N°d'ordre: ...../2022

République Algerienne Démocratique et Populaire
Ministère de l'enseignement supérieur et de la Recherche scientifique
Université Akli Mohand Oulhadj de Bouira
Laboratoire LIMPAF

Faculté des Sciences et Sciences appliquées

Thèse de Doctorat
En Sciences

Présentée par :

Mme BENSALAH Née AZIZOU Zahia

Filière : Informatique
Option : Informatique

Découverte et localisation de services dans l'Internet des Objets

---

Soutenu publiquement, le 22 / Juin / 2022 , devant le jury composé de :

M.	BENNOUAR Djamal	Professeur UAMOB	Président du jury
M.	BOUDRIES Abdelmalek	MCA Université Béjaia	Directeur de thèse
M.	AMAD Mourad	Professeur UAMOB	Co-Directeur de thèse
M.	SEMCHEDDINE Fouzi	Professeur Université sétif 1	Examinateur
M.	BAADACHE Abderrahmane	Professeur Université Alger 1	Examinateur
M.	ABBAS Akli	MCA UAMOB	Examinateur

Année Universitaire : 2021 - 2022

⁕ *Remerciements* ⁕

En premier lieu, je tiens à exprimer ma profonde gratitude à mon directeur de thèse, Mr Abdelmalek BOUDRIES maitre de conférence de l'Université de Béjaia, qui m'a donné la chance de réaliser ces travaux et de les mener à bien ainsi que pour la confiance qu'il m'a accordée durant toutes ces années.

Je tiens à remercier également Monsieur Mourad AMAD professeur à l'université de Bouira qui a codirigé ma thèse, pour ses orientations scientifiques précieuses et pour ses remarques enrichissantes.

J'adresse mes sincères remerciements à Monsieur Djamal BENNOUAR, Professeur à l'Université de Bouira qui m'a fait l'honneur de présider le jury de cette thèse.

J'exprime ma profonde reconnaissance à Monsieur Fouzi SEMCHEDDINE Professeur à l'université Sétif 1, Monsieur Abderrahmane BAADACHE Professeur à l'université d'Alger 1, et Monsieur Akli ABBAS maitre de conférence à université de Bouira pour l'intérêt qu'ils ont bien voulu porter à ce travail en acceptant de l'examiner.

Je remercie chaleureusement mon adorable époux pour son soutien, son écoute et ses encouragements tout au long de cette thèse sans oublier nos deux chers anges Ilyane et Lya.

Mes vifs remerciements vont à toutes les personnes ayant contribué de près ou de loin au bon déroulement et à l'aboutissement de cette thèse, tant sur le plan professionnel que sur le plan personnel.

⁂ *Dédicaces* ⁂

À la mémoire de ma chére grand-mère maternelle et mon cher père ;
À ma mère ;
À mon mari ;
À mes petits enfants ;
À toute ma famille ;
À tous ceux qui me sont chers.

RÉSUMÉ

Aujourd'hui, l'Internet des Objets permet de connecter des milliards de dispositifs hétérogènes, qui génèrent un grand nombre de services. Cependant ces objets ont des contraintes en ressources, en particulier une limitation de la taille de la mémoire et de la batterie. Ces contraintes et la prolifération du nombre d'objets connectés affectent considérablement la découverte de leurs services. Les divergences des caractéristiques exigent de nouvelles méthodes intelligentes afin d'assurer la découverte de services. Les solutions heuristiques deviennent obsolètes ou impuissantes pour satisfaire la requête d'un l'utilisateur, d'où une recherche de nouvelles méthodes devient obligatoire. Parmi ces solutions, nous avons celles basées sur les modèles bio-inspirés. Dans ce contexte, pour minimiser le nombre de sauts effectués, augmenter le taux du succès, et pour un meilleur passage à l'échelle, nous proposons deux approches bio-inspirées pour la découverte de services, elles reposent sur une architecture décentralisée. La première consiste à adapter la métaheuristique ACA. Une deuxième consiste à adapter la métaheuristique GWO. Les résultats expérimentaux montrent que les métaheuristiques proposées offrent de meilleures performances.

Mots clés : Découverte de services ; Internet des Objets ; Ant Colony Algorithm (ACA) ; Gray Wolf Optimizer (GWO) ; Approche décentralisée.

ABSTRACT

Today the Internet of Things connects billions of heterogeneous devices which generate a large number of services. However, these objects have resource constraints, in particular a memory size and battery limitation. These constraints and the proliferation of the number of connected objects significantly affect discovery of their services. Divergent features require new, intelligent methods to ensure service discovery. Heuristic solutions become obsolete or powerless to satisfy a user's query. Hence the need for an appropriate service discovery mechanisms to search services. Among these solutions, we have those based on bio-inspired models. In this context, to minimize the number of hops performed, increase the success rate and improve scalability, we suggest two bio-inspired approaches for service discovery ; they are based on a decentralized architecture. The first consists in adapting the ACA metaheuristics. The second consists in adapting the GWO metaheuristics. The experimental results show that the proposed metaheuristics provide better performance.

Key words : Service discovery ; Internet of Things ; Ant Colony Algorithm (ACA) ; Gray Wolf Optimizer (GWO) ; Decentralized approach.

ملخص

اليوم، يتيح إنترنت الأشياء توصيل مليارات الأجهزة غير المتجانسة، والتي توفر عددًا كبيرًا من الخدمات. إن هذه الأشياء لديها قيود في خصائصها ولا سيما الحد من الذاكرة والبطارية تؤثر هذه القيود وانتشار عدد الأشياء المتصلة بشكل كبير على اكتشاف خدماتها. تتطلب الاختلافات في الخصائص طرقًا جديدة وذكية لضمان اكتشاف الخدمة. تصبح الحلول الاستكشافية قديمة عاجزة عن تلبية طلبات المستخدم، وبالتالي يصبح البحث عن طرق جديدة إلزاميًا. من بين هذه الحلول، لدينا تلك التي تعتمد على الخوارزميات. في هذا السياق، لتقليل عدد القفزات التي يتم إجراؤها، وزيادة معدل النجاح، ولتحسين قابلية التوسع، تم اقتراح طريقتين لمعالجة هذه المشكلة واللتان تعتمدان على بنية لامركزية، الأولى هي تكييف خوارزمية مستعمرة النمل (ACA)أما في المساهمة الثانية نقترح تكيف خوارزمية الذئاب الرمادية (GWO) .

أظهرت النتائج التجريبية أن الخوارزميات المقترحة توفر أداءً أفضل لإكتشاف خدمات.

كلمات البحث: اكتشاف الخدمات، انترنت الأشياء، خوارزمية مستعمرة النمل (ACA)، خوارزمية الذئاب الرمادية(GWO)، النهج اللامركزي .

Table des matières

Table des figures

Liste des algorithmes

Liste des tableaux

Liste des abréviations

ABC Artificial Bee Colony optimization

ACA Ant Colony Algorithm

ACO Ant Colony Optimization

AE Algorithmes Evolutionnaires

AISA Artificial Immune System Algorithm

ALO Ant Lion Optimizer

AMFSS Adaptive Momentum Factor BP'S Service

ANN Artificial Neural Network

APFs Artificial Potential Fields

AS Ant System

BBO Biogeography-Based Optimization

BFOA Bacterial Foraging Optimization Algorithm

CoAP Constrained Application Protocol

COP Constraint Optimization Problems

CS Cuckoo Search

DHT Distributed Hash Table

DM Descent Method

DNS Domain Name System

DPWS Devises Profile for Web Services

DSDP Distributed Service Discovery Protocol

FA Firefly Algorithm

FSA Fish Swarm Algorithm

GA Genetic Algorithm

GPS Positioning System

GRASP Greedy Randomized Adaptive Search Procedure

GWO Gray Wolf Opimizer

HTTP Hyper Text Markup Language

IETF Internet Engineering Task Force

IoT Internet of Things

IPV Internet Protocol version 6

LSD Lightweight Service Discovery

MDM Multiple Dimensional Measuring

mDNS multicast DNS

mDNS-SD multicast DNS Service Discovery

MQTT Message Queuing Telemetry Transport

MQTT-RD MQTT Resource Discovery

MQTT-SN MQTT for Sensor Network

NFC Near Field Communication

OASIS Organization for the Advancement of Structured Information Standards

PSO Particle Swarm Optimization

REST Representational State Transfer

RFID Radio Frequency IDentification

S-CoAP Semantic Enrichment of CoAP

SFLA Shuffled Frog Leaping Algorithm

SI Swarm Intelligence

SLP Service Location Protocol

SOA Service Oriented Architecture

SOAP Simple Object Access Protocol

UDDI Universal Description, Discovery, and Integration

URI Uniform Resource Identifier

USPIOT Ubiquitous Services Platform for IoT

WSN Wireless Sensor Network

XML Extensible Markup Language

XMPP Extensible Messaging and Presence Protocol

Introduction générale

Avec l'évolution croissante du nombre d'objets connectés répartis dans des emplacements physiques, appelés des espaces intelligents (smart home, Smart Cities, etc.), les réseaux du futur doivent évoluer vers de nouvelles architectures. La majorité des objets connectés sont mobiles et peuvent se déplacer d'un espace intelligent vers un autre. Pour assurer une meilleure accessibilité et exploitation des objets connectés du monde réel et faire un lien entre le monde physique et le monde virtuel, un nouveau paradigme émerge, et appeler l'Internet des Objets ou en anglais Internet of Things (IoT).

Cependant, dans l'Internet des Objets, le nombre d'appareils connectés augmente au fur et à mesure, et chaque objet génère un ou plusieurs services nommés les services de l'Internet des Objets (services de l'IoT). Un service est définit comme un ensemble de fonctionnalités qui permettent l'accès, la recherche et le traitement de l'information, et aussi la possibilité de communiquer avec d'autres utilisateurs ou d'autres applications. Dans IoT, l'utilisateur sera donc entouré par un grand nombre de services offerts par ses objets connectés et aura besoin de les trouver pour interagir avec eux. Cependant, les environnements de l'Internet des Objets sont caractérisés par le nombre volumineux des objets connectés, leurs mobilités, leurs hétérogénéités, et leurs limitations de ressources. Ces caractéristiques rendent la découverte et la localisation de services difficile.

Considérant ce contexte, le mécanisme de découverte doit également répondre à certaines exigences pour être efficace. En effet, avec un large volume de services IoT, il serait difficile pour l'utilisateur de connaitre les caractéristiques de tous les services qui peuvent répondre à ses besoins. Ainsi, le mécanisme de découverte serait capable de retourner des services qui correspondent à la requête demandée par l'utilisateur. L'IoT est composé de milliards d'appareils connectés offrant des services, pour faire face à la très forte augmentation, il nécessite une approche qui permet une meilleure mise à l'échelle des services et des objets. Enfin, le mécanisme de découverte doit pouvoir être extensible par rapport à un nombre volumineux de services et d'objets considérés. Sur la base de ce qui précède, la question que nous nous posons est : Comment construire un mécanisme de recherche efficace qui répond à ces exigences tout en prenant en compte les contraintes de l'enivrement IoT ?

Le problème de découverte de services est l'un des plus important challenge dans l'Internet des Objets. La découverte de services est considérée comme un problème d'optimisation.
Pour résoudre et surmonter le problème de découverte de services, nous avons opté pour l'utilisation des métaheuristiques. Les métaheuristiques constituent une famille d'algorithmes inspirés de la nature. Ces algorithmes sont particulièrement utiles pour résoudre des problèmes où les algorithmes d'optimisation classiques sont incapables de produire des résultats satisfaisants. Parmi ces méthodes, nous avons utilisé l'algorithme de colonies de fourmis (ACA) et l'algorithme des loups gris (GWO) pour optimiser le nombre de sauts effectués et augmenter le taux du sucés. Ces deux métaheuristiques (ACA, GWO) sont connues pour leurs performances dans la résolution des problèmes d'optimisation et nous les avons adaptés à un système décentralisé pour une meilleure mise à échelle des services et des objets.

La première contribution [13], est basée sur l'adaptation de la métaheuristique ACA et l'étude des capacités de cette métaheuristique pour la résolution du problème de la découverte de services. Pour ce faire, il est indispensable de choisir la stratégie phéromonale appropriée.

La deuxième contribution, est basée sur l'adaptation de la métaheuristique GWO pour la résolution du problème de la découverte de services. La version initiale de GWO est utilisée pour les problèmes d'optimisation avec un espace de recherche continues (réels). Notre approche propose une version discrète adaptée au problème de découverte de services.

Ce manuscrit est organisé en quatre chapitres, la présente partie constitue une introduction générale incluant le contexte d'étude, la problématique de la thèse et l'organisation du manuscrit. Les quatre chapitres sont résumés comme suit :

1. Partie I : État de l'art. Cette partie inclut deux chapitres :

Le premier chapitre présente les concepts nécessaires à la compréhension générale de notre travail. Nous présentons, dans une première partie, le concept de l'Internet des Objets, objet connecté, Architecture de l'Internet des Objets, domaines d'application, et un large panel de Technologies utilisées. Nous explorons quelques challenges de recherche, dont la découverte de services fait partie. La découverte de services est considérée comme un problème d'optimisation combinatoire, de ce fait, la deuxième partie de ce chapitre introduit les concepts fondamentaux de l'optimisation ainsi que les différentes approches de résolution de cette classe de problèmes.

Le Chapitre 2, présente le concept service et les concepts clés de la découverte de services. Nous

présentons, par la suite, les différentes architectures existantes répondant à cette problématique. Une classification des différentes approches existantes relatives à la découverte de services dans l'IoT est présentée. Une étude comparative ainsi qu'une synthèse sont présentées à la fin du chapitre, dans laquelle nous identifions notamment les limitations de chaque approche étudiée.

2. Partie II : Contributions. Cette partie regroupe deux chapitres :

Dans le chapitre 3, nous présentons la première contribution, ainsi que l'évaluation, en simulation, de ses performances. Dans ce chapitre nous avons adapté l'algorithme ACA [13] pour la résolution du problème de découverte de services dans un réseau IoT, partant du phénomène naturel vers un modèle d'adaptation. La proposition est conçue de manière à rendre la découverte de services décentralisée (sans annuaire).

Dans le chapitre 4, nous avons présenté la deuxième contribution, ainsi que l'évaluation, en simulation, de ses performances. Dans ce chapitre nous avons proposé une variante discrète de l'algorithme GWO pour la résolution du problème de découverte de services dans un réseau IoT, et la solution proposée est purement décentralisée.

Nous terminons notre thèse par une conclusion générale et nous dresserons quelques perspectives qui peuvent être envisagées afin d'enrichir notre travail.

Partie I

Etat de l'art

Chapitre 1

Internet des Objets et généralités

Sommaire

1.1 Introduction

Avec les récentes avancées technologiques, les objets connectés (smartphones, tablettes numériques, montres, etc.) gagnent actuellement une place de plus en plus importante dans notre vie quotidienne. Ces objets sont devenus incontournables pour beaucoup d'entre nous en tant qu'outils d'assistance, de suivi, de surveillance, de communication, d'information, etc. Pour assurer une meilleure accessibilité et exploitation des objets connectés du monde réel et faire un lien entre le monde physique et le monde virtuel, un nouveau paradigme émerge, et appeler l'Internet des Objets ou en anglais Internet of Things (IoT).

Ce chapitre est organisé en deux parties, dans la première partie, nous décrivons le concept de l'Internet des Objets, objet connecté, architecture de l'Internet des Objets, domaines d'application, et un large panel de Technologies utilisées. Nous explorons quelques challenges de recherche, dont la découverte de services fait partie. La découverte de services est considérée comme un problème d'optimisation combinatoire, de ce fait, la deuxième partie de ce chapitre introduit les concepts fondamentaux d'un problème d'optimisation, et les méthodes de résolution de ces problèmes. Les méthodes de résolution peuvent être réparties en deux grandes classes : les méthodes exactes et les méthodes approchées. Les métaheuristiques constituent une classe de méthodes approximatives (approchées) adaptées à un grand nombre de problèmes d'optimisation, nous donnerons une classification des métaheuristiques selon leur nature de fonctionnement et leur chronologie d'apparition.

1.2 Internet des Objets

L'internet des Objets est une révolution technologique dans le domaine de l'informatique et des télécommunications [144, 43] car il permet de connecter les gens et les objets n'importe où, n'importe quand, par n'importe qui [73]. À l'origine, le terme Internet des Objets a été utilisé pour la première fois en 1999 par Kevin Ashton pour décrire des objets équipés de puces d'identification par radiofréquence (ou puce RFID). Chaque objet, identifié de manière unique et universelle et cela grâce au protocole Internet Protocol version 6 (IPV6) qui permet d'offrir une quantité d'adresses IP suffisante pour connecter des objets à Internet. Les propriétés, l'état courant, nature et localisation d'un objet sont alors des données échangées entre les objets, formant un nouveau réseau qui leur est dédié : l'Internet des Objets[9].
Le concept a toutefois évolué avec le temps et s'est généralisé vers une approche consistant à connecter un très grand nombre d'objets du quotidien au réseau Internet en utilisant d'autre technologie.

Plusieurs définitions de l'Internet des Objets ont été proposées dans la littérature, je donne ci-après, de façon non exhaustive, les définitions les plus pertinentes suivant leur ordre chronologique. Selon [18], l'IoT peut se définir aussi comme étant un réseau de réseaux qui permet, via des systèmes d'identification électroniques normalisés et unifiés, et des dispositifs mobiles sans fil, d'identifier directement et sans ambiguïté des entités numériques et des objets physiques et ainsi, de pouvoir récupérer, stocker, transférer et traiter les données sans discontinuité entre les mondes physiques et virtuels. D'après [29] qui a cherché à approfondir la définition de l'IoT, l'Internet des Objets représente une extension de l'Internet tel que nous le connaissons aujourd'hui en créant un réseau omniprésent et auto-organisé d'objets physiques connectés, identifiables et adressables permettant le développement d'applications au sein de secteurs verticaux clés et entre ces secteurs par le biais de puces, capteurs, déclencheurs et dispositifs miniatures intégrés et peu onéreux. D'après [57], l'Internet des Objets est un réseau qui relie et combine les objets avec l'Internet, en suivant les protocoles qui assurent leurs communication et échange d'informations à travers une variété de dispositifs.
Et selon la définition donnée par *Union Internationale des Télécommunications*, l'Internet des Objets est une infrastructure mondiale pour la société de l'information, qui permet de disposer des services évolués en interconnectant des objets (physiques ou virtuels) grâce aux technologies de l'information et de la communication interopérables existantes ou en évolution. D'après IEEE [97], l'IoT est un réseau qui connecte des objets qui possèdent des identifiants uniques à l'Internet. Les objets possèdent des capteurs, des actionneurs, et une capacité de calcul. À partir de l'identifiant, l'objet peut transférer des données et connaitre l'état de l'objet qui peut être changé à n'importe où, à n'importe moment, et par n'importe quel objet.

En conclusion, on peut dire que le paradigme de l'internet des Objets consiste à fusionner le monde virtuel et le monde physique, dans lequel les objets, les animaux et les personnes possèdent des identifiants uniques et peuvent transférer des données et communiquer sur un réseau sans nécessiter aucune interaction humaine, et cela grâce à un protocole standard.

1.3 Qu'est-ce qu'un objet connecté

Aujourd'hui, nous sommes entourés de ces appareils pouvant communiquer avec leur environnement et échanger des données, qui nous offrent de plus en plus de services facilitant nos activités, et avec lesquels nous interagissons fréquemment (voir la figure 1.2). Selon Dave Evans, auteur du rapport Cisco [42], il devrait y avoir environ 50 milliards d'objets connectés dans l'année 2020 (voir la figure 1.1).

Nombreuse définitions d'objet connecté ont été proposées, certaines définitions insistent sur les aspects techniques de l'objet, tandis que d'autres se concentrent plutôt sur les usages et les

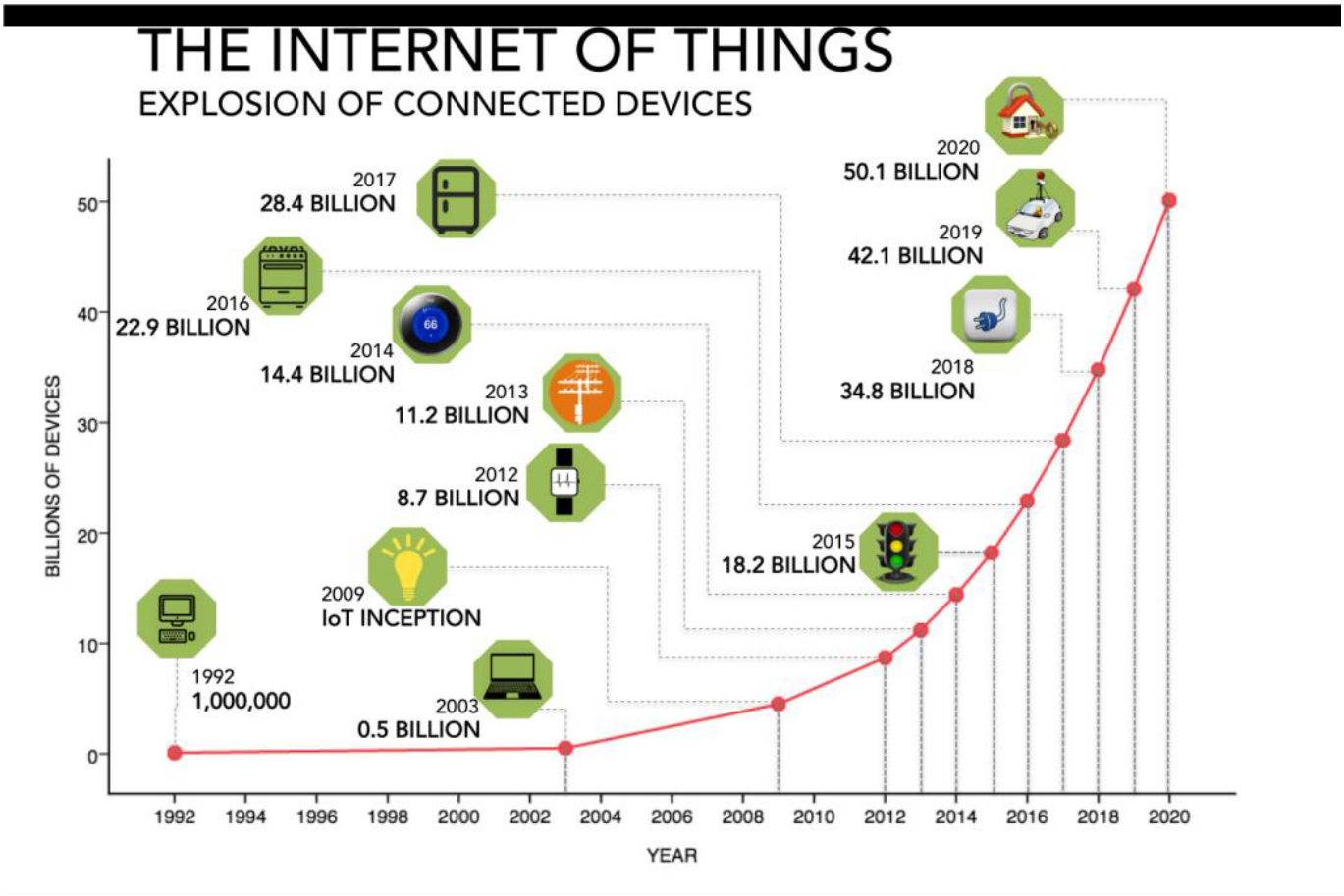


FIGURE 1.1 – Evolution technologiques de l'Internet des Objets

fonctionnalités. De nombreuses définitions s'accordent à dire qu'un objet possède des capacités de calcul, d'acquisition (capteur) et de traitement [98, 76], cependant cette définition exclut les objets inertes identifiés par RFID. En effet, une puce RFID classique ne peut pas être considérée comme un dispositif de calcul. Dans le contexte précis de l'Internet des Objets, quelle que soit la vision, cet objet possède au minimum un identifiant unique attaché à une identité (exemple d'une puce RFID), exprimant ses propriétés immuables (type, couleur, poids, etc.) et son état (position géographique, niveau de batterie, etc.) [11]. En plus, cet objet opère dans des espaces intelligents et utilise des interfaces intelligentes pour se connecter et communiquer au sein de contexte d'usages variés [132]. Il existe deux types d'objets ; des objets *actifs* qui sont capables d'accomplir des calculs et interagir avec l'environnement, comme le smartphone est un objet intelligent, il communique avec internet, il peut se localiser grâce à son GPS et se situer dans l'espace. Le deuxième type sont des objets *passifs* qui n'ont pas d'autres aptitudes que celles d'être suivis et détectés par des objets actifs, ils peuvent être un meuble, une pièce mécanique, ou même un être vivant, etc.

FIGURE 1.2 – : Quelques Objets Connectés mis en œuvre dans l'Internet des Objets (smartphone, montre, réfrigérateur, bracelet podomètre, raquette intelligente, assistant au sommeil, thermomètre).

1.4 Architecture de l'Internet des Objets

L'objectif principal de l'IoT est de connecter des milliards d'objets. Pour cela, il faut nécessairement une architecture pour représenter, organiser et structurer l'IoT de manière à lui permettre de fonctionner efficacement. En particulier, la nature hétérogène de l'IoT nécessite des matériels, et des applications capables de prendre en charge ces objets connectés à des ressources limitées, leurs services et les flux de données générés. Actuellement, il n'y a pas un consensus unique sur l'architecture pour l'IoT [134]. Différentes architectures ont été proposées [10] [125]. La conception architecturale pour l'IoT standardisée par l'IETF (Internet Engineering Task Force), est structurée en six couches [88] à savoir : application, transport,réseau, adaptation, MAC, et physique.

Les travaux de standardisation de l'IETF sur l'IoT ont également conduit à la mise en place des protocoles de communication légers adaptes aux objets connectés qui ont des ressources limitées. Ces protocoles sont entre autres, le CoAP (Constrained Application Protocol), le 6LoWPAN (IPv6 over Low-Power Wireless Personal Area Network), RPL (IPv6 Routing Protocol for Low-Power and Lossy Networks), etc.

La figure 1.3 [88] montre une comparaison entre le modèle TCP/IP et le modèle IoT de l'IETF en détaillant les différentes couches et les différents protocoles utilisés et adaptés.

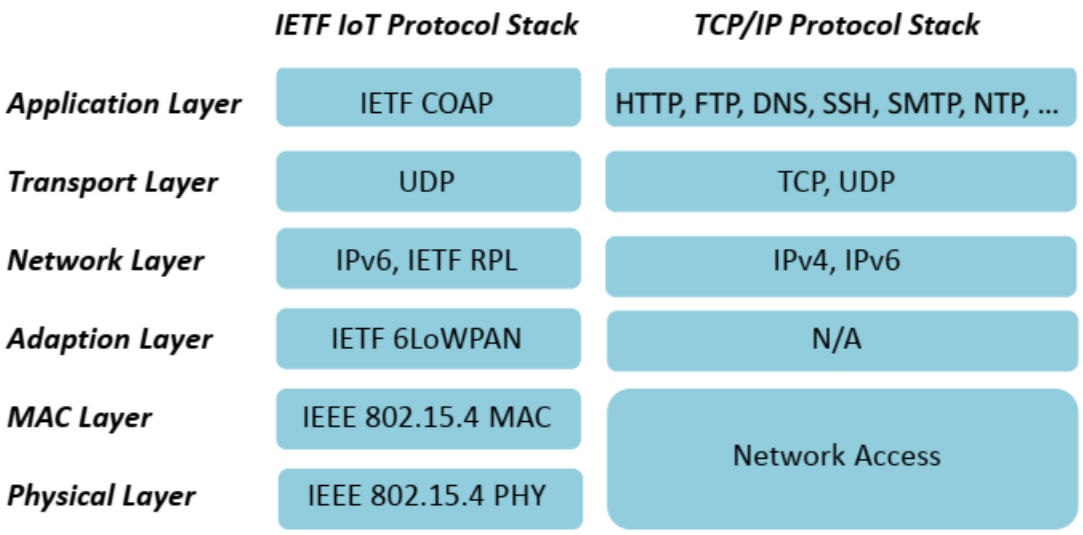


FIGURE 1.3 – Comparaison entre le modèle TCP/IP et l'architecture IoT de l'IETF

L'architecture IoT la plus couramment utilisée dans IoT, est divisée en trois couches [144, 90] : (i) couche application, (ii) couche réseau, et (iii) couche de perception, qui sont illustrées dans la figure 1.4 [90].

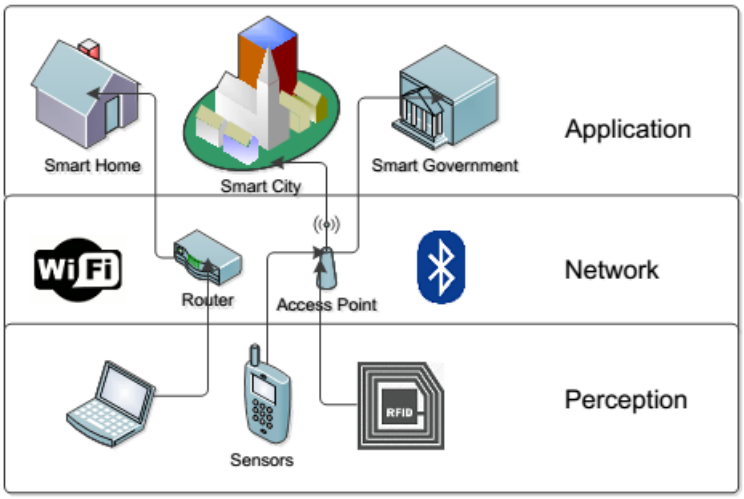


FIGURE 1.4 – Architecture de l'Internet des Objets

1.4.1 La couche de perception

Appelée aussi la couche de détection, se trouve au niveau bas dans la hiérarchie [11], son rôle consiste à connecter des objets au réseau IoT, identifier les objets, collecter et traiter les données, et transmettre les données traitées à la couche supérieure. Cette couche comprend ainsi le matériel

nécessaire pour parvenir à la collection de données contextuelles des objets connectés, à savoir les capteurs, les étiquettes RFID, caméras, GPS (Global Positioning System), etc.

1.4.2 La couche réseau

Appelée aussi la couche de transmission, c'est une couche intermédiaire dans l'architecture IoT [85]. Cette couche se charge de la transmission des données générées dans la couche perception ainsi que l'assurance de la connectivité inter-objets connectés, et entre objets intelligents et les autres hôtes de l'Internet. La couche réseau est la couche la plus importante dans l'architecture IoT, dont les divers périphériques (hub, commutation, passerelle, cloud computing, etc.), et divers technologies de communication (6LowPAN, Bluetooth, WiFi, Long-Term Evolution (LTE), etc.) sont intégrés dans cette couche. La couche réseau doit transmettre des données aux différents objets ou applications, à travers des interfaces ou passerelles entre réseaux hétérogènes, et en utilisant divers technologies et protocoles de communication.

1.4.3 La couche application

Aussi appelée la couche de gestion, c'est la couche supérieure dans l'architecture IoT [3], son rôle consiste à définir les profiles des services intelligents et les mécanismes de gestion de données de différents types provenant de différentes sources (différents types d'objets) en utilisant des protocoles comme (COAP, MQTT, DDS, etc.). La couche application reçoit les données transmises depuis la couche réseau et utilise ces données pour fournir des services ou des opérations. La couche application peut fournir un service stockage pour sauvegarder les données reçues dans une base de données, ou fournir un service d'analyse pour évaluer les données reçues afin de prédire l'état futur des dispositifs physiques. Un certain nombre d'applications existent dans cette couche, chacun ayant des exigences différentes. Comme exemple d'application : Le transport intelligent, villes intelligentes, etc.

1.5 Technologies de l'Internet des Objets

L'Internet des Objets se repose sur un large panel de technologies utilisées dans le fonctionnement de l'IoT, on met l'accent seulement sur quelques-unes qui sont : RFID, WSN, et NFC, qui sont définies ci-dessous.

1.5.1 RFID (Radio Frequency IDentification)

Le terme RFID englobe toutes les technologies qui utilisent les ondes radio pour identifier automatiquement des objets ou des personnes, elle est constituée d'un lecteur et une étiquette (étiquette aussi appelée puce ou bien tag), elle utilise le rayonnement radiofréquence pour identifier ou suivre les objets porteurs d'étiquettes. L'étiquette, composée d'une puce électronique et d'une

antenne, peut être collée ou incorporée à un objet. Pour identifier un objet, le lecteur envoie une onde radio, l'étiquette envoie à son tour une trame d'identification [108], une fois la puce alimentée. Il existe trois types d'étiquettes, les étiquettes *passives*, *actives*, et *semi-actives*. Les premières ne possédant pas de batteries : une petite quantité d'énergie leur est fournie par le champ magnétique induit par le lecteur au moment de l'identification. Les puces *actives* équipées d'une batterie interne et dotées de ressources matérielles plus importantes, ils sont capables d'envoyer eux-mêmes des informations d'identification sans sollicitation d'un lecteur. Les *semi-actifs* utilisent un mécanisme hybride : auto-alimentés, ils ne s'activent que si celui-ci est interrogé, permettant une plus faible consommation d'énergie que les tags actifs.

1.5.2 Wireless sensor Natwork WSN

Les réseaux de capteurs sans fil, ou wireless sensor network (WSN) en anglais sont des systèmes distribués composés d'un grand nombre d'appareils qui sont organisés en un réseau coopératif et capables d'acquérir des informations sur leur environnement au moyen de capteurs embarqués : température, luminosité, vibrations, présence, la pression, etc. Typiquement, ces dispositifs sont équipés d'une batterie et communiquent au moyen de liaisons sans fil bas débit et à courte portée, ce qui les rend particulièrement autonomes. En effet, dotés de capacités matérielles réduites, de fait, peu coûteux, ils sont destinés à être déployés en masse dans l'environnement et à produire des mesures en continu, ce avec un minimum d'intervention humaine. Les réseaux de capteurs sans fil ont été utilisés pour de nombreuses applications [102], par exemple dans le domaine médical [5], les maisons et villes intelligentes [142, 17], des applications militaires ou encore l'étude des populations animales [39].

1.5.3 Near Field Communication (NFC)

NFC est une technologie simple et intuitive qui permet d'utiliser un téléphone portable à des fins innovantes, elle permet donc d'établir une communication entre deux appareils compatibles à courte distance (10 cm maximum) [141]. NFC est basée sur le même principe que la RFID, c'est-à-dire l'identification par radio fréquence. Elle permet l'échange d'informations à courte distance entre deux objets (un lecteur et une carte) sans contact, et fonctionne suivant deux modes : le mode *passif* et le mode *actif*. En mode *passif*, le terminal de l'utilisateur émule une carte à puce et acquiert de l'énergie des radiations du lecteur (téléphone mobile par exemple). En mode *actif*, le terminal se comporte comme un lecteur d'étiquettes électroniques et possède sa propre source d'énergie (une batterie embarquée par exemple). NFC permet à l'utilisateur d'échanger des informations avec son environnement, notamment dans le domaine des transports, des loisirs, des achats, ou la lecture d'informations sur des panneaux d'affichage. L'utilisation de la NFC facilite la gestion des données de ventes dans la chaîne logistique, la gestion et la validation de tickets de bus dans le transport urbain [59, 106].

1.6 Domaines d'application

L'émergence de l'IoT offre un grand potentiel pour le développement de nouvelles applications dans plusieurs domaines (voir la figure 1.5) connectant le monde physique au monde virtuel.

1.6.1 Santé

L'IoT aura de nombreuses applications dans le secteur de la santé. Des capteurs corporels implantés dans le corps du patient récoltent des informations relatives aux paramètres médicaux, telles que la température, la glycémie, le rythme cardiaque et la tension artérielle [98]. Ces informations seront stockées et traitées sur Internet (dans le cloud) [111], et mises à la disposition du médecin qui pourra les consulter n'importe quand et depuis n'importe quel dispositif connecté à Internet (ex son smartphone ou sa tablette). Un médecin peut interroger ces capteurs pour avoir les valeurs actuelles afin d'optimiser la consommation de médicaments, prévenir des situations graves et certaines maladies, et de suivre à distance des patients atteints des maladies chroniques et agir rapidement si cela s'est avéré nécessaire. Le médecin est alerté en temps réel (en lui envoyant un mail ou un SMS) de tout changement brusque concernant l'état de son patient. Suivant le degré de gravité de la situation, le médecin réagit soit en se déplaçant chez le patient ou juste en le contactant et lui indiquant ce qu'il faut faire pour revenir à l'état normal.

1.6.2 Usine intelligente

L'IoT révolutionne le fonctionnement des usines, des capteurs, des étiquettes RFID et des contrôleurs embarqués, s'accroissent dans les systèmes de production industrielle. Ce qui permit d'optimiser le processus logistique des entreprises, suivre la chaîne de distribution, depuis les usines de fabrication des produits jusqu'à l'utilisateur final (acheteur de produit), et cela garantit une traçabilité. A cela s'ajoutent des contrôles de sécurité tout au long de la fabrication, qui permettent de rappeler un produit en cas de défaillance, de manière ciblée et plus rapidement. Ainsi, les usines connectées offrent une production plus flexible qui permet de s'adapter à la demande en temps réel, à l'aide des informations collectées auprès des différents points de ventes. Pour offrir plus de production et gagner de temps, des machines capables de contacter un spécialiste apte à les dépanner à distance, ou pour se mettre à jour et améliorer leurs performances. La production est optimisée en fonction du coût de l'énergie et de sa disponibilité au cours d'une journée, lorsqu'elle est moins chère ou lorsque les énergies alternatives sont utilisables. Une mise hors tension des machines est également effectuée si elles n'ont pas besoin de fonctionner. Les remontées d'informations peuvent aider à optimiser les consommations et participent ainsi à l'efficacité énergétique de l'usine. Ces usines permettront, en plus d'améliorer la sécurité et la santé au travail des collaborateurs, de valoriser l'humain en lui assignant des tâches à valeur ajoutée.

1.6.3 Maison intelligente (smart home)

L'Internet des Objets révolutionne les maisons sur les différents plans énergétique, confort et sécurité [30]. Des objets connectés de plus en plus innovants qui permettent d'accéder ou déclencher à distance par ses propriétaires via des smartphones, tablettes ou ordinateurs connectés (la porte, la télévision, le thermostat, le réfrigérateur, le chauffage, les montres, etc.), de telle sorte qu'une porte connectée informe les parents de la rentrée de leurs enfants ou les alertent sur une porte laissée ouverte par une notification envoyée sur un smartphone. Un thermostat intelligent connecté à internet de la maison permet de contrôler facilement la température sans aucune intervention, la température se règle de façon optimale en fonction des prévisions météorologiques ou en fonction de la géolocalisation, pour une amélioration du confort et une optimisation des économies énergétiques. Le réfrigérateur intelligent connecté à Internet et muni d'un système RFID traque les produits élémentaires qui y sont stockés et enregistre des informations pertinentes leur concernant (comme la durée du stockage et la date d'expiration). L'utilisateur peut l'interroger à distance pour savoir ce qui reste et ramener les produits manquant avant de rentrer à la maison. Ou alternativement, le réfrigérateur peut être programmé pour commander automatiquement les produits qui manquent.

1.6.4 Villes intelligentes (Smart Cities)

Les villes intelligentes sont le fruit de l'Internet des Objets [63], (les maisons, les routes, les bâtiments, les véhicules, les magasins, les parkings, les hôtels, les restaurants, etc.) ; seront tous connectés à Internet pour être gérer plus efficacement, par exemple pour faciliter et aider la vie des automobilistes gagner le temps en leur fournissant des informations pertinentes pour trouver un itinéraire, en temps réel, sur l'endroit où il trouve une place dans le plus proche parking, hôtel, restaurant, hôpital etc., et des informations d'ordre général sur la ville, comme la température, le taux d'humidité, etc., de même, les autorités de la ville intelligente trouveront une facilité de réalisation des tâches de contrôle de la pollution, l'éclairage urbain, etc., notons qu'une coexistence massive de multiples technologies est nécessaire pour la réalisation du projet de la ville intelligente, principalement les réseaux de capteurs.

1.6.5 Environnement

En effet, Internet des Objets permet d'obtenir une multitude d'informations sur l'environnement (température, luminosité, humidité, consommation énergétique, bruit, pollution etc.). Les objets connectés permettent de réduire les émissions de gaz comme le CO2, de réduire la consommation d'énergie (comme la consommation d'électricité, d'eau, de carburant,etc.) et ainsi une meilleure gestion d'énergie [19], en fonction de la demande des utilisateurs grâce aux nouveaux réseaux de capteurs connectés qui renvoient des données en temps réel, comme la qualité de l'air, le trafic routier, etc., afin d'offrir un meilleur environnement [22]. Les transports en commun

sont un élément crucial des villes et ils polluent énormément, de nombreuses alternatives aux carburants fossiles ont été découvertes notamment le biocarburant ou métros électriques qui diminuent fortement les émissions de gaz. Les assistants d'aide à la conduite pour les transports intelligents permettront sans doute de diminuer encore davantage la consommation d'énergie nécessaire pour le fonctionnement de ces véhicules. L'Internet des Objets pourrait donc permettre à l'Union européenne d'atteindre ses objectifs 20-20-20 (réduire de 20% d'émission de CO2, augmenter l'efficacité énergétique de 20% et atteindre 20% d'énergie renouvelable sur le territoire européen en 2020) [16].

L'Internet des Objets peut également prévenir de tous les dangers liés à la nature comme les tremblements de terre, les inondations, les avalanches en montagne, etc. [103]. Toutes les zones à risque pourraient être équipées de capteurs qui préviendraient les populations des dangers imminents liés à l'environnement qui les entourent.

1.6.6 Les applications militaires

Internet des Objets est également porteur de promesses pour les militaires, elle améliore la gestion globale des soldats et de l'infrastructure, et aussi en renforçant considérablement les possibilités d'automatiser la guerre grâce à la robotique et l'armement de drones. Les Armées opérant un très grand nombre de capteurs, réseaux, ordinateurs embarqués et systèmes d'information permettent d'envisager des applications sophistiquées pour l'exploration, la surveillance des champs de batailles et des frontières, ainsi que la poursuite et la détection d'intrusions, localisation de soldats, véhicules, armes, etc., sur un champ de bataille, sous l'eau, dans l'espace, dans le sol [122].

1.6.7 Réseau intelligent de distribution d'électricité (Smart grid)

L'un des domaines d'application de l'IoT est le secteur de la distribution d'énergie intelligente, des réseaux électriques seront développés grâce aux technologies de l'IoT, des capteurs installés sur l'ensemble du réseau indiquent instantanément les flux électriques et les niveaux de consommation. Les opérateurs du réseau peuvent alors réorienter les flux énergétiques en fonction de la demande[22], est d'assurer une livraison d'électricité économiquement viable et efficace à long terme, et envoyer des signaux de prix aux particuliers pour adapter leur consommation (volontairement ou automatiquement).

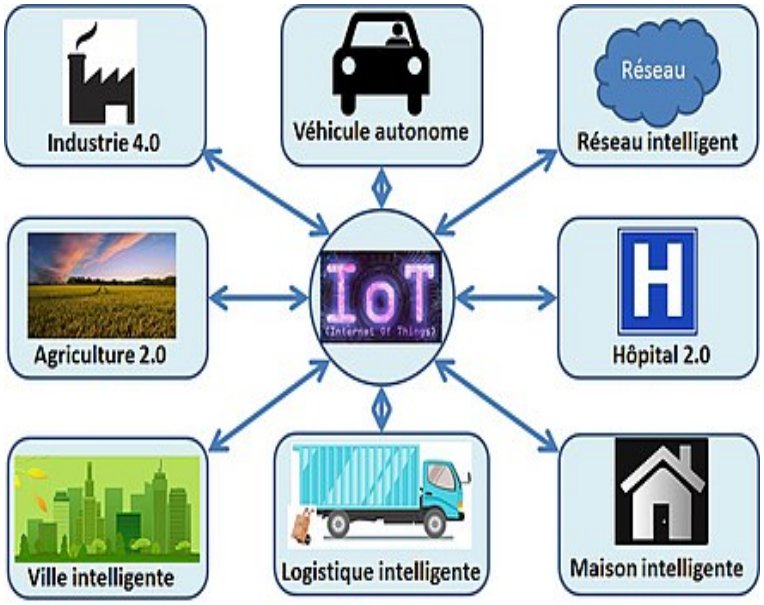


FIGURE 1.5 – Domaines d'application de l'Internet des Objets

1.7 Challenges dans Internet des Objets

Pour permettre à l'Internet des Objets d'atteindre son plein potentiel et concrétiser les scénarios qu'on décrit dans les différentes applications, plusieurs aspects doivent être étudiés. En effet, les concepts prometteurs imaginés par la communauté scientifique nécessitent de s'attaquer à un large éventail de challenges et résoudre un certain nombre de problématiques : passage à l'échelle, hétérogénéité, sécurité et vie privée, Auto-configuration, découverte du contexte et de services, et limitations de ressources.

1.7.1 Passage à l'échelle

Le passage à l'échelle ou la scalabilité dans l'Internet des Objets est la capacité de maintenir les fonctionnalités et les performances malgré l'augmentation du nombre d'objets et de services. Avec le très grand nombre d'objets connectés qui est estimé à 125 Milliards d'ici l'an 2030, selon les prévisions d'IHS Markit. Afin éviter la dégradation des performances et assurer une qualité acceptable des services fournis, l'Internet des Objets doit être doté de mécanismes robustes lui permettant de supporter le passage à l'échelle afin d'assurer ses fonctionnalités principales telles que la communication et la découverte de services.

1.7.2 Hétérogénéité des objets IoT

Les objets possèdent des capacités et des propriétés spécifiques et varient d'un objet à un autre (statique ou mobile, alimenté en continu ou par une batterie, ressources matérielles, capteurs ou actionneurs, etc.), de plus, ils utilisent des infrastructures de communication différentes (par ex.

WiFi, Bluetooth, ZigBee, UMTS, 3G, 4G, etc.) et une variété de plateformes logicielles (systèmes d'exploitation et les langages de programmation). Afin d'assurer une communication entre ces objets, l'Internet des Objets doit être doté de mécanismes qui permet aux objets d'interagir en toutes circonstances, quelles que soient les contraintes posées par leurs caractéristiques et leur environnement et pouvoir gérer l'hétérogénéité technologique et les normes d'objets couplées à des multitudes de besoins d'applications et des usages en terme de services.

1.7.3 Sécurité et vie privée

La sécurité est un problème critique sur Internet, et c'est le principal défi pour l'Internet des Objets. L'IoT devrait connecter un très grand nombre d'objets à Internet qui deviennent des objets intelligents qui rassemblent des données, interagissent entre eux et génèrent des quantités énormes de données. En effet, il y a une forte chance que des logiciels malveillants pénètrent dans les systèmes ou les données. Pour la préservation de l'intégrité des données, l'Internet des Objets doit être muni des politiques de protection de données et de contrôle d'accès aux services et aux objets.

1.7.4 Auto-configuration

L'IoT prévoit des milliards d'objets qui seront connectés à Internet. En conséquence, un défi majeur se pose lors de la connexion et de la configuration des objets pour une application. Pour avoir une connexion spontanée et de manière automatique, il doit y avoir un protocole automatisé ou au moins semi-automatisé pour connecter les objets d'une application afin d'accomplir un service, les objets peuvent obtenir une adresse en utilisant le protocole IPv6 de manière automatique via une auto-configuration. La configuration dépends des caractéristiques d'un objet (par exemple, les capacités des objets, les structures de données qu'ils produisent, matériel / pilote) et l'environnement.

1.7.5 La découverte du contexte

L'IoT est caractérisé par l'extrême hétérogénéité et un grand nombre d'objets qui peuvent être interconnectés, ainsi que par la nature spontanée des interactions. Pour traiter l'énorme quantité d'informations recueillies à partir de l'IoT, de nouvelles solutions sont ainsi nécessaires pour extraire les informations utiles et identifier les situations de contexte pertinentes pour rendre un service adapté pour une situation précise et pour une personne particulière. Le contexte couvre toutes les informations pouvant être utilisées pour caractériser la situation d'une entité (une personne, un lieu ou un objet) que l'on considère pertinent pour l'interaction entre un utilisateur et une application, incluant l'utilisateur et les applications elles-mêmes [1].

1.7.6 La découverte de services

L'Internet des Objets est caractérisé par le nombre volumineux des objets connectés, leurs mobilités et leurs hétérogénéités. Ces caractéristiques rendent la recherche de leurs services adjacents difficile. Pour enrichir les descriptions de services et rendre leur recherche précise et flexible, cela nécessite des mécanismes qui s'appuient sur l'utilisation des technologies du Web Sémantique et des méthodes d'optimisation afin d'améliorer la découverte de services dans l'IoT.

1.7.7 Les limitations de ressources

Les objets (les capteurs et les tags RFID) sont très limités en ressources de calcul, de stockage mémoire et d'énergie. A cet effet, les solutions (protocoles de communications ou de sécurité, technologies de transmission, etc.) destinées à l'Internet des Objets doivent prendre en considération ces contraintes et limitations.

La grande puissance de l'IoT repose sur le fait que ses objets communiquent, analysent, traitent et gèrent des données d'une manière autonome et sans aucune intervention humaine. Cependant, les challenges cités auparavant dans la section 1.7 freinent considérablement l'évolution et le déploiement rapide de cette haute technologie. Les objets connectés sont généralement très limités en capacité de calcul, de stockage, et d'énergie. Dès lors, on ne peut pas employer des mécanismes classiques. De ce fait, il faut utiliser des méthodes légères et robustes pour la résolution de ces challenges, tout en étant adapté aux capacités des objets et des technologies de communication.

1.8 Méthodes de résolution de problèmes d'optimisation

Un problème d'optimisation consiste à minimiser, ou maximiser, une fonction objectif f, dans Le but de trouver la (ou les) solution(s) optimale(s) x^* parmi un ensemble de solutions noté S, appelé espace de recherche ou espace de solutions.

Un problème de minimisation est définit comme suit :

$$f(x^*) \leqslant f(x), \forall x \in S, soit \quad \min_{x \in S} f(x) \tag{1.1}$$

Dans ce cas la fonction objectif f est connue comme une fonction de coût.
De même, pour un problème de maximisation :

$$f(x^*) \geqslant f(x), \forall x \in S, soit \quad \max_{x \in S} f(x) \tag{1.2}$$

Dans ce cas la fonction objectif f est connue comme une fonction d'utilité, de profit, ou de fitness.

Résoudre un problème d'optimisation, consiste alors à trouver la solution optimale(optimum global), qui est la meilleure solution possible x^* de la fonction objectif f. Cependant, il peut exister des solutions intermédiaires, qui sont également des optimums, mais uniquement pour un sous-espace restreint de l'espace de recherche, ces solution appelées optimums locaux. Afin d'illustrer cela, la figure 1.6 montre des optima locaux et un optimum global d'une fonction objectif dans le cas d'un problème d'optimisation à minimiser.

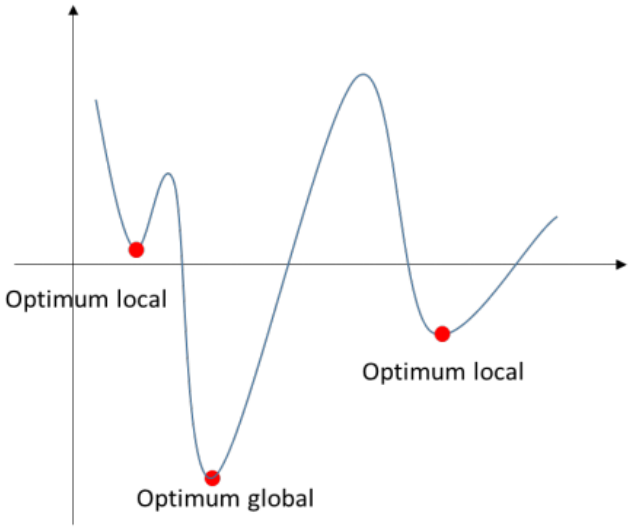


FIGURE 1.6 – Optima locaux et optimal global dans le cas d'un problème d'optimisation à minimiser

Les problèmes d'optimisation se divisent en deux classes : les problèmes avec des variables continues, et les problèmes avec des variables discrètes ou les problèmes *combinatoires*[113]. La solution pour un problème continu est souvent représentée par un ensemble de nombres réels, tandis que la solution pour un problème combinatoire est souvent représentée par ensemble fini de nombres entiers.

1.8.1 Problème d'optimisation combinatoire

Un problème d'optimisation combinatoire (Constraint Optimization Problems COP) est une discipline utilisant conjointement différentes techniques des mathématiques discrètes, de la recherche opérationnelle et de l'informatique, afin de résoudre des problèmes d'optimisation dont la structure sousjacente est discrète (généralement un graphe). L'optimisation combinatoire consiste à chercher une solution optimale dans un ensemble discret et fini, en théorie c'est facile car il suffit d'essayer toutes les solutions (combinaisons), et de comparer leurs qualités pour voir la meilleure. Cependant, en pratique, l'énumération de toutes les solutions peut prendre trop de

temps ; or, le temps de recherche de la solution optimale est un facteur très important et c'est à cause de lui que les problèmes d'optimisation combinatoire sont réputés si difficiles.

Un grand nombre de méthodes de résolution existe pour résoudre les problèmes d'optimisation combinatoire ; elles peuvent être réparties en deux grandes classes : les méthodes exactes et les méthodes approchées (voir la figure 1.7) ; les méthodes exactes garantissent une solution optimale pour un problème donné mais au prix de temps de calcul et/ou d'espace mémoire parfois prohibitifs, et les méthodes approchées offrent la possibilité de trouver souvent des solutions approchées de bonne qualité en un temps raisonnable.

1.8.2 Méthodes Exactes

Ce sont des méthodes efficaces qui garantissent l'obtention de la solution optimale (optimum global) du problème traité de petite taille. Elles consistent à effectuer une énumération explicite de toutes les solutions possibles pour assurer l'obtention de toutes les meilleures solutions optimales potentielles que la solution optimale trouvée au cours de la recherche. Parmi Les méthodes exactes, on peut citer : la programmation linéaire en nombres entiers, les procédures de recherche arborescente et la programmation dynamique, elles offrent en théorie la garantie de trouver une solution optimale mais sont coûteux en temps de calcul et d'espace mémoire.

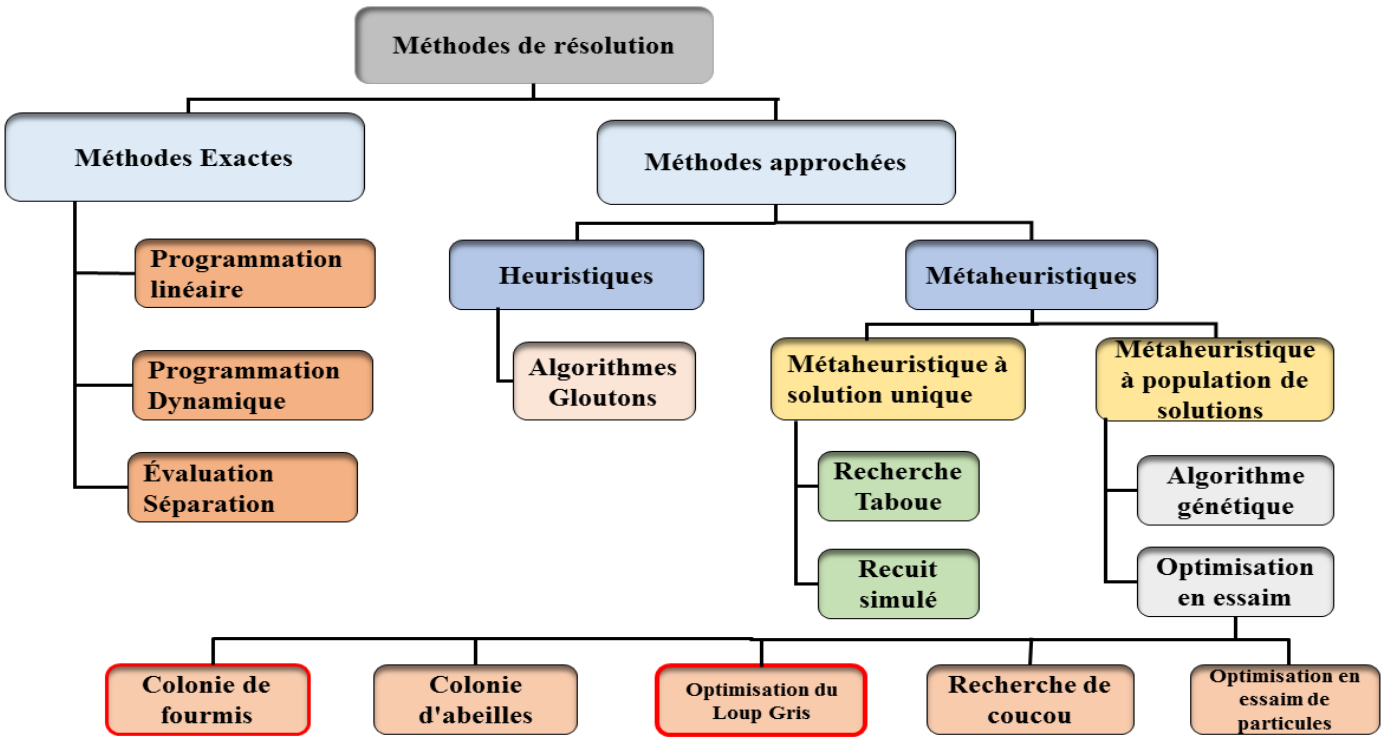


FIGURE 1.7 – Taxonomie des méthodes de résolution d'un problème d'optimisation

1.8.3 Méthodes approchées

Le problème de la recherche d'une solution optimale est souvent très complexe et très difficile pour déterminer rapidement les bonnes solutions. Par conséquent les méthodes approchées sont souvent utilisées pour accélérer le processus et trouver une bonne solution rapidement dans les cas où une recherche exhaustive est peu pratique. Les méthodes approchées offrent une solution de bonne qualité en un temps raisonnable. Ces méthodes peuvent être réparties en deux classes : Heuristiques et Métaheuristiques.

1.8.3.1 Heuristiques

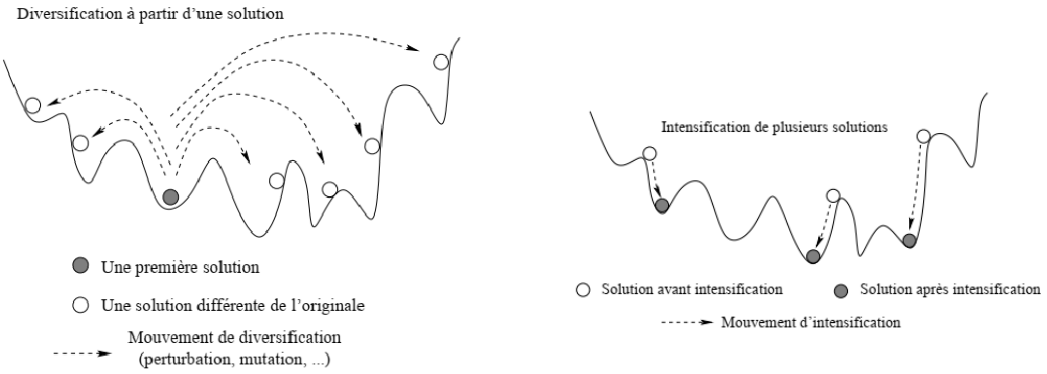
Une heuristique est des méthode de résolution spécialisée à un problème d'optimisation particulier dit difficile. Cette méthode a pour but de trouver une solution approchée et réalisable en un temps raisonnable, mais pas nécessairement optimale [81]. L'utilisation d'une heuristique est efficace pour trouver une solution approchée pour un problème difficile et accélère le processus de résolution. Une heuristique est spécifique au problème et ne peut pas être généralisée.

1.8.3.2 Métaheuristique

Les métaheuristiques sont des méthodes généralement inspirées de la nature, elles utilisent l'intelligence de plusieurs concepts pour explorer et exploiter tout l'espace de recherche de manière

plus efficace. Elles s'appliquent à plusieurs problèmes de nature différentes et sont dédiées particulièrement à la résolution des problèmes d'optimisation. Cependant, une métaheuristique ne peut pas résoudre tous les problèmes d'optimisation. Ces facteurs et d'autres ont conduit à une explosion du nombre de métaheuristiques inspirées de la nature au cours des dernières années.
Les performances d'une métaheuristique dans la résolution d'un problème d'optimisation est liée à sa capacité d'établir un certain équilibre entre la diversification (ou exploration) et l'intensification (ou exploitation). Ne pas préserver cet équilibre conduit à une convergence rapide vers des minima locaux (pas diversification) ou à une exploration trop longue (pas d'intensification).

1. **Intensification (exploitation)** : Dans l'intensification, les zones prometteuses sont explorées plus en profondeur afin de trouver de meilleures solutions (voir la figure 1.8b)[21].
2. **diversification (exploration)** : Dans la diversification, les zones non explorées doivent être visitées pour s'assurer que toutes les zones de l'espace de recherche sont explorées, et cela dans le but de trouver de nouvelles meilleures solutions (voir la figure 1.8a)[21].



(a) Diversification (b) Intensification

FIGURE 1.8 – Diversification et Intensification

Les métaheuristiques peuvent être réparties en deux classes :

1. **Métaheuristiques à solution unique :** Ces méthodes consistent à traiter et améliorer une seule solution. Afin de trouver la solution optimale, elles se basent généralement sur une recherche locale (recherche par voisinage), ce principe est illustré dans la figure 1.9 *a*. Parmi les métaheuristiques les plus populaires, on retrouve la méthode de Descente (DM : Descent Method) aussi appelée Hill climbing [113], le Recuit Simulé (SA : Simulated Annealing) [77], la recherche Tabou (Tabu Search TS) [52], et la Procédure de Recherche Gloutonne Aléatoire Adaptative (GRASP : Greedy Randomized Adaptive Search Procedure) [44].

Le principe générale de métaheuristique à solution unique est décrit dans l'algorithme 1[131].

Algorithm 1 Algorithme d'une métaheuristique à solution unique

Entrée(s) Une solution initiale S_0
Sortie(s) la meilleure solution
t=0
répéter
$Generate(C(S_t))$ Générer des solutions candidates à partir de S_t
$S_{t+1} = Select(C(S_t)$ Sélectionner une solution de $C(S_t)$ pour remplacer la solution courante S_t
$t = t + 1$
jusqu'à la satisfaction du critère d'arrêt

2. **Métaheuristiques à population de solutions :** Contrairement aux métaheuristiques à solution unique, elles font évoluer simultanément un ensemble de solutions dans l'espace de recherche, ce principe est illustré dans la figure 1.9 *b*. On distingue deux grandes calasses, les algorithmes évolutionnaires inspirées de la biologie de l'évolution des espèces (les méthodes d'héritage et de la sélection naturelle), et les algorithmes d'intelligence en essaim inspirés du comportement collectif des insectes sociaux ou d'autres animaux. L'algorithme 2 décrit ce principe général [131].

Algorithm 2 Algorithme d'une métaheuristique à population de solutions

Entrée(s) Une population de solutions initiale $P = P_0$
Sortie(s) la meilleure solution
$t = 0$
répéter
Générer une nouvelle population P'_t
p_{t+1} =Select-Population($P_t \cup P'_t$) Sélectionner une nouvelle population
$t = t + 1$
jusqu'à la satisfaction du critère d'arrêt

Dans le cadre de cette thèse, notre intérêt porte sur les métaheuristiques à base de population de solutions pour la résolution du problème de découverte de services dans IoT.

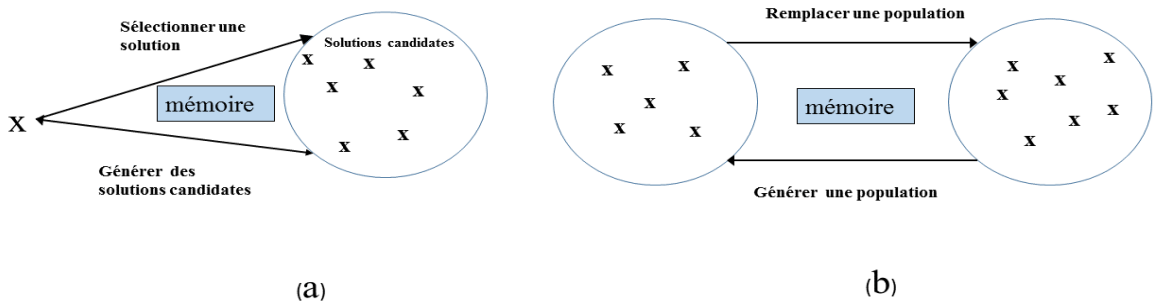


FIGURE 1.9 – Principes généraux d'une métaheuristique (a) à solution unique, (b) à population de solutions

Les deux classes les plus dominantes dans cette catégorie d'algorithmes impliquent les algorithme évolutifs et les algorithme à base d'essaims.

1.8.3.3 Algorithmes évolutionnaires

Les algorithmes évolutionnaires (AE) sont des algorithmes d'optimisation bio-inspirés [23] pour résoudre des problèmes divers. Les AE sont inspirées par la théorie darwinienne de l'évolution [32], ils se basent sur la théorie darwinienne de l'évolution des espèces(animale et végétale), et la capacité des individus à s'adapter aux changements environnementaux. Les algorithmes évolutionnaires utilisent une population d'individus gouvernés par des opérateurs (la mutation, croisement, selection). Le choix et l'ordre dans lequel ces opérateurs s'effectuent a donné naissance à plusieurs algorithmes évolutionnaires. On distingue, programmation génétique [82], programmation évolutive [47], les stratégies évolutionnaires [66] et les algorithmes génétiques (GA : Genetic Algorithm) [64].

Algorithme génétique

Les algorithmes génétiques sont peut-être les algorithmes évolutionnaires les plus populaires et leurs utilisation dans de nombreux domaines a fait ses preuves, notamment dans des problèmes combinatoires tels que les problèmes d'ordonnancement [105], l'optimisation de réseaux sans fil [69], et traitement des images [60].
Les algorithmes génétiques utilisent trois opérateurs pour définir une métaheuristique pour la résolution d'un problème d'optimisation combinatoire. L'idée est de faire évoluer une population

de combinaisons (une population de solutions), par trois opérateurs sélection, croisement (par reproduction) et mutation (par modification). La capacité d'adaptation d'une combinaison est évaluée par une fonction objectif à optimiser. L'algorithme 3 décrit ce principe général dont les principales étapes sont détaillées ci-après.

Algorithm 3 Algorithme génétique

Entrée(s) Une population avec un ensemble de combinaisons de E
Sortie(s) la meilleure combinaison
tant que critères d'arrêt non atteints **faire**
Sélectionner des combinaisons de la population
Créer de nouvelles combinaisons par croisement et mutation
Mettre à jour la population
fin tant que

Initialisation de la population : la population initiale est générée de façon aléatoire, selon une distribution uniforme afin d'assurer une bonne diversité des combinaisons.
Sélection : cette opération, consiste à choisir les combinaisons de la population selon leurs performances, qui seront ensuite croisées et mutées.
Croisement : cette opération consiste à générer de nouvelles combinaisons, à partir des combinaisons sélectionnées.
Mutation : cette opération consiste à modifier de façon aléatoire certains composants des combinaisons obtenues par croisement.

1.8.3.4 Algorithmes basés essaim

Les algorithmes d'intelligence par essaim (SI, Swarm Intelligence) basés sur l'intelligence des animaux où leur capacité individuelle est très limitée, peuvent conjointement effectuer de nombreuses tâches complexes. L'intelligence collective observée notamment chez les insectes sociaux (les fourmis et les abeilles), les animaux évoluant en groupe comme les oiseaux migrateurs, les loups, les lions,...,etc. D'autres intelligences collectives sont observées dans certains phénomènes biologiques comme les systèmes immunitaires. Ce sont des systèmes composés d'agents simples qui peuvent :

- Partager des informations entre eux ;
- S'auto-organiser et évoluer ensemble ;
- Être très efficace pour le co-apprentissage ;
- Faire du parallélisme pour des problèmes complexes et en temps réel.

plusieurs algorithmes ont été proposés ces dernières décennies, je donne ci-après, de façon non exhaustive, les différentes métaheuristiques selon l'ordre chronologique : Optimisation d'essaims de particules (PSO : Particle Swarm Optimization) [40], Optimisation par colonie de fourmis(ACO : Ant Colony Optimization) [37], optimisation par système immunitaire artificiel (AISA : Artificial Immune System Algorithm) [33], optimisation par essaim de poissons(FSA : Fish Swarm Algorithm) [87], algorithme d'optimisation bactérienne de la recherche de nourriture (BFOA : Bacterial Foraging Optimization Algorithm) [114], qui s'inspire du comportement de recherche de nourriture et de reproduction des bactéries, optimisation par sauts de grenouille partitionnés (SFLA : Shuffled Frog Leaping Algorithm) [41], optimisation par colonie d'abeilles(ABC : Artificial Bee Colony optimization) [71], l'algorithme d'optimisation basée sur la biogéographie insulaire (BBO : Biogeography-Based Optimization [128], Recherche de coucou (CS : Cuckoo Search) [148], algorithme de luciole (FA : Firefly Algorithm [147], optimisation des loup gris (GWO : Grey Wolf Optimizer) [100], l'algorithme d'optimisation des fourmilions (ALO : Ant Lion Optimizer) [99], etc.

1.9 Conclusion

L'Internet des Objets est une évolution de l'Internet, son objectif est de rendre le monde réel plus intelligent grâce à la connexion des objets à Internet, afin d'améliorer la qualité de vie des personnes sans aucune demande explicite de leur part en offrant d'une manière transparente des services adaptés aux besoins des usagers. Cette révolution facilite la création d'objets intelligents utilisant des technologies comme RFID, réseaux de capteurs, et NFC, pour les faire connecter à l'Internet. L'IoT offre un grand potentiel pour le développement de nouvelles applications dans de multiple domaines (maison intelligente, usine intelligente, ville intelligente et l'environnement intelligent). L'Internet des Objets pose divers problèmes, notamment du fait de sa très grande échelle, de l'hétérogénéité des objets et des systèmes, sécurité, découverte de services, et limitation de ressources, ces défis devraient être traités pour que l'Internet des Objets atteigne son plein potentiel. De nombreux problèmes peuvent être considérés comme des problèmes d'optimisation, les méthodes de résolution de ces problèmes peuvent être réparties en deux grandes classes : les méthodes exactes et les méthodes approchées. Les métaheuristiques constituent une classe de méthodes approximatives adaptées à un grand nombre de problèmes d'optimisation, leur utilisation dans les différents domaines montre leur efficacité à résoudre des problèmes complexe et offre la possibilité de trouver souvent des solutions approchées de bonne qualité en un temps raisonnable.

Chapitre 2

La découverte de services dans l'Internet des Objets

Sommaire

2.1 Introduction

Les services constituent des composants logiciels interopérables pouvant être réutilisés dans le développement d'applications distribuées. Aider le concepteur ou le développeur dans la recherche de composants nécessaires contribuera à baisser le coût de développement de nouvelles applications. Avec l'augmentation accélérée du nombre de services dans les espaces intelligents, trouver un service particulier à partir des milliards de services disponibles est la tâche fondamentale de toute approche de découverte de services. Dans la littérature, plusieurs approches de résolution du problème de découverte de services ont été proposées.

Ce chapitre vise à analyser les approches proposées dans la littérature en vue d'identifier des mécanismes qui constituent des points forts de ces approches. Une étude comparative ainsi qu'une synthèse sont présentées à la fin du chapitre, dans laquelle nous identifions notamment les limitations de certaines approches vis-à-vis du challenge de la découverte de services.

2.2 Service dans Internet des Objets

De nombreux exemples de services peuvent être trouvés dans le monde réel, on peut citer, par exemple, le téléphone, le courrier, la presse, la télévision, la radio, etc. Dans le monde informatique, un service est un ensemble de fonctionnalités qui permettent l'accès, la recherche et le traitement de l'information, et aussi la possibilité de communiquer avec d'autres utilisateurs ou d'autres applications. Plusieurs définitions du concept service ont été proposées dans la littérature, je donne ci-après, les définitions les plus pertinentes.

Définition 2.2.1. Dans la définition proposée par Papazoglou un service est vu comme une entité logicielle décrite par une interface formelle. Les services permettent d'offrir des fonctionnalités indépendamment des autres services, la technologie sous-jacente de son implémentation et de la plateforme de son exécution. D'autre part, le service ne connaît pas à priori le contexte dans lequel il sera utilisé par le client.

Définition 2.2.2. Une autre définition proposée par Arsanjani met l'accent sur les principales interactions, entre un fournisseur et un client, pour permettre l'utilisation des fonctionnalités offertes par un service. En effet, un service est vu comme une ressource logicielle possédant une description de service. Le client peut alors effectuer une découverte des services correspondant à ses besoins sur la base des descriptions des services existants [8].

Le consortium OASIS (Organization for the Advancement of Structured Information Standards) définit un service comme étant un mécanisme permettant d'accéder à une ou plusieurs

fonctionnalités. La description de ces fonctionnalités sont décrites dans une spécification, appelée interface de service, Il n'y a pas de détails sur la manière dont les fonctionnalités du service sont réalisées, un service est accessible au travers d'une interface conforme à un ensemble de contraintes et de politiques d'accès [89].

A travers ces définitions, nous définissons le concept de service comme suit : Un service est une unité autonome de fonctionnalités logicielles, conçue pour réaliser une tâche précise. Le service est décrit à travers une spécification. La spécification de service contient la description des fonctionnalités offertes par le service, et aussi des informations sur son comportement ou sur ses aspects non-fonctionnels. L'ensemble des fonctionnalités définies dans la spécification du service sont implémentées par le fournisseur. Une spécification de service est utilisée par les consommateurs pour rechercher un fournisseur qui correspond à ses besoins.

2.2.1 Les propriétés d'un service

Un service représente une brique logicielle dont les fonctionnalités, les propriétés non fonctionnelles ainsi que les conditions d'utilisation sont définies de manière déclarative par un descripteur de service. Ces services peuvent être publiés, découverts et invoqués pour être utilisables par d'autres services. Dans la description de service, on trouve des propriétés fonctionnelles et de propriétés non fonctionnelles.

2.2.1.1 Propriétés fonctionnelles

Les propriétés fonctionnelles d'un service sont des propriétés relatives à la fonctionnalité de service, elles sont décrites généralement par les paramètres d'entrées, les paramètres de sorties, les postconditions, les préconditions, et les effets du service. Elles spécifient principalement les conditions de l'invocation du service et le résultat attendu dans le cas d'une exécution réussie.

2.2.1.2 Propriétés non fonctionnelles

Les propriétés non fonctionnelles sont les contraintes sur les propriétés fonctionnelles. Elles se concentrent sur les attributs de qualité de services, tels que la métrique de coût, les mesures de performance (le temps de réponse, la précision, les attributs de sécurité, l'autorisation, authentification, intégrité (transactionnelle), fiabilité, la journalisation des accès, évolutivité et disponibilité).

2.3 Concepts clés de la découverte de services

La découverte ou la recherche de services permettent à chaque utilisateur (client) dan le réseau de trouver les services disponibles d'une manière automatique sans avoir besoin d'une configuration

manuelle. Les technologies de découverte de services sont caractérisées par la méthode de description d'un service, la centralisation ou la décentralisation des descriptions de services, la déclaration de ces descriptions, requête utilisateur, et la gestion de la dynamicité des services.

2.3.1 Description de service

La description de service permet de décrire les fonctionnalité fournies par les services. Le service doit être décrit d'une façon expressive pour que n'importe quel utilisateur(client) puisse comprendre clairement ces caractéristiques, et le protocole de découverte de service puisse correctement le choisir au sein d'un annuaire. Le format de description de service doit être standardisé afin de permettre l'interopérabilité et la compatibilité entre différentes structures et plat-formes de réseau.

2.3.2 Déclaration de service

La déclaration de service consiste à annoncer dans le réseau, la présence d'un nouveau service et la possibilité de l'utiliser. Il existe deux méthodes : la première consiste à enregistrer la description de service auprès d'un annuaire de service. La deuxième consiste à envoyer périodiquement des messages dans le réseau pour annoncer la disponibilité de service.

2.3.3 Annuaire de services

L'annuaire de services est un catalogue de description de services publiés par les fournisseurs. Après la publication et l'enregistrement des services dans l'annuaire, ce dernier est consulté par des clients pour chercher et découvrir des services désirés sans l'intervention manuelle. L'annuaire peut être centralisé, distribué, ou décentralisé. Ces types d'annuaire seront développés en détail dans la section 2.4.

2.3.4 Requête utilisateur

Une requête est une tâche explicitement spécifiée par un utilisateur. Il s'agit d'une demande de service formulée par un utilisateur à un instant donné. Après la découverte de cette requête, le système renvoie une liste de services jugés pertinents.

2.3.5 Découverte de services

La découverte de service est le mécanisme qui permet le partage des services dans un système entre le consommateur (client) et le fournisseur, en d'autres termes, la découverte des services est un processus de recherche et de localisation des services au sein d'un réseau, il consiste pour une entité x (client) à localiser un service s se trouvant sur une entité y (le fournisseur) [12]. Ce processus est déclenché lorsqu'un client x formule une requête dans laquelle il exprime les critères de choix de services qu'il veut récupérer. La découverte de services ne peut néanmoins se cantonner

à la seule fonction de localisation. Cependant Verveidis et Polyzos [137] définissent la découverte de services comme un processus qui permettent aux nœuds de réseau de :

- Annoncer leurs services ;
- Faires des requêtes sur les services fournis par d'autres entités ;
- Sélectionner les services les plus pertinent ;
- Invoquer des services.

Il existe deux type de mécanismes de découverte de services : les mécanismes de découverte active et les mécanismes de découverte passive. Les mécanismes de découverte active imposent au client d'envoyer sa requête au registre de services pour connaître les services disponibles dans le réseau. En revanche, les mécanismes de découverte passive permettent de notifier les clients sur des départs et des arrivées des services.

2.3.6 Dynamicité du réseau

Certains réseaux sont très dynamique comme l'environnement IoT, avec une multitude d'objets et services formant un système à grande échelle, ouvert, et dynamique, dans laquelle la disponibilité des services et des objets change fréquemment. Afin de simplifier le processus de découverte de services, il est nécessaire de gérer l'apparition et la disponibilité des services. La gestion se fait à travers notamment des mécanismes d'annonce et de communication par événements (notifications), et cela pour informer les clients sur le changement d'état et la disponibilité de services. Ces mécanismes ajoutent alors une dynamicité aux interactions entre les dispositifs.

2.4 Architecture pour la découverte de services

Dans la littérature, il y a trois types d'architectures pour les mécanismes de découverte de services [96, 7] : basé sur un annuaire (directory-based), sans annuaire (directory-less) et hybride.

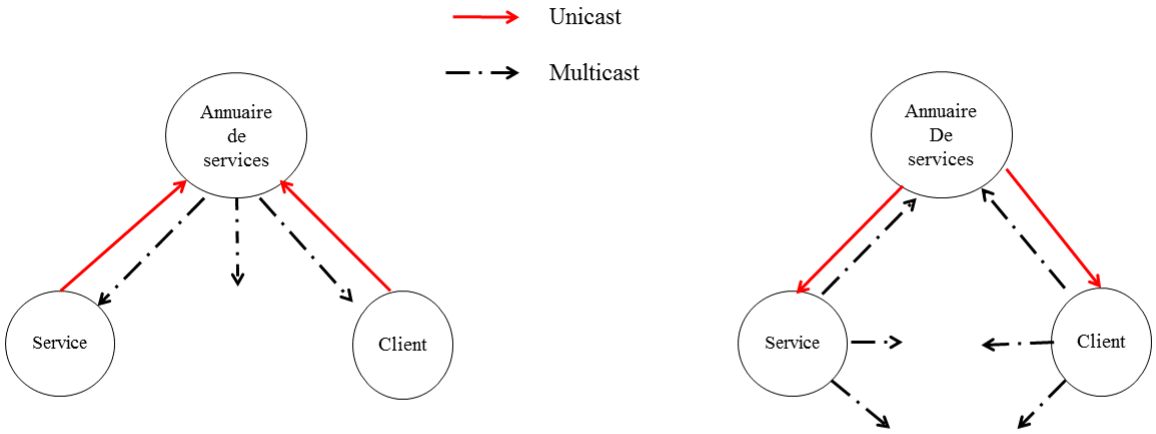
2.4.1 Modèles avec annuaire

Ce modèle se base sur un registre où sont stockés les descriptions de services et les références vers les fournisseurs de services. Dans un modèle basé sur un annuaire, un client x voulant invoquer un service s s'adresse à l'annuaire pour obtenir les informations relatives au(x) fournisseur(s) du service s. Dans le Modèle avec annuaire, les informations liées aux services et fournisseurs peuvent être stockées de façon centralisées ou bien distribuées.

2.4.1.1 Modèle centralisé

Le modèle avec annuaire est dit centralisé lorsque la l'intégrité de l'annuaire est stockée dans une entité unique, ce modèle permet d'avoir une connaissance globale des services enregistrés par

les fournisseurs et facilite la consultions des services. En revanche, une panne de l'entité hébergeant l'annuaire rendrait le système inopérant, et ainsi que la mobilité des dispositifs peut induire une impossibilité de communication entre un client et l'annuaire, ce qui complique davantage la tâche de découverte de services. Ce modèle est utilisé dans le standard le plus utilisé UDDI qui est un annuaire centralisé de services fondé sur XML (Extensible Markup Language) et plus particulièrement destiné aux services Web, il permet d'enregistrer et de localiser sur le réseau le service Web recherché. Dans l'approche centralisée, Il existe deux types de mécanismes de découverte : active et passive (voir la figure 2.1), ces deux dernières sont définies dans la section 2.3.5.



(a) Découverte passive **(b) Découverte active**

FIGURE 2.1 – Approche centralisée

2.4.1.2 Modèle distribué

Dans le modèle avec annuaire distribué, plusieurs nœuds du réseau sauvegardent une partie de l'annuaire. Plusieurs approches du modèle distribué existent [137] [96], je donne ci-après, les plus courantes.

L'approche multi-annuaires repose sur l'utilisation de plusieurs annuaires, chaque annuaire est sauvegardé par un nœud différent. Dans ce modèle le fournisseur a la possibilité de choisir un fournisseur sur lequel il enregistre ses services, et il ne nécessite pas une communication entre les annuaires. Nous pouvons citer comme exemple le système JINI/river qui est un protocole de découverte de services reposant sur une approche multi-annuaires[138]. Cette approche permet de réduire l'impact des défaillances qui peuvent survenir sur les nœuds où sont sauvegardés les

parties de l'annuaire.

L'approche par réplication, qui se base sur l'approche multi-annuaires avec utilisation du mécanisme de réplication de l'annuaire dans plusieurs nœuds, comme c'est une réplication, elle nécessite une synchronisation et une communication régulières entre les annuaires. Nous pouvons citer comme exemple le protocole SDP [121]

L'approche par ensemble dominant (backbone) repose également sur l'utilisation de plusieurs annuaires. Les nœuds hébergeant ces annuaires sont élus de façon distribuée parmi l'ensemble des entités du réseau de manière à former un sous-réseau connexe (le backbone). Contrairement à l'approche par réplication, l'approche par backbone n'utilise pas la synchronisation sur les annuaires. En revanche, ces nœuds communiquent entre eux afin de découvrir une requête de services. Ainsi, l'enregistrement ou la consultation d'un service n'ont besoin d'être effectués qu'auprès d'un unique annuaire, cet annuaire est choisi par des fonctions aléatoire. et si les services recherchés ne sont pas sauvegardés, la requête est transmise aux autres membres du backbone. Le protocole DSDP (Distributed Service Discovery Protocol) [83] utilise une fonction aléatoire pour choisir l'annuaire tandis que le protocole SSD [123] réutilise des informations gardées en cache pour sélectionner l'annuaire le plus pertinent.

L'approche hiérarchique (par cluster) repose sur le partitionnement du réseau en sous groupes possédant chacun son annuaire propre, comme l'approche par backbone. Cependant, dans cette approche chaque groupe possède son annuaire propre et ne peut communiquer qu'avec lui. La formation et le maintien des groupes requièrent également l'échange régulier de messages de contrôle entre les nœuds du réseau. Les groupes peuvent être constitués selon des différents critères comme le type de services [78], la proximité géographique des nœuds [79] où encore le type de mobilité des nœuds [124].

Dans *l'approche par DHT (Distributed Hash Table)*, l'annuaire est subdivisé en partie, chaque partie de l'annuaire est stockée dans un nœud du réseau. Afin de déterminer la partie de l'annuaire a été stockée dans un nœud, une fonction de de distance utilisée, basé sur identifiant unique pour chaque nœud. Cette fonction permet de connaitre les fournisseurs de services et elle permet de calculer une le nombre de sauts entre deux identifiants. Dans dans la littérature, plusieurs solution basé sur DHT, dans [93] utilise par exemple l'opérateur ou exclusif (XOR) comme fonction de distance. Chord [14] repose quant à lui sur un overlay en anneau.

Les approches par annuaire distribué reposent généralement sur l'utilisation de routes calculées par des protocoles de routage opérant sur le réseau sous-jacent afin d'établir les communications

entre clients, annuaires et fournisseurs de services. Les performances des protocoles de routage utilisés ont donc un impact sur celles des mécanismes de découverte de services mis en œuvre dans le réseau. Les différentes analyses montrent que le modèle distribué est adapté aux réseaux de grande taille ayant une faible mobilité [137] [96]. Les changements topologiques impliquent des modifications dans la structure du mécanisme de découverte (backbone, cluster, overlay).

2.4.2 Modèle décentralisé (sans annuaire)

Dans ce modèle, chaque fournisseur est responsable de l'annonce de ses propres services au reste des membres du réseau. Les communications entre clients et fournisseurs peuvent reposer sur des routes établies par des protocoles de routage. Elles peuvent aussi reposer exclusivement sur une stratégie d'inondation [137] [96]. Le modèle sans annuaire est plus tolérant à la mobilité que les approches présentées avec annuaire. En revanche, il n'est pas conçu pour être utilisé sur des réseaux de taille importante. Il existe plusieurs modes de fonctionnement pour ce modèle : push, pull et hybride.

2.4.2.1 Mode push

Dans ce mode, la découverte se fait d'une manière proactive, les fournisseurs de services diffusent leurs services par inondation sans que les clients fassent la demande (voir la figure2.2 .b). De ce fait, chaque client stocke un annuaire local (i.e., cache) de services disponibles annoncés par des fournisseurs. Ce cache est organisé sous forme d'un arbre pour classifier les services et pour faciliter la recherche des informations sur ces services. Cette approche est adoptée par le protocole DEAPspace [110], ainsi OLSR-SD [70] qui est un protocole de routage et de découverte proactif OLSR en mode push. Avec l'augmentation du nombre de fournisseur, il entraîne une croissance du trafic et une saturation des communications dans le réseau.

2.4.2.2 Mode pull

Dans ce mode, la découverte se fait d'une manière réactive, Lorsqu'un client x souhaite invoquer un service s se trouvant sur un fournisseur y, le client x émet une requête de découverte de services s par inondation, ensuite la requête est retransmise aux autres nœuds jusqu'à atteindre le fournisseur y, le fournisseur y répond avec un massage unicast. Ce principes est représenté dans la figure 2.2.a. Cette approche est adoptée par le protocole AODV-SD et DSR-SD qui sont des protocoles de routage et de découverte proactif en mode pull. L'augmentation du nombre de clients et de requêtes intensifie en effet considérablement la taille du trafic.

2.4.2.3 Mode hybride

Le mode hybride utilise à la fois le mode push et le mode pull en fonction de divers paramètres (nombre de clients, nombre de fournisseurs, nombre de services, nombre de requêtes). Ce mode est

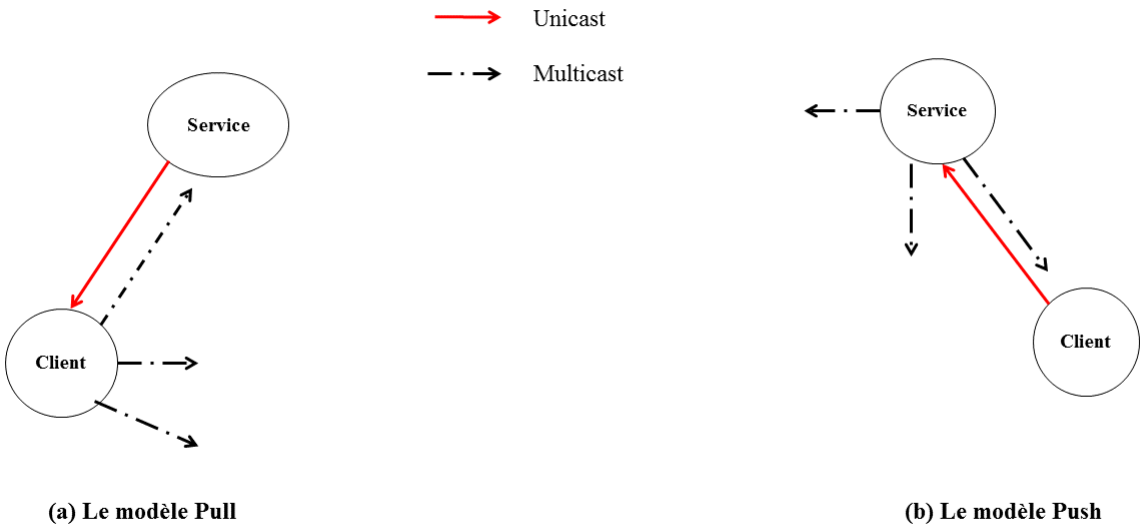


FIGURE 2.2 – Approche décentralisée (a) Pull, (b) Push

adopté par le protocoles de routage et de découverte M-ZRP [136], ainsi le protocole Konark [62], et le protocole HAID (Hybrid Adaptive Protocol for Integrated Discovery) proposé dans [112] se basent sur l'utilisation de la technique Push et la technique Pull où les informations sur les services sont disséminées sur les nœuds d'une zone donnée. Ce type de modèle vise à réduire la surcharge du réseau, en sélectionnant dynamiquement le mode le plus adapté au contexte du réseau[101].

2.4.3 Modèle hybride

Un modèle hybride fonctionne avec ou sans annuaire. Dans ce modèle, un client qui cherche un service s s'adresse en priorité à un annuaire pour effectuer sa requête de découverte. Si aucun annuaire n'est disponible ou si le service s n'est pas trouvé, le client utilise une stratégie sans annuaire pour obtenir la localisation du service s. Peu de systèmes hybrides existent dans la littérature, nous pouvons néanmoins citer LSD (Lightweight Service Discovery) [86] qui est une extension du protocole de routage proactif OLSR. Bien qu'il ait été conçu pour des réseaux filaires, le standard SLP (Service Location Protocol) est également un système de découverte de services hybrides.

2.4.4 Discussion

Il n'existe aucun consensus sur le choix d'une meilleure architecture [137]. Néanmoins certaines études ont montré que la mobilité des nœuds ainsi que la taille du réseau sont les paramètres

les plus cruciaux dans le choix d'une architecture pour la découverte de services et elles ont un impact significatif dans les performances du réseau. Mian et al. [96] estiment que les architectures avec annuaire basées sur un modèle distribué permettent un passage à des échelles importantes tandis que les architectures sans annuaire basées sur une stratégie d'inondation sont plus adaptées à des réseaux à forte mobilité. La figure 2.3 résume les résultats de ces analyses.

Internet des Objets consiste à faire interagir un grand nombre d'objets hétérogènes qui manipulent différents formats de données en utilisant diverses technologies. Cependant, l'accès aux objets et leurs services n'est pas forcement pratique. Pour cela, IoT adopte les deux architectures les plus connus : SOA et REST.

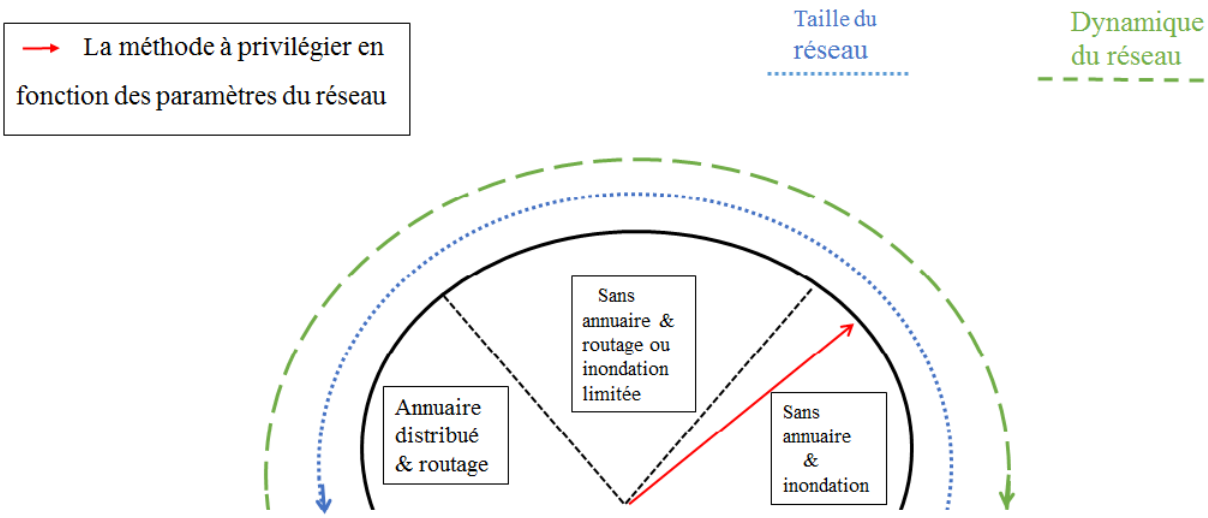


FIGURE 2.3 – Choix de l'architecture d'une solution de découverte de services

2.5 Architecture orientée services (SOA)

Afin d'assurer une interopérabilité entre les entités hétérogènes sur tous les systèmes et plate-formes, les chercheurs ont proposé l'architecture SOA (Service Oriented Architecture) pour la construction rapide et à faible coût de nouvelles applications logicielles [54]. L'architecture orientée services (SOA) est un paradigme largement adopté qui utilise les services comme éléments fondamentaux pour le développement des applications distribuées à l'aide de composants réutilisables et interopérables dont les interactions s'effectuent sur la base d'échanges de messages[117] L'idée principale d'une architecture orientée service est de décomposer une fonctionnalité en un ensemble de fonctions basiques, appelées services, qui sont fournies par des composants et de décrire finement le schéma d'interaction entre ces services. Chaque service représente une unité autonome, auto-descriptive et indépendante des plateformes sous-jacentes.

Ce service est peut-être publié, découvert et invoqué pour être utilisables par d'autres services ou utilisateur.

Pour le développement et l'intégration de systèmes, les fonctionnalités sont développées sous forme de services interopérables et indépendants. Généralement, le paradigme de l'architecture orientée service est basé sur un ensemble de principes qu'il faut respecter [65] :

— ***Couplage faible*** : Le couplage est le niveau d'interaction entre deux ou plusieurs composants logiciels, deux composants sont couplés s'ils échangent de l'information. Ce couplage est fort s'ils échangent beaucoup d'informations et il est faible dans le cas contraire. Dans une architecture SOA, le couplage entre les applications clientes et les services doit être faible dans le but de garder une faible liaison et de réduire les dépendances entre les services afin d'atteindre un haut degré de flexibilité dans l'architecture. Grâce à la réduction des dépendances entre les services, il sera facile de remplacer les services ou de les relier entre eux sans altérer le reste des services existants. Il sera facile en outre, en cas de défaillance, de remplacer le service défaillant avec un autre service sans arrêter tout le système.

— ***Interopérabilité*** : Elle est considérée comme une propriété principale, l'interopérabilité permet l'intégration simple et la communication entre différents types d'applications développées avec des langages de programmation différents et qui s'exécutent sur des plateformes différentes.

— ***Réutilisabilité*** : La réutilisation des services indique la possibilité de les réutiliser pour des tâches différentes et des utilisateurs différents. Le but de la réutilisabilité est de minimiser la redondance, réduire le coût de développement et facilite la maintenance de l'application.

— ***Découverte*** : Il s'agit de la localisation d'un service afin de pouvoir l'utiliser. La découverte peut être faites d'une manière automatique ou manuelle. La localisation automatique des services est souvent utilisée car elle fournit des informations contextuelles.

— ***Composition*** : C'est la propriété clé de SOA, les services peuvent invoquer ou utiliser d'autres services dont l'objectif est de créer de nouvelles fonctionnalités en combinant des fonctionnalités offertes par des services existants. Cette construction est rendue possible grâce au mécanisme de composition. SOA utilise les technologies et standards offerts par le système afin de créer de nouveaux services (services composites) en composant des services existants (services atomiques).

2.5.1 Acteur et modèle d'interactions dans SOA

L'Architecture Orienté Service (SOA)[56] [89][109] fait intervenir trois acteurs clés : le fournisseur, le consommateur appelé aussi utilisateur, et l'annuaire de services.

— ***Le fournisseur de service*** : Est une entité qui possède une adresse sur le réseau, cette entité développe un ensemble de fonctionnalités (fonctionnelles et non fonctionnelles) sous forme de services. Ces fonctionnalités sont décrites dans une spécification de service. Le Fournisseur publie et enregistre leurs spécifications dans un annuaire de services afin que les consommateurs de services puissent découvrir et accéder au service via une requête de service. Les fournisseurs de service enregistrent la description de leurs services dans l'annuaire.

— ***Le consommateur de services*** *ou client* : Appelé aussi utilisateur qui cherche et invoque un service, il envoie des requêtes à l'annuaire de services pour découvrir les services qui répondent à ses besoins. Après la découverte dans l'annuaire, le client ou utilisateur peut se connecter au fournisseur et utiliser le service.

— ***L'annuaire des services*** : Est un registre où sont stockés les descriptions de services et les références vers les fournisseurs de services. Il permet aux fournisseurs de services de publier des descriptions de services, et il permet aussi aux consommateurs de services de pouvoir consulter les différentes descriptions disponibles. L'annuaire joue le rôle d'un acteur intermédiaire entre les fournisseurs et les consommateurs de services.

Pour garantir une évolution et une meilleur intégration des applications hétérogènes au sein d'une SOA, l'approche à service prévoit un modèle d'interaction entre les trois acteurs (fournisseur, consommateur et annuaire) qui peuvent être résumées en trois primitives principales de communication : la publication, la découverte, et l'invocation (voir la figure 2.4). Nous décrivons ci-dessous le rôle de chaque primitive.

— ***Publication de service*** : Cette interaction s'effectue entre les fournisseurs et l'annuaire de services. Après que les services sont créés est déployés par les fournisseurs, leurs descriptions de services doivent être enregistrées dans l'annuaire de services, pour les rendre accessibles aux clients.

— ***Découverte de service*** : Cette interaction s'effectue entre le consommateur et l'annuaire de services. Elle permet aux consommateurs de service de rechercher dans l'annuaire de services pour découvrir les fournisseurs de services qu'ils demandent.

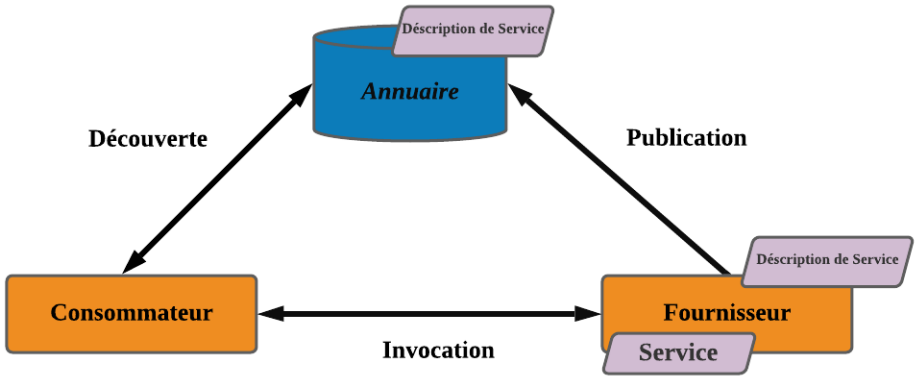


FIGURE 2.4 – Interactions dans l'architecture SOA

— ***Invocation de service*** : Cette interaction s'effectue entre le consommateur et le fournisseur de service. Après avoir consulté les descriptions de services découverts, le consommateur peut connecter avec le fournisseur du service sélectionné et procède à l'invocation de ce service en fonction des informations présentes dans la description de service. L'invocation de ce service peut, par exemple, s'effectuer via le protocole de communication standard SOAP (Simple Object Access Protocol).

Les protocoles standards de communication utilisés dans SOA sont conçus pour que les applications développées avec différents langages sur différentes plateformes puissent communiquer. Comme il s'agit d'un protocole, il impose des règles intégrées et des exigences spécifiques, et cela augmentent la complexité et les coûts. Cependant, il existe une forme de SOA encore moins restrictive qu'un transfert d'état de représentation de services, C'est le type REST.

2.6 REST (Representational State Transfer)

REST (Representational State Transfer) a fait l'objet de la thèse de R. Fielding [46]. La première idée pour résoudre la problématique de l'interopérabilité est d'utiliser l'approche SOA, cette approche propose une idée intéressante et efficace aux problèmes que rencontrent les entreprises en termes de réutilisabilité, d'interopérabilité et de réduction de couplage entre les différents systèmes qui implémentent leurs systèmes d'information. Cependant, le protocole principalement utilisé dans ces architectures (SOAP) augmentent la complexité. En effet, les messages SOAP échangés entre le client et le serveur sont des flux XML de taille non négligeable via le protocole applicatif HTTP (Hyper Text Markup Language). REST n'est pas un protocole, mais plutôt une approche architecturale (voir la figure 2.5), dont l'objectif est de faciliter la programmation d'applications orientées services en utilisant HTTP.

Dans REST, l'entité de base s'appelle une ressource, toute information peut être une ressource,

on peut citer par exemple, une photo, un service, etc. Une ressource dans REST possède un identificateur unique, dans le Web les identificateurs sont URI (Uniform Resource Identifier). REST met l'accent sur les rôles de client et de serveur, sur l'identification des ressources accessibles grâce à un adressage unique, et sur des échanges ne nécessitant pas d'états et qui contiennent toutes les informations nécessaires au traitement. Aussi, la faculté d'HTTP de transporter des informations et de disposer des verbes nécessaires à CRUD (Create, Retrieve, Update, Delete)[104].

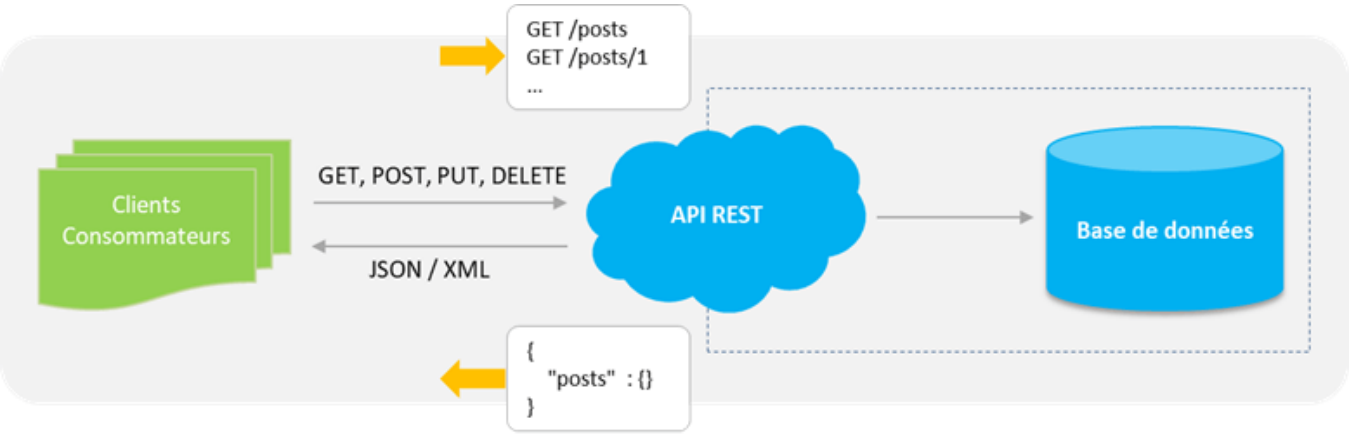


FIGURE 2.5 – Architecture REST

L'architecture REST est basée sur les éléments suivants :

— ***Identifiant de la ressource*** : chaque ressource dans REST possède un identificateur de ressource. Cet identificateur permet aux composants de l'architecture d'identifier les ressources qu'ils manipulent.

— ***Représentation de la ressource*** : Les composants de l'architecture traitent les ressources en se basant sur le transfert de leurs représentations. Sur le Web, nous trouvons aujourd'hui la plupart des représentations au format HTML, JavaScript Object Notation JSON ou Extensible Markup Language XML.

— ***Opération sur la ressource*** : Les ressources sont manipulées en transférant les représentations des ressources à travers une interface uniforme en utilisant l'identifiant de ressource. Chaque représentation doit implémenter une interface CRUD pour réaliser l'interface uniforme requise par l'architecture REST. Le protocole HTTP/1.1 définit huit méthodes, dont quatre permettent de définir l'interface : GET, PUT, POST et DELETE, ci-dessous, le rôle de chaque méthode.

- La méthode *GET* est la plus utilisée, elle permet de récupérer l'information d'une ressource ;

- La méthode *POST* permet d'envoyer, de mettre à jour ou d'insérer une ressource ;
- La méthode *PUT* permet de mettre à jour une ressource ;
- La méthode *DELETE* permet de supprimer une ressource.

Dans le contexte de systèmes IoT avec des objets très contraints, REST est un choix de plus en plus répandu. La plupart des opérations courantes peuvent être réalisées grâce à HTTP, plus simplement, de façon plus légère et concise [53]. L'utilisation des services pour représenter les dispositifs, offre des facilités pour l'interopérabilité entre les dispositifs et pour la découverte de services.

2.7 Principales techniques de découverte de services dans l'Internet des Objets

La problématique de la découverte de services a fait l'objet de plusieurs travaux de recherche comme en témoigne la littérature. On distingue cinq catégories d'approches : les approches basées sur la sémantique, les approches bio-inspirées, les approches basées sur les protocoles de découverte de services, les approche basées sur le contexte, les approches basées sur la qualité de service.

La figure 2.6 montre notre classification des différentes approches de découverte de services dans IoT.

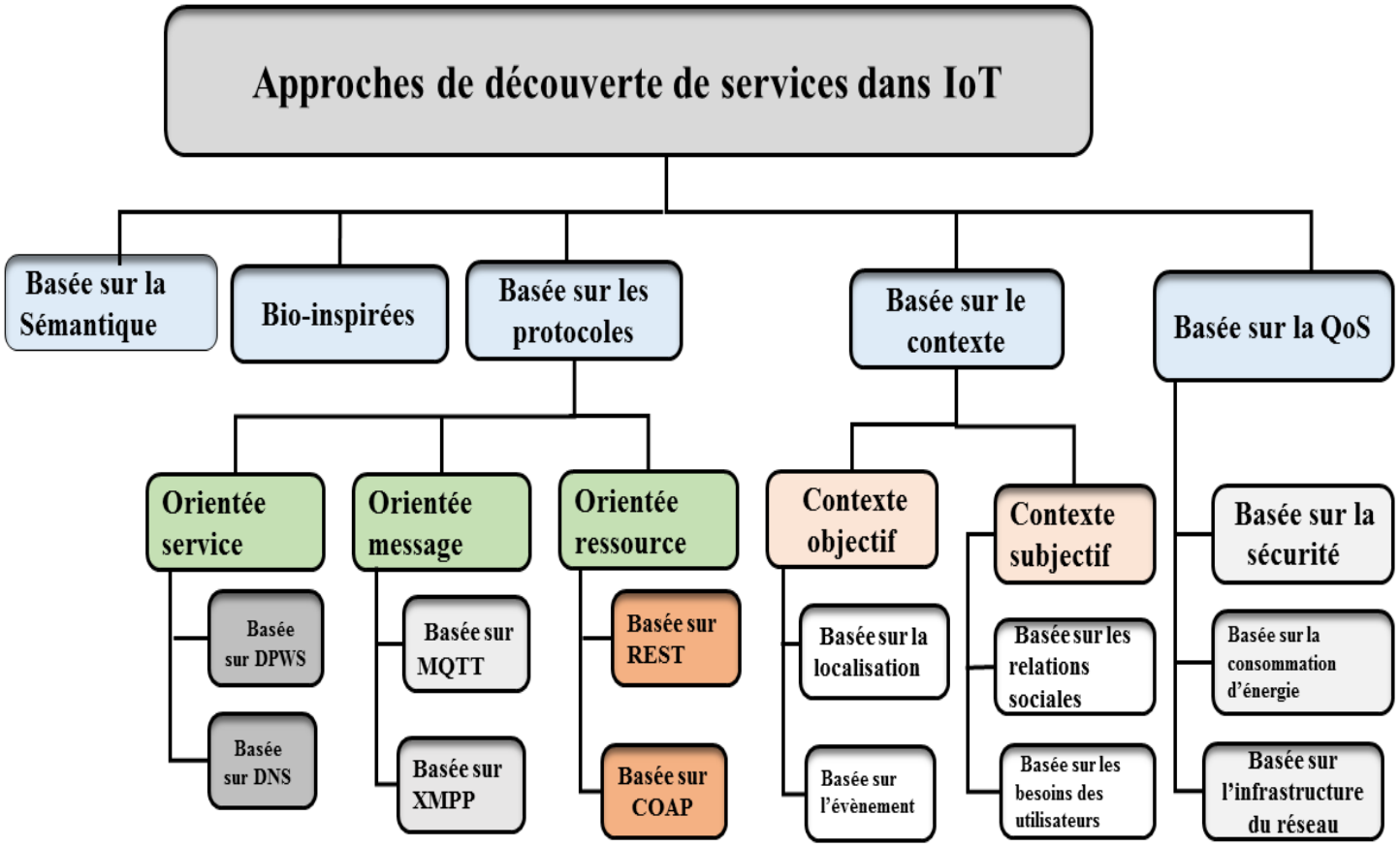


FIGURE 2.6 – Calssification des approches de découverte de services dans l'IoT

2.7.1 Approches basées sur la sémantique

Dans cette approche, la découverte d'un service dans Internet des Objets utilise les techniques de Web sémantique comme la similarité, la relation entre les concepts, les ontologies etc. Plusieurs travaux se sont basés sur ces notions de sémantique et d'ontologie pour la découverte de services.

Dans [48], les auteurs proposent une approche de découverte efficace et scalable des services de l'Internet des Objets, en se basant sur le Web Sémantique et en supportant des contextes dynamiques. L'approche se base sur des passerelles sémantiques distribuées qui représentent différents espaces intelligents et enregistrent leurs services associés. Le système de découverte intègre des mécanismes de groupement, d'agrégation de l'information et de routage sémantique. Le système proposé permet de trouver tous les services qui répondent à une requête donnée en limitant les opérations inutiles d'appariement sémantiques.

Dans [68], un modèle centralisé est proposé, structuré à quatre couches qui sont : la couche d'interface interactive, la couche d'analyse (parsing), la couche service matching (similrité), et la couche sémantique de données. Ils ont aussi proposé une méthode hybride pour calculer le degré d'appariement de service en utilisant la similarité sémantique, la relation logique entre les concepts pour chaque couche séparément. La découverte se fait en quatre étapes afin d'améliorer les performances de découverte de services.

Dans [150], les auteurs ont proposé un algorithme MDM (Multiple Dimensional Measuring) pour décrire les services IoT et mesurer le degré de similarité entre les services sur chaque dimension. Une dimension est un modèle sémantique qui contient une catégorie de services bien définie (des propriétés de services et les contraintes sur les propriétés). Ils ont utilisé la technique de clustering, basée sur les pics de densité pour répartir les services en clusters et cela en fonction du degré de similarité.

Discussion

Les principaux avantages de ces travaux concernent la possibilité de décrire et déployer les différents services de manière décentralisée et indépendante au sein de chaque domaine du réseau. Il est aussi possible de mieux répondre aux besoins d'un client à travers la restriction à un ensemble approprié de domaines du réseau qui correspond aux exigences du service demandé par le client. Cependant, Dans un réseau IoT plus particulièrement, le nombre de services est très élevé peut limiter le passage à l'échelle. De plus, Il est aussi possible d'avoir des services inadéquats par rapport aux attentes et aux exigences du client puisque l'usage des ontologies limite la précision.

2.7.2 Approches bio-inspirées

Les approches bio-inspirées sont largement utilisées dans plusieurs domaines notamment la découverte de services, Ils sont inspirés du processus d'adaptation biologique des êtres vivants à leur environnement. Certains travaux ont utilisé ce paradigme dans la découverte de services dans l'IoT.

Dans[15], les auteurs ont développé une plateforme USPIOT (Ubiquitous Services Platform for IoT), USPIOT qui s'appuie sur la méthode de description de service basée sur XML. La recherche et la sélection de services se fait avec l'algorithme AMFSS (Adaptive Momentum Factor BPS Service) qui est basé sur les réseaux de neurones artificiels (ANN) .

Dans [119], les auteurs ont proposé une solution décentralisée (sans annuaire) basée sur les champs potentiels artificiels (Artificial Potential Fields APFs), ces champs sont formés à chaque

requête de services, ici le service est une fonctionnalité. Après une requête de services envoyé par un utilisateur, les nœuds qui offrent le service recherché deviennent actifs nommés SPN (service provision nodes), les SPN sont gérés par des agents. La force de chaque APF dépend du pourcentage de services pouvant être offert après une requête de services. La découverte et la sélection sont alors conduites par des forces artificielles appliqués entre les nœuds demandant le service (requested service) et les SPN.

Discussion

Les approches bio-inspirées ont rencontré un vif succès grâce à leur simplicité, leur adaptabilité, leur flexibilité et leur capacité de trouver des solutions presque optimales. Ces méthodes sont très efficaces et largement utilisées dans la découverte de services.

2.7.3 Approches basées sur les protocoles

Les protocoles de découverte de services fournissent des mécanismes pour la découverte spontanée et dynamique de services disponibles dans le réseau. D'après [49], il existe trois approches principales dans l'enivrement IoT : l'approche orientée service, l'approche orientée message, et l'approche orientée ressource. Dans ce qui suit, nous donnons un aperçu sur les protocoles et les approches les plus utilisés dans l'Internet des Objets et nous montrons comment chaque protocole assure la découverte de services.

2.7.3.1 Approches orienté service (SOA)

Dans SOA, un service est une unité de fonctionnalités qui permettent l'accès, la recherche et le traitement de l'informations, et aussi la possibilité de communiquer avec d'autres utilisateurs ou d'autres applications. Au sein de cette classe, nous pouvons identifier deux classes d'approches, les approches basées sur DPWS, et les approches basées sur DNS.

DPWS (Devises Profile for Web Services)

DPWS est considéré comme la version 2 de UPnP (Universal Plug and Play), son objectif est d'offrir un accès facile et intuitif aux ressources basant sur les extensions standardisées WS, il emploie notamment les standard WSDL, SoAP, WS-Addressing, WS-Eventing, WS-Policy et WS-Discovery. DPWS est fondé sur trois entités : les équipements, les services, et les points de contrôle. Les équipements pour héberger des services, les points de contrôles permettent de découvrir les services disponibles sur le réseau, et de s'y abonner. La découverte de services se fait par envoi de message multicast et l'invocation de service se fait au mode passif et actif avec et sans annuaire de services.

Dans [54], les auteurs ont proposé un processus RSDPP (Real-World SD et un Provisioning Process) utilisé pour aider les développeurs dans la conception d'un SD. Il utilise les techniques DPWS pour enregistrer dynamiquement et découvrir les dispositifs et les services fournis via un API RESTful.

Dans [58], Son Han et al. ont étendu le DPWS en utilisant un proxy REST pour surmonter ses limites, réduire la latence, et simplifier la topologie globale d'utilisation des API Web RESTful.

Approches basées sur le DNS

Dans un réseau, les dispositifs sont identifiés par des adresses IP. Néanmoins, ces adresses sont plus difficiles à gérer et à manipuler, d'où la nécessité d'utiliser des DNS (Domain Name System) qui assurent la conversion entre les noms des dispositifs et les adresses IP. DNS étant une technologie relativement simple, très largement utilisé, il est un annuaire hiérarchique, contenant les descriptions des services enregistrés dans un serveur DNS. Cependant, le nombre d'objets connectés IOT augmente au fur et à mesure, passant de milliards à bientôt centaines de milliards. Si l'on considère que chaque dispositif IoT offre un ou plusieurs services, leur découverte est une tâche complexe du moment où le nombre de services devient très important. Pour remédier à cette situation, le protocole mDNS (multicast DNS) Service Discovery est développé afin d'assurer une découverte rapide des dispositifs et des services locaux en utilisant le multicast dans l'ensemble du réseau ou dans un sous-réseau. Nous citons ci-dessous quelques travaux utilisés mDNS-SD (multicast DNS Service Discovery) dans la découverte de services dans l'IoT.
Dans [91], les auteurs ont proposé un protocole mDNS/DNS-SD pour la découverte optimisée, distribuée et autonome pour les objets limités en terme d'énergie, de calcul, de stockage, et de communication.

Dans[130], les auteurs ont proposé une extension pour le protocole mDNS/DNS-SD, ils ont intégré le contexte de services dans la découverte des services. Cette amélioration peut réduire considérablement l'espace de stockage (utilisation du contexte au lieu de nom de domaine), et réduire aussi le nombre et la taille des messages échangés pendant la découverte de services. Cette approche assure la scalabilité dans les réseaux à grand échelle comme le réseau IoT.

2.7.3.2 Approche orientée message

Cette classe d'approches considère la fourniture et le transfert de données et de messages asynchrones entre les dispositifs distribués pour découvrir des services. La fiabilité et la qualité de service (QoS) des messages échangés sont contrôlées par des entités centralisées appelées courtiers. Parmi ces protocoles on peut citer MQTT et XMPP.

Approche basée sur MQTT

Le protocole MQTT (Message Queuing Telemetry Transport) est l'un des protocoles les plus utilisés dans l'IoT standardisé par l'organisme OASIS, il permet à deux dispositifs distants de communiquer par des messages de manière asynchrone avec une faible bande passante. MQTT utilise le mécanisme *publication/souscription*, chaque objet connecté crée pour lui un sujet (Topic) qui formera un canal (une chaîne d'information) sur laquelle il peut publier ses messages, cette objet, ainsi, est appelé publieur (publisher). Les Topics sont gérés par un broker MQTT (analogue à un serveur dans le cas de HTTP). De l'autre côté du broker, des abonnés (subscribers) peuvent lire les messages d'un Topic. Les Topics (ou les canaux d'informations) peuvent avoir une hiérarchie qui permet de sélectionner finement les informations que l'on désire. Il existe une variante de MQTT adaptée aux objets avec ressources limités appelés MQTT-SN (MQTT for Sensor Network), l'une de ses principales modification est l'utilisation d'UDP au lieu de TCP. Quelques travaux ont utilisé MQTT dans la découverte de services dans l'IoT.

Dans [75], les auteurs ont proposé une architecture pour la découverte de services avec une configuration automatique en utilisant le modèle de messagerie (publication/abonnement) de MQTT. De plus, une passerelle agit comme un broker, elle utilise le protocole DNS-SD /mDNS pour enregistrer et découvrir les services selon leurs contextes en utilisant la technologie du Web sémantique.

Dans [116], les auteurs ont proposé une approche distribuée pour la découverte de ressources basée sur MQTT nommée MQTT-RD (MQTT Resource Discovery). Cette approche permet l'auto-configuration des nouveaux dispositifs avec des opérations plug and play, les publishers et les subscribers peuvent rejoindre et quitter les brokers à tout moment. Ceci est particulièrement pertinent pour les appareils alimentés par batterie et qui ont besoin de se mettre en veille pour économiser leur énergie. D'autre part, l'approche permet aussi l'enregistrement et la recherche de leurs topics dans RD.

Approche basé sur XMPP

Extensible Messaging and Presence Protocol (XMPP) est un protocole de communication basé sur une architecture client-serveur permettant les échanges décentralisés de messages instantanés ou non, entre les clients, au format Extensible Markup Language (XML). Comme son nom l'indique, XMPP est extensible, offrant ainsi la possibilité d'avoir d'autre extensions. On trouve EXI (Efficient XML Interchange) utilisé pour applications IoT, dans l'échange décentralisé des données structurées entre deux ou plusieurs entités connectées au réseau en temps quasi réel. Quelques travaux ont utilisé XMPP dans la découverte de services dans l'IoT.

Dans [139], les auteurs proposent un protocole nommé lightweight XMPP, Protocole léger de messagerie instantanée de type publish/subscribe utilisé pour les ressources limitées de l'IoT afin d'effectuer des échanges périodiques de données. Ce protocole prend en charge la publication d'événements et la publication périodique de données dans l'IoT.

2.7.3.3 Approche orientée ressource

Dans cette approche, les protocoles sont basés sur l'architecture REpresentational State Transfer (REST). Il existe deux classes, REST et CoAP.

REST

Dans REST l'entité de base s'appelle une ressource, toute information peut être une ressource,

Une ressource dans REST possède un identificateur unique, dans le Web les identificateurs sont URI (Uniform Resource Identifier). REST met l'accent sur les rôles de client et de serveur, sur l'identification des ressources accessibles grâce à un adressage unique, et sur des échanges ne nécessitant pas d'états et qui contiennent toutes les informations nécessaires au traitement.

Dans [74] les auteurs ont développé un Framework pour définir, découvrir et composer les objets et leurs services. Le langage de modélisation utilisé est RESTful API (RAML). Le protocole de découverte de services est implémenté comme RESTful WS(Web service) à deux phases recherche et sélection. Pendant la phase de recherche, l'approche cherche des entités qui répondent à un ensemble de critères. Pendant la phase de sélection, l'approche utilise le résultat de la première phase de recherche afin de choisir et sélectionner les objets qui fournissent les services appropriés.

CoAP

Constrained Application Protocol (CoAP)[127], est un protocole applicatif conçu par le groupe

IETF CoRE pour les environnements contraints. Ce protocole a une forte similarité avec le protocole HTTP mais il utilise le protocole User Datagram Protocol (UDP), CoAP repose sur le modèle REST (modèle Client-Serveur), Pour cela, tout objet est vue comme une ressource REST, sur laquelle nous pouvons utiliser les verbes habituels de HTTP, à savoir GET, POST, PUT et DELETE. Ainsi, une ressource possède un identificateur unique, dans le web les identificateurs sont URI et une ressource est accessible via une URL.

Dans [135], les auteurs proposent le protocole S-CoAP (Semantic Enrichment of CoAP) pour la découverte de ressources, auquel ils ajoutent une dimension sémantique, des annotations sémantiques sous la forme d'attributs ont été ajoutés au protocole CoAP, et aussi ils ont proposé un Framework basé sur CoAP et le paradigme Internet Social des objets qui permet une découverte

plus efficace dans un environnement distribué.

Dans [133], les auteurs proposent une architecture distribuée basée sur le protocole CoAP. La découverte de services se fait par une table de hachage distribuée (DHT) maintenue par des passerelles déployées dans le réseau IoT.

Dans [45], les auteurs proposent un middleware hybride basé sur CoAP, se focalise sur l'architecture cloud computing. Cette approche permet de relever les défis de flexibilité, d'extensibilité et d'interopérabilité des réseaux IoT à des dispositifs très limité en ressources.

Discussion

Les protocoles de découverte de services présentés dans la section 2.7.3 sont proposés pour faciliter la coopération, la communication et l'interopérabilité des objets. Pratiquement, tous les protocoles cités précédemment fournissent des mécanismes similaires, à la base, des clients doivent retrouver des descriptions de services pertinentes, comprenant des informations suffisantes pour établir le contrat avec le service. Il existe deux mécanismes de base, l'annonce de service et la recherche de service qui se déroulent dans l'un des deux types de topologies : centralisée ou distribuée. Cependant les protocoles centralisés (MQTT, CoAP, DNS-SD) posent de nombreux problèmes par rapport aux protocoles distribués (m DNS), tels que la défaillance de l'annuaire et la congestion potentielle du réseau autour de l'annuaire. Pour ces raisons, les protocoles distribués sont plus couramment utilisés.

2.7.4 Approches basées sur le contexte

Plusieurs approches ont été proposées dans la littérature pour permettre la découverte de tels services avec l'intégration des informations contextuelles. Ces approches peuvent être classés en deux classes selon leurs contextes : Contexte objectif et Contexte subjectif. Dans le contexte objectif, la découverte de services utilise les techniques SD centralisé. Dans le contexte subjectif, la découverte de services se base principalement sur les relations sociales entre les utilisateurs et ainsi leurs relations avec les objets IoT.

2.7.4.1 Contexte objectif

La découverte de services basée sur le contexte objectif utilise plus souvent des propriétés sur la localisation, le temps, les dispositifs, le réseau, et le profil de l'utilisateur.

Approches basée sur la localisation

Cette approche se focalise sur la localisation d'un objet du contexte. Les objets sont spatialement organisés et les personnes se déplacent d'un endroit à un autre. Aujourd'hui, la plupart des

systèmes pervasifs prennent en compte cette approche.

Dans [26], les auteurs proposent une approche de découverte basée sur le contexte de la localisation pour les réseaux IoT, ils ont développé un framework décentralisé orienté services, ils ont intégré toutes les informations liées à la localisation de services qui sont stockées dans des modèles sémantiques basés sur des ontologies.

Approche basé sur évènement

Dans cette classe, la découverte de services se base sur les évènements générés par des entités. Une entité peut être une personne, un dispositif, un lieu ou une application informatique. Les entités sont en général caractérisées par des propriétés (niveaux de bruit, de luminosité, de température, d'humidité, de fuite de gaz...).

Dans [146], les auteurs développent un Framework pour la découverte dynamique de services dans un environnement ambiant. Ce Framework est composé de cinq unités (Ambient services discovery, ambient services classification, user task specification, events handling, et ambient services selection). L'unité découverte de services effectue une souscription auprès des répertoires de services pour écouter passivement les annonces spontanées relatives aux événements de publication, de modification ou de suppression de services via le protocole Active Ambient Services Discovery Algorithm (A2SDA) . Par ailleurs, du fait de la possibilité de connexions et de déconnexions imprévisibles des répertoires de services, la requête active est renouvelée périodiquement par le module de découverte de services via le Passive Ambient Services Discovery Algorithm (PASDA).

Dans [55], les auteurs développent une architecture EDSOA (Event-driven service-oriented architecture), c'est une extension de SOA. Par opposition à l'architecture SOA où un fournisseur rend un service à la demande d'un client (consommateur), en architecture orientée évènement, un service prévient par émission d'un événement qu'il vient de réaliser une opération donnée. Ensuite, c'est aux clients potentiels (i.e. consommateurs d'événements) de traiter cet événement. Les auteurs constatent que l'architecture proposée est la meilleurs pour IoT.

Dans [28], les auteurs proposent une plateforme de coordination de services basée sur les évènements. Avec l'utilisation des automates et la coordination des différents services basée sur les évènements, les auteurs ont développé un algorithme de détection d'événements pour une situation donnée. Ils ont utilisé le modèle de messagerie (publication/abonnement) pour faciliter la communication asynchrone entre les différentes entités, et la distribution à la demande des données sensorielles dans un environnement IoT distribué à grande échelle.

2.7.4.2 Contexte subjectif

La découverte de services basée sur le contexte subjectif utilise plus souvent les relations sociales et les besoins et intérêts des utilisateurs.

Approche basé sur les relations sociales

L'augmentation du nombre d'objets dans l'environnement IoT, introduit des problèmes d'interopérabilité et limite l'aspect évolutif des réseaux d'objets communicants. Ceci a favorisé l'émergence d'une nouvelle approche appelée Social Internet of Things (SIoT), dans lequel les objets peuvent se mettre en relation comme les êtres humains pour offrir ou consommer un service donné.

Dans [118], les auteurs développent une plate-forme SIoT pour un environnement IoT, où les objets sont capables de créer leurs propres relations basées sur des règles définies par leurs propriétaires, tout en minimisant l'intervention humaine. Cette approche permet de créer des communautés d'objets régies par des relations sociales. Ces relations peuvent affecter les interactions sociales entre les objets communicants surtout aux niveaux de la confiance entre les partenaires, et de la fiabilité des services.

Dans [145], les auteurs ont proposé un mécanisme décentralisé pour la découverte de services basé sur les relation sociales et sémantiques dans un environnement IoT à grande échelle nommé SLSA (Social Like Semantic-Aware service discovery). Dans SLSA, les services sont décrits au moyen d'ontologies OWL. Ils ont considéré à la fois la similarité sémantique et les relations sémantiques. Ils ont utilisé la logique floue pour calculer les coefficients de corrélation pour classer les objets. Le SLSA utilise la diffusion adaptative, dans laquelle la requête de services est transférée à un sous-ensemble sélectionné d'objets voisins ordonnés. Les auteurs ont montré que cette approche permet une découverte rapide d'un service demandé dans un réseau grand échelle.

Dans [149], les auteurs ont proposé une architecture décentralisée, et auto-organisée pour les réseaux sociaux en ligne pour la découverte efficace des services dans un environnement IoT social et dynamique. La principale caractéristique de ce modèle est l'utilisation du phénomène de l'homophilie pour intégrer les relations et les intérêts des utilisateurs. Ce modèle est capable d'identifier des voisins ayant une certaine mesure de similarité avec le fournisseur de services potentiel qui possède un nombre élevé de connexions. En outre, pour obtenir une recherche efficace, un nouvel algorithme OSS (Olfactory Sensitive Search) est proposé. Cet algorithme est utilisé pour trouver le plus court chemin avec un maximum de services demandés.

Approche basée sur les besoins des utilisateurs

Cette approche se focalise sur les besoins des utilisateurs en termes de leurs intérêts et leurs contextes pour la découverte de services les plus appropriés.

Dans [115], les auteurs ont proposé une approche nommée ProSA (Progressive Search Algorithm). Cet algorithme établit des correspondances entre les besoins de l'utilisateur avec les attributs des ressources IoT, les services fournis ainsi que les résultats de recherche. Ils ont classé les besoins de l'utilisateur en deux types (essentiels et facultatifs). L'algorithme utilise deux stratégies de recherche PSS (Primitive Search Strategy) et ESS (Elaborate Search Strategy). Le PSS est utilisé pour réduire l'espace de recherche et avoir les services selon les besoins essentiels de l'utilisateur en fonction de leur similarité. L'ESS est utilisé pour récupérer les services de recherche personnalisés selon les besoins facultatifs de l'utilisateur.

Dans [25], les auteurs ont proposé un protocole de découverte de services adaptatif aux contextes (utilisateur et réseau) pour l'IoT nommé TRENDY. Ce protocole utilise CoAP pour la description des services, et la technique APPUB (Adaptive Piggybacked Publish) pour trouver un équilibre entre le délai d'invocation du service et l'efficacité du réseau. De plus, un mécanisme de mise en cache adaptatif est intégré pour les services les plus demandés afin de réduire les coûts en énergie induits par la communication. Comme les applications sont dépendantes de leurs contexte, TRENDY exploite le regroupement des dispositifs en se basant sur les contextes des utilisateurs, ce qui permet au réseau IoT une meilleure mise à l'échelle des services. Les résultats de simulation montrent que le protocole TRENDY contrôle la surcharge du réseau et réduit la consommation d'énergie.

Discussion

Les approches basées sur le contexte pour la découverte de services requièrent l'identification de l'ensemble des informations de contexte (la localisation de l'utilisateur, les relations sociales les évènements, les besoins des utilisateurs) lié à un service. La prise en compte du contexte lors du processus de découverte permet la découverte de services plus pertinents et plus proches des besoins de l'utilisateur. Cependant, une limite naturelle de l'adaptation du contexte est la nécessité de disposer d'une représentation variée et riche du contenu, ce qui n'est pas toujours le cas. La précision de la recherche est liée à la quantité d'information dont dispose le système. En plus, avec le très grand nombre d'objets connectés au réseau, entraîne une explosion du volume des données. En conséquence, des solutions de stockage et d'acquisition sont nécessaires afin de permettre l'exploitation de ces données par des mécanismes d'adaptation.

2.7.5 Approche basée sur QoS

La Qualité de Service (QoS : Quality of Service) désigne la capacité à découvrir des services conformément à des exigences appelées attributs de QoS et de sécurité, ces exigences doivent être implémentées au niveau des différentes couches de l'architecture IoT afin de fournir une QoS et une sécurité de bout en bout pour l'utilisateur des services IoT.

2.7.5.1 Approche basée la sécurité

Afin de garantir la sécurité dans un environnement IoT où les menaces issues de l'interconnexion d'un grand nombre d'objets ainsi que l'hétérogénéité des équipements, des mécanismes de sécurité pour découvrir des services sont implémentés, tels que l'authentification, la confiance, la confidentialité et le contrôle d'accès aux objets.

Dans [84], l'approche proposée définit un système ISS (IoT Service Store) de découverte et l'interaction de services IoT, ISS est un registre où sont enregistrés les services, il gère les informations confidentielle et les préférences des utilisateurs, il permet également d'inspecter les propriétés de confidentialité et de contrôler les données collectées à partir des capteurs tout en évaluant les risques potentiels sur la confidentialité.

Dans [31], l'approche proposée détaille une architecture SD sécurisée et distribuée appliquée aux réseaux EPCglobal, cette approche est basée sur les tables de Hachage Distribuées la DHT (Distributed Hash Table), les DHT fournissent une association générale entre une clef et toute sorte d'information (comme l'identité d'un nœud ou une position). L'idée est de mettre des ONS (Object Naming Service) locaux dans les différentes sociétés inscrites au réseau EPCglobal, ONS joue un rôle important dans le partage des données des étiquettes RFID. Il va permettre de récupérer des adresses de services disponibles en fonction de l'identifiant sur une étiquette RFID (code EPC - Electronic Product Code) contenu dans la requête. L'ONS est une sorte d'annuaire qui permet à une entreprise de suivre la progression et le statut des marchandises depuis leur production, jusqu'à leur espace de vente. Les informations liées à l'EPC (Electronic Product Code) sont stockées dans un nœud DHT où les abonnés EPCglobal peuvent les récupérer. Ces améliorations visent à remédier les inconvénients majeurs de l'architecture centralisée d'ONS qui a prouvé sa robustesse et son efficacité dans les systèmes Peer-to-Peer (P2P).

2.7.5.2 Approche basée sur la consommation d'énergie

La consommation d'énergie est également un problème de qualité de service. Les auteurs présentent de nouvelles techniques sensibles à l'énergie et à la qualité de service afin de résoudre

le problème de découverte de services.
Dans [126], l'approche proposée est basée sur l'utilisation des UAVs (Unmanned Aerial Vehicle) dans le cadre de réseau IoT et BSNs (Body Sensor Networks). Une architecture est développée basée sur l'énergie et le trafic dans le but de construire un réseau tolérant aux pannes et d'améliorer les performances de déchargement et d'équilibrage de charge. Les auteurs ont montré que l'architecture proposée offre une solution éco énergétique pour la découverte des dispositifs tout en minimisant l'itinéraire entre les appareils terminaux, ainsi que l'équilibrage de charge dans le réseau.

2.7.5.3 Approche basée sur l'infrastructure du réseau

Dans [92], l'approche proposée se base sur le déploiement d'un ensemble de nœuds (fog nodes) et une ou plusieurs passerelles (Fog Smart Gateway FSG) qui ont des capacités de communication et de traitement plus intelligente de données issues du trafic IoT, et elles répondent aux exigences des applications de niveau supérieur et aux contrainte des nœuds sous-jacents. Les auteurs ont utilisé cinq méthodes d'optimisation (Random, k-means, k-median, Recuit Simulé, et Greedy) afin d'optimiser le nombre de nœuds (fog nodes) déployés, ensuite une étude comparative a été faite, les résultats de l'étude révèle que la méthode Recuit Simulé offre des meilleures performances en terme de latence.

Dans [6], les auteurs ont proposé une approche zeroconf (zero-configuration) pour la découverte de services qui supporte les réseaux IoT à faible consommation et avec perte (Lowpower and Lossy Networks LLN), LLN est un type spécial de réseau sans fil dans lequel les nœuds sont largement limités en ressources. En effet, l'approche se base sur la diffusion locale qui utilise le filtre Bloom pour détecter les paquets reçus en double et évite de créer des boucles dans la topologie.

Discussion

Les approches de découverte de services basées sur la qualité de service améliorent considérablement la qualité et la fiabilité de la prestation des services requis. Ainsi, pour répondre à certaines exigences IoT (des objets limités en terme d'énergie, de calcul, de stockage, et communication).

Le tableau 2.1 synthétise les approches présentées dans la section 2.7 en les classant selon les critères suivants : l'architecture, avec ou sans annuaire, description de service, méthode de recherche, évolutivité, mode push ou pull, et la portée de découverte.
Nous constatons que :

1. La plupart des travaux présentés utilisent l'architecture distribuée (plusieurs annuaires) pour

la découverte de services. Cette architecture de SD offre principalement l'avantage de dépasser le problème de *Un point de défaillance unique*, permet d'éviter les goulots d'étranglements comme lors de l'utilisation d'une architecture centralisée avec un seul annuaire, et offre une meilleure scalabilité. En contrepartie, si le nombre de ces annuaires devient trop important, la découverte de services devient une tâche complexe et son coût peut devenir pesant.

2. L'utilisation des approches bio-inspirées sont rarement utilisées pour la résolution du problème de découverte de services dans IoT. Les approches bio-inspirées ont rencontré un vif sucées grâce à leur simplicité, leur adaptabilité,leur flexibilité et leur capacité de trouver des solutions presque optimales. Ces méthodes sont très efficaces et largement utilisées dans différents domaines, et parmi ces approches, il y a des méthodes ne nécessitant aucun annuaire (décentralisées).

Dans notre travail, nous avons pris en compte ces aspects, en développant des approches bio-inspirées pour résoudre le problème de découverte de services , Nous proposons donc, à cette fin, les algorithmes suivants : SD-ACA et SD-GWO.

Catégorie	Article	Architecture	Avec annuaire	Description de service	Méthode de recherche	Evolutivité	Push ou Pull	Portée de découverte
sémantique	[48]	Hiérarchique	oui	Sémantique	Système recommandé	oui	Push	Remote
	[68]	Centralisée	oui	Sémantique	similrité	oui	Push	Remote
	[150]	hiérarchique	oui	Sémantique	Clustring	oui	Push	Remote
Bio-inspiré	[15]	Centralisée	oui	XML	ND	oui	Push	Remote
	[119]	Décentralisée	non	Syntaxique	APF	oui	Pull	Remote
Protocole	[54]	Centralisée	oui	WSDL	RSDPP	oui	Pull	Remote
	[58]	Centralisée	oui	WSDL	DPWS	oui	Push	Local
	[91]	Distribuée	oui	Syntaxique	DNS-SD/mDNS	non	Pull	Local
	[130]	Distribuée	oui	Syntaxique	DNS-SD/mDNS	non	Pull	Local
	[75]	Distribuée	oui	Sémantique	DNS-SD/mDNS et MQTT	oui	Pull	Local
	[116]	Distribuée	oui	Syntaxique	MQTT-RD	non	Pull	Local/remote
	[139]	Centralisée	oui	Syntaxique	XMPP	non	Pull	Local
	[74]	Centralisée	oui	Syntaxique	ARML	non	Pull	Local
	[135]	Centralisée	oui	Sémantique	SD-CoAP	non	Pull	remote
	[133]	Distribuée	–	Syntaxique	ND	oui	Pull	local/remote
	[45]	Centralisée/Distribuée	oui	Sémantique	SD-CoAP	oui	Pull	remote
Contexte	[146]	Hiérarchique	oui	Sémantique	A2SDA/PASDA	oui	Pull/Push	Remote
	[55]	Centralisée	oui	Sémantique	Global directory	non	Pull	Remote
	[28]	Distribuée	oui	Sémantique	Publish/Subscribe	oui	Pull	Remote
	[118]	Hiérarchique	non	Syntaxique	mDNS DNS/SD	non	Push	Remote
	[145]	Hiérarchique	non	Sémantique	SLSA	oui	Push	Local
	[149]	Décentralisée	non	Sémantique	OSS	oui	Push	Local
	[115]	Centralisée	–	Sémantique	ProSA	non	Pull	Local
QoS	[84]	Centralisée	oui	Syntaxique	UDDI	non	Pull	Local
	[31]	Distribuée	oui	—	Publish/Subscribe	oui	Push	Remote
	[126]	Distribuée	oui	Syntaxique	Publish/Subscribe	oui	Push	Remote
	[92]	Hiérarchique	—	Syntaxique	Coordinating DEVICE	oui	Push	Remote
	[6]	Décentralisée	non	Syntaxique	mDNS DNS/SD	non	Push	Remote

Tableau 2.1 – Comparaison entre les différentes techniques de découverte de services dans IoT

2.8 Conclusion

Dans ce chapitre, nous avons présenté le service et les concepts clé de la découverte de services ainsi que leur architecture, Ensuite, nous avons fait une classification des différentes approches existantes relatives à la découverte de services dans l'IoT. D'après notre étude, nous avons constaté que les approches bio-inspirées ont rencontré un vif succès grâce à leur simplicité, leur adaptabilité, leur flexibilité et leur capacité de trouver des solutions presque optimales. Ces méthodes sont très efficaces et largement utilisées.
Nous avons constaté que les métaheuristiques sont peu exploitées dans la résolution du problème de découverte de services. Dans le chapitre suivant, un intérêt particulier sera porté à la méta heuristique ACA.

Partie II

Contributions

Chapitre 3

Découverte et localisation de services basée sur l'algorithme ACA

Sommaire

3.1 Introduction

Dans ce chapitre nous décrivons notre approche proposée pour résoudre le problème de découverte de services dans l'IoT. Dans la première partie, nous présentons d'abord nos motivations, nous allons présenter en détail notre première contribution, qui consiste à l'adaptation de l'algorithme ACA. Dans la deuxième partie, nous présentons les corpus de données et scénarios utilisés pour évaluer les performance de l'approche proposée, et nous montrons les différentes expérimentations menées tout en discutant les résultats obtenus. Enfin, nous terminons le chapitre par une conclusion qui résume l'essentiel de notre contribution.

3.2 Motivations

Ces dernières années, la plupart des métaheuristiques présentées dans la littérature pour résoudre des problèmes combinatoires sont inspirées des phénomènes naturels comme l'intelligence des animaux ou l'intelligence collective observée notamment chez les insectes sociaux et les animaux. Comme nous l'avons vu précédemment au chapitre 1, les algorithmes à population, basés sur l'intelligence en essaim, ou les algorithmes évolutionnaires, ont provoqué un fort intérêt de recherche pour la résolution de problèmes d'optimisation. Certains sont basés sur la sélection des meilleurs individus, tandis que d'autres s'intéressent aux comportements d'une population peu intelligente mais extrêmement communicante, qui va échanger de l'information et collaborer. Ces comportements ont été décrits à travers les métaheuristiques, d'essaim particulaire, de colonie d'abeilles artificielles, ou de colonie de fourmis. Plus précisément, nous avons vu dans la section 2.7.2, qu'ils ont été aussi appliqués avec efficacité dans le domaine de la découverte de services. Cependant, pour autant que nous sachions, ACA n'a pas été très exploité dans le domaine du problème de découverte de services dans l'IoT, alors que, certains travaux récents démontrent sa performance comme celui de [72] ou encore [143]. ACA a un meilleur contrôle d'équilibre de la stratégie de recherche intensive locale (intensification)et une exploration (diversification) plus efficace de l'ensemble de l'espace de recherche, et cette recherche est décentralisée et guidée par la fonction heuristique et la quantité de phéromones déposés. Aussi, le nombre réduit de paramètres fait d'ACA un algorithme moins complexe. Nous n'avons recensé que quelques papiers [107] et [61] appliquant ACA pour la découverte de services, cependant, les auteurs n'ont cependant mené aucune expérimentation pour ajuster les paramètres d'ACA.

Motivés par ces observations, nous proposons [13], une adaptation de l'algorithme ACA pour la résolution du problème de découverte de services dans l'IoT.

3.3 Métaheuristique ACA (Ant Colony Algorithm)

La métaheuristique d'optimisation par colonies de fourmis a été initialement introduite par Dorigo, Maniezzo et Colorni [36], [37] et a été inspirée par les études sur le comportement des fourmis réelles effectuées par Deneubourg et al [34]. ACA est considérée comme une métaheuristique de construction de solutions qui s'inspire du comportement des fourmis afin d'optimiser la longueur de leur chemin entre la colonie et la source de nourriture (figure 3.1). A l'origine, l'optimisation par colonie de fourmis a été conçue pour résoudre le problème du voyageur de commerce "Travelling Salesman Problem (TSP)" en proposant le premier algorithme ACA : Ant System (AS) [35].

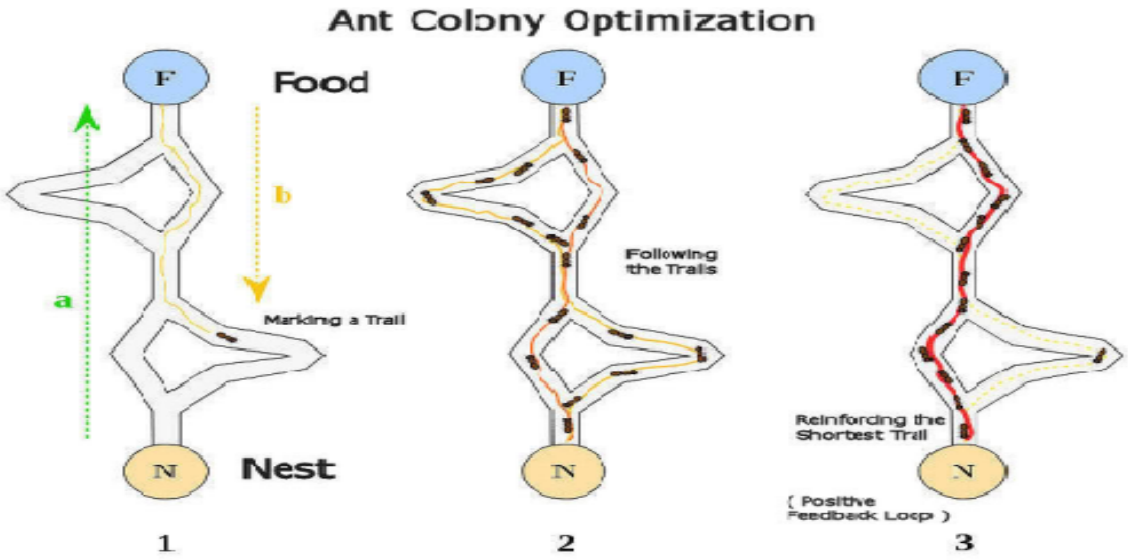


FIGURE 3.1 – Optimisation du chemin par les fourmis, au cours des itérations

Algorithm 4 Algorithme ACA

Entrée(s) Une population initiale de solutions
Sortie(s) la meilleure solution
  Initialiser les paramètres
  Initialiser matrice des phéromones
  tant que critères d'arrêt non atteints **faire**
    Construire nouvelle solution
    Evaluer la solution
    Mettre à jours la matrice des phéromones
  fin tant que

Par la suite, les ACA sont devenus rapidement un domaine de recherche mature qui s'est développé au travers de modèles de plus en plus sophistiqués pour résoudre un grand nombre de problèmes d'optimisation combinatoire. Elle a été appliquée à plusieurs problèmes combinatoires

[38] comme l'affectation quadratique [95], le routage des véhicules [24], le problème de satisfaction de contraintes [129], sac à dos multidimensionnel [4] les problèmes d'ordonnancement [20], les problèmes de planification de projet [94]. Le principe de l'algorithme ACA est résumé dans l'algorithme 4.

3.4 Random Walk

Random Walk (RW) ou (marche aléatoire), consiste à transmettre une requête de services d'un nœud à un de ses voisins aléatoirement choisi selon une probabilité jusqu'à la satisfaction de la requête de services, ce principe est illustré dans la figure 3.2. Cependant, trouver l'information en utilisant le Random Walk peut ne pas être évident car le choix aléatoire des nœuds voisins peut conduire à des chemins avec un nombre important de sauts, car on ne doit pas chercher l'information jusqu'à l'infini. Pour pallier cet inconvénient, la recherche de l'information doit être limitée par une valeur seuil, un seuil TTL (Time To Live) est défini. La valeur de TTL est décrémentée à chaque saut. Si TTL atteint la valeur zéro, le processus de recherche s'arrête. l'algorithme RW est très utilisé dans la découverte de services dans IoT [80] et dans les dans les réseaux paire à paire (P2P Peer to Peer) [51].

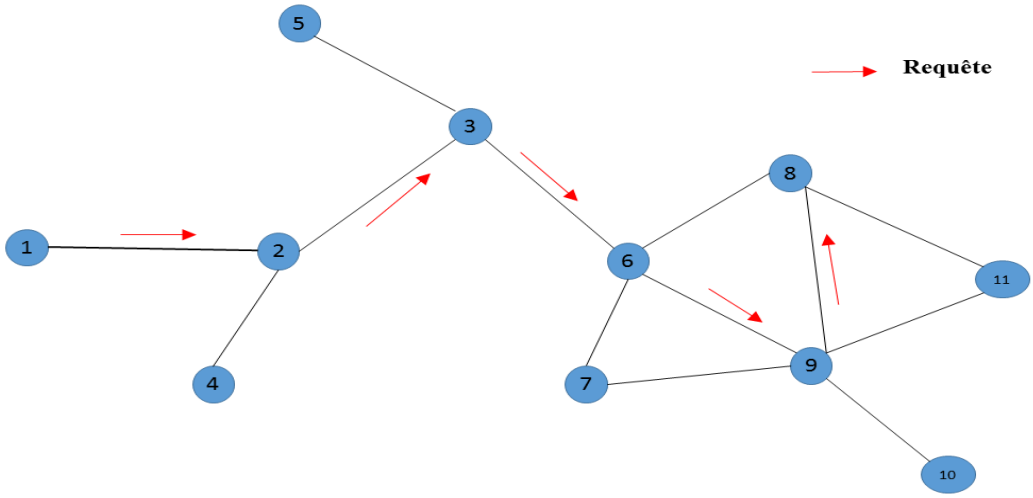


FIGURE 3.2 – Approche Random Walk

3.5 Flooding

Flooding ou la diffusion dans un environnement mobile se base sur le mécanisme d'inondation afin de garantir un taux de réception important, ce principe est illustré dans la figure 3.3. Dans la découverte de services, quand un utilisateur initie une requête de services, cette dernière est diffusée

aux nœuds voisin (ou à un sous-ensemble de ses voisins) qui eux-mêmes re-diffusent l'information en espérant atteindre les fournisseurs de services dans le réseau, ceux offrant le service demandé répondent au client en lui envoyant un message unicast. La technique de Flooding est très utilisée dans les protocoles de découverte de services comme le Gnutella [2] et AODV-SD [50], et d'autres approches [120], [27].

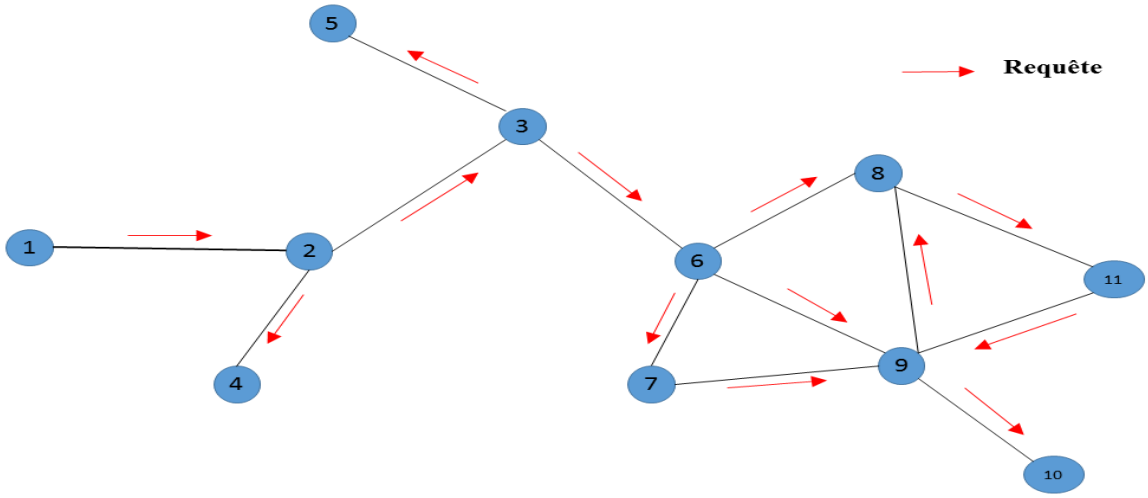


FIGURE 3.3 – Approche Flooding

3.6 Notions de base

Définition 3.6.1. (Graphe). Un graphe non-orienté $G = (V, E)$ est composé d'un ensemble V de sommets (ou nœuds) et d'un ensemble E de paires (non ordonnées) de sommets nommées arêtes (ou liens).

Définition 3.6.2. (Graphe connexe). Un graphe $G = (V, E)$ est connexe si, quels que soient les sommets u et v de V, il existe un chemin de u vers v.

Définition 3.6.3. (Graphe complet). Un graphe non orienté $G = (V, E)$ est dit complet si quelque soit la paire (u, v), il existe un arc $\in E$ reliant les deux sommets u et v. Un graphe complet de n sommets contient $\frac{n(n-1)}{2}$ arcs.

Nous adoptons les notations suivantes : N représente le nombre de sommets ($N = |V|$) et M le nombre d'arêtes ($M = |E|$), le graphe est dit d'ordre N et de taille M.

Définition 3.6.4. (Voisinage). Le voisinage d'un nœud est l'ensemble de tous ses nœuds adjacents. Autrement dit, l'ensemble des voisins d'un sommet $v \in V$, notée $N(v)$, est défini comme $N(v) = \{u \in V | (v, u) \in E\}$.

Définition 3.6.5. (Distance). La distance entre deux sommets u et v est la longueur de la plus courte chaîne entre ces sommets.

Définition 3.6.6. (Chemin). Le chemin entre deux sommets (nœuds) est une séquence de liens consécutifs dont ils sont les extrémités ; la longueur de ce chemin sera le nombre de liens qu'il comporte ; la distance entre deux sommets sera le minimum des longueurs de chemins allant de l'un à l'autre.

Définition 3.6.7. (Matrice d'adjacence). Une matrice d'adjacence A d'un graphe G d'ordre N est une représentation matricielle exactement équivalente au graphe. Cette matrice $(N \times N)$ est binaire, $A_{i,j} = 1$ s'il existe un lien entre les nœuds x_i et x_j, $A_{i,j} = 0$ dans le cas contraire.

3.7 Adaptation de la métaheuristique ACA au problème de découverte de services SD-ACA

Dans cette partie, nous présentons une adaptation de l'algorithme ACA nommé SD-ACA (la découverte de services basée sur ACA) pour résoudre le problème de découverte de services dans l'IoT en transformant le problème en un problème d'optimisation avec une fonction objectif. Nous allons optimiser le nombre de sauts pour la découverte d'une requête de services en utilisant l'algorithme de recherche ACA. Optimisation par colonie de fourmis est considérée comme une métaheuristique de construction de solutions qui s'inspire du comportement des fourmis afin d'optimiser la longueur de leur chemin entre la colonie et la source de nourriture.

La version originale est conçue pour résoudre le problème du voyageur de commerce [35]. Lorsque une fourmi commence par un déplacement aléatoire dans un graphe connexe, à partir d'une ville A à la ville B , en passant par l'arc qui connecte ces deux villes, elle y dépose une quantité de phéromone. Les autres fourmis qui suivent cette fourmi auront le choix de considérer le phéromone déposé et passer sur cet arc ou de prendre un arc aléatoirement selon les paramètres et la fonction heuristique, qui peut être l'inverse de la distance entre les villes.
Dans notre adaptation , nous avons défini la fonction heuristique comme suit :

Fonction heuristique : est le nombre de services offerts par un voisin.

Dans ce qui suit, nous montrons comment le réseau assure la découverte de services dans une architecture purement décentralisée.

3.7.1 Formalisation

Pour formaliser le problème de découverte de services dans l'IoT, nous devons concevoir un graphe complet et une fonction objectif qui calcule le nombre de Sauts utilisés pour la découverte de services. Le réseau IoT est modélisé par un graphe $G = (W, L)$.
$W = O \bigcup S$,$O = O_1, O_2, ..., O_M$ est l'ensemble d'objets dans IoT.
$S = S1, S2, ...S_{Nb\_S}$ est l'ensemble de services offerts par les objets.
$L = \{L_{i,j} / (O_i, O_j) \in O\}$ est l'ensemble des liens entre les objets.
Il est trivial que le principal objectif dans la recherche d'un service dans un réseau IoT, après la localisation bien évidemment, est le fait de pouvoir minimiser et optimiser le nombre de sauts effectués, et pour cela nous avons défini une fonction objectif qui calcule le nombre de sauts, elle est défini comme suit :

$$f(L^s) = NB\_SAUTS(O_i, O_j) \tag{3.1}$$

L^s C'est le chemin entre O_i et O_j (la solution), comprend une liste des objets.
O_j C'est le dernier objet dans L^s.
O_i C'est l'objet qui a initié la requête de services.

3.7.1.1 Codification de la solution

Une solution pour le problème de découverte de services est codifiée sous forme d'un vecteur d'entiers (chemin). Chaque élément du vecteur correspond à un objet choisi parmi les objets candidats(voisins).
Dans la figure 3.4, le vecteur d'entiers représente la codification d'une solution possible avec trois sauts pour une requête $REQ = \{S_2, S_7\}$, initié par l'objet O_1, où chaque élément du vecteur indique l'identifiant d'un objet. Le graphe est définit par une matrice d'adjacence définit comme suit :

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

.

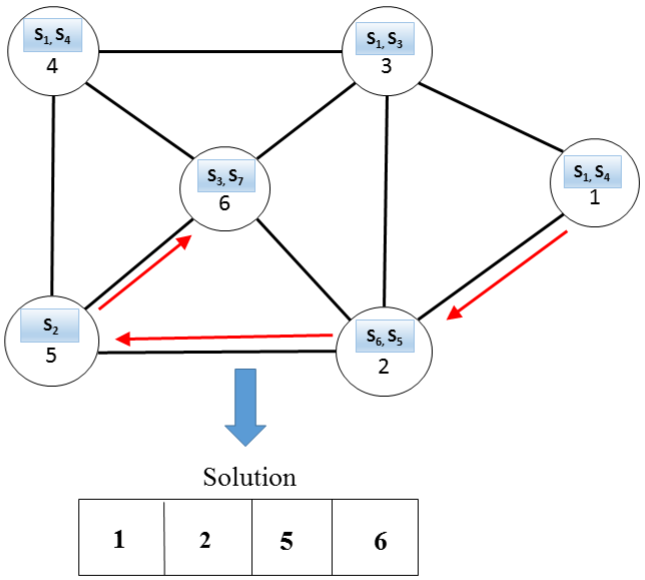


FIGURE 3.4 – Codage d'une solution

3.7.2 Algorithme de découverte de services basée sur l'algorithme ACA (SD-ACA)

Les différentes notations utilisées sont définies dans le tableau 3.1

Notation	Description
L^s	Le plus court chemin
S_i	La liste de services à rechercher
S_r	La liste des services non trouvés
V_h	L'ensemble des voisins de l'objet h
p	Le voisin choisi à mettre dans la liste solution
ns	Le nombre de services non trouvés

Tableau 3.1 – Récapitulatif des notations utilisées dans l'approche SD-ACA

Le pseudo code de l'algorithme SD-ACA est donné ci-dessous :

Algorithm 5 SD-ACA ACA-based service discovery : la découverte de services basée sur l'algorithme ACA

Entrée(s) Req (S_i, O_i) requête de l'utilisateur, S_i ensemble de services demandés par O_i
Sortie(s) L^s Le plus court chemin

1: **pour tout** $e(i,j) \in E$, $(i,j) \in O$, $i \leqslant M$, $j \leqslant M$ **faire**
2: $\tau_{ij}(0) = c$
3: **fin pour**
4: **pour** t =1...N **faire**
5: **pour** k=1..Nb_Ants **faire**
6: $S_r = S_i$
7: L^k=i
8: $h = dernier(L^k)$ Placer le dernier objet de la liste dans h
9: V_h Ensemble des voisins de l'objet h $e(h,j) \in E$,$h, j \in O$, $j \leqslant M$
10: Calculer $P_{ij}^k(t)$ $j \in V_h$ avec la formule 3.2
11: $L^k = L^k \bigcup \{p\}$, $P \in V_h$
12: $ns = length(S_r)$
13: **pour** t=1...ns **faire**
14: **si** service $S_r(t)$ offert par P **alors**
15: $S_r = S_r - S_r(t)$
16: **fin si**
17: **fin pour**
18: **si** $S_r \neq \varnothing$ **alors**
19: Aller à la ligne 8
20: **sinon**
21: Calculer la taille de L^K
22: **si** length $(L^k) < L^s$ **alors**
23: $L^s = L^k$
24: **fin si**
25: Calculer $\Delta\tau_{ij}^k$ avec la formule 3.4
26: Mettre à jour $\tau_{ij}(t)$ for each $E(i,j) \in L^k$ avec la formule 3.3
27: **fin si**
28: **fin pour**
29: Mettre à jour $\tau_{ij}(t)$ par la formule 3.5
30: **fin pour**

3.7.3 Probabilité de transition

Une fourmi k placée sur l'objet O_i à l'instant t va choisir l'objet O_j de destination en fonction de la visibilité η_{ij} de cet objet et de la quantité de phéromones $\tau_{ij}(t)$ déposée sur le lien reliant ces deux objets. Ce choix sera réalisé de manière aléatoire, avec une probabilité de choisir l'objet j donnée par la formule suivante :

$$P_{ij}^k(t) = \begin{cases} \frac{[\tau_{ij}(t)]^\alpha \times [\eta_{ij}]^\beta}{\sum_{y \in N_i^k} [\tau_{iy}(t)]^\alpha \times [\eta_{iy}]^\beta} & Sij \in N_i^k \\ 0 & Sinon \end{cases} \tag{3.2}$$

où on a définit $\tau_{ij}(t)$ comme la quantité de phéromones déposée sur le lien $E(i,j)$ reliant les deux objets O_i et O_j à l'itération t.
N_i^k comme étant l'ensemble des objets que la fourmi k, placée sur l'objet O_i , n'a pas encore visité à l'instant t dans le cycle courant.
α et β sont deux paramètres qui contrôlent l'importance relative entre phéromones et visibilité(information heuristique). Ainsi si α est égal à 0, le choix se fera uniquement en fonction de la visibilité (si β est différent de zero. A l'inverse, si $\beta = 0$ alors seules les traces de phéromone sont prises en compte pour définir les probabilités de transitions.Pour éviter une sélection trop rapide d'un trajet, un compromis entre ces deux paramètres est nécessaire, en jouant sur les comportements de diversification et d'intensification.
η_{iy} est le facteur heuristique associé à l'objet O_i, on a définit comme le nombre de services offerts par l'objet voisin y.

3.7.4 Mise à jour des phéromones

Une fois que chaque fourmi a construit un chemin(combinaison d'objets), les valeurs de phéromone sont mises à jour durant l'exécution d'un algorithme de SD-ACA pour orienter la recherche vers de bonne solutions. Cela se fait en deux étapes complémentaires : dépôt de phèromone et évaporation de phéromone.

3.7.4.1 Dépôt de phéromone

il consiste à augmenter les valeurs de phéromone des composants de solution qui apparaissent dans un ensemble de bonnes solutions construites dans cette itération ou dans les itérations précédentes. La forme générale de cette opération s'exprime par la formule suivante :

$$\tau_{ij}(t+1) = \sum_{k=1}^{Nb\_Ants} \Delta\tau_{ij}^k(t) \tag{3.3}$$

Nb_Ants c'est le nombre de fourmis. $\Delta\tau_{ij}^k$ c'est la quantité de phéromone déposé sur l'ensemble du chemin. Cette quantité dépend de la qualité de la solution trouvée, et elle est définie par

$$\Delta\tau_{ij}^k = \begin{cases} \frac{\varrho}{|L^k|} & Si\ (i,j) \in L^k \\ 0 & Sinon \end{cases} \tag{3.4}$$

L^k est la liste des objets déjà visitées, et $|L^k|$ est la longueur du tour généré par la fourmi k afin de découvrir la requête de services

ϱ est une constante.

On voit bien ici que les quantités de phéromones sont régulés en fonction de la qualité de la solution obtenue car plus L^k est faible plus l'arc sera mis à jour en phéromones.

3.7.4.2 Evaporation de phéromone

Enfin, il est nécessaire d'introduire un processus d'évaporation des phéromones. En effet, pour éviter de se faire piéger dans des solutions sous-optimales (minimums locaux), il est nécessaire qu'une fourmi "oublie" les mauvaises solutions. La réduction des valeurs de phéromones peut être définie par

$$\tau_{ij}(t+1) = (1-\rho) \times \tau_{ij} + \sum_{k=1}^{Nb\_Ants} \Delta\tau_{ij}^{k} \tag{3.5}$$

$\rho \in [0\,;\,1]$ est un coefficient définissant la vitesse d'évaporation des phéromones sur les liens entre l'instant t et l'instant $t+1$.

3.7.5 Selection de meilleur chemin

pour chaque itération t, une meilleur solution construite est sélectionnée selon la formule

$$L^{s} = Min(hops(L^{k}))/K = 1..Nb\_Ants \tag{3.6}$$

$hops(L^{k}$ retourne le nombre de sauts utilisés pour la découverte de la requête de services.

3.7.6 Fonctionnement de l'algorithme SD-ACA

Nous allons maintenant préciser les différentes phases de l'algorithme proposé , on décrit d'abord les entrées et sorties. La première phase constitue une phase d'initialisation, suivis d'une phase de recherche des solutions, ensuite une phase de sélection de la meilleure solution, enfin la phase finale qui retourne la solution pour le problème.

3.7.6.1 Entrées et sorties

En entrée, l'algorithme démarre avec M objets qui sont repartis aléatoirement, Nb_Ants fourmis, et NB_S services, chaque objet offre de 1 à 3 services. Une requête de recherche $REQ(S_i, O_i)$ initiée de par l'objet O_i pour découvrir l'ensemble de services S_i
En sortie, l'algorithme fournit une solution (un chemin) proche de l'optimale.

3.7.6.2 Initialisation de l'algorithme

Les paramètres s'agencent de la manière suivante au début de l'algorithme :

1. les NB_ants fourmis sont placées dans l'objet O_i (O_i c'est objet qui a initié la requête de services).
2. Pour chaque fourmi k , une liste L^{k} qui modélise sa mémoire à l'itération t, et elle contient au départ l'objet qui a initié la requête de services.

3. Les pistes de phéromones sont initialisées comme suit : $\tau_{ij}(0) = c$, où c est une petite constante positive, qui ne peut être nulle.

3.7.6.3 Phase de recherche

Dans la phase de recherche

1. Les fourmis explorent la zone de recherche, et si un objet offre l'un des services demandés, la fourmi met à jour L^k.
2. A la fin de chaque tour (après la découverte de tous les services REQ), les valeurs $\Delta_{ij}(t)$ sont calculées conformément à la formule 3.4.
3. Les variables de phéromone $\tau ij(t)$ sont mises à jour suivant la formule 3.5. En d'autres termes, les fourmis recommencent un nouveau tour, toujours au départ d'objet O_i laquelle elles avaient été placées au début de l'algorithme.

3.7.6.4 Phase de sélection

Á la fin de chaque itération k, une meilleure solution est sélectionnée en utilisant la fonction objectif donnée dans la formule 3.7.

$$L^s = Min(hops(L^k))/K = 1..Nb\_Ants \tag{3.7}$$

$hops(L^k$ retourne le nombre de sauts utilisés pour la découverte de la requête de services.

3.7.6.5 Phase finale

On arrête l'algorithme après un nombre de cycles égal à une constante N, c'est à partir de cet instant, toutes les fourmis font le même tour, l'algorithme s'interrompt, on est dans une situation de stagnation où le programme arrête de chercher des alternatives. L'algorithme donne en retour le meilleur tour mémorisé.

3.8 Etude de la complexité de l'algorithme SD-ACA

Dans cette partie, nous présentons la complexité temporelle de l'algorithme proposé SD-ACA. Cette complexité dépend de quelques paramètres : NB_ants (le nombre du fourmis), M(le nombre d'objets dans le réseau), NB_V (le nombre de voisins pour un objet). Le tableau 3.2 présente la complexité temporelle dans les différentes phases de l'algorithme.

Phase de l'algorithme	Complexité	Détails
Phase d'initialisation	$O(M^2 + NB\_ants\ )$	Initialiser la liste de chaque fourmi est de l'ordre NB_ants et initialiser aussi les pistes de phéromones dans chaque lien du réseau
Phase de recherche	$O(NB\_ants.\|L\|) + O(\|L\| + NB\_ants.\|L\|) = O(NB\_ants.\|L\|)$	L est l'ensemble de chemin $\|L\| = M.NB_V/2$, $O(\|L\|) = M.NB\_V$
Phase de sélection	$O(NB\_ants.M)$	comparer les NB_ants solutions, au maximum il y a M objets dans la solution

Tableau 3.2 – Complexité temporelle de l'algorithme SD-ACA

La complexité globale est obtenu en additionnant la complexité de la phase d'initialisation, au produit de nombre total d'itérations (N) par la complexité de la phase de recherche et la complexité de la phase de sélection. la complexité générale de l'algorithme est :
$O(M^2 + NB\_ants + N.NB\_ants.|L| + N.NB\_ants.M)$.
Comme $O(|L|) = M.NB\_V$, la complexité devient donc :
$O(M^2 + NB\_ants + N.NB\_ants.M.NB\_V + N.NB\_ants.M)$.
$maximum(NB\_ants; N.NB\_ants.M.NB\_V; N.NB\_ants.M) = N.NB\_ants.M.NB\_V$
donc :$O(M^2 + N.NB\_ants.M.NB\_V)$
La complexité globale est du l'ordre $O(N.NB\_ants.M.NB\_V)$

3.9 Évaluation des performances de l'algorithme SD-ACA

Dans cette partie, nous décrivons les résultats de l'évaluation, en simulation de l'algorithme proposé. Pour ce faire, nous présentons en premier lieu l'environnement de simulation, nous décrivons ensuite le corpus de données et scénarios d'évaluation suivi des métriques à considérer pour l'évaluation et enfin, nous montrons à travers une série de résultats de simulation, l'apport de l'algorithme proposé en terme du nombre de sauts et le taux de succès

3.9.1 Environnement de simulation

Les expérimentation ont été menées sur une machine dotée d'un processeur core 2 Duo avec une fréquence de 2.4GHz et une RAM de 2GB, exécutant Matlab R2016b sous système Windows 7 entreprise 64 bits.
Les différents paramètres de notre simulation sont décrits dans le tableau 3.3.

Portée de transmission R	25m
Nombre maximum d'itérations N	100
Nombre de fourmis Nb_Ants	100
Nombre de services offerts par objet	1-3
la constante ϱ	70
ρ le degré d'évaporation de phéromone	0.9

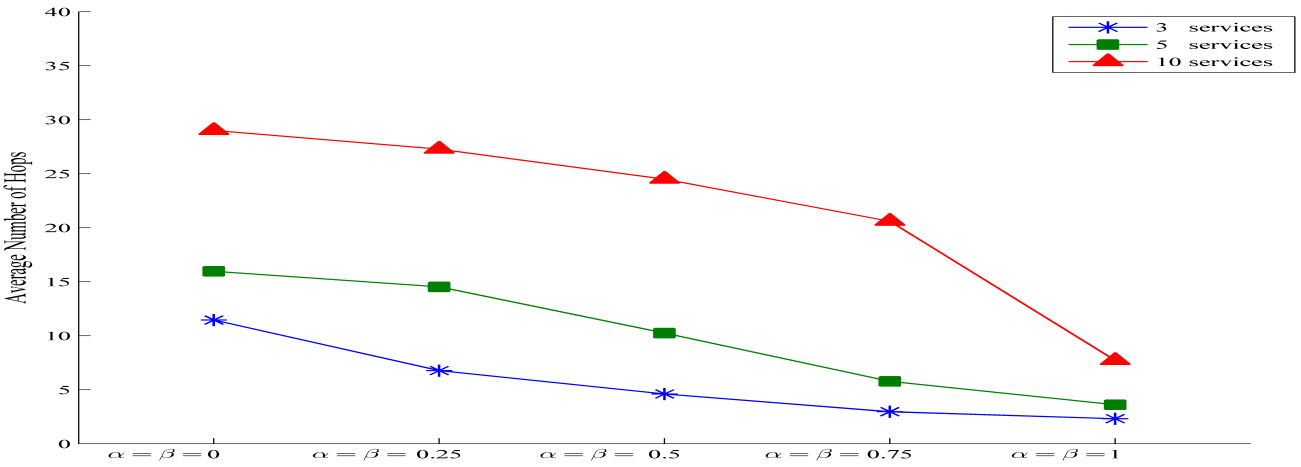
Tableau 3.3 – Paramètres de simulation 1

3.9.2 Corpus de données et scénarios d'évaluation

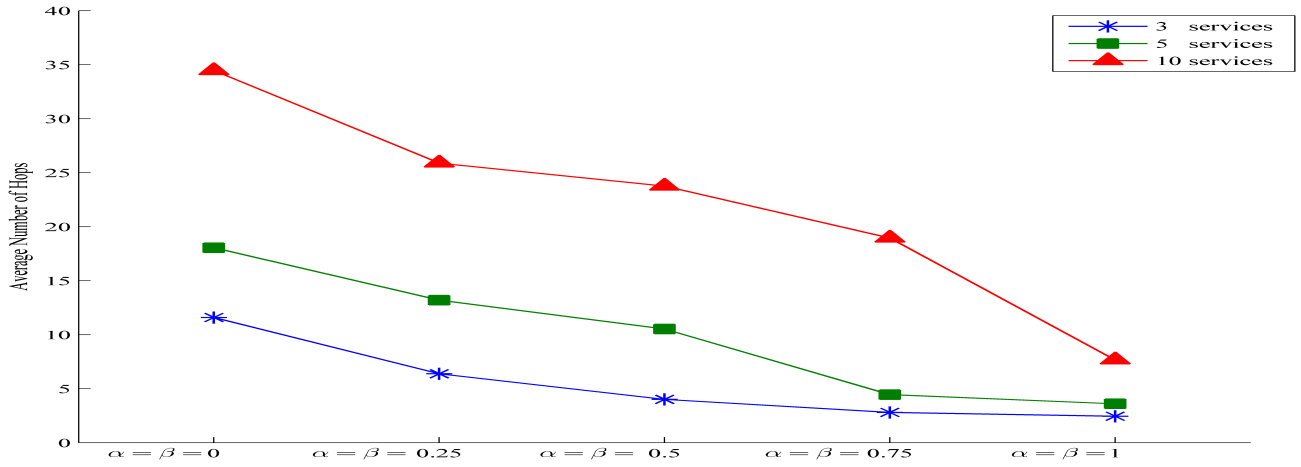
Pour évaluer les performances de notre proposition, nous avons utilisé une collection de données générées aléatoirement appliquées sur diffèrent scénarios de découverte de services. Ces scénarios sont générés en faisant varier les valeurs des paramètres d'ACA , le nombre d'objets, et le nombre d'objets défaillants.

3.9.3 Choix des paramètres

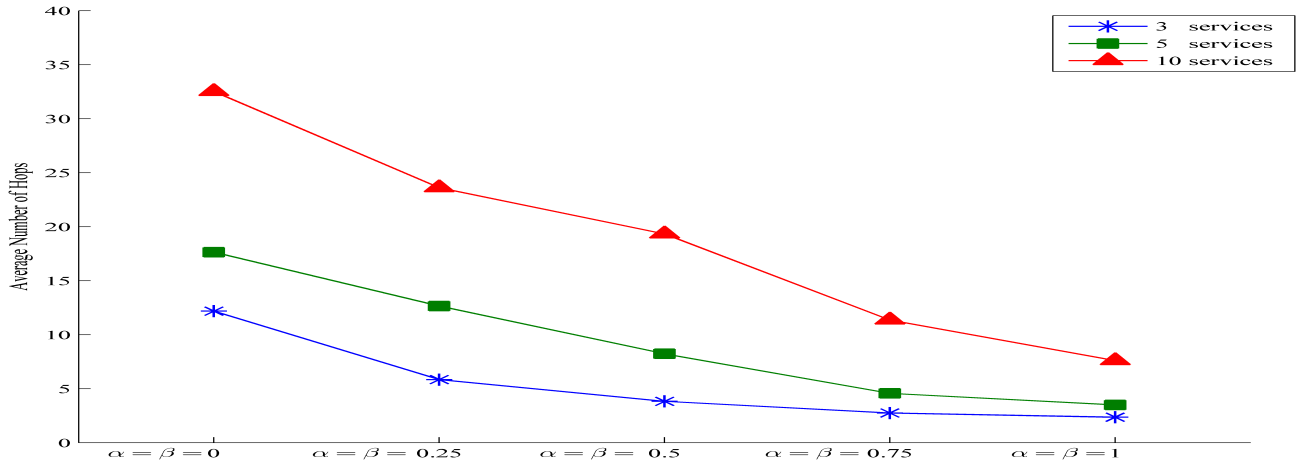
Une grande faiblesse de l'algorithme ACA est le nombre élevé de paramètres en jeu. De ce fait, il est très difficile de les fixer subtilement. En l'absence de modèle mathématique, nous nous sommes basés sur des résultats expérimentaux pour fixer les paramètres de l'algorithme proposé SD-ACA. La synthèse de ces résultats est présentée dans la section suivante.
Dans cette simulation nous varions les valeurs α et β sur l'intervalle [0,1] dans différentes tailles du réseau (50,100,1000) objets et des requêtes contiennent (3,5,10) services à découvrir. Toutes les mesures des simulations représentent les résultats moyens pris de 100 exécutions successives.



(a) Avec 50 objects



(b) Avec 100 objects



(c) Avec 1000 objects

FIGURE 3.5 – Nombre moyen de sauts effectués dans la découverte de services pour les différentes de valeurs de α et β (avec 1000 objects)

La figure 3.5 montre que la recherche est plus efficace avec un nombre minimum de sauts quand les fourmis communiquent entre elles, c'est-à-dire quand les quantités de phéromones ont un grand impact sur la politique de décision de la fourmi. Ceci correspond à prendre $\alpha = 1$ et $\beta = 1$. dans le cas $\alpha = 0$ et $\beta = 0$, l'approche se comporte comme une recherche aléatoire avec un nombre très important de sauts.

3.9.4 Métriques et méthodologie de simulation

Les performances de l'algorithme proposé sont comparées à celle de deux algorithmes, la diffusion (Flooding) et Random Walk. Les métriques considérées pour l'évaluation des performances sont :

Le nombre de sauts : C'est le nombre de sauts nécessaires effectués par l'algorithme de recherche pour atteindre les services demandés et le plus court chemin est celui qui compte moins de sauts.

Le taux de succès : il représente le pourcentage du nombre de requêtes découvertes sur le nombre total de requêtes.

3.9.4.1 Les résultats de simulation

Cette partie illustre une comparaison du nombre de sauts entre les trois approches. Le nombre de sauts est mesuré en fonction du nombre d'objets, nous avons effectué différents tests pour les trois approches en faisant varier le nombre d'objets (50,100,500,1000) et une requête qui contient 5 services. Toutes les mesures des simulations représentent les résultats moyens pris de 100 exécutions successives.

La Figure 3.6 montre que lorsque le nombre d'objets augmente, le nombre de sauts effectués pour la découverte des services dans les deux approches (Flooding et SD-ACA) est comparables (avec une légère infériorité pour Flooding). Cependant, dans l'approche basée sur Flooding, le réseau peut être saturé en raison des nombreuses rediffusions qui doivent être opérées par l'ensemble des objets intermédiaires qui relaient le message provenant d'un objet et sa destination. Cela peut également augmenter le taux de collision et augmente aussi la consommation d'énergie. Par contre, l'approche basée sur Random walk donne des mauvais résultats car elle est purement aléatoire vu sa nature stochastique.
Nous constatons aussi que notre approche fournit des résultats proches de ceux obtenus avec l'approche Flooding (moins d'un saut), et aussi notre approche garantit le passage à l'échelle du réseau.

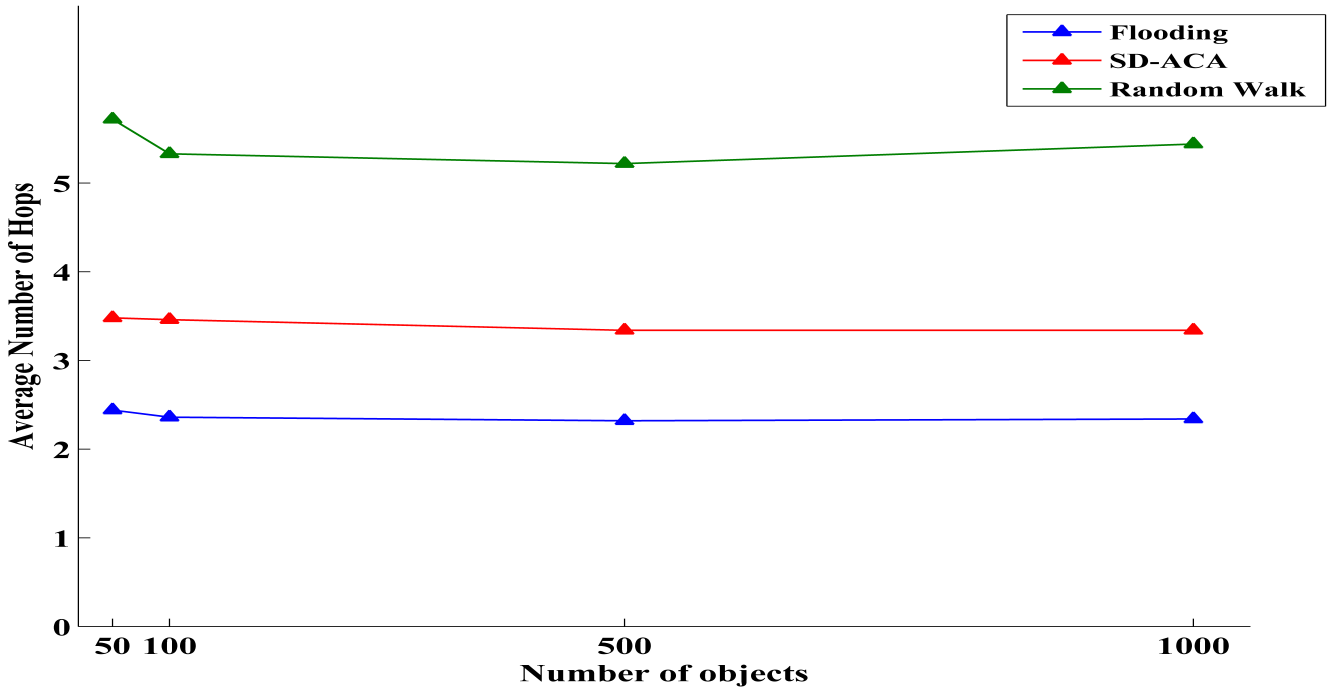
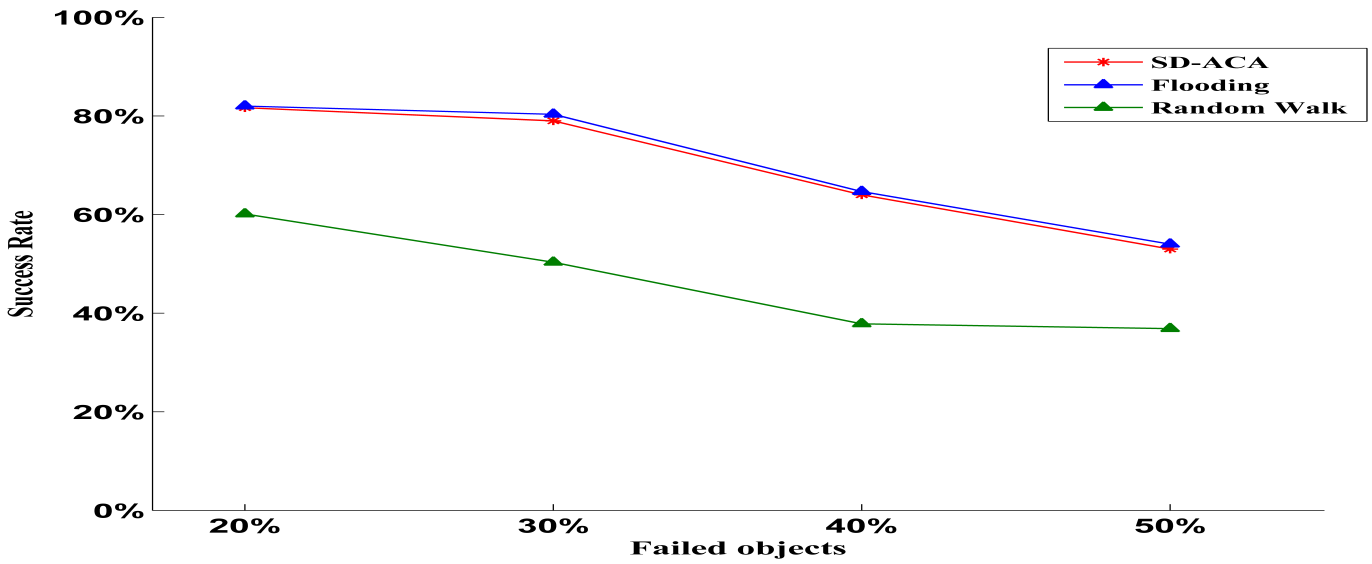


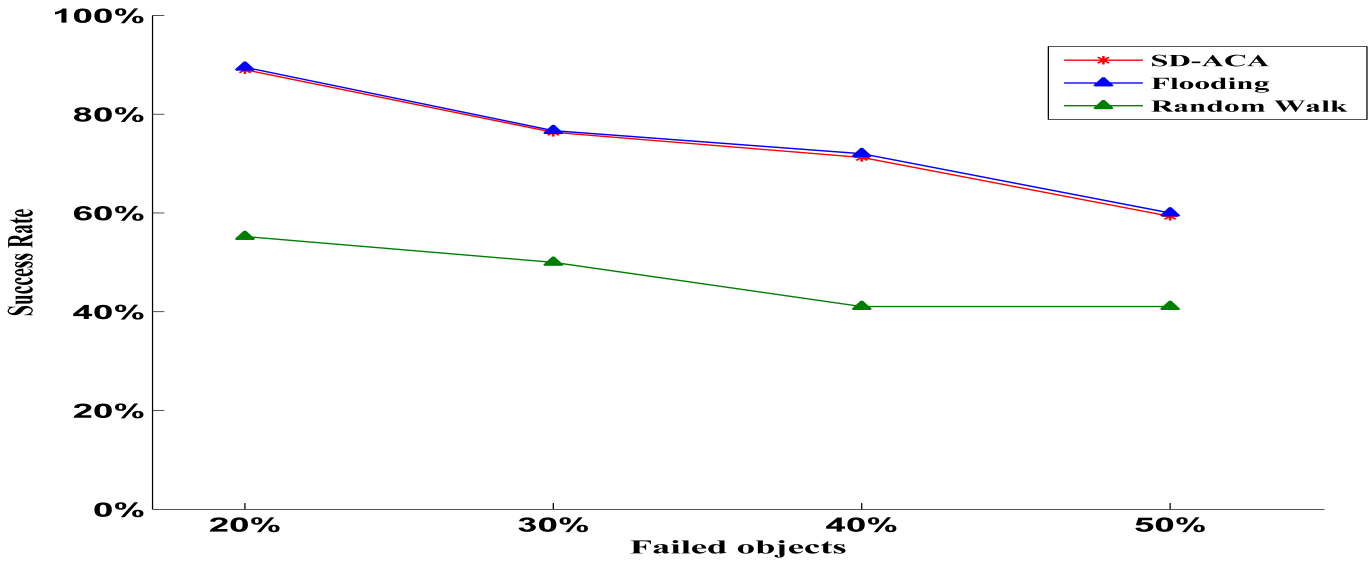
FIGURE 3.6 – Nombre moyen de sauts effectués dans découverte de services pour les différentes approches SD-ACA, Flooding, et Random Walk

Le taux de succès : Cette partie illustre une comparaison du taux de succès entre les trois approches. Le taux de succès est mesuré en fonction d'objets défaillants dans notre système, nous avons effectué différents tests pour les trois approches en faisant varier le pourcentage d'objets défaillants (20%,30%,40%,50%) et une requête qui contient cinq services et la valeur de TTL égal à 5. Toutes les mesures des simulations représentent les résultats moyens pris de 100 exécutions successives.

Les figures 3.7a et 3.7b montrent que lorsque le nombre d'objets défaillants augmente, le taux de succès diminue dans les trois approches. Nous constatons que notre approche fournit des résultats similaires à ceux obtenus avec l'approche Flooding. Par contre l'approche Random Walk donne de mauvais résultats car elle est purement aléatoire.



(a) Avec 50 objects



(b) Avec 100 objects

FIGURE 3.7 – Taux du sucés

3.9.5 Synthèse

En résumé, nos résultats de simulation ont montré que le taux de succès, le nombre de sauts requis pour la découverte de services en utilisant SD-ACA sont très proches à ceux de l'algorithme Flodding, L'avantage clé de notre solution est qu'elle ne nécessite pas une table de routage mais seulement le maintien d'une liste des voisins et aussi notre proposition se base sur une probabilité pour choisir le prochain voisin à suivre, ce qui réduit le nombre de messages dans le système, alors que l'approche Flooding se base sur une stratégie d'inondation, et cela n'est pas adapté aux

objets avec ressources limités. En plus, un modèle décentralisé permet un passage à des échelles importantes.

3.10 Conclusion

ACA est une métaheuristique qui requiert de plus en plus l'attention de la communauté scientifique. Cette métaheuristique a prouvé ses performances pour plusieurs problèmes d'optimisation combinatoire. L'utilisation des traces de phéromones permet d'exploiter l'expérience de recherche acquise par les fourmis et ainsi renforcer l'apprentissage pour la construction de solutions dans les itérations futures de l'algorithme. En même temps, l'information heuristique peut guider les fourmis vers les zones prometteuses de l'espace de recherche.
Dans ce chapitre nous avons présenté notre contribution, où nous avons adapté la métaheuristique ACA à la découverte de services, et nous avons opté pour une approche décentralisée pour une meilleure mise à l'échelle des services et des objets. Les résultats de simulation offre de meilleurs performances en terme de nombre de sauts et le taux de succès.
Le chapitre suivant présentera une deuxième contribution dans laquelle on a adopté une variante discrète de la métaheuristique GWO pour la résolution du problème de découverte de services.

Chapitre 4

Découverte et localisation de services basée sur l'algorithme GWO

Sommaire

4.1 Introduction

Dans ce chapitre, nous allons présenter une nouvelle approche de découverte de services basée sur le même principe, qui consiste en la transformation du problème de découverte de services en un problème d'optimisation combinatoire. Cette approche est basée sur une méthode d'optimisation nommée (GWO : Gray Wolf Optimizer) en mode discret heuristique récemment développée. La première partie de ce chapitre est dédiée à la formulation du problème de découverte de services et la définition de la fonction objectif utilisée.
Dans la deuxième partie de ce chapitre, après avoir présenté nos motivation, nous allons présenter en détail notre contribution basée sur la métaheuristique GWO(Gray Wolf Optimizer), après nous montrons les différentes expérimentations menées, et finalement nous discutons les résultats obtenus.

4.2 Motivations

Les métaheuristiques sont des algorithmes itératifs qui parcourent l'espace de recherche avec des différentes techniques, dans le but de générer des solutions. Ces algorithmes sont souvent inspirés par des systèmes physiques, ou biologiques. Ils ont prouvé leurs efficacités dans les différents domaines de l'optimisation. Plus précisément, nous avons vu dans la section 2.7.2, qu'ils ont été aussi appliqués avec efficacité dans le domaine de la découverte de services. Cependant, pour autant que nous sachions, GWO n'a pas été très exploité dans le domaine du problème de découverte de services dans IoT, alors que, certains papiers récents démontrent sa performance comme celui de Wang et al. [140]
L'algorithme GWO a suscité beaucoup notre attention en raison de sa simplicité, de sa facilité d'implémentation et le nombre réduit de paramètres de contrôle utilisés. Il peut également être généralisé à des problèmes à grande échelle. De plus, il a une capacité d'avoir un compromis entre l'exploration et l'exploitation pendant la recherche de solutions, ce qui conduit à une convergence favorable vers l'optimum global.
Motivés par ces observations, nous proposons une adaptation de l'algorithme GWO pour la résolution du problème de découverte de services dans l'IoT.

4.3 GWO : Gray Wolf Optimizer

L'algorithme des loups gris est introduit par Mirjalili et al. [100], il imite la hiérarchie de leadership des loups qui sont bien connus pour leur chasse en groupe, il a été développé à partir d'une méthode de recherche optimisée inspirée par les activités de chasse des loups gris, Le loup gris est un animal social qui vit dans un pack de 5 à 12 membres. Dans cet algorithme la population

est divisée en quatre groupes : alpha (α), bêta (β), delta (δ) et oméga (ω)(figure 4.1). Le loup alpha α est considéré comme le loup dominant de la meute et ses ordres doivent être suivis par les membres de la meute et est considéré comme le meilleur candidat. Les loups bêta β sont des loups subordonnés, qui aident le loup α dans la prise de décision et sont considérés comme le meilleurs candidats après le loup alpha. Les loups δ doivent se soumettre aux loups α et β, mais ils dominent les loups ω. Les loups Omega ω sont considérés comme le bouc émissaire de la meute, sont les individus les moins importants de la meute et sont les derniers loups autorisés à manger.

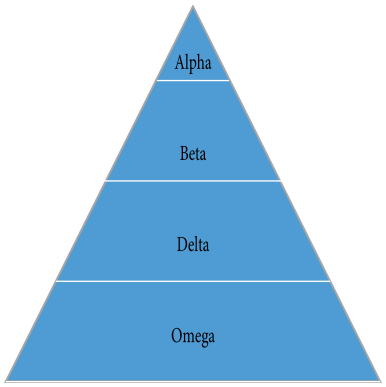


FIGURE 4.1 – Hiérarchie des loups gris

4.4 Discrétisation de l'algorithme GWO

La nature des problèmes d'optimisation peut être associée à celle de leur espace de recherche (type de solution). Il existe deux type d'espace de recherche ; le premier est l'espace continu contenant des solutions caractérisées par leurs variable réelles. Tandis que, le deuxième type est discret (combinatoire). Le problème de découverte de services est un problème discret. Malgré que l'algorithme GWO original soit continu pour résoudre les problèmes d'optimisation de nature continu, nous allons proposer une adaptation à notre problématique. Pour réaliser cette adaptation, la solution est codifiée sous forme d'un vecteur d'entiers, chaque élément du vecteur correspond à un objet choisi parmi les objets les plus proches aux loups (individus de la population). Le vecteur d'entiers est comme solution candidate pour le problème de découverte de services.
Dans la figure 4.2, le vecteur d'entiers représente la codification d'une solution possible avec deux sauts pour une requête $REQ = \{S_2, S_7\}$, initié par l'objet O_1, où chaque élément du vecteur indique l'identifiant d'un objet.

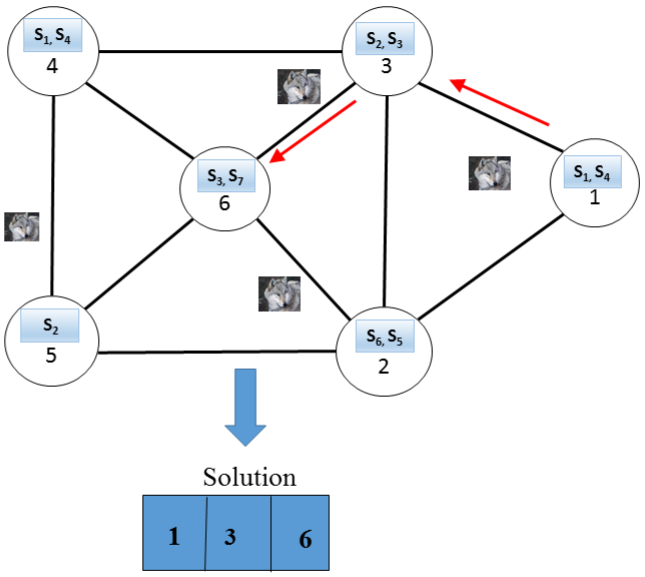


FIGURE 4.2 – Discrétisation et codage d'une solution

4.5 Formalisation

Pour formaliser le problème de découverte de services dans IoT, nous devons concevoir un graphe et une fonction objectif qui calcule le nombre de Sauts utilisés pour la découverte de services. Le réseau IoT est modélisé par un graphe $G = (W, L)$.
$W = O \bigcup S$,$O = O_1, O_2, ..., O_M$ est ensemble d'objets dans IoT. $S = S1, S2, S_{Nb\_S}$ est l'ensemble de services offerts par les objets.
$L = \{L_{i,j} / (O_i, O_j) \in O\}$ est ensemble des liens entre les objets.

Le principal objectif dans la recherche d'un service dans un réseau IoT, après la localisation bien évidement, est le fait de pouvoir minimiser et optimiser le nombre de sauts effectués. La fonction objectif est définit comme suit :

$$f(LS) = NB\_SAUTS(O_i, O_j) \tag{4.1}$$

LS c'est la solution, comprend une liste des objets.
O_j C'est le dernier objet dans LS.
O_i C'est l'objet qui a initié la requête de services.
Les différentes notations utilisées sont définies dans le tableau 4.1

Notation	Description
S_j	Les services à découvrir
X_α	Le meilleur agent de recherche
X_β	Le deuxième meilleur agent de recherche
X_δ	Le troisième meilleur agent de recherche
O_h	L'objet qui offre le service recherché
ns	Le nombre de services non trouvés

Tableau 4.1 – Récapitulatif des notations utilisées dans l'approche SD-GWO

4.5.1 Recherche de proies (Exploration)

Les loups gris recherchent principalement la proie selon les positions des loups alphas, bêtas et deltas. Ils divergent les uns des autres pour explorer la position de la proie, ensuite ils convergent pour attaquer les proies. Afin de modéliser mathématiquement la divergence, nous utilisons le paramètre A avec des valeurs aléatoires supérieures à 1 ou inférieures à 1. Si $A > 1$, les agents de recherche divergent de la proie. Afin de contrôler la valeur de ce paramètre, un autre paramètre clé nommé a qui diminue linéairement de 2 à 0. C est un autre paramètre de GWO qui favorise l'exploration. Il contient des valeurs aléatoires comprises dans l'intervalle [0, 2]. Cette contribution est forte lorsque $C < 1$, la solution gravite davantage vers les proies, favorisant l'exploration et évitant les optima locaux.

4.5.2 Encercler la proie

La première étape du processus de chasse est l'encerclement de la proie, elle correspond à l'exploration (recherche globale)et l'attaque à l'exploitation (recherche locale).
Afin de modéliser mathématiquement le comportement d'encerclement, les équations suivantes sont proposées [100] :

$$\vec{D} = |\vec{C}.\vec{X_p}(t) - \vec{X(t)}| \tag{4.2}$$

$$\vec{X(t+1)} = \vec{X_p}(t) - \vec{A}.\vec{D} \tag{4.3}$$

Où t indique l'itération courante, $\vec{X_p}$ est le vecteur de position de la proie, tandis que le vecteur $\vec{X}$ est le vecteur de position d'un loup gris.
$\vec{A}$ et $\vec{C}$ sont deux vecteurs de coefficients, qui peuvent être calculés comme suit :

$$\vec{A} = 2.\vec{a}.\vec{r_1} - \vec{a} \tag{4.4}$$

$$\vec{C} = 2.\vec{r_2} \tag{4.5}$$

Où $\vec{r_1}$ et $\vec{r_2}$ sont deux vecteurs aléatoires générés dans l'intervalle [0, 1] et les composants de $\vec{a}$ diminuent linéairement de 2 à 0 au cours des itérations.

Lorsque la proie cesse de bouger alors le loup gris termine la chasse en attaquant la proie et cela est modélisé mathématiquement par la diminution de la valeur de $\vec{a}$. $\vec{A}$ est une valeur aléatoire générée dans l'intervalle $[-2a, 2a]$, où a est diminué de 2 à 0 au cours des itérations.

4.5.3 Chasse

Dans GWO, il est évident que alpha, bêta et delta sont toujours les trois meilleures solutions obtenues. L'optimum global dans les problèmes d'optimisation est toujours inconnu, il a été supposé que alpha, bêta et delta ont une meilleure connaissance de l'emplacement potentiel des proies car ce sont les meilleures solutions dans l'ensemble de la population. Par conséquent, les autres loups devraient être obligés de mettre à jour leurs positions (figure 4.3).

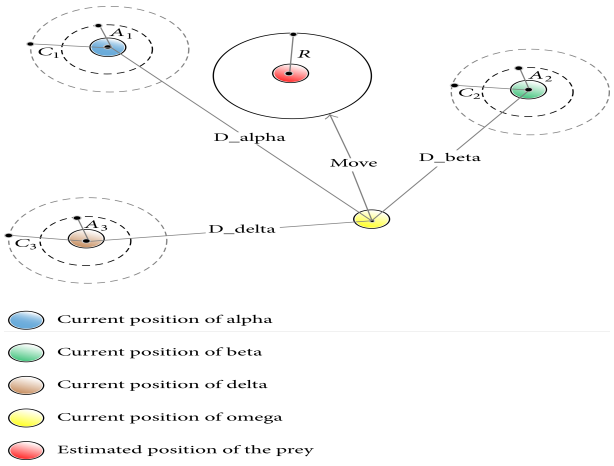


FIGURE 4.3 – Mis à jour des positons des loups gris

Le modèle mathématique de la chasse peut être exprimé comme suit :

$$\vec{D_\alpha} = |\vec{C_1}.\vec{X_\alpha} - \vec{X}|, \quad \vec{D_\beta} = |\vec{C_2}.\vec{X_\beta} - \vec{X}|, \quad \vec{D_\delta} = |\vec{C_3}.\vec{X_\delta} - \vec{X}| \tag{4.6}$$

$$\vec{X_1} = \vec{X_\alpha} - \vec{A_1}.(\vec{D_\alpha}), \quad \vec{X_2} = \vec{X_\beta} - \vec{A_2}.(\vec{D_\beta}), \quad \vec{X_3} = \vec{X_\delta} - \vec{A_3}.(\vec{D_\delta}) \tag{4.7}$$

$$\vec{X(t+1)} = \frac{\vec{X_1} + \vec{X_2} + \vec{X_3}}{3} \tag{4.8}$$

4.5.4 Attaque de proies (Exploitation)

Les loups gris terminent la chasse en attaquant la proie lorsqu'elle cesse de se déplacer. l'exploitation est favorisée lorsque $-1 < A < 1$. Un bon équilibre entre l'exploitation et l'exploration est nécessaire pour trouver l'optimum global avec précision, et cela à l'aide du paramètre a qui diminue linéairement de 2 à 0.

4.6 Algorithme de découverte de services basé sur l'algorithme GWO SD-GWO

Le pseudo code de l'algorithme SD-GWO est donné ci-dessous :

Algorithm 6 SD-GWO GWO-based service discovery : la découverte de services basée sur l'algorithme GWO

Entrée(s) Req (S_j, O_i) requête de l'utilisateur, S_j est l'ensemble de services demandés par O_i
Sortie(s) LO La meilleure solution

initialiser la position X_k la population de loups gris (agents de recherche) (k=1,2,..N) où N le nombre de loups
initialiser a, A, et C
$X_\alpha \leftarrow$ Le meilleur agent de recherche
$X_\beta \leftarrow$ Deuxième meilleur agent de recherche
$X_\delta \leftarrow$ Troisième meilleur agent de recherche
$LO = inf$ Le nombre de sauts de la meilleure solution
tant que $t <= Max\_iter$ **faire**
  pour k =1..N **faire**
    Mis à jour de la position de l'agent de recherche k par l'équation 4.8
    chercher l'objet O_h, O_h est l'objet le plus proche de l'agent k
    si O_h offre le service S_j **alors**
      Calculer la fonction objectif pour O_h par l'équation 4.1
      Mis à jour LO
    fin si
  fin pour
  Mettre à jour a, A, et C
  Mettre à jour X_α,X_β et X_δ
  t=t+1 ;
fin tant que
retourner LO

4.6.1 Fonctionnement de l'algorithme

Nous allons maintenant décrire les différentes phases de algorithme proposé, on décrit d'abord les entrées et sorties. La première phase constitue une phase d'initialisation, suivi d'une phase de recherche des solutions, ensuite une phase de sélection de la meilleure solution, enfin la phase finale qui retourne la solution pour le problème.

4.6.1.1 Entrées et sorties

En entrée, l'algorithme démarre avec M objets qui sont repartis aléatoirement, N agents de recherche (loups)qui sont repartis aléatoirement, et NB_S services, chaque objet offre de 1 à 3

services. Une requête de recherche $REQ(S_i, O_i)$ initiée de par l'objet O_i pour découvrir l'ensemble de services S_i.
En sortie, l'algorithme fournit une solution (un chemin) proche de l'optimale.

4.6.1.2 Initialisation de l'algorithme

Les paramètres s'agencent de la manière suivante au début de l'algorithme :

1. Initialiser aléatoirement la position X loups gris (agents de recherche)
2. initialiser les paramètres a, A et C.
3. Initialiser
   $X_\alpha \leftarrow$ Le meilleur agent de recherche
   $X_\beta \leftarrow$ Deuxième meilleur agent de recherche
   $X_\delta \leftarrow$ Troisième meilleur agent de recherche

4.6.1.3 Phase de recherche

Dans la phase de recherche les loups gris recherchent souvent la proie selon les positions de X_α, X_β et X_δ. Ils divergent les uns des autres pour explorer la position de la proie et convergent ensuite pour attaquer la proie.

4.6.1.4 Phase de Remplacement

À la fin de chaque itération, t, les trois premières meilleures solutions (X_α,X_β, et X_δ) sont mises à jour, et les autres agents sont obligés de mettre à jour leurs positions en fonction de la position des meilleurs agents de recherche.

4.6.1.5 Phase finale

On arrête l'algorithme après un nombre de cycles égal à une constante Max_iter, c'est à partir de cet instant, que tous les loups (agents de recherche) vont converger, l'algorithme n'interrompe. On est dans une situation de stagnation où le programme arrête de chercher des alternatives. L'algorithme donne en retour la meilleure solution sauvegardée.

4.7 Étude de la complexité de l'algorithme SD-GWO

Dans cette partie, nous présentons la complexité temporelle de l'algorithme proposé SD-GWO. Cette complexité dépend de quelques paramètres : N (le nombre de loups), M (le nombre d'objets), et Max_iter(le nombre maximum d'itérations). Le tableau 4.2 présente la complexité temporelle dans les différentes phases de l'algorithme.

Phase de l'algorithme	Complexité	Détails
Phase d'initialisation	$O(3+N+3) = O(N)$	Initialiser les trois paramètres de l'algorithme, les positions des N loups, et les trois meilleures solutions
Phase de recherche	$O(N + N + M) = O(N+M)$	Mis à jour des positions des loups et leurs évaluations avec la fonction objectif
Phase de remplacement	$O(N+3) = O(N)$	mis a jours des trois meilleurs solutions et les trois paramètres

Tableau 4.2 – Complexité temporelle de l'algorithme SD-GWO

La complexité globale est obtenu en additionnant la complexité de la phase d'initialisation, au produit de nombre total d'itérations Max_iter par la complexité de la phase de recherche et la complexité de la phase de remplacement. la complexité générale de l'algorithme est :
$O(N) + O(Max\_iter * (N+M)) + O(Max\_iter * N)$ Après la simplification, on aura la complexité est de l'ordre $O(Max\_iter * (N+M))$

4.8 Évaluation des performances de l'algorithme SD-GWO

Dans cette partie, nous décrivons les résultats de l'évaluation, en simulation de l'algorithme proposé. Pour ce faire, nous présentons en premier lieu l'environnement de simulation, nous décrivons ensuite le corpus de données et scénarios d'évaluation suivi des métriques à considérer pour l'évaluation et enfin, nous montrons à travers une série de résultats de simulation, l'apport de l'algorithme proposé en terme du nombre de sauts et le taux du succès.

4.8.1 Environnement de simulation

Les expérimentations ont été menées sur machine HP dotée d'un processeur Core i5 et une RAM de 6GB, exécutant Matlab R2016b sous le système Windows 10 professionnel 64 bits.
Les différents paramètres de notre simulation sont décrits dans le tableau 4.3.

Portée de transmission R	25m
Nombre maximum d'itération Max_iter	100
Nombre d'objets Nb_Ants	50,100,5000,1000
Nombre de services offerts par objet	1-3

Tableau 4.3 – Paramètres de simulation 2

4.8.2 Corpus de données et scénarios d'évaluation

Pour évaluer les performances de notre proposition, nous avons utilisé une collection de données générées aléatoirement, appliquées sur diffèrent scénarios de découverte de services. Ces scénarios sont générés en faisant varier le nombre d'objets.

4.8.3 Métriques et méthodologie de simulation

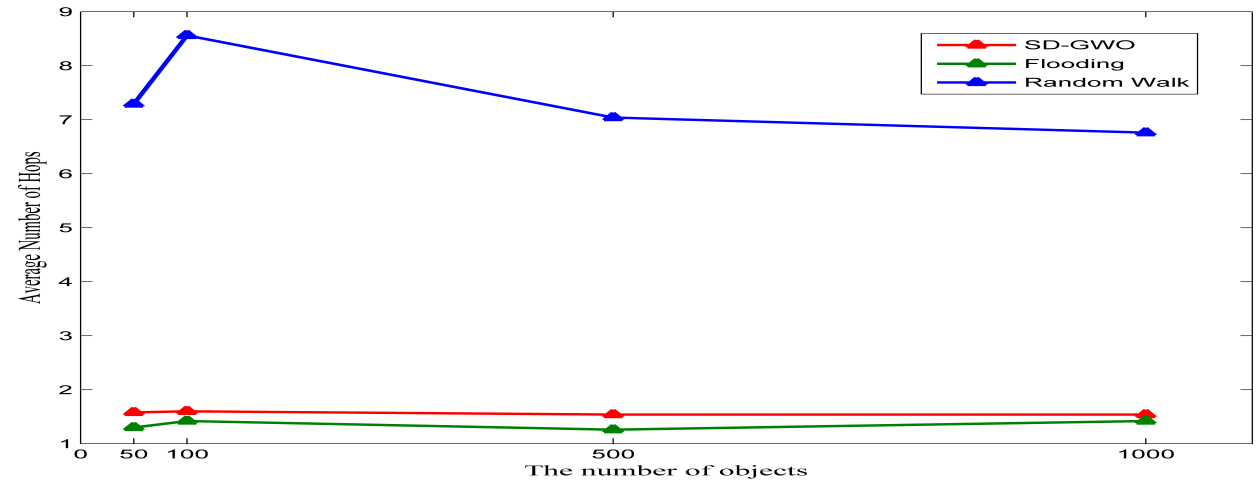
Les performances de l'algorithme proposé sont comparées à celle de deux algorithmes : la diffusion (Flooding) et Random Walk. Les métriques considérées pour l'évaluation des performances sont : le nombre de sauts et le taux de succès qui sont déjà définis dans la section 3.9.4.

4.8.4 Les résultats de simulation

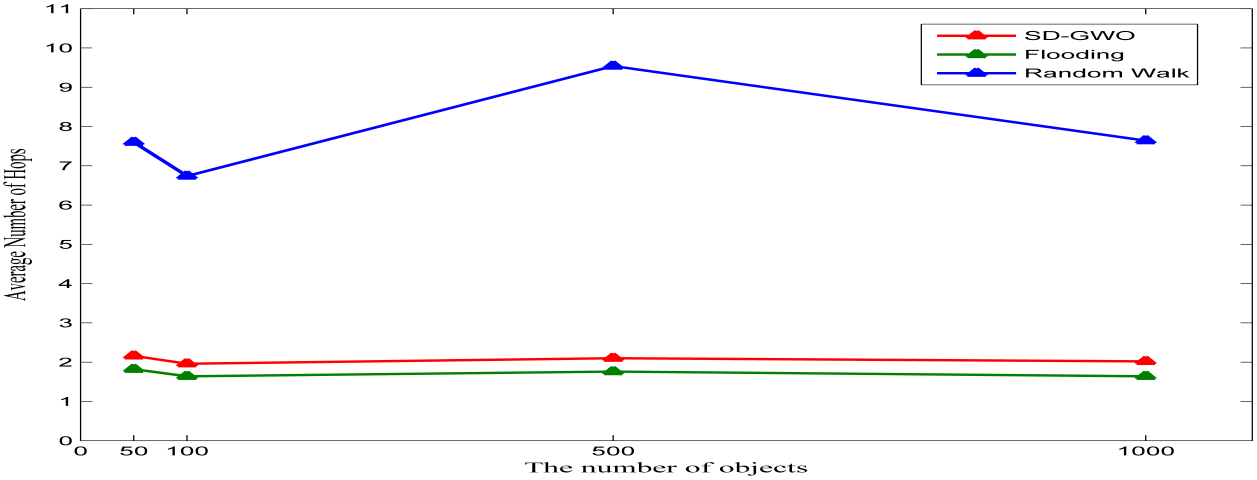
Cette partie illustre une comparaison du nombre de sauts entre les trois approches. Le nombre de sauts est mesuré en fonction du nombre d'objets, nous avons effectué différents tests pour les trois approches en faisant varier le nombre d'objets (50,100,500,1000) et une requête qui contient (1,2,3) services. Toutes les mesures des simulations représentent les résultats moyens pris de 100 exécutions successives.

Le nombre de sauts

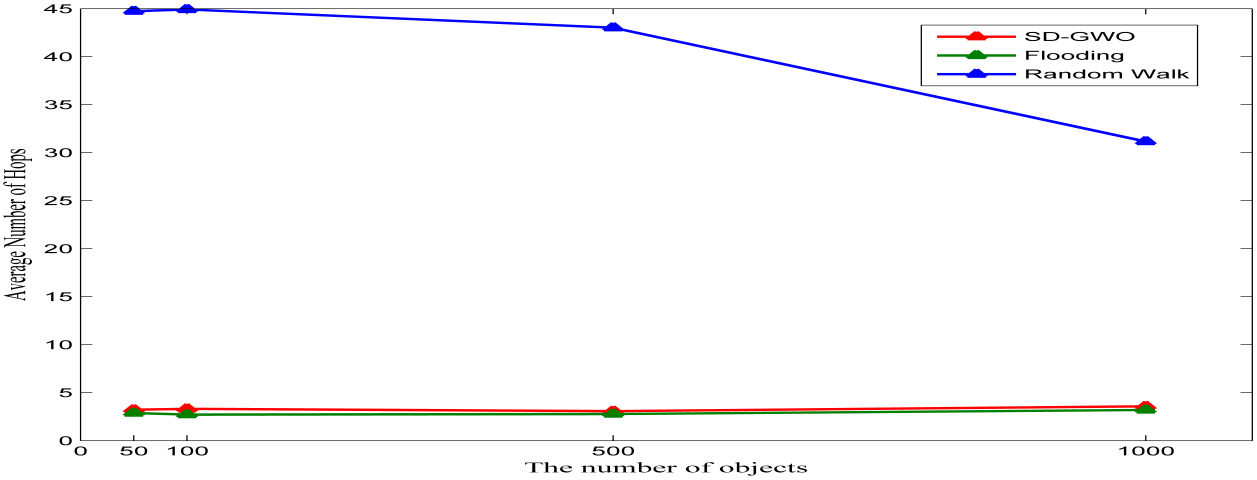
Les figures 4.4a, 4.4b, et 4.4c montrent que lorsque le nombre d'objets augmente, le nombre sauts effectués pour la découverte de services dans les deux approches (Flooding et SD-ACA) est comparables (avec une petite légère infériorité pour Flooding). Cependant, dans l'approche basée sur Flooding, le réseau peut être saturé en raison des nombreuses rediffusions qui doivent être opérées par l'ensemble des objets intermédiaires qui relaient le message provenant d'un objet et sa destination. Cela peut également augmenter le taux de collisions et augmente aussi la consommation d'énergie. Par contre, l'approche basée sur Random walk donne des mauvais résultats car elle est purement aléatoire. Nous constatons aussi que notre approche garantit le passage à l'échelle du réseau et garde ses performances malgré le nombre élevé d'objets.



(a) Avec un service dans la requête



(b) Avec deux services dans la requête



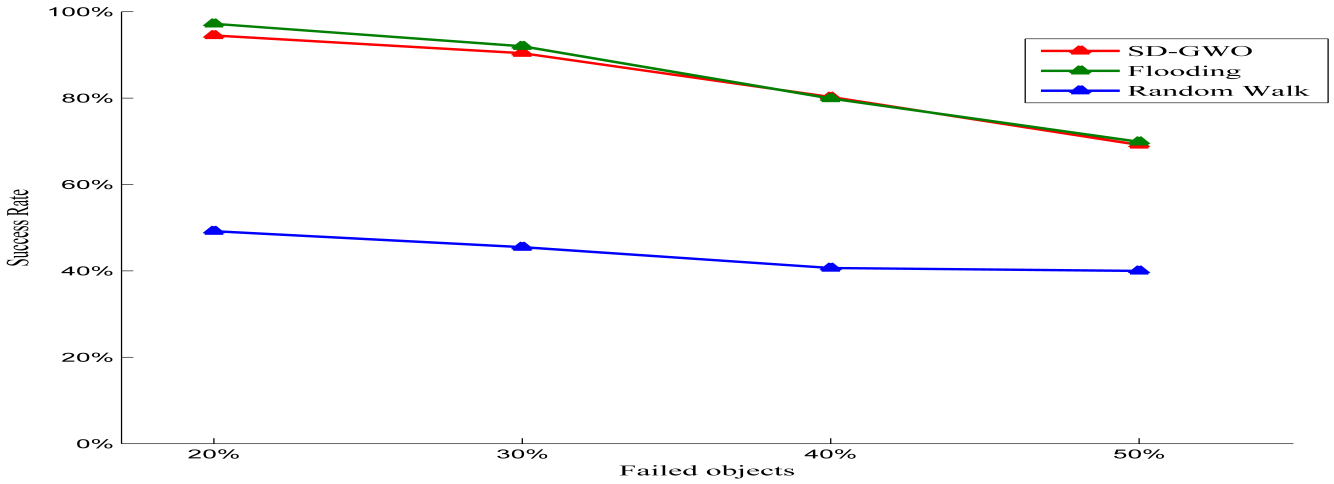
(c) Avec trois services dans la requête

FIGURE 4.4 – Nombre moyen de sauts effectués dans la découverte de services pour les différentes approches GWO, Flooding et Random Walk

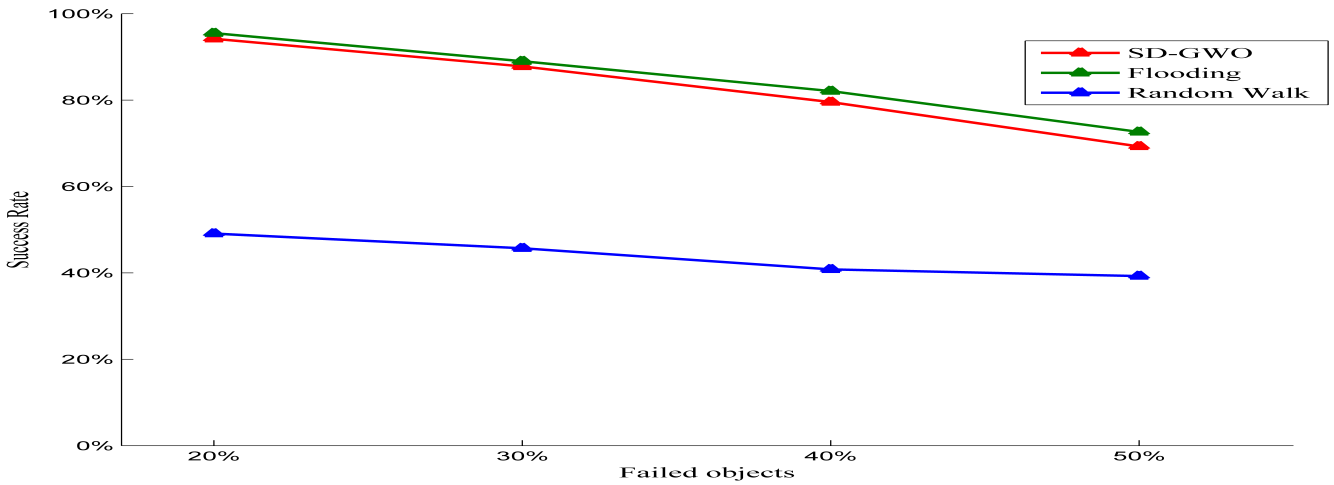
Le taux de succès

Cette partie illustre une comparaison du taux de succès entre les trois approches. Le taux de succès est mesuré en fonction d'objets défaillants dans notre système, nous avons effectué différents tests pour les trois approches en faisant varier le pourcentage d'objets défaillants (20%,30%,40%,50%) et une requête qui contient un service et la valeur de TTL égal à 3. Toutes les mesures des simulations représentent les résultats moyens pris de 100 exécutions successives.

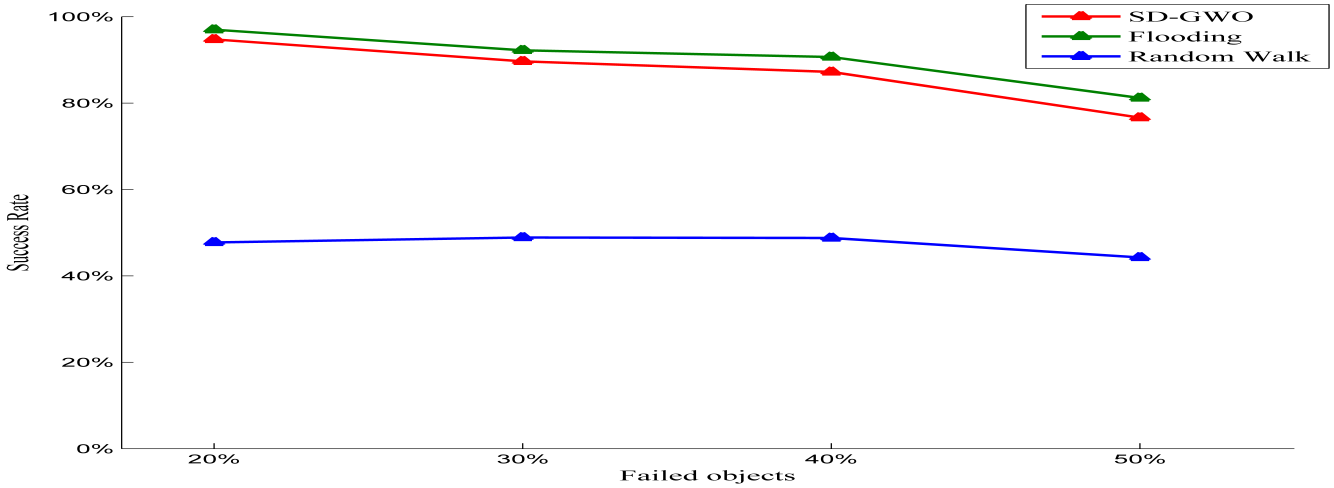
Les figure 4.5a et 4.5b montrent que lorsque le nombre d'objets défaillants augmente, le taux de succès diminue dans les trois approches. Nous constatons que notre approche fournit des résultats similaires à ceux obtenus avec l'approche Flooding. Par contre l'approche Random Walk donne de mauvais résultats car elle est purement aléatoire et vu sa nature stochastique.



(a) Avec 50 objects



(b) Avec 100 objects



(c) Avec 500 objects

FIGURE 4.5 – Taux du succès

4.8.5 Synthèse

En résumé, nos résultats de simulation ont montré que le taux de succès, le nombre de sauts requis pour la découverte de services sont très proches à ceux de l'algorithme Flodding, l'avantage clé de notre solution est qu'elle ne nécessite pas une table de routage mais seulement de déployer des agents de recherche (loups) qui utilisent deux mécanismes (Exploration et exploitation) pour une efficace recherche. Cependant, l'approche Flooding se base sur une stratégie d'inondation, et cela n'est pas adapté aux objets avec ressources limités. En plus, le modèle proposé est décentralisé, ce qui permet le passage à des échelles importantes.

4.9 Conclusion

Nous avons présenté dans ce chapitre notre contribution pour le problème de découverte de services. Nous avons proposé une approche de découverte basée sur La métaheuristique de GWO (Grey Wolf Optimization), elle est inspirée de la hiérarchie de leadership et le mécanisme de chasse des loups gris dans la nature. Dans ce chapitre nous avons présenté notre contribution, nous avons adapté la metaheuristique GWO en mode discret au problème de découverte de services, et nous avons opté pour une approche décentralisée pour une meilleure mise à l'échelle des services et des objets.
A partir de résultats expérimentaux, nous avons montré que notre solution offre de meilleurs performances en terme de nombre de sauts et le taux du succès.

Chapitre 5

Conclusion et perspectives

L'Internet des Objets est défini comme une infrastructure complexe qui supprime les frontières entre le monde physique et le monde virtuel. En effet, l'IoT regroupe plusieurs domaines tels que les réseaux de capteurs sans fils, les villes intelligentes, etc. De ce fait, IoT est un système hétérogène utilisant les différentes technologies de l'information et de la communication pour interconnecter ces dispositifs et pour leur permettre d'échanger leurs données pour fournir une variété de services qui peuvent être découverts grâce à des mécanismes de découverte. Cependant, avec la prolifération et l'évolution exceptionnelle du nombre d'objets connectés et leurs services, la découverte de services est l'une des problématiques les plus importantes dans l'Internet des Objets. En effet, les techniques traditionnelles de recherche de services ne suffisent plus face à la diversité et au nombre croissant de services disponibles aujourd'hui dans IoT.

L'objectif de cette thèse est de proposer des solutions pour découvrir de meilleurs services satisfaisants les besoins élémentaires d'une requête avec un minimum de sauts sans s'adresser à l'annuaire (architecture décentralisée). Ce problème est considéré comme un problème d'optimisation en raison de multitude de services disponibles avec différents paramètres de qualité de service dans le réseau IoT.

5.1 Synthèse

Le premier chapitre de cette thèse à été réservé à la présentation des concepts fondamentaux. Dans la premier partie, nous avons décrit les concepts de base, l'Internet des Objets, object connecté, Architecture de l'Internet des Objets, domaines d'application, et un large panel de Technologies utilisées. Nous avons, par la suite, passé en revue les challenges d'IoT, dont la découverte de services fait partie et nous avons constaté que de nombreux problèmes peuvent être considérés comme des problèmes d'optimisation. Dans la deuxième partie, nous avons présenté les deux grandes classes : les méthodes exactes et les méthodes approchées. Les métaheuristiques

constituent une classe de méthodes approximatives adaptées à un grand nombre de problèmes d'optimisation, leur utilisation dans les différents domaines montre leur efficacité à résoudre des problèmes complexes et offre la possibilité de trouver souvent des solutions approchées de bonne qualité en un temps raisonnable. Nous nous sommes particulièrement intéressés à la deuxième catégorie de cas qui constitue un axe majeurs de recherche pour la résolution du problème de la découverte de services.

Dans le chapitre 2, nous avons présenté le service et les concepts clé de la découverte de services ainsi que leur architecture. Ensuite, nous avons fait une classification des différentes approches existantes relatives à la découverte de services dans IoT pour dégager les atouts et les limites de chaque approche, cette classification nous a permis de constater que, d'une part, les techniques traditionnelles de recherche de services ne suffisent plus face à la diversité et au nombre croissant de services disponibles aujourd'hui sur IoT. D'autre part, les approches bio-inspirées ont rencontré un vif succès grâce à leur simplicité, leur adaptabilité, leur flexibilité et leur capacité de trouver des solutions presque optimales. Ces méthodes sont très efficaces et largement utilisées.

Pour résoudre le problème de découverte de services, nous avons opté pour l'utilisation des métaheuristiques, connus pour leurs performances dans la résolution des problèmes. Deux contributions sont alors proposées.

Dans le chapitre 03, nous avons présenté la premier contribution, elle consiste à l'adaptation de l'algorithme ACA (Ant Colony Algorithm) pour la découverte de services, et nous avons opté pour une approche décentralisée pour une meilleure mise à l'échelle des services et des objets.

Les résultats d'évaluation en simulation de cette contribution, nous ont permis de constater d'une part l'algorithme proposée est plus prometteuse en terme de nombre du sauts, et le passage à l'échelle. D'autre part, les comparaisons avec l'algorithme Flooding indiquent que l'algorithme Flooding présente une infine légère supériorité que SD-ACA, néanmoins Flooding se base sur une stratégie d'inondation, et cela n'est pas adapté aux objets avec ressources limités.

Dans le quatrième chapitre une deuxième contribution a été présentée. Nous avons adapté l'algorithme GWO(Gray Wolf Optimizer) en mode discret au problème de découverte de services.

Les résultats d'évaluation en simulation de cette contribution montre que l'algorithme proposé est plus prometteur et terme de nombre de sauts, et le passage à l'échelle.

5.2 Perspectives

Dans notre thèse, nous avons proposé des solutions bio-inspirées pour la découverte de services dans IoT. Dans nos contributions, nous avons obtenu des résultats satisfaisants mais cela pourrait conduire à plusieurs améliorations et enrichissements. Nous énonçons quelques perspectives intéressantes qui pourraient améliorer ce travail :

- prendre en compte l'aspect non fonctionnel (description sémantique) des services, en proposant une approche qui permet d'ajouter de l'information sémantique à la description classique du service. Cette information peut ensuite être traitée et intégrer lors du processus de découverte de services.
- Explorer d'autres métaheuristiques qui s'inspirent de la nature comme l'algorithme Cuckoo Search (CS), et l'algorithme de recherche Tabou.

- Prendre en compte d'autres paramètres de qualité de service lors de la découverte de services.

- La découverte de services cloud doit être étendue pour tenir en compte des services IAAS (Infrastructure As A Service) en plus des services traditionnels SAAS (Software As A Service). les services SAAS représentent des services Web traditionnels accessibles avec un compte sur une plateforme de cloud computing.

- Une autre piste à suivre est celle des inspirations naturelles. La nature est une source inépuisable d'inspiration. Une partie de nos futures recherches va se concentrer sur l'observation de nouveaux phénomènes naturels qui peuvent améliorer réellement et remarquablement la performance des métaheuristiques. L'autre partie, examinera la possibilité d'hybrider plusieurs métaheuristiques existantes afin de cumuler leurs avantages sous un seul modèle (hybride).

Liste des contributions

Dans le cadre de cette thèse, nous avons réalisé les contributions scientifiques suivantes :

1. Z. BENSALAH AZIZOU, A. BOUDRIES, and M. AMAD. Decentralized service discovery and localization in internet of things applications based on ant colony algorithm. International Journal of Computing and Digital Systems, 9(5) :941950, 2020
2. Z. BENSALAH AZIZOU, A. BOUDRIES, and M. AMAD. Grey Wolf Optimizer-based decentralized service discovey in Internet of Things applications, has been submitted to Wireless Personal Communications

Bibliographie

[1] G. D. Abowd, A. K. Dey, P. J. Brown, N. Davies, M. Smith, and P. Steggles. Towards a better understanding of context and context-awareness. In *International symposium on handheld and ubiquitous computing*, pages 304–307. Springer, 1999.

[2] E. Adar and B. A. Huberman. Free riding on gnutella. 2000.

[3] A. Al-Fuqahaand, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash. Internet of things : A survey on enabling technologies, protocols, and applications. *IEEE communications surveys & tutorials*, 17(4) :2347–2376, 2015.

[4] I. Alaya, C. Solnon, and K. GHéDIRA. Ant algorithm for the multidimensional knapsack problem. In *International conference on Bioinspired Methods and their Applications (BIOMA 2004)*, pages 63–72, 2004.

[5] H. Alemdar and C. Ersoy. Wireless sensor networks for healthcare : A survey. *Computer networks*, 54(15) :2688–2710, 2010.

[6] M. Antonini, S. Cirani, G. Ferrari, P. Medagliani, M. Picone, and L. Veltri. Lightweight multicast forwarding for service discovery in low-power iot networks. In *2014 22nd International conference on software, telecommunications and computer networks (SoftCOM)*, pages 133–138. IEEE, 2014.

[7] F. M. Anwar, S.W. Yoo, and K. H. Kim. Survey on service discovery for wireless sensor networks. In *2010 Second International Conference on Ubiquitous and Future Networks (ICUFN)*, pages 17–21. IEEE, 2010.

[8] A. Arsanjani. Service-oriented modeling and architecture. *IBM developer works*, 1 :1–15, 2004.

[9] K. Ashton et al. That 'internet of things' thing. *RFID journal*, 22(7) :97–114, 2009.

[10] L. Atzori, A. Iera, and G. Morabito. The internet of things : A survey. *Computer networks*, 54(15) :2787–2805, 2010.

[11] L. Atzori, A. Iera, G. Morabito, and M. Nitti. The social internet of things (siot)–when social networks meet the internet of things : Concept, architecture and network characterization. *Computer networks*, 56(16) :3594–3608, 2012.

[12] V. Autefage. *Découverte de services et collaboration au sein d'une flotte hétérogène et hautement dynamique d'objets mobiles communicants autonomes*. PhD thesis, Université de Bordeaux, 2015.

[13] Z. BENSALAH AZIZOU, A. BOUDRIES, and M. AMAD. Decentralized service discovery and localization in internet of things applications based on ant colony algorithm. *International Journal of Computing and Digital Systems*, 9(5) :941–950, 2020.

[14] H. Balakrishnan. Chord : A scalable peer-to-peer lookup service for internet applications. In *ACM SIGCOMM*. Citeseer, 2001.

[15] J. Bao, Y. Ding, and H. Hu. A new service selection algorithm in uspiot. In *2012 IEEE International Conference on Computer Science and Automation Engineering (CSAE)*, volume 2, pages 22–26. IEEE, 2012.

[16] J. M. Barroso. Objectif 20-20-20 en 2020. `https://www.lalibre.be/economie/entreprises-startup/2008/01/28/objectif-20-20-20-en-2020-RITD2BIDI5FLVD5V2IDF6KNWNQ/`, 2008 (consulté le 17 Juillet 2021).

[17] D. Basu, G. Moretti, G. S. Gupta, and S. Marsland. Wireless sensor network based smart home : Sensor selection, deployment and monitoring. In *2013 IEEE Sensors Applications Symposium Proceedings*, pages 49–54. IEEE, 2013.

[18] P. J. Benghozi, S. Bureau, and F. Massit-Folléa. *L'Internet des objets/The Internet of Things : Quels Enjeux Pour L'Europe ?/What Challenges for Europe ?* Les Editions de la MSH, 2009.

[19] B. Benhamou. L'internet des objets : Défis technologiques, économiques et politiques. *Esprit (1940-)*, pages 137–150, 2009.

[20] C. Blum. Beam-aco-hybridizing ant colony optimization with beam search : An application to open shop scheduling. *Computers & Operations Research*, 32(6) :1565–1591, 2005.

[21] J. C. Boisson. *Modélisation et résolution par métaheuristiques coopératives : de l'atome à la séquence protéique*. PhD thesis, Université Lille 1, 2008.

[22] E. Borgia. The internet of things vision : Key features, applications and open issues. *Computer Communications*, 54 :1–31, 2014.

[23] I. Boussaïd, J. Lepagnot, and P. Siarry. A survey on optimization metaheuristics. *Information sciences*, 237 :82–117, 2013.

[24] B. Bullnheimer, R. F. Hartl, and C. Strauss. An improved ant system algorithm for thevehicle routing problem. *Annals of operations research*, 89 :319–328, 1999.

[25] T. A. Butt, I. Phillips, L. Guan, and G. Oikonomou. Adaptive and context-aware service discovery for the internet of things. In *Internet of things, smart spaces, and next generation networking*, pages 36–47. Springer, 2013.

[26] P. C. Calcina-Ccori, L. C. De Biase, C. E. L. De Oliveira, G. Fedrecheski, F. C. da Silva, and M. K. Zuffo. Describing services geolocation in iot context. In *2019 IEEE International Conference on Consumer Electronics (ICCE)*, pages 1–2. IEEE, 2019.

[27] D. Chakraborty, A. Joshi, Y. Yesha, and T. Finin. Gsd : A novel group-based service discovery protocol for manets. In *4th International workshop on mobile and wireless communications network*, pages 140–144. IEEE, 2002.

[28] B. Cheng, D. Zhu, S. Zhao, and L. Chen. Situation-aware iot service coordination using the event-driven soa paradigm. *IEEE Transactions on Network and Service Management*, 13(2) :349–361, 2016.

[29] H. R. chindler, J. A. K. Cave, N. Robinson, V. Horvath, P. J. Hackett, S. Gunashekar, M. Botterman, S. Forge, and H. Graux. *Europe's Policy Options for a Dynamic and Trustworthy Development of the Internet of Things : SMART 2012/0053*. RAND, 2013.

[30] G. Chong, L. Zhihao, and Y. Yifeng. The research and implement of smart home system based on internet of things. In *2011 International Conference on Electronics, Communications and Control (ICECC)*, pages 2944–2947. IEEE, 2011.

[31] A. Dahbi, M. G. Khair, and H. T. Mouftah. Secured distributed discovery services in the epcglobal network. In *2013 IEEE International Conference on Communications (ICC)*, pages 2959–2963. IEEE, 2013.

[32] C. Darwin. *On the origin of species, 1859*. Routledge, 2004.

[33] D. Dasgupta and S. Forrest. An anomaly entection algorithm inspired by the immune syste. In *Artificial immune systems and their applications*, pages 262–277. Springer, 1999.

[34] J. L. Deneubourg, J. M. Pasteels, and J. C. Verhaeghe. Probabilistic behaviour in ants : a strategy of errors ? *Journal of theoretical Biology*, 105(2) :259–271, 1983.

[35] M. Dorigo and L. M. Gambardella. Ant colonies for the travelling salesman problem. *biosystems*, 43(2) :73–81, 1997.

[36] M. Dorigo, V. Maniezzo, and A. Colorni. The ant system : An autocatalytic optimizing process. 1991.

[37] M. Dorigo, V. Maniezzo, and A. Colorni. Ant system : optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 26(1) :29–41, 1996.

[38] M. Dorigo and T. Stützle. The ant colony optimization metaheuristic : Algorithms, applications, and advances. In *Handbook of metaheuristics*, pages 250–285. Springer, 2003.

[39] V. Dyo, S. A. Ellwood, D. M. Macdonald, A. Markham, N. Trigoni, R. Wohlers, C. Mascolo, B. Pásztor, S. Scellato, and K. Yousef. Wildsensing : Design and deployment of a sustainable sensor network for wildlife monitoring. *ACM Transactions on Sensor Networks (TOSN)*, 8(4) :1–33, 2012.

[40] R. Eberhart and J. Kennedy. A new optimizer using particle swarm theory. In *MHS'95. Proceedings of the sixth international symposium on micro machine and human science*, pages 39–43. Ieee, 1995.

[41] M. M. Eusuff and K. E. Lansey. Optimization of water distribution network design using the shuffled frog leaping algorithm. *Journal of Water Resources planning and management*, 129(3) :210–225, 2003.

[42] D. Evans. The internet of things : How the next evolution of the internet is changing everything. *CISCO white paper*, 1(2011) :1–11, 2011.

[43] M. A. Feki, F. Kawsar, M. Boussard, and L. Trappeniers. The internet of things : the next technological revolution. *Computer*, 46(2) :24–25, 2013.

[44] T. A Feo and M. G. C. Resende. Greedy randomized adaptive search procedures. *Journal of global optimization*, 6(2) :109–133, 1995.

[45] R. Ferdousi and P. K Mandal. Loamy : A cloud-based middleware for coap-based iot service discovery. In *2019 Second international conference on advanced computational and communication paradigms (ICACCP)*, pages 1–6. IEEE, 2019.

[46] R. T. Fielding. *Architectural Styles and the Design of Network-based Software Architectures*. PhD thesis in Information and Computer Science, UNIVERSITY OF CALIFORNIA, IRVINE, October 2018.

[47] L. J. Fogel, A. J. Owens, and M. J. Walsh. Intelligent decision making through a simulation of evolution. *Behavioral science*, 11(4) :253–272, 1966.

[48] S. B. Fredj, M. Boussard, D. Kofman, and L. Noirie. Efficient semantic-based iot service discovery mechanism for dynamic environments. In *2014 IEEE 25th annual international symposium on personal, indoor, and mobile radio communication (PIMRC)*, pages 2088–2092. IEEE, 2014.

[49] K. Fysarakis, I. Askoxylakis, O. Soultatos, I. Papaefstathiou, C. Manifavas, and V. Katos. Which iot protocol ? comparing standardized approaches over a common m2m application. In *2016 IEEE Global Communications Conference (GLOBECOM)*, pages 1–7. IEEE, 2016.

[50] J. A. Garcia-Macias and D. A. Torres. Service discovery in mobile ad-hoc networks : better at the network layer ? In *2005 International Conference on Parallel Processing Workshops (ICPPW'05)*, pages 452–457. IEEE, 2005.

[51] M. Ghorbani, M. R. Meybodi, and A. M. Saghiri. A new version of k-random walks algorithm in peer-to-peer networks utilizing learning automata. In *The 5th Conference on Information and Knowledge Technology*, pages 1–6. IEEE, 2013.

[52] F. Glover. Future paths for integer programming and links to artificial intelligence. *Computers & operations research*, 13(5) :533–549, 1986.

[53] D. Guinard and V. Trifa. Towards the web of things : Web mashups for embedded devices. In *Workshop on Mashups, Enterprise Mashups and Lightweight Composition on the Web (MEM 2009), in proceedings of WWW (International World Wide Web Conferences), Madrid, Spain*, volume 15, page 8, 2009.

[54] D. Guinard, V. Trifa, S. Karnouskos, P. Spiess, and D. Savio. Interacting with the soa-based internet of things : Discovery, query, selection, and on-demand provisioning of web services. *IEEE transactions on Services Computing*, 3(3) :223–235, 2010.

[55] P. Gupta, T. P. Mokal, D. D. Shah, and K. V. V. Satyanarayana. Event-driven soa-based iot architecture. In *International Conference on Intelligent Computing and Applications*, pages 247–258. Springer, 2018.

[56] D. Haas., F. McCabe, E. Newcomer, M. Champion, C. Ferris, and D. Orchard. Web services architecture. `https://www.w3.org/TR/ws-arch/.`, 2004.

[57] M. Han and H. hang. Business intelligence architecture based on internet of things. *Journal of Theoretical & Applied Information Technology*, 50(1) :90–95, 2013.

[58] S. N. Han, S. Park, G. M. Lee, and N. Crespi. Extending the devices profile for web services standard using a rest proxy. *IEEE Internet Computing*, 19(1) :10–17, 2014.

[59] E. Haselsteiner and K. Breitfuß. Security in near field communication (nfc). In *Workshop on RFID security*, volume 517, page 517. sn, 2006.

[60] S. Hashemi, S. Kiani, N. Noroozi, and M. E. Moghaddam. An image contrast enhancement method based on genetic algorithm. *Pattern Recognition Letters*, 31(13) :1816–1824, 2010.

[61] A. Heidari and N. J. Navimipour. A new sla-aware method for discovering the cloud services using an improved nature-inspired optimization algorithm. *PeerJ Computer Science*, 7, 2021.

[62] S. Helal, N. Desai, V. Verma, and C. Lee. Konark-a service discovery and delivery protocol for ad-hoc networks. In *2003 IEEE Wireless Communications and Networking, 2003. WCNC 2003.*, volume 3, pages 2107–2113. IEEE, 2003.

[63] A. Hernandez-Bravo and J. Carretero. Approach to manage complexity in internet of things. *Procedia Computer Science*, 36 :210–217, 2014.

[64] J. H. Holland. An introductory analysis with applications to biology, control, and artificial intelligence. *Adaptation in Natural and Artificial Systems. First Edition, The University of Michigan, USA*, 1975.

[65] M. N. Huhns and M. P. Singh. Service-oriented computing : Key concepts and principles. *IEEE Internet computing*, 9(1) :75–81, 2005.

[66] A. Huning. Evolutionsstrategie. optimierung technischer systeme nach prinzipien der biologischen evolution, 1976.

[67] IEEE. MARCH 2014.

[68] B. Jia, W. Li, and T. Zhou. A centralized service discovery algorithm via multi-stage semantic service matching in internet of things. In *2017 IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC)*, volume 1, pages 422–427. IEEE, 2017.

[69] S. Jin, M. Zhou, and A. S. Wu. Sensor network optimization using a genetic algorithm. In *Proceedings of the 7th world multiconference on systemics, cybernetics and informatics*, pages 109–116. Citeseer, 2003.

[70] J. L. Jodra, M. Vara, J. M. Cabero, and J. Bagazgoitia. Service discovery mechanism over olsr for mobile ad-hoc networks. In *20th International Conference on Advanced Information Networking and Applications-Volume 1 (AINA'06)*, volume 2, pages 6–pp. IEEE, 2006.

[71] D. Karaboga and B. Basturk. A powerful and efficient algorithm for numerical function optimization : artificial bee colony (abc) algorithm. *Journal of global optimization*, 39(3) :459–471, 2007.

[72] S. Kashef and H. Nezamabadi-pour. An advanced aco algorithm for feature subset selection. *Neurocomputing*, 147 :271–279, 2015.

[73] R. Khan, S. U. Khan, R. Zaheer, and S. Khan. Future internet : the internet of things architecture, possible applications and key challenges. In *2012 10th international conference on frontiers of information technology*, pages 257–260. IEEE, 2012.

[74] F. Khodadadi, A. V. Dastjerdi, and R. Buyya. Simurgh : A framework for effective discovery, programming, and integration of services exposed in iot. In *2015 International Conference on Recent Advances in Internet of Things (RIoT)*, pages 1–6. IEEE, 2015.

[75] G. Kim, S. Kang, J. Park, and K. Chung. An mqtt-based context-aware autonomous system in onem2m architecture. *IEEE Internet of Things Journal*, 6(5) :8519–8528, 2019.

[76] D. Kiritsis. Closed-loop plm for intelligent products in the era of the internet of things. *Computer-Aided Design*, 43(5) :479–501, 2011.

[77] S. Kirkpatrick, C. D. Gelatt Jr, and M. P. Vecchi. Optimization by simulated annealing. *science*, 220(4598) :671–680, 1983.

[78] M. Klein and B. König-Ries. Multi-layer clusters in ad-hoc networks ?an approach to service discovery. In *International Conference on Research in Networking*, pages 187–201. Springer, 2002.

[79] M. Klein, B. Konig-Ries, and P. Obreiter. Service rings-a semantic overlay for service discovery in ad hoc networks. In *14th International Workshop on Database and Expert Systems Applications, 2003. Proceedings.*, pages 180–185. IEEE, 2003.

[80] S. Kosunalp and K. Demir. Sarl : A reinforcement learning based qos-aware iot service discovery model. *Journal of Electrical Engineering*, 71(6) :368–378, 2020.

[81] A. Kout. *Contributions à la résolution du problème de routage dans les réseaux mobiles Ad-hoc par les méthodes bio-inspirées*. PhD thesis, Abdelhamid Mehri - Constantine 2, 2017.

[82] J. R. Koza et al. Evolution of subsumption using genetic programming. In *Proceedings of the first European conference on artificial life*, pages 110–119. MIT Press Cambridge, MA, 1992.

[83] U. C. Kozat and L. Tassiulas. Service discovery in mobile ad hoc networks : an overall perspective on architectural choices and network layer support issues. *Ad Hoc Networks*, 2(1) :23–44, 2004.

[84] H. Lee, R. Chow, M. R. Haghighat, H. M. Patterson, and A. Kobsa. Iot service store : A web-based system for privacy-aware iot service discovery and interaction. In *2018 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, pages 107–112. IEEE, 2018.

[85] M. Leo, F. Battisti, M. Carli, and A. Neri. A federated architecture approach for internet of things security. In *2014 Euro Med Telco Conference (EMTC)*, pages 1–5. IEEE, 2014.

[86] L. Li and U. Lamont. A lightweight service discovery mechanism for mobile ad hoc pervasive environment using cross-layer design. In *Third IEEE International Conference on Pervasive Computing and Communications Workshops*, pages 55–59. IEEE, 2005.

[87] X. L. Li. An optimizing method based on autonomous animats : fish-swarm algorithm. *Systems Engineering-Theory & Practice*, 22(11) :32–38, 2002.

[88] H. Lin and BN. W. ergmann. Iot privacy and security challenges for smart home environments. *Information*, 7(3) :44, 2016.

[89] C. M. MacKenzie, K. Laskey, F. McCabe, P. F. Brown, R. Metz, and B. A Hamilton. Oasis reference model for service oriented architecture 1.0, oasis standard. *OASIS standard*, octobre 2006.

[90] R. Mahmoud, T. Yousuf, F. Aloul, and I. ualkernan. Internet of things (iot) security : Current status, challenges and prospective measures. In *2015 10th International Conference for Internet Technology and Secured Transactions (ICITST)*, pages 336–341. IEEE, 2015.

[91] M. Mahyoub, A. Mahmoud, and T. Sheltami. An optimized discovery mechanism for smart objects in iot. In *2017 8th IEEE Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, pages 649–655. IEEE, 2017.

[92] P. Maiti, H. K. Apat, B. Sahoo, and A. K Turuk. An effective approach of latency-aware fog smart gateways deployment for iot services. *Internet of Things*, 8 :100091, 2019.

[93] P. Maymounkov and D. Mazieres. Kademlia : A peer-to-peer information system based on the xor metric. In *International Workshop on Peer-to-Peer Systems*, pages 53–65. Springer, 2002.

[94] D. Merkle, M. Middendorf, and H. Schmeck. Ant colony optimization for resource-constrained project scheduling. *IEEE transactions on evolutionary computation*, 6(4) :333–346, 2002.

[95] L. M.Gambardella, É. D. Taillard, and M. Dorigo. Ant colonies for the quadratic assignment problem. *Journal of the operational research society*, 50(2) :167–176, 1999.

[96] A. N. Mian, R. Baldoni, and R. Beraldi. A survey of service discovery protocols in multihop mobile ad hoc networks. *IEEE Pervasive computing*, 8(1) :66–74, 2008.

[97] R. Minerva, A. Biru, and D. Rotondi. Towards a definition of the internet of things (iot) ieee internet initiative, torino, italy, 2015.

[98] D. Miorandi, S. Sicari, F. De Pellegrini, and I. Chlamtac. Internet of things : Vision, applications and research challenges. *Ad hoc networks*, 10(7) :1497–1516, 2012.

[99] S. Mirjalili. The ant lion optimizer. *Advances in engineering software*, 83 :80–98, 2015.

[100] S. Mirjalili, S. M. Mirjalili, M. Seyed, and A. Lewis. Grey wolf optimizer. *Advances in engineering software*, 69 :46–61, 2014.

[101] U. Mohan, K. C. Almeroth, and E. M. Belding-Royer. Scalable service discovery in mobile ad hoc networks. In *International Conference on Research in Networking*, pages 137–149. Springer, 2004.

[102] L. Mottola and G. P. Picco. Programming wireless sensor networks : Fundamental concepts and state of the art. *ACM Computing Surveys (CSUR)*, 43(3) :1–51, 2011.

[103] S. C. Mukhopadhyay and N. K. Suryadevara. Internet of things : Challenges and opportunities. In *Internet of Things*, pages 1–17. Springer, 2014.

[104] G. Mulligan and D. Gračanin. A comparison of soap and rest implementations of a service based interaction independence middleware framework. In *Proceedings of the 2009 Winter Simulation Conference (WSC)*, pages 1423–1432. IEEE, 2009.

[105] R. Nakano and T. Yamada. Conventional genetic algorithm for job shop problems. In *ICGA*, volume 91, pages 474–479, 1991.

[106] S. M. Nasution, E. M. Husni, and A. I. Wuryandari. Prototype of train ticketing application using near field communication (nfc) technology on android device. In *2012 International Conference on System Engineering and Technology (ICSET)*, pages 1–6. IEEE, 2012.

[107] L. Ndlovu, O. P. Kogeda, and M. Lall. Enhanced service discovery model for wireless mesh networks. *Journal of Advanced Computational Intelligence and Intelligent Informatics*, 22(1) :44–53, 2018.

[108] E. W. T. Ngai, K. K. Moon, F. J. Riggins, and Y. Y. Candace. Rfid research : An academic literature review (1995–2005) and future research directions. *International Journal of Production Economics*, 112(2) :510–520, 2008.

[109] D. Nickul, L. Reitman, J. Ward, and J. Wilber. Service oriented architecture (soa) and specialized messaging patterns. *Adobe Article, available at : www. adobe. com/enterprise/pdfs/Services_Oriented_Architecture_from_Adobe. pdf*, 2007.

[110] M. Nidd. Service discovery in deapspace. *IEEE personal communications*, 8(4) :39–45, 2001.

[111] D. Niewolny. How the internet of things is revolutionizing healthcare. *White paper*, pages 1–8, 2013.

[112] C. S. Oh, Y. B. Ko, and Y. S. Roh. An integrated approach for efficient routing and service discovery in mobile ad hoc networks. In *Second IEEE Consumer Communications and Networking Conference, 2005. CCNC. 2005*, pages 184–189. IEEE, 2005.

[113] C. H. Papadimitriou and K. Steiglitz. *Combinatorial optimization : algorithms and complexity*. Courier Corporation, 1998.

[114] K. M. Passino. Biomimicry of bacterial foraging for distributed optimization and control. *IEEE control systems magazine*, 22(3) :52–67, 2002.

[115] S. Pattar, D. S. Kulkarni, D. Vala, R. Buyya, K. R. Venugopal, S. S. Iyengar, and L. M. Patnaik. Progressive search algorithm for service discovery in an iot ecosystem. In *2019 international conference on Internet of Things (iThings) and IEEE green computing and communications (GreenCom) and IEEE cyber, physical and social computing (CPSCom) and IEEE smart data (SmartData)*, pages 1041–1048. IEEE, 2019.

[116] E. M. Pereira, R. Pinto, J. P. C. dos Reis, and G. Gonçalves. Mqtt-rd : A mqtt based resource discovery for machine to machine communication. In *IoTBDS*, pages 115–124, 2019.

[117] C. Preist. A conceptual architecture for semantic web services. In *International Semantic Web Conference (ISWC)*, volume 7-11, pages 395–409, Hiroshima, Japan, November 2010.

[118] M. Pruthvi, S. Karthika, and N. Bhalaji. A novel framework for siot college. In *2019 International Conference on Computational Intelligence in Data Science (ICCIDS)*, pages 1–4. IEEE, 2019.

[119] E. Rapti, A. Karageorgos, C. Houstis, and E. Houstis. Decentralized service discovery and selection in internet of things applications based on artificial potential fields. *Service Oriented Computing and Applications*, 11(1) :75–86, 2017.

[120] O. Ratsimor, D. Chakraborty, A. Joshi, and T. Finin. Allia : Alliance-based service discovery for ad-hoc environments. In *Proceedings of the 2nd international workshop on Mobile commerce*, pages 1–9, 2002.

[121] V. Raychoudhury, J. Cao, W. Wu, Y. Lai, C. Chen, and J. Ma. K-directory community : Reliable service discovery in manet. *Pervasive and mobile computing*, 7(1) :140–158, 2011.

[122] S. Sahraoui. *Mécanismes de sécurité pour l'intégration des RCSFs à l'IOT (Internet of Things))*. PhD thesis, Université de Batna 2, 2016.

[123] F. Sailhan and V. Issarny. Scalable service discovery for manet. In *Third IEEE International Conference on Pervasive Computing and Communications*, pages 235–244. IEEE, 2005.

[124] G. Schiele, C. Becker, and K. Rothermel. Energy-efficient cluster-based service discovery for ubiquitous computing. In *Proceedings of the 11th workshop on ACM SIGOPS European workshop*, pages 14–es, 2004.

[125] P. Sethi and S. R. Sarangi. Internet of things : architectures, protocols, and applications. *Journal of Electrical and Computer Engineering*, 2017, 2017.

[126] V. Sharma, F. Song, I. You, and M. Atiquzzaman. Energy efficient device discovery for reliable communication in 5g-based iot and bsns using unmanned aerial vehicles. *Journal of Network and Computer Applications*, 97 :79–95, 2017.

[127] Z. Shelby, K. Hartke, and C. Bormann. The constrained application protocol (coap). 2014.

[128] D. Simon. Biogeography-based optimization. *IEEE transactions on evolutionary computation*, 12(6) :702–713, 2008.

[129] C. Solnon. Ants can solve constraint satisfaction problems. *IEEE transactions on evolutionary computation*, 6(4) :347–357, 2002.

[130] M. Stolikj, P. J. L. Cuijpers, J. J Lukkien, and N. Buchina. Context based service discovery in unmanaged networks using mdns/dns-sd. In *2016 IEEE international conference on consumer electronics (ICCE)*, pages 163–165. IEEE, 2016.

[131] E. G. Talbi. *Metaheuristics : from design to implementation*, volume 74. John Wiley & Sons, 2009.

[132] L. Tan and N. Wang. Future internet : The internet of things. In *2010 3rd international conference on advanced computer theory and engineering (ICACTE)*, volume 5, pages V5–376. IEEE, 2010.

[133] G. Tanganelli, C. Vallati, and E. Mingozzi. Edge-centric distributed discovery and access in the internet of things. *IEEE Internet of Things Journal*, 5(1) :425–438, 2017.

[134] V. Tyagi and A. Kumar. Internet of things and social networks : A survey. In *2017 International Conference on Computing, Communication and Automation (ICCCA)*, pages 1268–1270. IEEE, 2017.

[135] C. P. Vandana and A. A. Chikkamannur. S-coap : Semantic enrichment of coap for resource discovery. *SN Computer Science*, 1(2) :1–7, 2020.

[136] C. N. Ververidis and G. C. Polyzos. Routing layer support for service discovery in mobile ad hoc networks. In *Third IEEE International Conference on Pervasive Computing and Communications Workshops*, pages 258–262. IEEE, 2005.

[137] C. N. Ververidis and G. C. Polyzos. Service discovery for mobile ad hoc networks : a survey of issues and techniques. *IEEE Communications Surveys & Tutorials*, 10(3) :30–45, 2008.

[138] J. Waldo. The jini architecture for network-centric computing. *Communications of the ACM*, 42(7) :76–82, 1999.

[139] H. Wang, D. Xiong, P. Wang, and Y. Liu. A lightweight xmpp publish/subscribe scheme for resource-constrained iot devices. *IEEE Access*, 5 :16393–16405, 2017.

[140] R. Wang and J. Lu. Qos-aware service discovery and selection management for cloud-edge computing using a hybrid meta-heuristic algorithm in iot. *Wireless Personal Communications*, pages 1–14, 2021.

[141] R. Want. Near field communication. *IEEE Pervasive Computing*, 10(3) :4–7, 2011.

[142] T. Watteyne and K. S. J. Pister. Smarter cities through standards-based wireless sensor networks. *IBM Journal of Research and Development*, 55(1.2) :7–1, 2011.

[143] J. M. T. Wu, J. Zhan, and J. C. W. Lin. An aco-based approach to mine high-utility itemsets. *Knowledge-Based Systems*, 116 :102–113, 2017.

[144] M. Wu, T.J. Lu, F.Y. Ling, J. Sun, and H.Y. Du. Research on the architecture of internet of things. In *2010 3rd international conference on advanced computer theory and engineering (ICACTE)*, volume 5, pages V5–484. IEEE, 2010.

[145] H. Xia, C. Hu, F. Xiao, X. Cheng, and Z. Pan. An efficient social-like semantic-aware service discovery mechanism for large-scale internet of things. *Computer Networks*, 152 :210–220, 2019.

[146] A. Yachir, Y. Amirat, A. Chibani, and N. Badache. Event-aware framework for dynamic services discovery and selection in the context of ambient intelligence and internet of things. *IEEE Transactions on automation science and engineering*, 13(1) :85–102, 2015.

[147] X. S. Yang. Firefly algorithms for multimodal optimization. In *International symposium on stochastic algorithms*, pages 169–178. Springer, 2009.

[148] X. S. Yang and S. Deb. Cuckoo search via lévy flights. In *2009 World congress on nature & biologically inspired computing (NaBIC)*, pages 210–214. Ieee, 2009.

[149] B. Yuan, L. Liu, and N. Antonopoulos. Efficient service discovery in decentralized online social networks. *Future Generation Computer Systems*, 86 :775–791, 2018.

[150] S. Zhao, L. Yu, B. Cheng, and J. Chen. Iot service clustering for dynamic service matchmaking. *Sensors*, 17(8) :1727, 2017.

RÉSUMÉ

Aujourd'hui, l'Internet des Objets permet de connecter des milliards de dispositifs hétérogènes, qui génèrent un grand nombre de services. Cependant ces objets ont des contraintes en ressources, en particulier une limitation de la taille de la mémoire et de la batterie. Ces contraintes et la prolifération du nombre d'objets connectés affectent considérablement la découverte de leurs services. Les divergences des caractéristiques exigent de nouvelles méthodes intelligentes afin d'assurer la découverte de services. Les solutions heuristiques deviennent obsolètes ou impuissantes pour satisfaire la requête d'un l'utilisateur, d'où une recherche de nouvelles méthodes devient obligatoire. Parmi ces solutions, nous avons celles basées sur les modèles bio-inspirés. Dans ce contexte, pour minimiser le nombre de sauts effectués, augmenter le taux du succès, et pour un meilleur passage à l'échelle, nous proposons deux approches bio-inspirées pour la découverte de services, elles reposent sur une architecture décentralisée. La première consiste à adapter la métaheuristique ACA. Une deuxième consiste à adapter la métaheuristique GWO. Les résultats expérimentaux montrent que les métaheuristiques proposées offrent de meilleures performances.

Mots clés : Découverte de services ; Internet des Objets ; Ant Colony Algorithm (ACA) ; Gray Wolf Optimizer (GWO) ; Approche décentralisée.

ABSTRACT

Today the Internet of Things connects billions of heterogeneous devices which generate a large number of services. However, these objects have resource constraints, in particular a memory size and battery limitation. These constraints and the proliferation of the number of connected objects significantly affect discovery of their services. Divergent features require new, intelligent methods to ensure service discovery. Heuristic solutions become obsolete or powerless to satisfy a user's query. Hence the need for an appropriate service discovery mechanisms to search services. Among these solutions, we have those based on bio-inspired models. In this context, to minimize the number of hops performed, increase the success rate and improve scalability, we suggest two bio-inspired approaches for service discovery; they are based on a decentralized architecture. The first consists in adapting the ACA metaheuristics. The second consists in adapting the GWO metaheuristics. The experimental results show that the proposed metaheuristics provide better performance.

Key words : Service discovery; Internet of Things; Ant Colony Algorithm (ACA); Gray Wolf Optimizer (GWO); Decentralized approach.

ملخص

اليوم، يتيح إنترنت الأشياء توصيل مليارات الأجهزة غير المتجانسة، والتي توفر عددًا كبيرًا من الخدمات. إن هذه الأشياء لديها قيود في خصائصها ولا سيما الحد من الذاكرة والبطارية تؤثر هذه القيود وانتشار عدد الأشياء المتصلة بشكل كبير على اكتشاف خدماتها. تتطلب الاختلافات في الخصائص طرقًا جديدة وذكية لضمان اكتشاف الخدمة. تصبح الحلول الاستكشافية قديمة عاجزة عن تلبية طلبات المستخدم، وبالتالي يصبح البحث عن طرق جديدة إلزاميًا. من بين هذه الحلول، لدينا تلك التي تعتمد على الخوارزميات. في هذا السياق، لتقليل عدد القفزات التي يتم إجراؤها، وزيادة معدل النجاح، ولتحسين قابلية التوسع، تم اقتراح طريقتين لمعالجة هذه المشكلة واللتان تعتمدان على بنية لامركزية، الأولى هي تكييف خوارزمية مستعمرة النمل (ACA)أما في المساهمة الثانية نقترح تكيف خوارزمية الذئاب الرمادية (GWO) .

أظهرت النتائج التجريبية أن الخوارزميات المقترحة توفر أداءً أفضل لإكتشاف خدمات.

كلمات البحث: اكتشاف الخدمات، انترنت الأشياء، خوارزمية مستعمرة النمل (ACA)، خوارزمية الذئاب الرمادية(GWO)، النهج اللامركزي .