiliser ce service. À l'issue de la négociation, un contrat de service est conclu entre l'utilisateur et le fournisseur. La plupart du temps, ce contrat de service contracte les termes d'application du service par l'usager sans couverture complète de résultat. Au moyen des renseignements fournis dans la description du service, l'utilisateur du service pourrait créer le lien et appeler les fonctionnalités du service.

Annuaire de services

Il accueille et enregistre d'une part les descriptions de services publiées par les fournisseurs. D'autre part, il reçoit et réagit selon les recherches envoyées par les clients. Il s'agit donc d'un répertoire de descriptions qui donne aux fournisseurs de services les méthodes d'annoncer des descriptions de services et de référencer leurs services sur le réseau. Il facilite pratiquement aux clients de se référer aux différentes descriptions disponibles. En plus des caractérisations de service, le registre possède des références aux fournisseurs de services. Cela permettra aux utilisateurs de situer les fournisseurs de caractérisations de service qui conviennent à leurs exigences. Ainsi, l'annuaire agit comme un acteur entre les fournisseurs de services et les clients.

2.5.2 Modèle d'Interactions dans SOA

Les interactions entre les principaux acteurs de l'architecture SOA se concentrent principalement sur la publication de services, la recherche et la découverte de services, ainsi que l'invocation de services.

Publication de service

Après avoir structuré sémantiquement son service, le fournisseur passe ensuite à la publication de ce service afin d'être découvert puis utilisé par d'autres utilisateurs ou machines. La publication du service se fait par l'UDDI (Universal Description, Discovery, and Integration), un annuaire de services fondé sur XML. Toutefois, avant qu'il soit publié, le service doit être décrit, cette description se faisant dans le langage WSDL (Web Service Description Langage).

Recherche et découverte d'un service

Une fois le service publié, le client pourrait consulter l'annuaire (UDDI) avec des mots clés permettant d'obtenir les URL (Uniform Resource Locator) des services demandés et un ensemble de descriptions (WSDL) contenant tous les renseignements essentiels pour appeler le service. Le recueil renvoie via un message SOAP (Simple Object Access Protocol) un répertoire de services qui répondent à la demande du client. Le client n'a qu'à en choisir un dans le répertoire.

Invocation de service

Après avoir référencé l'URL et la description du service détecté et appelé le fournisseur du service sélectionné, le consommateur de service appelle ce service en se référant aux renseignements fournis dans la description du service. Cette interaction entre le consommateur et le fournisseur de services a lieu durant l'accomplissement de l'application. Le service du fournisseur de services obtient la demande, la traite, formule la réponse SOAP et l'expédie au client.

2.6 Representational State Transfer (REST)

À la fin de la période dominée par SOA, le besoin de solutions plus légères et flexibles a émergé, conduisant à une transition vers REST (Representational State Transfer). Alors que SOA se concentrait sur l'intégration via des services complexes et souvent lourds, REST a introduit une approche plus simplifiée en utilisant des URI (Uniform Resource Identifiers) comme

ressources et en exploitant les méthodes standard HTTP pour faciliter l'accès et la manipulation des données. Cette évolution a favorisé une meilleure adaptabilité aux environnements distribués et aux exigences croissantes en matière d'interopérabilité.

REST a été introduit en 2000 dans la thèse de Roy Fielding [24], l'un des créateurs du protocole HTTP. Ce paradigme représente une approche architecturale pour la conception de systèmes distribués, tels que le World Wide Web. Contrairement à une idée répandue, REST n'est pas un format, un protocole ni un standard spécifique. C'est plutôt une manière de concevoir des systèmes informatiques basée sur des principes architecturaux comme l'interopérabilité, la scalabilité et la simplicité.

En utilisant REST, les développeurs peuvent créer des services web plus flexibles, évolutifs et interopérables. Cela facilite l'échange de données entre différentes applications et plateformes en mettant l'accent sur les rôles distincts de client et de serveur. Les ressources sont identifiées de manière unique à l'aide d'un système d'adressage standardisé, et les échanges entre client et serveur sont conçus pour être sans état, c'est-à-dire que chaque requête contient toutes les informations nécessaires au traitement, sans nécessiter de maintien d'état côté serveur.

L'architecture REST est illustrée dans la Figure 2.2 [25].

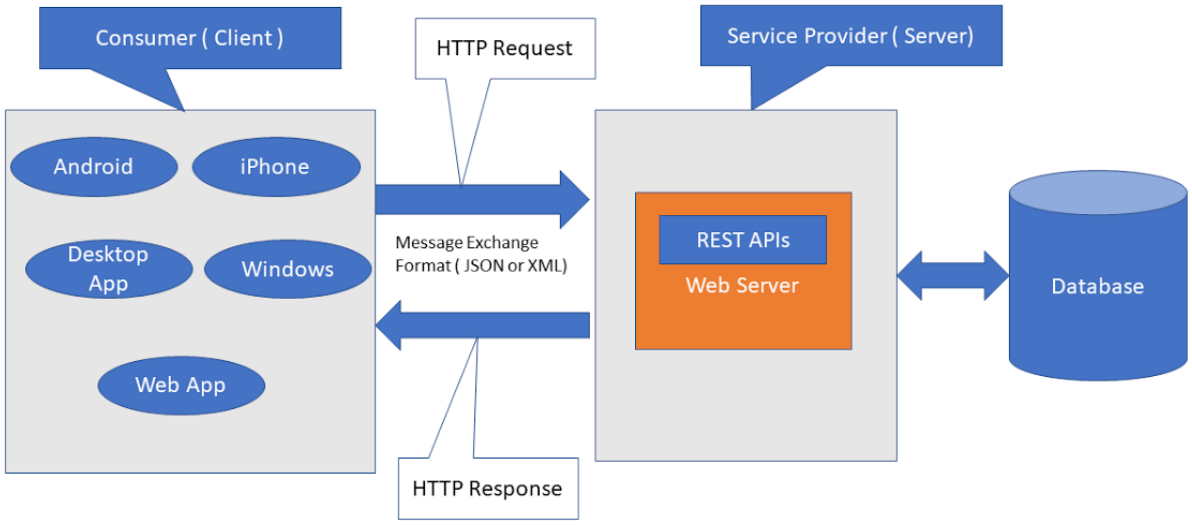

FIGURE 2.2 – Architecture REST

2.6.1 Principes clés de l'architecture REST

L'architecture REST repose sur plusieurs principes essentiels :

Identification des ressources : Chaque ressource dans une architecture REST est identifiée de manière unique par un URI. Ces identifiants permettent aux différents composants de l'architecture de distinguer et de manipuler les ressources de façon précise et uniforme.

Représentation des ressources : Les composants de l'architecture REST interagissent en transférant des représentations de ressources. Ces représentations peuvent prendre différentes formes telles que HTML, JSON ou XML. Cela permet aux systèmes hétérogènes de communiquer de manière efficace en utilisant des formats adaptés à leurs besoins spécifiques.

Opérations sur les ressources : Les ressources sont manipulées à travers une interface uniforme en utilisant les méthodes standardisées du protocole HTTP, telles que GET, POST, PUT et DELETE. Cette interface CRUD (Create, Read, Update, Delete) permet de gérer les ressources de manière cohérente et prévisible, simplifiant ainsi leur intégration et leur interaction au sein de l'architecture REST :

— **Create (POST) :** La méthode POST est utilisée pour envoyer, mettre à jour ou insérer une nouvelle ressource.

— **Read (GET) :** La méthode GET est couramment utilisée pour récupérer des informations à partir d'une ressource.

— **Update (PUT) :** La méthode PUT est utilisée pour mettre à jour une ressource existante.

— **Delete (DELETE) :** La méthode DELETE est utilisée pour supprimer une ressource.

2.7 Architectures pour la découverte de services

Dans le domaine de la recherche, on distingue trois types principaux d'architectures pour les systèmes de découverte de services : celles basées sur un annuaire, celles sans annuaire, et enfin, les architectures hybrides qui combinent les deux approches précédentes.

2.7.1 Modèles avec annuaire

Ce système est basé sur un registre qui conserve les détails des services et les coordonnées des fournisseurs. Quand un client a besoin d'accéder à un service, il se réfère à ce registre pour obtenir les informations requises sur le fournisseur du service. Dans ce modèle, les informations

peuvent être centralisées ou décentralisées.

Modèle centralisé

Dans une structure centralisée, toutes les informations de l'annuaire sont conservées au sein d'une seule entité. Cela offre une vue d'ensemble des services enregistrés, facilitant ainsi leur accès. Cependant, ce modèle présente des inconvénients notables, tels que le risque de défaillance de l'entité hébergeant l'annuaire, ce qui peut rendre le système inutilisable. De plus, la mobilité des appareils peut entraîner des difficultés de communication avec l'annuaire, ce qui complique la découverte des services. Un exemple caractéristique de ce modèle est la norme UDDI, un annuaire centralisé basé sur XML destiné aux services Web, permettant l'enregistrement et la localisation des services Web sur le réseau.

Modèle déstribué

Dans le modèle d'annuaire distribué, l'annuaire est réparti entre plusieurs nœuds du réseau. Ce modèle offre différentes approches, parmi lesquelles les suivantes sont les plus courantes :

Approche multi-annuaires : L'approche multi-annuaires implique l'utilisation de plusieurs annuaires, chacun étant stocké sur un nœud distinct. Dans ce schéma, le fournisseur a la liberté de choisir l'annuaire dans lequel il souhaite enregistrer ses services, sans nécessité d'établir une communication entre les différents annuaires.

Approche par réplication : La stratégie de réplication repose sur le modèle multi-annuaires, mais elle intègre un mécanisme de duplication de l'annuaire sur plusieurs nœuds. Cette technique nécessite une synchronisation et un échange fréquents d'informations entre les annuaires pour garantir l'uniformité des données.

Approche par ensemble dominant (backbone) : La méthode par ensemble dominant, ou backbone, fait également appel à de multiples annuaires. Les nœuds qui hébergent ces annuaires sont sélectionnés de façon distribuée parmi toutes les entités du réseau, créant ainsi un sous-réseau connecté (le backbone). À la différence de l'approche par réplication, cette technique ne requiert pas de synchronisation entre les annuaires. Cependant, ces nœuds échangent des

informations entre eux pour gérer les demandes de services. L'inscription ou la recherche d'un service s'effectue auprès d'un unique annuaire, choisi de façon aléatoire. Si les services demandés ne sont pas disponibles, la requête est alors acheminée vers les autres membres du backbone.

Approche hiérarchique (par cluster) : L'approche hiérarchique, aussi appelée par cluster, divise le réseau en plusieurs sous-ensembles, chacun possédant son propre annuaire, similaire à l'approche par backbone. Cependant, dans cette stratégie, chaque cluster utilise et communique exclusivement avec son annuaire local. La création et le maintien des clusters nécessitent un échange continu de messages de contrôle entre les nœuds du réseau. Les clusters peuvent être formés selon divers critères tels que les types de services, la localisation géographique des nœuds ou encore la mobilité des nœuds.

Approche par DHT (Table de hachage distribuée) : Dans cette approche, l'annuaire est segmenté en différentes sections, chacune étant conservée sur un nœud spécifique du réseau. Pour déterminer la section de l'annuaire stockée dans un nœud donné, on utilise une fonction de distance basée sur un identifiant unique attribué à chaque nœud. Cette fonction facilite la localisation des fournisseurs de services ainsi que le calcul du nombre de sauts nécessaires entre deux identifiants.

2.7.2 Modèle sans annuaire (décentralisé)

Dans ce modèle, chaque fournisseur est responsable de faire connaître ses propres services aux autres membres du réseau. Les communications entre les clients et les fournisseurs peuvent suivre des routes établies par des protocoles de routage ou se baser sur une stratégie d'inondation. Comparé aux approches utilisant des annuaires, le modèle sans annuaire est particulièrement adapté à la mobilité. Cependant, il n'est pas optimal pour les réseaux de grande taille. Ce modèle peut fonctionner selon différents modes : push, pull et hybride.

Mode push

Dans ce mode, la découverte des services est proactive : les fournisseurs diffusent leurs offres par inondation sans que les clients en fassent explicitement la demande. Chaque client maintient en mémoire un annuaire local, souvent appelé cache, des services disponibles annoncés

par les fournisseurs. Ce cache est organisé sous forme d'arbre pour structurer les services et faciliter leur recherche.

Mode pull

Dans ce mode, la découverte des services est réactive : lorsqu'un client souhaite utiliser un service spécifique chez un fournisseur donné, le client émet une requête de découverte de ce service par inondation. Cette requête est propagée à travers les nœuds du réseau jusqu'à atteindre le fournisseur concerné. Le fournisseur répond ensuite pour fournir les détails du service demandé.

Mode hybride

Le mode hybride combine à la fois le mode push et le mode pull en fonction de plusieurs paramètres tels que le nombre de clients, de fournisseurs, de services et de requêtes.

2.7.3 Modèle hybride

Un modèle hybride permet de choisir entre l'utilisation d'un annuaire ou de s'en passer. Lorsqu'un client recherche un service, il commence par interroger un annuaire pour la découverte. Si l'annuaire est indisponible ou si le service n'est pas trouvé, le client bascule alors vers une stratégie sans annuaire pour localiser le service.

2.8 Travaux de recherche sur les méthodes de découverte de services dans l'Internet des Objets

La recherche sur la découverte de services a attiré une attention significative dans la littérature académique. Elle est généralement regroupée en cinq catégories : les approches sémantiques, bio-inspirées, basées sur les protocoles, contextuelles et basées sur la qualité de service (voir Figure 2.3 [26]).

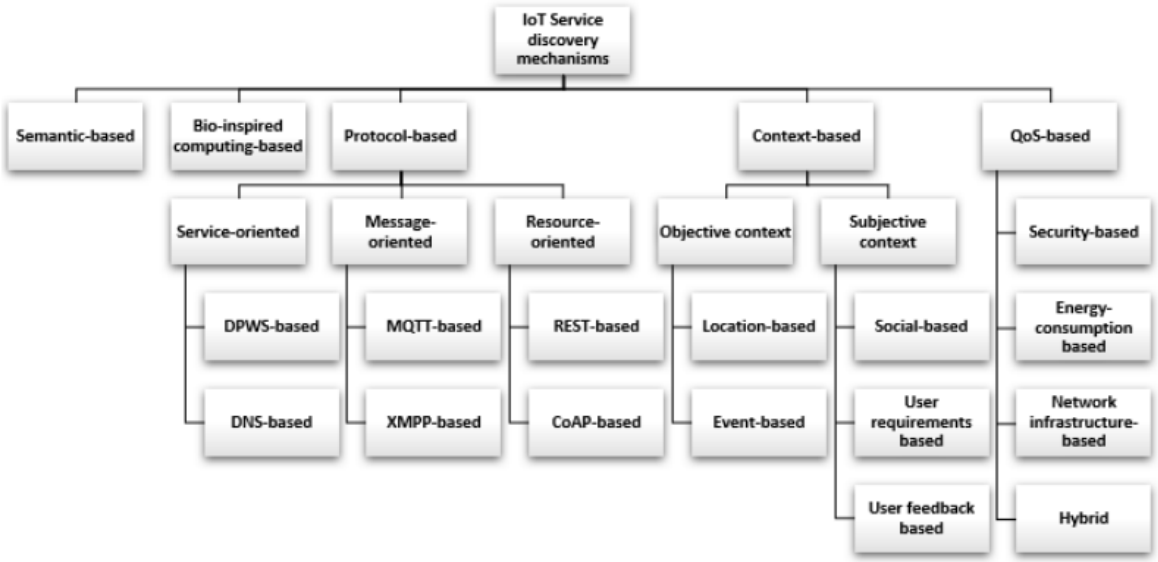

FIGURE 2.3 – Classification des approches de découverte de services dans l'IoT

2.8.1 Approches sémantiques

Cette catégorie utilise des techniques du web sémantique pour découvrir des services IoT, telles que la similarité et la relativité sémantiques, les ontologies, etc. Seuls quelques travaux peuvent être mentionnés dans cette catégorie.

Les chercheurs [27] proposent une nouvelle approche pour créer des écosystèmes IoT sémantiquement interopérables. Ils utilisent des requêtes SPARQL (SPARQL Protocol and RDF Query Language) pour découvrir et accéder aux appareils IoT publiant des données hétérogènes. L'approche, appelée eWoT, repose sur des descriptions d'objets et des mappings WoT (Web of Things) pour traduire les données en RDF (Resource Description Framework). Les expérimentations montrent qu'eWoT (Enterprise Web of Things) fournit des réponses complètes et correctes sans affecter le temps de réponse et offre une évolutivité linéaire dans les grands écosystèmes.

Quant à eux [28], traitent de l'Intenet des Objets et de la nécessité pour les applications d'interagir avec de nombreux dispositifs pour récupérer les données contextuelles produites par ces objets. Ces données doivent être découvertes et sélectionnées a priori. QoDisco est un service de découverte sémantique qui répond à ce besoin en IoT. Il utilise des référentiels pour stocker les descriptions des ressources selon un modèle d'information basé sur l'ontologie et permet

des requêtes multi-attributs et en gamme. Des expériences ont été menées pour évaluer les performances de QoDisco, notamment en ce qui concerne la réduction des coûts de recherche sémantique.

En outre, dans [29], Zhao et al ont introduit une approche basée sur un modèle multi-dimensionnel pour évaluer la similarité entre les services IoT. Ce modèle prend en compte la complexité et la diversité des services opérant dans un espace ternaire (utilisateur, cyberespace et espace physique). Chaque dimension du modèle construit une représentation sémantique des services, incluant une classification précise, les propriétés des services et les contraintes des propriétés. L'algorithme MDM (Multiple Dimensional Measuring) est alors présenté pour calculer la similarité entre les services sur chaque dimension, en tenant compte à la fois de la structure et de la description du modèle. Cette approche garantit un matchmaking agile et une découverte efficace des services dans l'IoT, en tenant compte de la dynamique et de la complexité de cet environnement.

2.8.2 Approches bio-inspirées

Les paradigmes informatiques bio-inspirés ont émergé en raison de leur capacité inhérente à fonctionner sans contrôle central, de leur gestion efficace des ressources et de leur auto-organisation.

Azizou et al. [30] proposent un nouvel algorithme décentralisé de découverte de services IoT basé sur la métaheuristique Grey Wolf optimisation (GWO), une méta-heuristique inspirée de l'intelligence en essaim. GWO utilise la hiérarchie sociale et le mécanisme de chasse des loups gris pour rechercher et découvrir des services IoT avec un nombre minimal d'étapes. Les simulations montrent que cette approche réussit bien et s'adapte à l'augmentation du nombre d'objets dans les réseaux IoT décentralisés. Comparé aux méthodes basées sur le flooding et le random walk, GWO optimise efficacement le nombre de sauts et maintient une haute scalabilité.

Dans un autre contexte similaire, Azizou et al. [31] proposent un modèle de découverte de services décentralisé basé sur l'algorithme de colonie de fourmis (ACA) pour les réseaux IoT. L'ACA, inspiré par le comportement des fourmis à trouver de la nourriture en utilisant des phéromones, est utilisé pour localiser efficacement les services avec un nombre réduit de sauts.

Leur approche tire parti des mécanismes de dépôt de phéromones pour orienter les décisions et converger vers des solutions optimisées. Les résultats de simulation montrent que cette méthode à un taux de succès élevé dans la découverte de services, même à grande échelle.

Rapti et al. [32] ont proposé une solution complémentaire basée sur les champs de potentiel artificiels (APF), où chaque appareil du réseau représente un nœud de fourniture de services (SPN). Chaque SPN offre sa fonctionnalité par le biais de services gérés en interne par des agents logiciels. Pour leur étude, les auteurs ont présenté une approche décentralisée fondée sur les APFs, sans recours à un annuaire central. Chaque demande de service déclenche la formation d'un APF, agissant comme un guide pour sélectionner les services en fonction de leur disponibilité et de leur pertinence pour l'utilisateur. Les APFs sont générés par les SPNs en réponse à une requête de service et sont gérés par des agents logiciels. La force de chaque APF est déterminée par divers facteurs, tels que le pourcentage de services que peut fournir le SPN après une demande de service. La découverte et la sélection des services sont orchestrées par des forces artificielles appliquées entre les nœuds demandant le service et les SPNs.

2.8.3 Approches basées sur des protocoles

Les protocoles de découverte de services fournissent des mécanismes permettant la découverte spontanée et dynamique des services disponibles au sein du réseau.

Dans [33], Vandana et al. ont présenté un enrichissement CoAP basé sur la sémantique pour la découverte de ressources (S-CoAP), améliorant les attributs du format de lien CORE pour permettre un traitement sémantique de la demande de l'utilisateur et des ressources.

Parallèlement, dans [34], Pereira et al. ont développé une architecture distribuée de découverte de ressources basée sur MQTT (Message Queuing Telemetry Transport) pour la communication M2M (MQTT-RD : MQTT for Resource Discovery), introduisant des capacités plug and play aux dispositifs IoT sans nécessiter de configuration préalable.

En complément, Mahyoub et al. [35] ont proposé des mécanismes de découverte optimisés, distribués et autonomes pour les objets intelligents. Leur approche vise à minimiser la surcharge générée par les opérations des protocoles mDNS/DNS-SD, tout en améliorant le trafic réseau et la gestion de l'énergie.

2.8.4 Approches contextuelles

Cette classe permet de découvrir des services en fonction du contexte, qui peut être divisé en deux sous-classes : le contexte objectif, qui fait référence aux techniques de développement durable centrées sur l'environnement physique de l'utilisateur, et le contexte subjectif, qui regroupe les approches principalement axées sur les utilisateurs et leurs relations sociales avec les objets de l'IoT.

Dans [36], Tao et al. ont proposé une recommandation de services de confiance basée sur la localisation dans l'Internet des véhicules (IoV). Cette approche utilise la communication coopérative pour satisfaire les exigences de service personnalisées et précises des utilisateurs, prenant en compte les caractéristiques spatio-temporelles ainsi que les relations directes et indirectes entre les utilisateurs.

Pruthvi et al. [37] ont mis en œuvre une plateforme SIoT pour l'environnement des collèges intelligents, où les objets peuvent créer leurs propres relations basées sur des règles définies par leurs propriétaires. Cette autonomie réduit l'intervention humaine et favorise la création de communautés d'objets. La plateforme établit des profils sociaux pour les objets en utilisant les données reçues de l'application IoT.

En outre, [38] ont proposé un cadre orienté service, centré sur l'utilisateur et conscient des événements, pour les services ambiants et la surveillance des événements dans les environnements ambiants. Ce cadre permet l'auto-adaptation aux changements imprévisibles en remplaçant les services et en replanifiant en cas de défaillance.

2.8.5 Approches basées sur la Qualité de Service

Dans cette catégorie, les services sont découverts pour les utilisateurs en fonction de leurs exigences de qualité de service (exigences non fonctionnelles de l'IoT) à l'aide de paramètres spécifiques. Un grand nombre de recherches et d'efforts ont été consacrés à l'amélioration de la qualité de service dans le cadre du développement durable de l'IoT.

Les chercheurs [39] proposent une approche innovante pour optimiser la découverte de services dans l'Intenet des Objets, intégrant la Qualité de Service (QoS) via l'algorithme d'op-

timisation des baleines et les algorithmes génétiques. Leur méthode vise à améliorer l'efficacité des processus de sélection de services IoT, réduisant ainsi le temps d'accès aux données, optimisant l'utilisation de l'énergie et améliorant la rentabilité selon les simulations réalisées. Cette approche ouvre la voie à des infrastructures IoT plus intelligentes et adaptables, capables de répondre efficacement aux exigences variées et dynamiques des environnements connectés modernes.

Parallèlement, [40] proposent une méthode de découverte sécurisée et basée sur un courtier pour l'Intenet des Objets. Utilisant un courtier centralisé, cette approche optimise la découverte de services en prenant en compte le contexte et la Qualité de Service (QoS), ainsi que la fiabilité des fournisseurs. Les utilisateurs soumettent des requêtes chiffrées au courtier, qui gère les valeurs de confiance des objets connectés et surveille la QoS. En se basant sur ces critères, le courtier associe chaque requête à un fournisseur approprié, sécurisant les communications ultérieures par la distribution de clés de session. Cette méthode efficace et légère utilise la cryptographie pour sécuriser les échanges, minimisant ainsi les ressources nécessaires et les retards, comme le démontrent les simulations réalisées.

De plus, dans [41], une approche est proposée pour résoudre les problèmes de confidentialité dans les services de l'Intenet des Objets. Cette approche consiste en la conception et la mise en œuvre d'un système web appelé IoT Service Store (ISS). ISS agit comme un registre auquel sont enregistrés les services IoT, gérant les informations confidentielles et les préférences des utilisateurs. Son rôle est également d'évaluer les risques potentiels sur la confidentialité des données collectées à partir des capteurs. Ce système permet aux utilisateurs de parcourir facilement les services IoT à proximité, de comprendre les implications en termes de confidentialité et de contrôler la collecte et l'utilisation des données des capteurs. Il affiche des informations détaillées sur les données personnelles pouvant être déduites des données des capteurs, permettant aux utilisateurs de noter chaque service en fonction du compromis entre utilité et confidentialité. ISS communique avec les services IoT pour modifier leurs pratiques de collecte et d'utilisation des données selon les préférences de confidentialité de l'utilisateur. Son objectif est de renforcer la confiance des utilisateurs dans leurs décisions de souscrire à des services IoT tout en atténuant leurs préoccupations concernant les risques de confidentialité.

2.9 Conclusion

La découverte de services dans l'Intenet des Objets est essentielle pour assurer une interaction efficace et transparente entre les objets connectés. En développant des approches innovantes et en favorisant la collaboration, nous pouvons créer un environnement IoT plus cohérent et mieux adapté aux besoins de notre société connectée.

Dans le chapitre suivant de ce mémoire, nous présentons notre approche basée sur l'algorithme d'optimisation par colonie d'abeilles artificielles, inspirée par la nature, visant à améliorer la découverte et la localisation des services IoT. Cette méthode met en évidence l'efficacité des techniques bio-inspirées pour résoudre les défis complexes du domaine de l'Intenet des Objets.

Chapitre 3

Contribution à la découverte et à la localisation de services dans l'Internet des Objets

3.1 Introduction

Ce chapitre présente notre contribution à l'amélioration de la découverte des services dans l'Intenet des Objets en utilisant l'algorithme d'optimisation par colonie d'abeilles artificielles, inspiré par le comportement des abeilles.

Nous commençons par examiner les motivations qui justifient l'utilisation des métaheuristiques, en particulier l'algorithme d'optimisation par colonie d'abeilles artificielles, pour la découverte de services IoT. Ensuite, nous introduisons notre adaptation de cet algorithme pour l'IoT, nommée SD-ABCA (Service Discovery based on Artificial Bee Colony Optimization), conçue pour optimiser la découverte de services en tenant compte des caractéristiques spécifiques des réseaux IoT. Nous comparons ensuite notre approche SD-ABCA à la méthode traditionnelle de flooding en termes de performances. À cet effet, nous décrivons notre cadre de simulation, les ensembles de données utilisés, ainsi que les mesures et méthodologies employées. Enfin, nous analysons les résultats pour fournir une évaluation comparative des performances de SD-ABCA et du flooding dans le contexte de la découverte de services IoT.

3.2 Motivations

L'optimisation par colonie d'abeilles artificielles (ABCA) est une métaheuristique inspirée de la nature qui a démontré son efficacité dans la résolution de problèmes d'optimisation dans

plusieurs domaines [42] [43] [44] [45] [46] [47] [48] [49] [50]. L'ABCA pourrait être particulièrement adaptée à la découverte de services IoT pour plusieurs raisons :

— **Efficacité de la recherche :** L'algorithme ABCA a prouvé son efficacité dans la recherche de solutions optimales grâce à sa capacité à explorer de larges espaces de solutions tout en convergeant rapidement vers les solutions optimales.

— **Adaptabilité :** L'algorithme ABCA est moins complexe et nécessite un nombre réduit de paramètres, ce qui facilite son adaptation à divers contextes de découverte de services IoT.

— **Exploration et exploitation équilibrées :** L'algorithme ABCA équilibre efficacement l'exploration et l'exploitation. Cet équilibre est important pour la découverte de services IoT, où il est nécessaire de découvrir de nouveaux services (exploration) tout en utilisant efficacement les services déjà connus (exploitation).

Inspirés par ces constatations, nous mettons en avant une adaptation de l'algorithme ABCA pour aborder la problématique de découverte de services dans l'Intenet des Objets.

3.3 Métaheuristique ABCA (Artificial Bee Colony Algorithm)

L'algorithme de colonie d'abeilles artificielles est une méta-heuristique basée sur un essaim, introduit par Karaboga en 2005 [51] pour optimiser les problèmes numériques, en inspectant les comportements des abeilles réelles pour trouver la source de nourriture, qui s'appelle le nectar, et partager l'information des sources de nourriture aux autres abeilles du nid. L'algorithme est spécifiquement basé sur le modèle théorique proposé par Tereshko et Loengarov [52].

Dans cet algorithme, les abeilles artificielles sont classifiées en trois groupes :

— Abeilles employées : abeilles qui recherchent des sources de nourriture.

— Abeilles spectatrices : abeilles d'observation qui évaluent les sources trouvées.

— Abeilles éclaireuses : sont chargées de trouver de nouvelles sources nourritures.

Les abeilles employées et spectatrices suivent le procédé d'exploitation dans l'espace de recherche d'une part et les abeilles éclaireuses commandent le procédé d'exploration d'autre

part. Une position de source de nourriture représente une solution possible au problème à optimiser.

Chaque cycle de recherche se compose de trois étapes principales : le déplacement des abeilles employées et des spectatrices sur les sources de nourriture, le calcul de leurs montants de nectar et enfin la détermination des sources abandonnées et charger les abeilles éclaireuses de chercher des sources possibles de nourriture.

On associe pour chaque source de nourriture une abeille employée ; c'est-à-dire, le nombre d'abeilles employées est égal au nombre de sources de nourriture. Si l'abeille employée, représentant une source de nourriture, n'est pas améliorée par un nombre prédéterminé d'épreuves, elle va être forcée de devenir une abeille éclaireuse pour une recherche aléatoire de nouvelles sources de nourriture. Cette action (une abeille est choisie comme abeille éclaireuse) est commandée par un paramètre de commande appelé la « limite ».

Les abeilles employées partagent l'information avec les abeilles spectatrices dans la ruche de la sorte que les abeilles spectatrices puissent choisir une source de nourriture pour l'explorer. La quantité de nectar d'une source de nourriture correspond à la qualité de la solution.

Les spectatrices sont placées sur les sources de nourriture en employant un processus de sélection basé sur la probabilité. À mesure que la quantité de nectar d'une source de nourriture augmente, la valeur de probabilité, avec laquelle la source de nourriture est préférée par les spectatrices, augmente aussi.

Ces étapes sont présentées dans l'Algorithme 1 [50]. Les notations utilisées dans l'Algorithme 1 sont présentées dans le tableau 3.1 ci-dessous :

Annotation	**Description**
E	Nombre d'abeilles employées
O	Nombre d'abeilles spectatrices
S	Nombre d'abeilles éclaireuses
Xi	Solution individuelle dans la population
Pi	Probabilité associée à chaque solution pour les abeilles spectatrices
vi	Nouvelle solution générée pour les abeilles spectatrices

TABLE 3.1 – Notations utilisées dans l'algorithme de la colonie d'abeilles artificielles

Algorithm 1 Algorithme de colonie d'abeille artificielle (ABCA)

Require: S, E, O

Ensure: Meilleure qualité de nourriture

1: Initialiser une population de $E + O$ solutions aléatoires.
2: Évaluer le fitness de chaque solution dans la population.
3: Mémoriser la meilleure solution dans la population.
4: **while** le critère d'arrêt n'est pas satisfait **do**
5: **for** chaque solution dans la population **do**
6: Effectuer une itération de l'algorithme de recherche de nouvelle source.
7: Évaluer le fitness de la population après l'itération.
8: Garder la meilleure solution trouvée entre la solution actuelle et la solution candidate.
9: **end for**
10: Calculer la probabilité (P_i) pour chaque solution (X_i).
11: Générer de nouvelles solutions (v_i) pour les abeilles spectatrices en fonction de P_i.
12: Calculer le fitness de toutes les nouvelles solutions dans la population.
13: **for** chaque solution dans la population **do**
14: **if** la solution n'a pas été améliorée au cours des itérations **then**
15: Sauvegarder la solution et la remplacer par une solution aléatoire.
16: **end if**
17: **end for**
18: Trouver S solutions aléatoires et remplacer les S membres de la population qui ont les mauvais fitness.
19: Mémoriser la meilleure solution trouvée dans la population.
20: **end while**
21: Retourner La meilleure qualité de nourriture.

3.4 Flooding

Le Flooding, également appelé diffusion, est une technique de communication largement utilisée dans les réseaux mobiles et les réseaux de capteurs [53] [54] [55] [56] [57]. Chaque nœud du réseau retransmet chaque message qu'il reçoit à tous ses voisins, à l'exception de celui qui lui a envoyé le message. Cette méthode est particulièrement efficace pour assurer un taux de réception élevé. Dans le contexte de la découverte de services IoT, une requête de services initiée par un utilisateur est diffusée à ses nœuds voisins. Ces derniers retransmettent l'information, visant à atteindre le plus grand nombre possible de fournisseurs de services dans le réseau. Les fournisseurs qui proposent le service demandé répondent ensuite directement au client en lui envoyant un message unicast. Cependant, malgré sa simplicité et sa capacité à garantir une couverture complète, le Flooding peut entraîner une utilisation inefficace des ressources du réseau en raison de la redondance des messages, surtout dans les grands réseaux. Par conséquent, des stratégies d'optimisation sont souvent nécessaires pour minimiser son impact sur les performances du réseau [58].

3.5 Adaptation de la métaheuristique ABCA pour la découverte de services dans l'Internet des Objets : SD-ABCA en mode discret

Dans cette section, nous présentons une adaptation de l'algorithme ABCA, nommée SD-ABCA (Service Discovery based on ABCA), pour résoudre le problème de découverte de services dans l'IoT en le traitant comme un problème d'optimisation doté d'une fonction objectif.

Notre objectif est d'optimiser le nombre de sauts nécessaires à la découverte d'une requête de services en utilisant l'algorithme ABCA. Cette approche utilise l'optimisation par colonie d'abeilles artificielles, une métaheuristique qui s'inspire du comportement des abeilles pour améliorer l'efficacité de la recherche de meilleur chemin entre la colonie et la source de nourriture.

3.5.1 Formalisation

Dans le contexte de l'IoT, la représentation de la découverte de services peut être formalisée à l'aide d'un graphe complet, où chaque nœud représente un objet spécifique offrant un ensemble distinct de services, et chaque arête symbolise une connexion directe entre deux objets. Ce graphe est défini comme $G = (k, L)$ avec :

- $k = \{O_1, O_2, \ldots, O_n\}$ représentant l'ensemble de tous les objets du réseau IoT.
- $L = \{L_{i,j} \mid (O_i, O_j) \in k\}$ est l'ensemble de toutes les connexions entre les objets.
- $S = \{s_1, s_2, \ldots, s_n\}$ est l'ensemble des services qui peuvent être fournis par les objets IoT.
- À chaque objet O_i dans k est associé un ensemble de services $F(O_i)$, où $F(O_i) \subseteq S$. Cela signifie que $F(O_i)$ est un sous-ensemble des services disponibles, décrivant les services spécifiques que l'objet O_i peut offrir.
- La fonction $F : k \to 2^S \setminus \{\emptyset\}$ attribue à chaque objet O_i son ensemble de services $F(O_i)$.

Lors de la recherche d'un service dans un réseau IoT, notre objectif principal, après avoir localisé le service, est de minimiser le nombre de sauts nécessaires pour atteindre ce service. Pour cela, nous avons défini une fonction objectif qui calcule le nombre de sauts. Cette fonction est formulée comme suit :

$$F(B) = \text{NBR\_SAUTS}(O_i, O_j) \tag{3.1}$$

Ici, B est le chemin de l'objet O_i (l'objet qui a initié la requête de service) à l'objet O_j (le dernier objet dans le chemin B). Cette fonction objectif nous aide à optimiser le processus de découverte de services en minimisant le nombre de sauts effectués.

Codification de la solution

Pour résoudre le problème de découverte de services, une solution est représentée par un vecteur d'entiers (chemin), où chaque élément du vecteur correspond à un objet sélectionné parmi les objets candidats (voisins).

Par exemple, dans la Figure 3.1, considérons une requête REQ ={5, 7, 12}, initiée par l'objet O_1. Une solution possible pour cette requête pourrait être représentée par le vecteur d'entiers [1, 4, 5, 6]. Chaque élément de ce vecteur représente l'identifiant d'un objet dans le réseau.

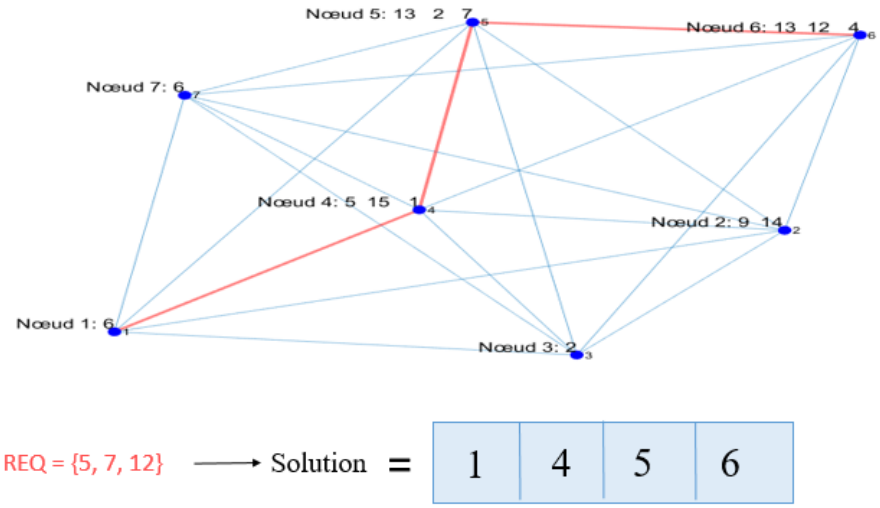

FIGURE 3.1 – Codage d'une solution

Le réseau lui-même peut être représenté par une matrice d'adjacence. Dans une matrice d'adjacence, chaque ligne et chaque colonne correspond à un objet dans le réseau. Si un objet i est connecté à un objet j, alors la cellule à l'intersection de la i-ème ligne et de la j-ème colonne contient un 1. Sinon, elle contient un 0.

Le graphe de la Figure 3.1 est défini à l'aide d'une matrice d'adjacence, formulée comme suit :

$$\begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 \end{pmatrix}$$

3.5.2 Principe de l'algorithme SD-ABCA

L'algorithme de colonie d'abeilles artificielles classique, utilisé principalement pour résoudre des problèmes d'optimisation, a été adapté pour la découverte de services dans l'Intenet des Objets. Dans ce contexte, les abeilles cherchent à découvrir des chemins optimaux qui minimisent le nombre de sauts nécessaires pour satisfaire une requête de services spécifiques initiée par un objet IoT.

Entrées et Sorties

L'algorithme commence avec N_O objets distribués aléatoirement, N abeilles, et N_S services. Chaque objet offre entre 1 et 3 services. Une requête de recherche Req(S_i, O_i) est initiée par l'objet O_i pour découvrir l'ensemble de services S_i. En sortie, l'algorithme produit une solution (un chemin) qui est proche de l'optimal.

Initialisation

L'algorithme commence par l'initialisation d'une population de N chemins aléatoires. Ces chemins représentent des solutions potentielles respectant les contraintes des liens et des ser-

vices, avec les valeurs discrétisées en utilisant la fonction de round pour assurer la compatibilité avec les ressources disponibles.

1. **Population Initiale** : Génération de N chemins aléatoires avec des valeurs discrétisées en utilisant la fonction de round.
2. **Évaluation Initiale** : Calcul des fitness de chaque chemin à l'aide de la formule 3.1, qui mesure la qualité des chemins en termes de nombre de sauts.
3. **Sélection du Meilleur Chemin** : Identification du meilleur chemin initial (X_{Best}).

Cycles de Recherche

L'algorithme itère jusqu'à atteindre un critère d'arrêt, tel qu'un nombre maximum d'itérations ($MaxT$).

Phase des Abeilles Employées

1. Chaque abeille employée explore le voisinage de son chemin actuel pour générer de nouvelles solutions en appliquant la formule de mutation :

$$v_{ij} = x_{ij} + \phi_{ij}(x_{ij} - x_{kj}) \tag{3.2}$$

   Cette formule génère de nouvelles solutions en ajustant les positions selon les différences avec d'autres chemins, favorisant l'amélioration itérative et la recherche locale. Le terme aléatoire ϕ_{ij} (entre [-1, 1]) introduit de la diversité, évitant une convergence prématurée. L'index k, choisi aléatoirement et différent de i, assure que chaque nouvelle solution est influencée par une autre, permettant une adaptation dynamique. Cette formule équilibre exploration et exploitation, optimisant ainsi la découverte de services IoT en évitant les optima locaux.
2. Calcul de la fitness des nouveaux chemins.
3. Adoption du mécanisme de sélection gourmande pour remplacer l'ancien chemin par le nouveau si ce dernier est meilleur.
4. Mise à jour des compteurs d'essais (Trial).

Phase des Abeilles Spectatrices

1. Les abeilles spectatrices évaluent les solutions basées sur les informations partagées par les abeilles employées.
2. La probabilité de sélection d'un chemin est calculée par :

$$P_i = \exp\left(-\frac{F_i}{\sum_{n=1}^{N} F_n}\right) \tag{3.3}$$

   où P_i est la probabilité de sélection du chemin i, F_i est la fitness du chemin i, et $\sum_{n=1}^{N} F_n$ est la somme des fitness de tous les chemins. Cette formule signifie que la probabilité est égale à l'exponentielle négative de chaque fitness divisée par la somme totale des fitness. Elle permet de calculer des probabilités normalisées basées sur les valeurs de fitness, où les valeurs plus élevées de fitness conduisent à des probabilités plus basses, favorisant ainsi la sélection des chemins avec une fitness minimale.
3. Génération de nouvelles solutions pour les abeilles spectatrices et application du mécanisme de sélection gourmande.

Phase des Abeilles éclaireuses

Les abeilles éclaireuses remplacent les chemins stagnants, qui n'ont pas été améliorés depuis un certain nombre d'itérations (limit), par de nouveaux chemins générés aléatoirement, respectant les contraintes des liens et des services.

Mise à Jour et Sélection de la Meilleure Solution

1. Mise à jour de la meilleure solution trouvée (X_{Best}).
2. Vérification que la solution couvre tous les services IoT demandés.
3. Retour de la meilleure solution (chemin) et de son fitness.

Toutes ces étapes sont détaillées dans l'organigramme de la Figure 3.2.

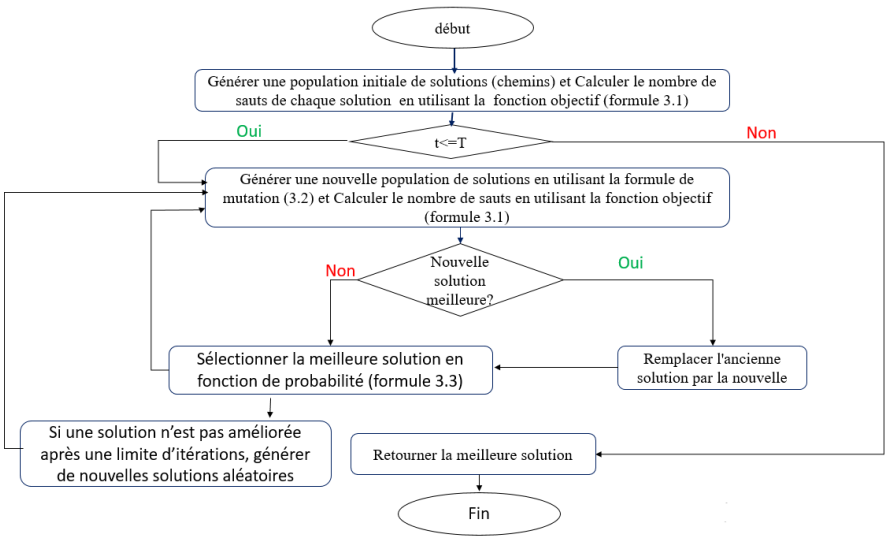

FIGURE 3.2 – Organigramme de l'algorithme SD-ABCA

3.5.3 Pseudo-code de SD-ABCA pour l'Optimisation de la Découverte de Services IoT

Le Tableau 3.2 ci-dessous présente les notations utilisées dans notre approche, suivi du pseudo-code de l'algorithme adapté SD-ABCA (Algorithme 2) :

Notation	Description
Req(S_i, O_i)	Requête de l'utilisateur, S_i ensemble de services demandés par O_i
X_{Best}	Meilleur chemin trouvé
N	Nombre total de chemins dans la population
Fitness	Mesure de la qualité d'un chemin
t	Compteur d'itérations
MaxT	Nombre maximum d'itérations
$X[i]$	Chemin i dans la population
$V[i]$	Nouveau chemin généré pour le chemin i
FitnessV	Fitness du nouveau chemin V
Trial[i]	Compteur de tentatives pour le chemin i
limit	Limite après laquelle un chemin est remplacé s'il n'a pas été amélioré
$P[i]$	Probabilité associée au chemin i
Random()	Fonction générant un nombre aléatoire entre 0 et 1

TABLE 3.2 – Notations utilisées dans l'algorithme SD-ABCA

Algorithm 2 SD-ABCA : la découverte de services basée sur l'algorithme ABCA

Require: Req(S_i, O_i)

Ensure: X_{Best}

Initialiser la population avec N chemins aléatoires respectant les contraintes des liens et des services.
Évaluer les fitness de chaque chemin dans la population initiale avec la formule 3.1.
Mémoriser le meilleur chemin dans la population initiale : $X_{\text{Best}} = X[1]$
for i de 2 à N **do**
    if Fitness[i] < Fitness(X_{Best}) **then**
        $X_{\text{Best}} = X[i]$
    end if
end for
while $t \leq$ MaxT **do**
    Phase des abeilles employées
    for i de 1 à N **do**
        Générer un nouveau chemin $V[i]$ à partir de $X[i]$ en appliquant la formule 3.2
    end for
    Calculer les fitness de tous les nouveaux chemins dans la population (formule 3.1).
    Garder le meilleur chemin entre le chemin actuel et le chemin candidat
    for i de 1 à N **do**
        if FitnessV[i] < Fitness[i] **then**
            $X[i] = V[i]$
            Fitness[i] = FitnessV[i]
            Trial[i] = 0
        else
            Trial[i] = Trial[i] + 1
        end if
    end for
    Calculer la probabilité (P_i) pour chaque chemin (X_i) avec la formule 3.3
    Phase des abeilles spectatrices
    for i de 1 à N **do**
        if Random() < $P[i]$ **then**
            Générer un nouveau chemin à partir de $X[i]$ en appliquant la formule 3.2
        end if
    end for
    Calculer les fitness de tous les nouveaux chemins dans la population (formule 3.1).
    Phase des abeilles éclaireuses
    Déterminer si un chemin a été abandonné, le remplacer par un nouveau chemin aléatoire si nécessaire
    for i de 1 à N **do**
        if Trial[i] > limit **then**
            $X[i]$ = Générer un chemin aléatoire respectant les contraintes des liens et des services
            Fitness[i] = Calculer les fitness de $X[i]$
            Trial[i] = 0
        end if
    end for
end while
Retourner le meilleur chemin (X_{Best}).

3.6 Évaluation des performances de l'algorithme SD-ABCA

Nous avons évalué l'algorithme proposé à travers une série de simulations détaillées. Nous débutons par la description de l'environnement de simulation utilisé pour tester l'algorithme. Ensuite, nous expliquons le jeu de données et les scénarios d'évaluation que nous avons utilisé afin de mesurer la performance de l'algorithme. Nous détaillons également les métriques considérées pour évaluer son efficacité. Enfin, nous présentons les résultats des simulations qui mettent en évidence les avantages de l'algorithme proposé, notamment en termes de réduction du nombre de sauts et d'amélioration du taux de succés.

3.6.1 Cadre de Simulation

Les tests ont été réalisés sur un ordinateur équipé d'un processeur Intel® Core™ i5-4300U avec une fréquence de 1.90 GHz (pouvant aller jusqu'à 2.50 GHz) et une mémoire RAM de 4 Go. L'ordinateur exécutait Matlab R2024a sur un système d'exploitation Windows 10 Entreprise 64 bits.

Les paramètres utilisés pour notre simulation, comme indiqué dans le tableau 3.3, jouent un rôle essentiel dans l'adaptation de l'algorithme de colonie d'abeilles artificielles à la découverte de services IoT. La portée de transmission R de 25 mètres détermine la distance maximale de communication des objets IoT, assurant une couverture réseau adéquate. Le nombre maximum d'itérations T de 100 limite le temps d'exécution tout en permettant une exploration approfondie des solutions. Avec 100 abeilles, l'algorithme explore robustement l'espace de recherche tout en évitant une convergence prématurée vers des solutions locales. Enfin, le nombre de services par objet (de 1 à 3) reflète la diversité des capacités des objets IoT, testant ainsi l'efficacité de l'algorithme dans différentes configurations. Cette combinaison optimise l'algorithme pour relever les défis de la découverte de services IoT, garantissant efficacité et adaptabilité.

Paramètre	Valeur
Portée de transmission R	25m
Nombre maximum d'itérations T	100
Nombre d'abeilles	100
Nombre de services offerts par objet	1-3

TABLE 3.3 – Paramètres de simulation

3.6.2 Ensemble de données et cas d'évaluation

Pour évaluer les performances de notre approche, nous avons utilisé une série de données générées de manière aléatoire, appliquées à divers scénarios de découverte de services. Ces scénarios ont été construits en variant le nombre d'objets, le nombre de services dans la requête, ainsi que le nombre d'objets défaillants.

3.6.3 Mesures et méthodologie de simulation

Nous comparons les performances de l'algorithme proposé à celles de la méthode de diffusion (Flooding). Les critères utilisés pour évaluer ces performances sont les suivants :

Nombre de sauts : Ce critère mesure le nombre de sauts nécessaires pour que l'algorithme atteigne les services demandés. Un chemin optimal est caractérisé par le moins de sauts possible.

Taux de succès : Ce critère indique le pourcentage de requêtes découvertes par rapport au nombre total de requêtes envoyées.

3.6.4 Analyse des résultats de simulation

Cette section présente une comparaison du nombre de sauts entre les approches étudiées. Nous avons mesuré le nombre de sauts en fonction du nombre d'objets, en réalisant des tests variés pour chaque approche avec des configurations comprenant 50, 100, 300 et 500 objets, ainsi que des requêtes contenant 1, 2 ou 3 services. Toutes les mesures sont basées sur les résultats moyens obtenus à partir de 100 exécutions successives.

Le nombre de sauts

Les Figures 3.2, 3.3 et 3.4 mettent en évidence que l'augmentation du nombre d'objets entraîne une comparaison du nombre de sauts effectués pour la découverte de services entre les approches Flooding et SD-ABCA, avec une légère infériorité pour Flooding. Toutefois, dans l'approche Flooding, le réseau peut se saturer en raison des multiples rediffusions nécessaires par les objets intermédiaires pour relayer le message de l'objet source à sa destination. Cela peut conduire à une augmentation du taux de collisions et de la consommation d'énergie.
Par contraste, notre approche maintient la scalabilité du réseau et ses performances, même avec un nombre élevé d'objets.

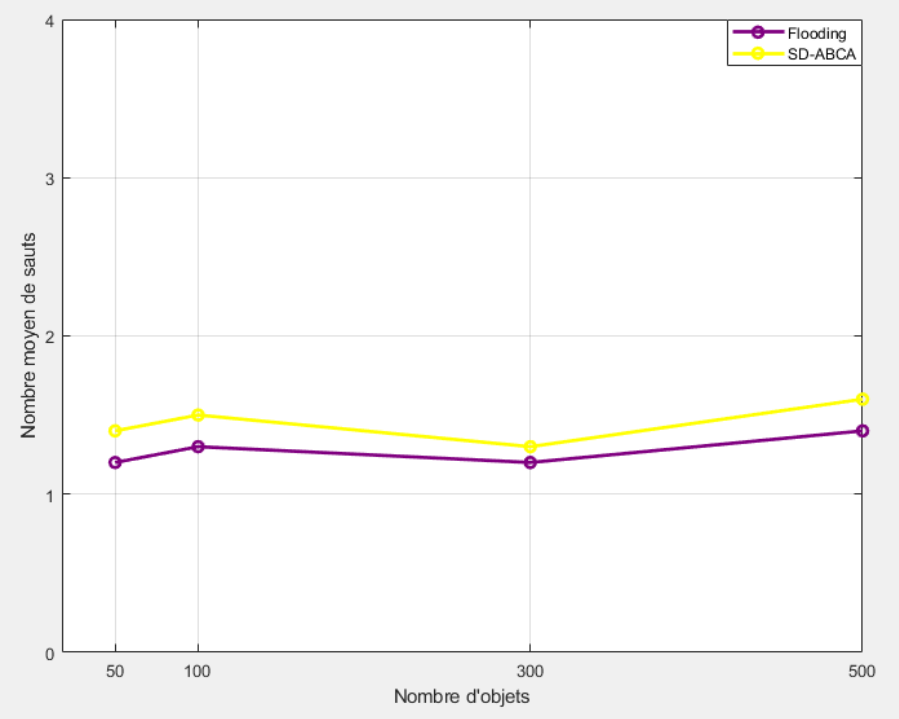

FIGURE 3.3 – Nombre moyen de sauts effectués lors de la découverte de services pour les approches SD-ABCA et Flooding avec 1 service dans la requête

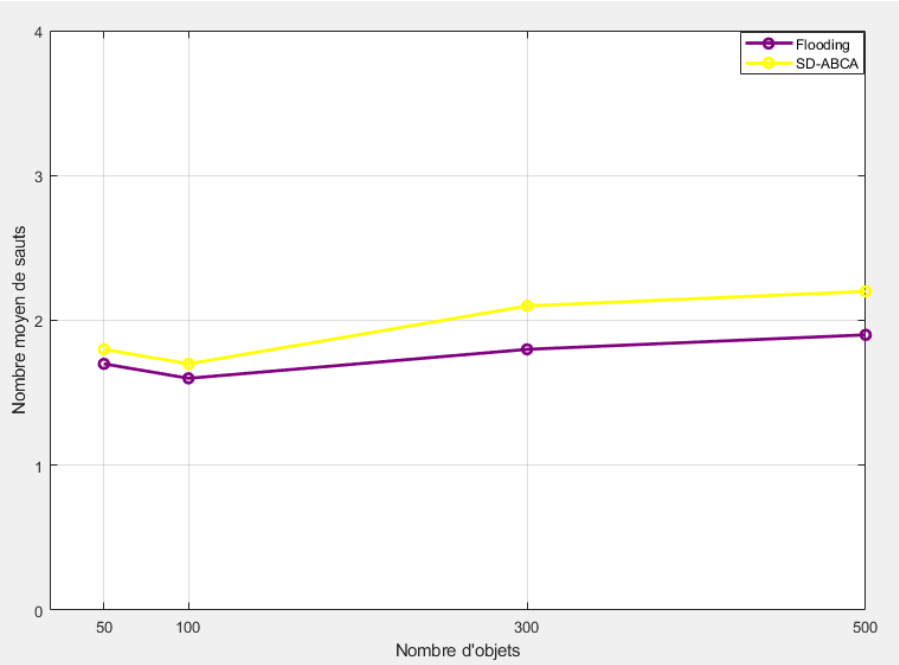

FIGURE 3.4 – Nombre moyen de sauts effectués lors de la découverte de services pour les approches SD-ABCA et Flooding avec 2 services dans la requête

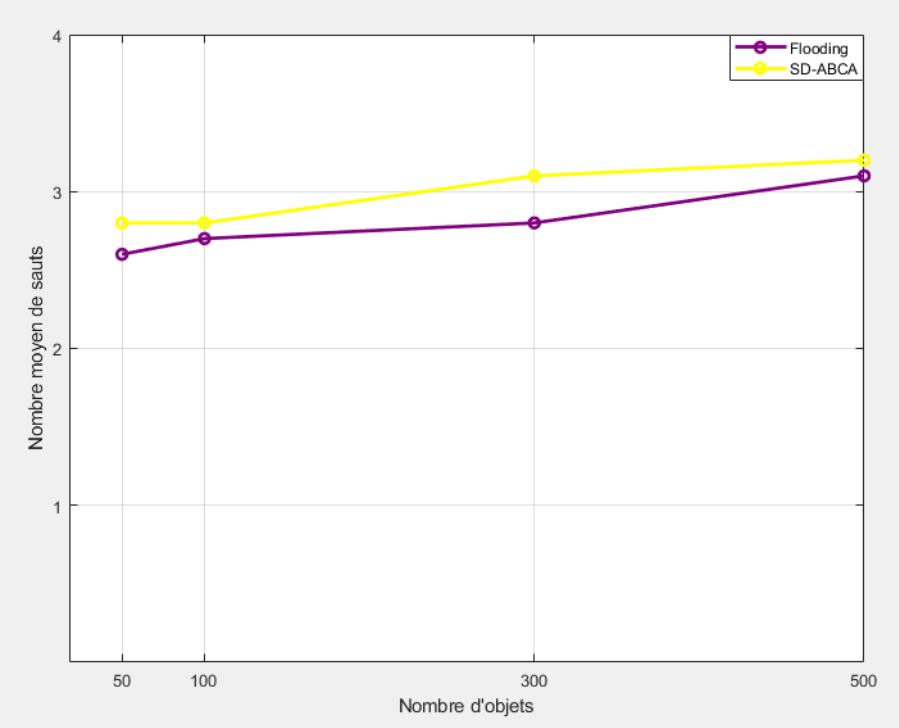

FIGURE 3.5 – Nombre moyen de sauts effectués lors de la découverte de services pour les approches SD-ABCA et Flooding avec 3 services dans la requête

Le taux de succès

Dans cette section, nous examinons le taux de succès des approches SD-ABCA et Flooding, en tenant compte des objets défaillants dans notre système. Nous avons mené divers tests pour ces deux méthodes en variant le pourcentage d'objets défaillants $(20\%, 30\%, 40\%, 50\%)$ et en utilisant une requête comprenant cinq services. Toutes les mesures de simulation sont basées sur les résultats moyens obtenus à partir de 100 exécutions consécutives.

Les Figures 3.5, 3.6 et 3.7 illustrent que le taux de succès diminue pour les deux approches à mesure que le nombre d'objets défaillants augmente. Nous constatons que notre méthode présente des résultats comparables à ceux de l'approche Flooding.

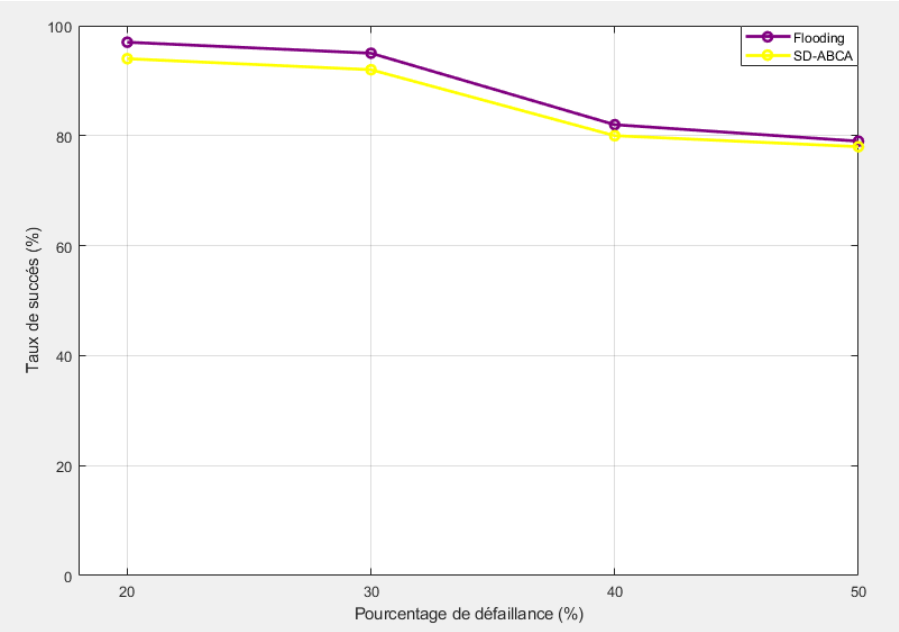

FIGURE 3.6 – Taux du succès avec 50 objets

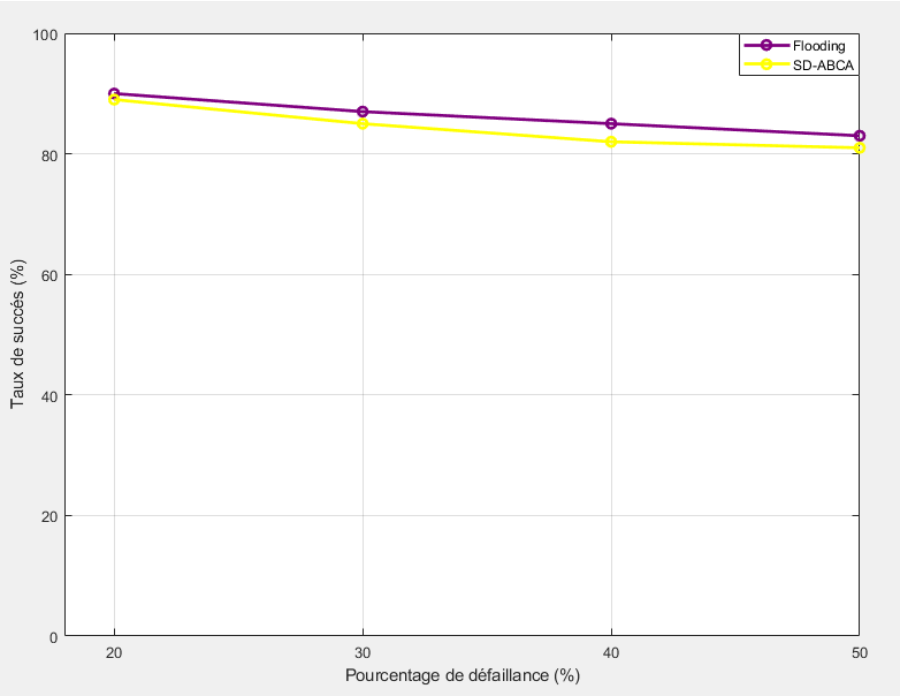

FIGURE 3.7 – Taux du succès avec 100 objets

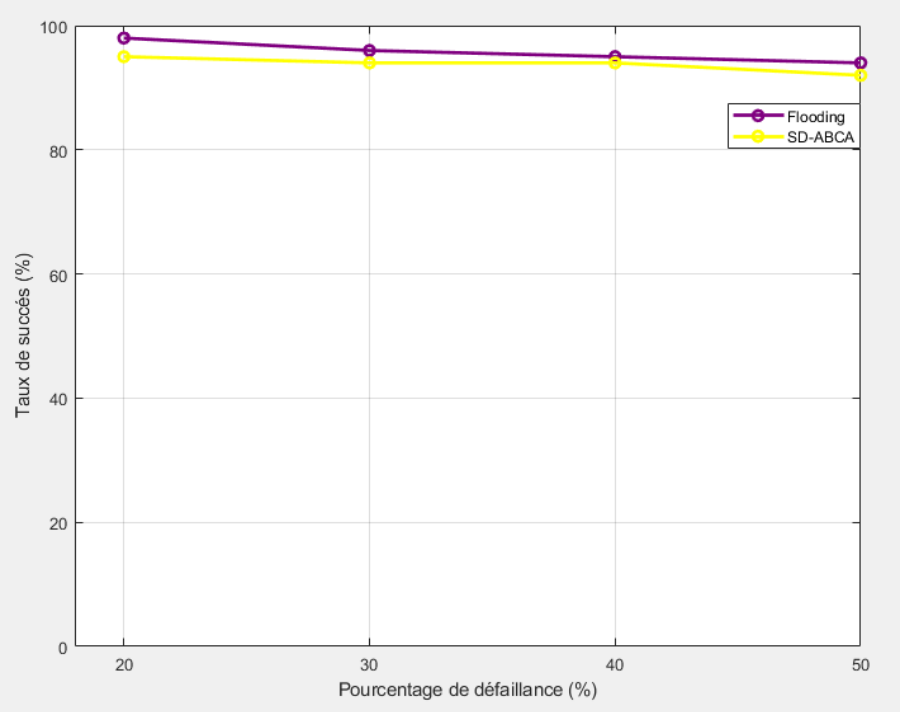

FIGURE 3.8 – Taux du succès avec 500 objets

3.6.5 Évaluation comparative de SD-ABCA et Flooding dans le contexte de la découverte de services IoT

Nos simulations ont démontré que l'efficacité de la découverte de services à l'aide de SD-ABCA est comparable à celle de l'algorithme Flooding en termes de taux de succès et de nombre de sauts nécessaires. Cependant, notre approche présente un avantage significatif : elle ne nécessite pas de table de routage, mais seulement la maintenance d'une liste de voisins. Cela se traduit par une réduction du nombre de messages circulant dans le système. En revanche, l'approche Flooding repose sur une stratégie d'inondation, ce qui n'est pas idéal pour les objets dotés de ressources limitées. De plus, notre modèle décentralisé facilite une mise à l'échelle plus aisée vers des environnements plus vastes.

3.7 Conclusion

Dans ce chapitre, nous avons exposé notre proposition, dans laquelle nous avons adapté la métaheuristique ABCA à la découverte de services. Nous avons choisi une approche décentralisée pour une meilleure mise à l'échelle des services et des objets. Les résultats de la simulation montrent une amélioration des performances en termes de nombre de sauts et de taux de succès.

Conclusion générale et perspectives

Dans ce mémoire, nous avons exploré en profondeur le domaine dynamique de l'Intenet des Objets à travers trois chapitres distincts, chacun apportant une contribution significative à notre compréhension et à notre capacité à optimiser les systèmes IoT.

Le premier chapitre a posé les fondements essentiels en examinant les concepts fondamentaux de l'IoT et les méthodes d'optimisation pertinentes. Nous avons discuté en détail des diverses applications de l'IoT et des stratégies clés pour résoudre les défis spécifiques à ce domaine en pleine expansion.

Le deuxième chapitre s'est concentré sur un aspect important de l'IoT : la découverte de services. À travers l'exploration de différentes architectures et approches, telles que l'Architecture Orientée Services (SOA) et les principes de REST, nous avons examiné comment identifier et localiser efficacement les services offerts par les objets connectés. Ce chapitre a souligné l'importance des mécanismes de découverte de services pour exploiter pleinement le potentiel de l'IoT dans différents contextes d'application.

Le troisième chapitre a présenté notre contribution qui consiste à adapter l'algorithme d'optimisation par colonie d'abeilles artificielles à la découverte de services IoT en mode discret (SD-ABCA). Nous avons détaillé le développement spécifique de cet algorithme, ses principes sous-jacents et les résultats de son évaluation comparative, démontrant son efficacité accrue dans la réduction du nombre de sauts nécessaires pour la découverte de services, ainsi que sa pertinence pour les environnements IoT réels.

Pour l'avenir, plusieurs perspectives stimulantes s'ouvrent. L'exploration d'autres métaheuristiques inspirées de la nature, comme celles observées dans les écosystèmes marins, pourrait enrichir nos approches. L'hybridation des métaheuristiques représente également une voie prometteuse à explorer.

Nous proposons d'effectuer des simulations et des tests en utilisant des services IoT réels. L'intégration de la mobilité des nœuds et la prise en compte de la qualité de service émergent comme des pistes de recherche importantes. L'ajout de contraintes supplémentaires, telles que la bande passante limitée, pourrait rendre nos modèles plus pertinents pour les environnements IoT réels.

Enfin, nous revenons à l'approche de découverte des services proposée, nous proposons l'implémentation et le test de prototype dans un réseau largement distribué. Cela inclurait l'intégration de l'algorithme de la colonie d'abeilles artificielles dans un cadre plus vaste de gestion des services IoT. ouvrant ainsi de nouvelles possibilités pour l'optimisation et la fiabilité des réseaux IoT avancés.

Bibliographie

[1] Cluster of European Research Projects on the Internet of Things. Vision and Challenges for Realising the Internet of Things. Mars 2010.

[2] Yacine Challal. « Sécurité de l'Internet des Objets : vers une approche cognitive et systémique ». Réseaux et télécommunications [cs.NI]. Thèse de doctorat. Compiègne, France : Université de Technologie de Compiègne, 2012.

[3] International Telecommunication Union. The Internet of Things. Report. Nov. 2005.

[4] Ala Al-Fuqaha et al. « Internet of Things : A Survey on Enabling Technologies, Protocols, and Applications ». In : IEEE Communications Surveys & Tutorials 17.4 (2015), p. 2347-2376. doi : 10.1109/COMST.2015.2444095. url : http://dx.doi.org/10.1109/COMST.2015.2444095.

[5] J. A. Stankovic. « Wireless sensor networks ». In : IEEE Computer Society 41.10 (2008), p. 92-95.

[6] N. Daniel, R. Marcel et K. Daniel. Livre blanc Machine To Machine enjeux et perspectives. Orange Business Services, Syntec informatique, Fing, 2006, p. 40.

[7] Younes Abbassi et Habib Benlahmer. « Un aperçu sur la sécurité de l'internet des objets (IOT) ». In : ffhal-03593723. IUT d'Aix-Marseille. Marseille, France, mars 2021.

[8] Alain COULON. L'Internet des Objets Un gisement à exploiter. La Lettre d'ADELI n°78 – Hiver 2010. (PDF, 5 p [page 26 à 30]). 2010.

[9] A. El Dor. « Perfectionnement des algorithmes d'Optimisation par Essaim Particulaire. Applications en segmentation d'images et en électronique ». Thèse de doctorat. France : École doctorale Mathématiques et STIC, Université Paris-Est, 2012.

[10] Michel Sakarovitch. Optimisation combinatoire : Méthodes mathématiques et algorithmiques. 270 pages. Paris : Dunod, 1984.

[11] Patrick Siarry et Yann Collette. Optimisation multiobjectif. Eyrolles, 2002.

[12] Amira Gherboudj. « Méthodes de résolution de problèmes difficiles académiques ». Thèse de doct. 2013.

[13] Edward A. Feigenbaum et Julian Feldman. Computers and Thought. New York, NY, USA : McGraw-Hill, Inc., 1963.

[14] Ibrahim H. Osman et Gilbert Laporte. Metaheuristics : A General Framework for Combinatorial Optimization Problems. 1996.

[15] Yassine Hammal et al. « Formal techniques for consistency checking of orchestrations of semantic Web services ». In : Journal of Computational Science 44 (2020), p. 101165. doi : 10.1016/j.jocs.2020.101165. url : http://dx.doi.org/10.1016/j.jocs.2020.101165.

[16] Sana Ben Fredj et al. « Efficient semantic-based IoT service discovery mechanism for dynamic environments ». In : Washington DC, USA : IEEE, sept. 2014, p. 2088-2092. doi : 10.1109/PIMRC.2014.7136516. url : http://dx.doi.org/10.1109/PIMRC.2014.7136516.

[17] Pedro Gomes et al. « A Federated Discovery Service for the Internet of Things ». In : Vancouver, BC, Canada : ACM, déc. 2015, p. 25-30. doi : 10.1145/2836127.2836129. url : http://dx.doi.org/10.1145/2836127.2836129.

[18] Sorin Chirila, Camelia Lemnaru et Mihaela Dînsoreanu. « Semantic-based IoT device discovery and recommendation mechanism ». In : Cluj-Napoca, Romania : IEEE, sept. 2016, p. 111-116. doi : 10.1109/ICCP.2016.7737131. url : http://dx.doi.org/10.1109/ICCP.2016.7737131.

[19] Bin JIA, Wei LI et Tao ZHOU. « A Centralized Service Discovery Algorithm via Multi-Stage Semantic Service Matching in Internet of Things ». In : t. 1. Guangzhou, China : IEEE Computer Society, juill. 2017, p. 422-427. DOI : 10.1109/CSE-EUC.2017.82. URL : http://dx.doi.org/10.1109/CSE-EUC.2017.82.

[20] Bo YUAN, Liang LIU et Nick ANTONOPOULOS. « Efficient service discovery in decentralized online social networks ». In : Future Generation Computer Systems 86 (2018), p. 775-791. DOI : 10.1016/j.future.2017.04.022. URL : http://dx.doi.org/10.1016/j.future.2017.04.022.

[21] Hongyu XIA et al. « An efficient social-like semantic-aware service discovery mechanism for large-scale Internet of Things ». In : Computer Networks 152 (2019), p. 210-220. DOI : 10.1016/j.comnet.2019.02.006. URL : http://dx.doi.org/10.1016/j.comnet.2019.02.006.

[22] Haiyan LI et al. « Study and Application of Urban Flood Risk Map Information Management System Based on SOA ». In : Journal of Software 10.2 (2015), p. 180-189.

[23] Bassam SOUKKARIEH. « Technique de l'internet et ses langages : vers un système d'information web restituant des services web sensibles au contexte ». Thèse de doctorat. Toulouse, France : Université de Toulouse, Université Toulouse III-Paul Sabatier, 2010.

[24] R. T. FIELDING. « Architectural Styles and the Design of Network-based Software Architectures ». PhD thesis in Information and Computer Science. Thèse de doct. Irvine, CA : University of California, Irvine, 2000.

[25] Abdr ALOSMAN. What is Web API ? Types, Usage, and Examples. https://medium.com/@abdr.alosman/what-is-web-api-types-usage-and-examples-d5557b216c29. 2023.

[26] Meriem ACHIR, Abdelkrim ABDELLI et Lynda MOKDAD. « A taxonomy of service discovery approaches in IoT ». In : 2020. DOI : 10.1109/WINCOM50532.2020.9272512.

[27] Andrea CIMMINO, María POVEDA-VILLALÓN et Raúl GARCÍA-CASTRO. « eWoT : A Semantic Interoperability Approach for Heterogeneous IoT Ecosystems Based on the

Web of Things ». In : Sensors 20.17 (2020), p. 4822. DOI : 10.3390/s20174822. URL : https://doi.org/10.3390/s20174822.

[28] Porfírio GOMES et al. « A semantic-based discovery service for the Internet of Things ». In : Journal of Internet Services and Applications 10.1 (2019), p. 1-22. DOI : 10.1186/s13174-019-0108-8. URL : https://doi.org/10.1186/s13174-019-0108-8.

[29] S. ZHAO et al. « IoT service clustering for dynamic service matchmaking ». In : Sensors 17.8 (2017), p. 1727.

[30] Zahia Bensalah AZIZOU, Abdelmalek BOUDRIES et Mourad AMAD. « Grey Wolf Optimizer-Based Decentralized Service Discovery in the Internet of Things Applications ». In : Int. J. Sens. Wirel. Commun. Control (). DOI : 10.2174/0122103279252457231018060854.

[31] Zahia Bensalah AZIZOU, Abdelmalek BOUDRIES et Mourad AMAD. « Decentralized Service Discovery and Localization in Internet of Things Applications Based on Ant Colony Algorithm ». In : International Journal of Computing and Digital Systems 9.5 (sept. 2020). DOI : 10.12785/ijcds/090514. URL : http://dx.doi.org/10.12785/ijcds/090514.

[32] E. RAPTI et al. « Decentralized service discovery and selection in internet of things applications based on artificial potential fields ». In : Service Oriented Computing and Applications 11.1 (2017), p. 75-86.

[33] C. P. VANDANA et A. A. CHIKKAMANNUR. « S-COAP : Semantic Enrichment of COAP for Resource Discovery ». In : SN Computer Science 1.1 (2020), p. 1-11. DOI : 10.1007/s42979-019-0010-1. URL : https://doi.org/10.1007/s42979-019-0010-1.

[34] E. M. PEREIRA et al. « MQTT-RD : A MQTT based resource discovery for machine to machine communication ». In : 2019, p. 115-124.

[35] Moutaz MAHYOUB, Ahmed MAHMOUD et Tariq SHELTAMI. « An Optimized Discovery Mechanism for Smart Objects in IoT ». In : 2017, p. 312-318. DOI : 10.1109/IEMCON.2017.8117210. URL : https://doi.org/10.1109/IEMCON.2017.8117210.

[36] Ming TAO, Wei WEI et Shilin HUANG. « Location-based trustworthy services recommendation in cooperative-communication-enabled Internet of Vehicles ». In : Network and

Computer Applications 126 (2019), p. 1-11. DOI : 10.1016/j.jnca.2018.10.017. URL : https://doi.org/10.1016/j.jnca.2018.10.017.

[37] M. PRUTHVI, S. KARTHIKA et N. BHALAJI. « A novel framework for SIOT college ». In : IEEE. 2019, p. 1-4.

[38] Amine YACHIR et al. « Event-Aware Framework for Dynamic Services Discovery and Selection in the Context of Ambient Intelligence and Internet of Things ». In : IEEE Transactions on Automation Science and Engineering 13.1 (2016), p. 85-102. DOI : 10.1109/TASE.2015.2497313. URL : https://doi.org/10.1109/TASE.2015.2497313.

[39] Xiao LIU et Yun DENG. « A new QoS-aware service discovery technique in the Internet of Things using whale optimization and genetic algorithms ». In : Journal of Engineering and Applied Science 71.4 (2024). DOI : 10.1186/s44147-023-00334-1. URL : https://doi.org/10.1186/s44147-023-00334-1.

[40] Ali H. AHMED, Nagwa M. OMAR et Hosny M. IBRAHIM. « Secured Service Discovery Technique in IoT ». In : Journal of Communications 14.1 (2019), p. 40-46. DOI : 10.12720/jcm.14.1.40-46.

[41] H. LEE et al. « IoT service store : A web-based system for privacy-aware IoT service discovery and interaction ». In : IEEE. 2018, p. 107-112.

[42] Pei-Wei TSAI et al. « Enhanced Artificial Bee Colony Optimization ». In : International Journal of Innovative Computing, Information and Control 5.12 (déc. 2009), 1–ISII08-247.

[43] Dervis KARABOGA et Bahriye BASTURK. « A powerful and efficient algorithm for numerical function optimization : artificial bee colony (ABC) algorithm ». In : Journal of Global Optimization 39 (2007), p. 459-471. DOI : 10.1007/s10898-007-9149-x.

[44] Mustafa SONMEZ. « Artificial Bee Colony algorithm for optimization of truss structures ». In : Elsevier (). Department of Civil Engineering, School of Engineering, Aksaray University, Aksaray 68100, Turkey.

[45] Jun-Hao LIANG et Ching-Hung LEE. « A Modification Artificial Bee Colony Algorithm for Optimization Problems ». In : Mathematical Problems in Engineering (2015). DOI : 10.1155/2015/581391. URL : http://dx.doi.org/10.1155/2015/581391.

[46] Weifeng Gao, Sanyang Liu et Lingling Huang. « A global best artificial bee colony algorithm for global optimization ». In : Elsevier (). Department of Applied Mathematics, Xidian University, Xi'an 710071, China.

[47] Efrén Mezura-Montes et Omar Cetina-Domínguez. « Empirical Analysis of a Modified Artificial Bee Colony for Constrained Numerical Optimization ». In : (). Laboratorio Nacional de Informática Avanzada (LANIA A.C.), Rébsamen 80, Centro, Xalapa, Veracruz, 91000, Mexico.

[48] Thomas Alves et al. « Modèle d'Interruption pour la Répartition des Tâches : Application à une simulation de colonie d'abeilles ». In : Angers, France, juin 2020. URL : https://hal.archives-ouvertes.fr/hal-03211869.

[49] Efrén Mezura-Montes, Mauricio Damián-Araoz et Omar Cetina-Domínguez. « "Smart Flight and Dynamic Tolerances in the Artificial Bee Colony for Constrained Optimization" ». In : Sep. (2010).

[50] Dervis Karaboga et Bahriye Basturk. « Artificial Bee Colony (ABC) Optimization Algorithm for Solving Constrained Optimization Problems ». In : 2007, p. 789-798.

[51] D. Karaboga. An idea based on honey bee swarm for numerical optimization. Technical Report TR06. Erciyes, Turkey : Erciyes University, Engineering Faculty, Computer Engineering Department, 2005.

[52] V. Tereshko et A. Loengarov. « Collective decision-making in honey bee foraging dynamics ». In : Computing and Information Systems 9.3 (2005), p. 1-7.

[53] Yi-Hsuan Chen et al. « A Spatiotemporal-Oriented Deep Ensemble Learning Model to Defend Link Flooding Attacks in IoT Network ». In : Sensors 21.4 (2021), p. 1027.

[54] Tie Qiu et al. « A Search Strategy of Level-Based Flooding for the Internet of Things ». In : Sensors 12.8 (2012), p. 10163-10195. DOI : 10.3390/s120810163. URL : http://www.mdpi.com/journal/sensors.

[55] O. Mendoza-Cano et al. « Experiments of an IoT-based wireless sensor network for flood monitoring in Colima, Mexico ». In : Journal of Hydroinformatics 23.3 (2021),

p. 385-401. DOI : 10.2166/hydro.2021.126. URL : https://doi.org/10.2166/hydro.2021.126.

[56] M. Shoyeb SAYYAD et al. « IoT Based Early Flood Detection and Avoidance ». In : IRE Journals 3 (12 2020), p. 50.

[57] Wen-Tsai SUNG, Ihzany Vilia DEVI et Sung-Jung HSIAO. « Early warning of impending flash flood based on AIoT ». In : Journal of Wireless Communications and Networking 2022.15 (2022). DOI : 10.1186/s13638-022-02096-5. URL : https://doi.org/10.1186/s13638-022-02096-5.

[58] Emmanuel BACCELLI et Philippe JACQUET. Flooding Techniques in Mobile Ad Hoc Networks. Research Report RR-5002. inria-00077040f. INRIA, 2003.