Improving Shared Access to Cloud of Things Resources

Ahmed Salim Al Rawahi

School of Science and Technology

A thesis submitted in partial fulfilment of the requirements of Nottingham Trent University for the degree of

Doctor of Philosophy

October 2019

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements. This dissertation contains fewer than 80,000 words including appendices, bibliography, footnotes, tables and equations and has fewer than 150 figures.

I dedicate this thesis to my parents, wife and children.

Acknowledgements

This work was carried out during the years 2016-2019 at Nottingham Trent University, UK, at The School of Science and Technology.

I am deeply grateful to my supervisor Prof. Ahmad Lotfi, for his unlimited supervision, support and encouragement. The work presented in this thesis would hardly have been completed without his supervision. I owe my warmest gratitude to my former supervisor and current external advisor, Dr. Kevin Lee, for his technical supervision, permanent guidance and academic career coaching. His continuous support helped me significantly in improving my research knowledge, skills and scientific writing. I have greatly benefited from both supervisors dedication, knowledge and professionalism.

I would like to express the deepest appreciation to my wife, Iman, who has provided me with endless emotional, moral and life support throughout my postgraduate studies. Further thanks go to my son, Almuneeb and my daughter, Maab, for their childish encouragement to finish my PhD earlier.

Special thanks go to my parents and other family members, who always keep me in their prayers and provided me with considerable emotional and moral support.

I owe a very important debt to Dr. Abu Baker Elbaudi for his invaluable feedback on my work, his non-academic support at the university and outside the university. I would like also to thank Mr. Pedro Baptista Machado for facilitating some of the experimental setup.

I would like to deeply thank Ministry of Higher Education, Oman, for the full scholarship (number PGE023347) they awarded me. Without such financial support, I would not be able to pursue my PhD.

Abstract

Cloud of Things (CoT) is an emerging paradigm that integrates Cloud Computing and Internet of Things (IoT) to support a wide range of real-world applications. Resource allocation plays a vital role in CoT, especially when allocating IoT physical resources to Cloud-based applications to ensure seamless application execution. Due to the heterogeneity and the constrained capacities of IoT resources, resource allocation is a challenge. This complexity leads to missing/limiting shared access to the IoT physical resources and consequently lessen the reusability of the resources across multiple applications. This issue results in, 1) replicating IoT deployments making them expensive and not feasible for many prospective users, 2) existing IoT infrastructures are over-provisioned to meet the unpredictable application requirements in which resources may be significantly underutilised, and 3) the adoption of CoT is slowed.

Improving shared access to CoT resources can provide efficient resource allocation, improve resource utilisation and likely to reduce the cost of IoT deployments. Existing solutions include small-scale, hardware and platform-dependent mechanisms to enable or improve shared access to IoT resources. The research presented in this thesis considers trading CoT resources in a marketplace as an approach to improve shared access to CoT resources. It proposes a solution to Cot resource allocation that re-imagines CoT resources as commodities that can be provided and consumed by the marketplace participants.

The novel contributions of the research presented in this thesis are summarised as follows: 1) a model to describe and quantify the value

of CoT resources, 2) a resource sharing and allocation strategy called Exclusive Shared Access (ESA) to CoT resources, 3) a QoS-aware optimisation model for trading CoT resources as a single and multiple-objective optimisation problem, and 4) a marketplace architecture and experimental evaluation to verify its performance and scalability.

Publications

The following are selected fully peer-reviewed journal, conference and workshop publications that have been submitted/published as a direct result of this thesis. Publications sorted based on the most recent first.

Journal Papers

Alrawahi, Ahmed Salim, Kevin Lee, and Ahmad Lotfi. "AMACoT: A Marketplace Architecture for Trading Cloud of Things Resources." IEEE Internet of Things Journal. 2020. DOI:10.1109/JIOT.2019.2957441

Alrawahi, Ahmed Salim, Kevin Lee, and Ahmad Lotfi. "A Multiobjective Model for Trading Cloud of Things Resources." IEEE Internet of Things Journal. 2019. DOI:10.1109/JIOT.2019.2943284.

Conference Papers

Alrawahi, Ahmed Salim, Kevin Lee, and Ahmad Lotfi. "Adaptive Trading of Cloud of Things Resources" 2019 IEEE 7th International Conference on Future Internet of Things and Cloud (FiCloud). IEEE, 2019.

Alrawahi, Ahmed Salim, et al. "An evaluation of optimisation approaches in Cloud of Things resource trading." 2018 IEEE 6th International Conference on Future Internet of Things and Cloud (FiCloud). IEEE, 2018.

Alrawahi, Ahmed Salim, Kevin Lee, and Ahmad Lotfi. "Trading of cloud of things resources." 2017 ACM 2nd International Conference on Internet of things, Data and Cloud Computing. ACM, 2017.

Workshop Papers

Al Rawahi, Ahmed Salim, et al. "Enabling exclusive shared access to Cloud of

Things resources." ACM 2018 Workshop on Theory and Practice for Integrated Cloud, Fog and Edge Computing Paradigms. ACM, 2018.

Contents

Nomenclature

$3G$ The Third Generation of Wireless Mobile Telecommunications Technology.

$4G$ The Fourth Generation of Wireless Mobile Telecommunications Technology.

A Set of CoT Applications

$AMACoT$ A Marketplace Architecture for Trading Cloud of Things Resources

AU Aggregated Utility Function

$B2B$ Business-to-Business

$B2B2C$ Business to Business to Consumer

$B2C$ Business-to-Consumer

BH Basin Hopping Algorithm

$BPMS$ Business Process Management Systems

C Group of Consumers

$C2C$ consumer-to-consumer

$CIaaS$ City Infrastructure as a Service

$CMA-ES$ Covariance Matrix Adaptation Evolution Strategy Algorithm

CPU A Central Processing Unit

$CPaaS$ City Platform as a Service

CS Total cost of resources

$CSaaS$ City Software as a Service

CoT Cloud of Things

Cv Resource Coverage

DE Differential Evolution Algorithm

DoS Denial of Service

E Energy consumption of allocated resources

ESA Exclusive Shared Access

$ESACoT$ Exclusive Shared Access to Cloud of Things Resources

Ep Initial power supply of a resource

Er Estimated power consumption requested by a consumer

Et_{max} Maximum Transmission Power of a Resource

F_t Fault tolerance of allocated resources

$GDE3$ The Third Evolution Step of Generalised Differential Evolution Algorithm

HV Hyper-volume Indicator

I/O Input/Output

$IBEA$ Multi-Objective Indicator-Based Evolutionary Algorithm

$IaaS$ Infrastructure as a Service

IoT Internet of Things

L_{ij} The latency between consumer i and provider j

M Smart Object as a Service

$MOEA/D$ A Multiobjective Evolutionary Algorithm based on Decomposition

M_g The marketplace profit

$NP-hard$ Non-Deterministic Polynomial-Time Hard

$NSGA2$ Non-dominated Sorting Genetic Algorithm II

$NSGAIII$ Non-dominated Sorting Genetic Algorithm III

OS Operating System

P Group of Providers

PR Provider Profit

PSO Particle Swarm Optimisation Algorithm

$PaaS$ Platform as a Service

$Pyro4$ Python Remote Object v4

QoS Quality of Service

R Set of Resources

RA Set of Resource Attributes

$RFID$ Radio-Frequency Identification

RQ Set of Requests

R_t Response time of allocated resources

$SAaaS$ Sensing and Actuation as a Service

$SLAs$ Service Level Agreements

$SOaaS$ Smart Object as a Service

$SPEA2$ Strength Pareto Evolutionary Algorithm 2

S_o Aggregated QoS objective function

$SaaS$ Software as a Service

T_{ij} The latency between resource and application.

V Set of constraints

$WSNs$ Wireless Sensor Networks

WoT Web of Things

ΔRt The difference between the current response time after failure and the average response time $avg(Rt)$

βRt The current response time after failure.

ac Available resource components

b A bid from a consumer

cm Marketplace commission

cp The total capacity of a provider

cr Communication reliability during faliures

dl_ij The distance between a requested location of a resource and its actual location.

mc the marketplace charges

mu Multiple communication interfaces

pt The proprietary technologies of the provider

pu the requested resource utilisation time

py Provider's policy

rc Cost of a Resource

rc Cost of a resource

rp Provider reputation

rr Redundant or standby resources

s Sensing Range of a Resource

se The security requirements of an application

t Requested Lease Time of a Resource

t_{ack} The time of receiving an acknowledgement from a provider j

t_{qd} Estimated queuing and transmitting delays

t_{start} Time when consumer i submits a request

uc Actually utilised resource components

w Weighting factor

Acronyms

PPC Particles per cell

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview

Cloud Computing and Internet of Things (IoT) have evolved to meet the requirements of many real-world applications. Many of these requirements can not be fulfilled by using either technology separately. In order to fulfil such diverse requirements, the integration of Cloud Computing and IoT is emerging as a new paradigm called Cloud of Things (CoT). While CoT provides various benefits for an increasing number of applications, the potential of CoT is not yet realised due to the challenges in sharing and reusing IoT physical resources across multiple applications. IoT physical resources are still computationally limited and cannot be shared as other Cloud resources yet.

There is a limited number of existing solutions that aim to improve shared access to CoT resources. The solutions include developing operating systems [7] and middleware [8] that enable multiple access to IoT devices. Most of the proposed approaches are limited to small-scale and hardware-specific resources without considering IoT and Cloud Computing integration as well as the heterogeneity of IoT resources.

The remainder of this chapter is organised as follows: Section 1.2 describes CoT in details, Section 1.3 establishes the need for shared access to CoT resources, Section 1.4 introduces the proposed approach, aims and objectives of this research are presented in Section 1.5, Section 1.6 discusses the research contributions and

Section 1.7 provides the thesis structure.

1.2 Cloud of Things

Cloud Computing and the IoT have evolved and developed independently from each other. The following sections describe Cloud Computing, IoT and CoT.

1.2.1 Cloud Computing

Cloud Computing is a model of offering computing capabilities as metered services over the Internet rather than physical products. It is widely adopted in many applications such as e-learning, e-business, health, logistics and manufacturing. Cloud's offering is characterised to be provisioned on-demand elastically, ubiquitously accessed and pooled as a part of shared resources [21].

Cloud Computing is delivered under one of the following traditional service models. Software as a Service (SaaS), Platform as a Service (PaaS) and Infrastructure as a Service (IaaS) [15]. SaaS refers to the delivery of software to consumers through the Internet without the need to install or maintain software on users' local infrastructure. Similarly, PaaS provides the consumers (e.g. software developers) with the capability to implement and deploy their applications into the Cloud without maintaining the required infrastructure. IaaS refers to the delivery of virtualised physical capacities (e.g. storage, processing, networking) via the Internet where the consumers do not require to install or maintain traditional data centres [104].

Besides the service model, Cloud Computing is deployed under one of the following models. Public Cloud, Private Cloud, Community Cloud and Hybrid Cloud. Public Cloud is deployed to be accessed and utilised by the general public, while private Cloud is exclusively deployed for private use. Community Cloud refers to a Cloud that is exclusively deployed for a community of consumers with a shared interest. Hybrid Cloud is a combination of two or more of the above-mentioned models [104].

Along with its technical value, Cloud Computing has a significant economic impact. Cloud resources are usually provisioned on-demand automatically or with minimal human intervention. This reduces the cost of resources management, enables pay-per-use only and reduces upfront investment on new computing infrastructure [28]. These economic and technical features attracted large deployments globally that become the current trend for many businesses. It is estimated that the value of the global public cloud market to reach $331 Billion through 2023 and expect continuous growth afterwards [34].

1.2.2 Internet of Things

The IoT is a recent and less mature technology than Cloud Computing. IoT is described as world-wide interconnected and interactive objects (things) that can be identified, monitored and controlled over the internet [24, 28]. The IoT is heavily dependent on the development of sensor networks. Sensor Networks are composed of tiny computers known as motes with embedded CPUs, low-cost sensors and low-power radios [138]. These motes often form wireless networks that are capable of sensing the physical world. Sensor networks collect data from sensors, collate, aggregate and transfer this in forms of data streams to back-end computers for processing. Those streams of data are used to support IoT applications.

IoT applications spread over many domains such as logistics, transportation, defence, public safety, home automation, industrial control and environmental monitoring. The deployment of IoT has rapidly increased in the last few years. It is estimated that the potential economic impact of IoT to be between $4-11 trillion by 2025 [75]. It is predicted that IoT will need five to ten years for mainstream deployments with over 20 billion connected things in 2020, increasing from 6.4 billion in 2016 [12].

IoT applications can be categorised into two groups as follows: 1) Latency-sensitive, and 2) None latency-sensitive applications. Latency- sensitive applications (e.g. military, emergency services) that benefit from the wide coverage of IoT resources in monitoring their operations [108], and None time-sensitive applications (e.g. marketing, planning) utilise a widely distributed IoT resources to produce big data that can be analysed and processed to aid long-term decision-making [139].

1.2.3 Cloud of Things

Despite the recent advances of Cloud Computing and IoT in terms of their computing capabilities, both technologies have been pushed to their limits by new real-world scenarios. The physical scope of Cloud Computing is limited because it is focused on data-centres and does not interact with the physical world. The main limitation of IoT is its constrained computational resources. This raises new challenges in which new applications are unlikely to be supported by separate deployments of either Cloud or IoT. As a result, considerable research efforts argue for a new paradigm that integrates both technologies to support a wide range of new applications [48, 95].

Although there is an increasing focus on the new paradigm, there is no standard name for it yet. Various names found in the literature include Internet of Things Cloud (IoT Cloud) [116], Cloud of Things (CoT) [1] and web of things (WoT) [145]. CoT will be used throughout this thesis because it is the most commonly used term. Furthermore, there is no standard definition or description of CoT. In this thesis, CoT refers to the integration of Cloud and IoT to form a new distributed paradigm of connected IoT technologies to Clouds via the internet to provide new services [29]. This also includes extending Cloud's coverage to support more distributed and flexible real-world applications (e.g. environmental monitoring, emergency) that are far away from Cloud data centres [94], as well as where IoT is utilising Cloud-based resources such as communication, processing and storage capabilities to extends its limited resources.

Despite the increasing interest in integrating Cloud Computing and IoT, there are still many open challenges including security, interoperability and resource management [9, 42, 101]. One of the significant issues related to resource management is how efficiently CoT resources can be shared, especially the constrained IoT physical resources. The complexity of this issue resides here for two reasons. The first is due to the heterogeneity of IoT resources which is difficult to quantify their value leading to the involvement of multifaceted variables and decisions. The second is due to the constrained nature of IoT resources in terms of computing capabilities which is challenging to enable an efficient sharing mechanism to the IoT physical resources. The following section discusses this issue in details.

1.3 Shared Access to CoT Resources

Prior to Cloud Computing, computing resources were dedicated to their users. In Cloud Computing, resources are shared by multiple users across different levels over a network or the Internet (e.g. network, host, application) [27]. Resource sharing mechanisms in Cloud Computing matured over time while approaches to sharing IoT resources are still emerging. One of the major differences between the two types of resources is their capabilities. Cloud resources are usually hosted in powerful large-scale data-centres to provide virtually unlimited, elastic and on-demand computing resources. Conversely, IoT resources are widely distributed across the application area with constrained computational and power resources that make it challenging if not impossible to share those resources among multiple users concurrently.

A solution to this challenge is to enable shared access to IoT resources [19]. In this context, shared access refers to the authorised access of multiple users to utilise an IoT resource simultaneously or non-simultaneously [170]. The complexity of shared access to IoT resources comes from IoT dynamism. IoT dynamism involves constrained and heterogeneous resources, complex application requirements, unpredictable resource mobility, uncertain resource availability, and scalable IoT systems. These challenges lead to missing/limiting shared access to the IoT physical resources and consequently lessen the reusability of the resources across multiple applications. These also result in over-provisioned IoT infrastructure to meet the unpredictable application requirements in which resources are underutilised significantly. Another drawback is the dependency of IoT applications development on the infrastructure deployment where each IoT application requires dedicated infrastructure. This makes expensive replications of IoT deployments and makes IoT adoption infeasible to many prospective users and emerging applications.

A number of solutions emerge to improve shared access to IoT resources. The solutions include proposing new IoT architectures [19, 62, 90] and developing new

IoT operating systems and middleware [7, 8]. These solutions aim to improve the reusability of IoT resources by enabling multiple access to IoT devices. The proposed solutions are restricted to small-scale applications or benefit hardware-specific resources.

1.4 Trading CoT Resources

To improve the reusability of CoT resources, commoditising them using market-based mechanisms is proposed as an alternative approach. This approach has been successfully used in other large-scale computing infrastructures such as Cloud Computing, Grid Computing and Wireless Sensor Networks [54, 93]. The approach of trading CoT resources is motivated as follows. IoT deployments usually require considerable investment in hardware, software and maintenance. Such investment is not affordable to many communities, and it slows down the rate of IoT adoption [132]. For the commoditisation of CoT resources to work efficiently, access to these resources needs to be global, purchasable and efficient.

Many technical and business benefits also motivate the commoditisation of CoT resources. Small and medium vendors are likely to invest in IoT commodities reducing the chance of monopoly and market dominance by large vendors similar to the Cloud market. Competition in the emerging market is expected to improve providers' Service Level Agreements (SLAs). It is also expected to enable hardware and software innovations when a large number of software developers and hardware makers respond to the requirements of the CoT market. The commoditisation of CoT resources will likely to reduce the overall costs, enable sharing and reusing of IoT resources, motivate for new services and applications.

One approach to achieving those goals is the creation of a non-vendor marketplace that potentially can automate the trading between CoT resources and CoT applications. A CoT marketplace can improve shared access to CoT resources by providing efficient resource allocation and deal with the complex issues present in the CoT. The research presented in this thesis proposes a solution that re-imagines CoT resources as commodities rather than as organisational assets. It considers the business model of a marketplace whereby consumers request access (lease) to providers' resources that is priced using Cloud pay-per-use pricing model.

1.5 Aims and Objectives

The work presented in this thesis aims to improve shared access to CoT resources by proposing a novel market-based approach to commoditise CoT resources. In order to achieve the aim of this research, the following objectives are identified:

- Investigating the market-based mechanisms of commoditising CoT resources with the focus on trading physical CoT resources.
- Exploring the potential of various optimisation algorithms in trading CoT resources. Optimisation algorithms replace traditional auctioneers to map resources to requests and perform resource allocation and scheduling.
- Describing CoT resources generically to quantify their value.
- Formulating the problem of trading CoT resources as an optimisation problem and proposing the required objective functions.
- Designing and developing a CoT marketplace system architecture to validate the optimisation-based approach and to simulate the trading environment.
- Proposing a model to support Quality of Service (QoS) requirements of commoditised CoT resources.
- Measuring and comparing the performance of the evaluated optimisation algorithms as well as the proposed system architecture to assess the feasibility of the proposed approach.

The identified objectives of this thesis will support answering the following research questions:

- Can market-based mechanisms improve the shared access to CoT resources?

- Are optimisation algorithms effective and efficient in trading CoT resources to improve their shared access?

1.6 Research Contributions

The following are the main contributions of this thesis:

- Presenting a state of the art literature review on shared access mechanisms to CoT resources.
- Proposing a novel description model for CoT resources that include all required vocabularies to quantify and monetise their value.
- Introducing a novel shared resource access and allocation strategy called Exclusive Shared Access to CoT resources to enable shared access to computationally constrained CoT resources.
- Developing a novel marketplace architecture for trading CoT resources referred to as AMACoT.
- Proposing a novel QoS optimisation-based model to optimise QoS requirements while trading CoT resources.
- Evaluating the proposed models and architecture experimentally to validate their feasibility and efficiency.

1.7 Thesis Outline

This thesis is structured as follows:

Chapter 2: Literature Review. This chapter presents a state of the art review on Cloud of Things and its existing challenges. The focus is on integrating Cloud Computing and IoT, shared access to CoT resources, trading CoT resources, resource allocation in CoT and QoS models in CoT. This chapter closes with a comprehensive gap analysis that discusses the existing gap in the literature and how this research is filling that gap.

Chapter 3: Trading of Cloud of Things Resources. This chapter provides detailed background on the motivations for trading CoT resources, discusses the requirements for CoT marketplaces, presents the proposed approach of trading CoT resources, introduces ESACoT strategy and explains how it works and finally provides several use cases for the proposed approach.

Chapter 4: A Multi-Attribute Description Model for CoT Resource. The chapter presents a generic description model for CoT resources. It introduces the necessary vocabularies needed for trading CoT resources and defines them. It also describes how the proposed model quantifies the value of CoT resources based on their properties.

Chapter 5: AMACoT: A Marketplace Architecture for Trading Cloud of Things Resources. This chapter presents the design and implementation of AMACoT. The experimental evaluation performed includes system performance verification and the evaluation of optimisation algorithms used by the system. Experiments evaluate the optimality of trading CoT resources solutions in terms of resource cost, resource utilisation, provider lock-in and provider profit. A threat analysis for the proposed architecture is also conducted to identify the potential threats and vulnerabilities of the system.

Chapter 6: A Multiobjective QoS Model for Trading Cloud of Things Resources. The research presented in this chapter proposes a multiobjective model to optimise trading of CoT resources based on five QoS objectives. The objectives optimised are resource cost, energy consumption, response time, fault tolerance and resource coverage. A comprehensive single-objective, bi-objective and multiple-objective evaluation are conducted to validate the performance of the proposed model.

Chapter 7: Conclusions and Future Work. This chapter provides concluding remarks of the work discussed in the thesis by reiterating the main contributions and presents some insights for future work on improving shared access and trading CoT resources.

Chapter 2

Literature Review

2.1 Introduction

Cloud Computing and IoT have evolved independently to support a wide range of real-world applications. Due to the limitations of both paradigms, considerable research argues for a new paradigm that integrates both technologies [95]. Both technologies are viewed as complementary to each other, where each technology helps expanding the capabilities of the other. One of the main issues inherited from IoT is the missing/limited shared access to IoT physical resources. This chapter presents the literature review on shared access to CoT resources in order to understand this research problem, the existing solutions and their limitations, and the solution presented in this thesis.

The remainder of this chapter is structured as follows: Section 2.2 reviews the motivation and approaches to integrating Cloud Computing and IoT, a revision on resource allocation techniques in CoT is presented in Section 2.3, Section 2.4 describes the market-based mechanisms used to commoditise CoT resources, Section 2.5 investigates the QoS requirements and parameters for CoT, a gap analysis is provided in Section 2.6 to analyse the limitations of existing solutions, the proposed approach in this research and to conclude this chapter.

2.2 Integrating Cloud Computing and Internet of Things

Cloud Computing and IoT continue to emerge as revolutionary paradigms to support a wide range of real-world scenarios. Cloud Computing delivers software, hardware and platforms as services over the internet. These services are metered and billed by pay-per-use pricing models. Cloud Computing is offered under one of three service models. SaaS model provides online access to software. PaaS model provides an environment for developing and deploying applications. IaaS model provides access to physical computing resources as virtualised services. Clouds are also deployed in one of the following deployment models. Public Clouds are accessible online by the public, while Private Clouds are only accessible within an organisation. Community Clouds enable shared-access to individuals or organisations in a community with a shared interest (e.g. Academic, Government). Hybrid Clouds use two or more deployment models to integrate their services [104].

IoT is a paradigm that enables heterogeneous physical objects (things) to be interconnected to monitor and control a wide range of real-world events [171]. These may include weather, environmental, traffic, and health. Things can be devices, people, machines, vehicles and many other objects. Things may connect and interact with each other at a global-scale network (e.g. the internet). IoT relies on technologies including RFID, Bluetooth, 3G/4G and WiFi [97]. Examples of application domains include smart homes, healthcare, agriculture, transportation and military [9, 108].

Both technologies still have open challenges despite their rapid advances. Traditional service models of Cloud Computing have limited interaction with the physical world. Thus, Cloud Computing is left with limited scope and flexibility [94]. The main limitation of IoT is the limited computing capabilities of its resources (e.g. network, CPU, memory, storage) [42, 112]. The convergence of both technologies is therefore considered as a potential solution to overcome technical issues and also as an enabler for a wide range of new emerging applications [2]. The new emerging paradigm is commonly called Cloud of Things (CoT) [2, 5, 43, 44, 98, 121, 129, 157, 159].

2.2.1 Motivations for Cloud of Things

Considerable research has motivated the integration of Cloud and IoT to improve the limitations of each technology. Motivations can be categorised into functional properties, computing capabilities and business values.

2.2.1.1 Functional Properties

A significant amount of reviewed literature focuses on the limitations and missing properties of IoT resources/systems that hinder its deployments. This involves limited or lacking interoperability, scalability, flexibility, reliability and availability [24, 42]. Security is also challenged to a great extent [152]. These limitations are considered inevitable due to the high heterogeneity of IoT in terms of hardware, software, platforms and communication protocols deployed [144]. Those functional properties are considered as an integral part of any recent Cloud offerings [95]. IoT would, therefore, benefit significantly from the integration with the Cloud by improving its functionalities [157].

2.2.1.2 Computing Resources

Recent research is extensively focused on one or more of limited IoT capabilities. This includes limited energy resources, basic computation capabilities, limited or no storage available and limited communication capacity.

- **Constrained Energy and Computation:** IoT devices are usually powered by batteries or have constrained power-supply. This limits the computational capabilities available for IoT devices. Thus, IoT nodes collect data and transfer it to more powerful back-end nodes for extensive processing and analysis. These limitations cause two issues for IoT; 1) real-time analysis and responding to some critical scenarios (e.g. emergency) are either not possible or very limited, 2) scalability to meet dynamic application requirements with poor processing resources is very challenging [87]. This may answer why IoT deployments in time-sensitive applications such as emergency, security and military scenarios are very challenging [52]. Cloud can lift the bar for IoT by acting as its back-end aggregator and processor. Thus, it enables scalability and real-time processing for more complex real-world implementations [24, 123, 169].

- **Limited Storage:** Data produced by IoT devices is characterised by its size (volume), types (variety) and generation frequency (velocity) [178]. IoT by nature is a big data producer but with very limited or no storage capacities [135]. This motivates the integration with the Cloud where Cloud offers virtually unlimited, on-demand, cost-effective and scalable storage capacities to accommodate IoT storage requirements [118, 137]. This would result in new technical and business opportunities as well, including ubiquitous access to data, Cloud-level security [148, 152] and the ability to share IoT devices and data with third parties [131].

2.2.1.3 Business Values

Along with the technical aspects that motivate integrating Cloud with IoT, business benefits play a crucial role in motivating the integration of both paradigms. This is because the current Cloud business model reduces the investments in IT infrastructure and the operational costs while IoT does not. The deployments of IoT remain costly despite the increasing demand for IoT applications and the reduction in software and hardware costs. IoT deployments currently require significant investment to deploy the IoT infrastructure, to manage and maintain the IoT infrastructure, and to develop the IoT applications [7]. This makes IoT adoption unfeasible to many prospective users and emerging applications, resulting in slow adaption rate of IoT [132]. Furthermore, the business risks of managing Cloud resources are shifted to Cloud providers from the end-users, motivating the integration of both technologies further [24]. These business motivators help IoT by improving its trustworthiness, business value and reducing costs to attract new deployments and users.

Several academic and commercial examples consider business values as the main motivation for the integration of Cloud and IoT. Authors in [132] propose a model that creates a trading-based value for IoT sensing resources. The presented model aims to support an emerging Cloud service model called

"Sensing as a Service". A survey presented in [31] investigates the challenges of mobile Cloud-based Business Process Management Systems (BPMS) for IoT. It analyses the surveyed challenges to drive solutions for future cloud-based IoT business systems. ClouT project is proposed in [159] to enable the stakeholders of the smart cities to overcome business-related challenges, including energy management and economic growth. The project integrates IoT services in the Cloud to provide the needed CoT infrastructure and platforms for the city stakeholders to develop their CoT applications. Commercial solutions include Sensor-Cloud [70], Google Cloud IoT [68] and ThingsSpeak [72].

2.2.2 Integration Approaches

Although there are many attempts to integrate Cloud and IoT, there is still no standard approach, and the process is always complex and challenging [161]. Different approaches can achieve the integration of Cloud and IoT. The most common approaches are categorised in [28] into minimal integration, partial integration and full integration. The three approaches are described as follows:

- **Minimal integration:** In this approach, Cloud has no real changes to its service or deployment models. An IoT platform or middleware is deployed into Infrastructure as a Service Cloud or Platform as a Service Cloud to utilise the Cloud services [23]. Examples of Cloud services utilised by IoT using this approach include virtualisation, data processing, data analysis and Cloud storage [28]. Existing solutions that demonstrate this approach are proposed by [46] and [58]. Another example is a new addressing scheme based on IPv6 that is proposed in [177] to enable IoT and Cloud integration. The scheme is implemented to integrate IoT sensors with SaaS Cloud.

- **Partial Integration:** Changes to the deployment of both Cloud and IoT is performed to some extent in this approach to achieve a higher level of integration compared to the minimal integration approach. The IoT middleware or platform is deployed into the Cloud to provide new service models based on the abstractions of IoT things [23]. Examples for new service models resulted from this integration approach are Sensing as a

Service (SaaS) [18, 133, 172], Sensing and Actuation as a Service (SAaaS) [44,50] and Smart Object as a Service (SOaaS) [84].

- **Full Integration:** This approach aims to achieve the highest level of integration between Cloud and IoT by extending the traditional Cloud service models (SaaS, PaaS, IaaS) to include the functionalities of IoT resources. This would provide IoT services as an integral part of Cloud services [23, 111]. Proposed solutions based in this approach are City Infrastructure as a Service (CIaaS), City Software as a Service (CSaaS) and City Platform as a Service (CPaaS) [53].

Existing deployment techniques of sharing Cloud-based IoT resources can be categorised into service-oriented approaches and software-oriented approaches. Service-oriented approaches are mainly focused on sharing data and/or the virtualised IoT resources as services [5,98]. The shared data is sent from various distributed IoT resources to a back-end Cloud for further processing by multiple users. Similarly, virtualised IoT services are built on the top of physical IoT resources as Cloud services where multiple users utilise the virtualised resources but not the physical ones. Software-oriented approaches focus on enabling sharing IoT resources by enabling multiple applications access to IoT devices using middleware [7,116,149].

2.3 Resource Allocation in CoT

Despite the integration efforts and the benefits that CoT promise, there are still several open challenges such as security, performance, heterogeneity and resource management [2, 152]. Many resource management problems in large-scale computing infrastructures are non-deterministic polynomial-time hard (NP-hard) [60]. This means there are no best or exact solutions to such problems in a reasonable time due to the complexity, scalability and uncertainty of users' requirements. CoT is a large-scale computing infrastructure by nature and its resource management aspects are challenging [9,101].

Resource allocation is a vital aspect of CoT resource management due to its distributed nature. Resource allocation techniques in the IoT ecosystem are still emerging. A considerable amount of research investigates resource allocation in IoT as part of other systems (e.g. Cloud Computing, CoT, WSNs). An early attempt to integrate wireless sensor networks (WSNs) and Cloud Computing is discussed and implemented in [95]. Their proposed architecture enables WSNs tasks to be offloaded to the Amazon EC2 Cloud. Another architecture is developed in [166] to integrate WSNs sensors to be integrated into the Cloud. The architecture uses dynamic proxies that host middleware to connect sensors to the Cloud. A four-layer architecture is introduced in [3] to enable Cloud-assisted remote sensing. The architecture enables consumers to collect sensory data, share collected data, utilise cloud resources and use cloud-based pricing model.

A framework is presented in [168] to enable collaboration between smart devices and Clouds. The framework uses real-world case studies to elaborate on the benefits of integrating smart devices and Cloud Computing. A scalable CoT architecture is developed in [4] along with two algorithms to discover and virtualise IoT resources. The proposed algorithms are developed to minimise the number of physical resources deployed and communication overhead. A detailed theoretical model for integrating sensors and Cloud Computing is provided in [107] to evaluate the cost-effectiveness and performance of the sensor-cloud architecture. A heuristic algorithm is presented in [20] to perform task mapping for IoT resources. The algorithm aims to achieve a fair allocation of mapped tasks among IoT nodes with consideration of IoT extension to Cloud.

Authors in [40] categorise approaches of resource allocation in IoT into three categories; namely, Cloud only approaches, IoT only approaches, and IoT Cloud approaches. IoT Cloud approaches focus on integrating IoT resources into a Cloud as part of its services. These approaches aim to enable on-demand provisioning of shared IoT resources via the Cloud of Things. The scope of this section is limited to the discussion of IoT Cloud approaches only.

A consensus-based framework is developed in [134] to allocate IoT resources in the Cloud. The goal of the allocation algorithm is to improve the lifetime of the connected resources. A three-tier CoT architecture is proposed along with the development of multi-objective scheme to optimise task allocation in CoT [98]. The scheme aims to minimise energy consumption and latency.

Another three-tier architecture is designed in [169] to enable sharing of Cloud resources in IoT vehicular networks where vehicles are considered the Things of IoT. The proposed system intends to reduce service dropping rates. A resource allocation algorithm is proposed in [115] to enable Cloud providers optimising the throughput, occupancy and utilisation of the IoT requests.

An architecture that integrates sensors and Cloud Computing for military operations is developed and implemented in [108]. Resource allocation in the proposed architecture is based on user prioritises to improve the performance and availability of resources for priority users in military operations. A model is presented in [162] to cooperate between the airborne sensor network and back-end Cloud. The model applies heuristics to minimise the travel time of the drones and failures in meeting their deadlines.

2.4 Commoditisation of CoT Resources

A solution to the resource allocation problem in CoT is to enable efficient resource sharing. One of the main obstacles to achieving this goal is the lack of support to share CoT resources. An emerging trend argues for market-based mechanisms to trade resources in large-scale infrastructures similar to CoT (e.g. Grids, Clouds, WSNs, Vehicular Networks) [45, 93]. Market-based mechanisms for trading Cloud and IoT resources are intensively studied. Cloud-based approaches are more mature than the ones dedicated to IoT that are still emerging. As CoT relies heavily on IoT resources, this section focuses on both IoT and CoT related market-based mechanisms by reviewing the recent literature in the following sections:

2.4.1 IoT Trading Mechanisms

A conceptual model is proposed in [132] to argue for the creation of trading-based value for IoT resources. The model aims to enable sharing and reusing IoT resources by trading them similarly as Cloud resources. The design and implementation of a market-based model are presented in [91]. The three-tier model considers the Cloud as a broker for IoT resources where resource allocation is formulated as a multi-objective optimisation problem. It aims to allocate traded resources with the minimum response time of the requests, minimum energy consumption of the system and maximum profit of the broker. A federation model for Cloud IoT providers is proposed in [49] to implement market-based mechanisms. The goal of the proposed model is to satisfy providers' requirements and improve the rate of resource utilisation of assigned tasks.

A combinatorial auction algorithm is developed in [33] to allocate CoT resources. The objective of the proposed algorithm is to maximise the providers' profit and the rate of job completion. A reputation-based framework for CoT architectures is presented in [78]. The framework uses an auction procedure to select physical resources for sensing tasks and made payments by users. An auction model is designed in [85] to map CoT computation resources to the consumers. The model targets performance improvement when allocating distributed IoT resources. Another auction-based algorithm is developed in [55] to support resource allocation in CoT environments. The proposed algorithm aims to maximise the providers' profit while maintaining their capacity constraints.

Several market-based approaches are depicted and discussed in [122]. The study also proposes a game-theory based model to study the pricing of two IoT sensing services. Another set of market-based mechanisms is investigated in [174]. This includes an analysis of IoT marketplace incentives, service patterns, information timeliness and social impacts. Two bidding algorithms to support IoT resource trading are introduced in [141]. The first algorithm aims to maximise the provider's revenue while the second is to lock the highest bid in the recurrent auction. Both algorithms are intended to protect the marketplace from collapsing in specific trading scenarios.

A semantic matching model for IoT marketplace is presented in [32]. The model facilitates the matching process between providers' offerings and consumers' requests in a marketplace of the BIG IoT project [73]. A composition mechanism for IoT offerings has been presented in [160]. The approach is based on a web-semantic model to describe IoT things and services for trading.

A feedback mechanism has been proposed in [128] to support IoT data marketplaces. The blockchain-based approach enables consumers to rate the providers who have to maintain and improve their reputation based on feedback received. A marketplace model has also been proposed in [110] to support the quality of trading. The proposed marketplace introduces a credibility rating mechanism for providers based on the quality of their data. Another IoT marketplace model based on Stackelberg game is presented in [175] to model the trading processes in IoT environments. The model aims to minimise the complexities for IoT consumers while trading with IoT providers.

A blockchain-based automated payment system is proposed in [154] to support automation in IoT trading. The system uses Ethereum contracts to automate payments without a need to intermediaries. An architecture is presented in [119] to support Service Level Agreements (SLAs) while trading IoT data. The architecture uses three criteria model to improve satisfaction, payment and the SLA. Security mechanisms for protecting the IoT data marketplace is studied in [22]. The study also proposes authentication and authorisation model to control access to the traded resources.

2.4.2 IoT Marketplace Architectures

Alongside with the market-based mechanisms for trading Cloud and IoT resources, the concept of a marketplace for IoT-related resources is gaining prominence. This section discusses both commercial and non-commercial solution for trading IoT-related resources.

A marketplace architecture for trading IoT data in real-time is proposed in [88]. The architecture enables providers to offer their IoT data streams for consumption by IoT applications. The proposed work differs from others by implementing the architecture and addressing various aspects, including scalability and compatibility. A generic Cloud-based marketplace architecture is proposed in [77] to enable trading of IoT deployments. The architecture addresses the Cloud-IoT integration and vendor lock-in issues. A marketplace for IoT resources is introduced in [25] as part of the broader architecture of the IoT ecosystem. The trading model of IoT information and functions is presented as a solution with five IoT interoperability patterns. The patterns are cross-platform access, cross-application domain, platform independence, platform-scale independence and higher level service facades.

A different approach is taken in [136] to establish a decentralised marketplace for IoT data based on blockchain technology. The proposed marketplace architecture uses simple contracts to simplify the trading of IoT data among participants. IDMoB is another decentralised marketplace for IoT data built on blockchain [127]. The marketplace enables trading of IoT data for none time-sensitive IoT applications. DataBroker DAO is another blockchain-based marketplace implemented to trade IoT data [120]. IoT generated data is traded using smart contracts among buyers and sellers via the Ethereum network.

A decentralised peer-to-peer marketplace for IoT data is presented in [37]. The proposed architecture differs from others by adopting fog computing model and blockchain technology. IoT data is prepared (e.g. filtered, processed) at a fog nodes layer while traded directly among the marketplace participants at the application layer. Another decentralised architecture is designed in [109] with the focus of IoT traffic metering and contract compliance. The presented system aims to improve transparency, fairness and interoperability while reducing the cost. The study also conceptualises a tracking model for the traded IoT data flows between IoT and Cloud Computing.

Various commercial solutions for trading IoT resources exist, including ones from large vendors. For instance, PTC marketplace is a platform that monetises access to PTC IoT applications and solutions [69]. Dawex [155] is another marketplace for IoT data. The marketplace is vendor-independent and open to global trading of IoT data. Terbine [71] is a global exchange for IoT data where the system obtains IoT data feeds from various resources and enables consumers to utilise them. A marketplace for IoT data and applications is available from Exosite [67]. Consumers can either consume streamed IoT data or develop their IoT applications based on reusable components from the marketplace.

2.5 Quality of Service in Cloud of Things

Quality of Service (QoS) refers to the description of the perceived performance of a particular service that can be tangible or non-tangible [14]. It measures a level of many performance qualities for a particular resource, service or application offered by a provider or requested by a consumer. In CoT, this may include reliability, cost, reputation [83], availability, response time and throughput [117]. Managing QoS is vital to resource allocation, especially in trading CoT resources environments.

Defining appropriate QoS attributes for a new domain plays a key role in supporting QoS in that domain. In the Cloud, there are SLAs that aim to define QoS parameters. Performance, dependability and cost are presented in [147] to measure QoS of online Cloud services. Various QoS metrics are considered in designing a Cloud SLA model [146]. These include performance metrics (e.g. response time), availability metrics (e.g. rate of completed service requests), reliability metrics (e.g. recovery time from failure) and cost metrics (e.g. service cost). There are also several attempts at QoS in Cloud, with a particular focus on supporting different workloads and capacities [14]. A QoS-aware framework is proposed in [126] to prevent Cloud consumers from being locked-in by specific providers. Supporting QoS in virtualisation-based environments is particularly challenging, especially in trust and security-related issues [102].

For IoT, QoS-aware architecture presented in [47] with the focus on information collection and analysis of QoS aspects in the IoT system. The proposed architecture addresses various QoS requirements and parameters for IoT systems. Parameters include service time, delay, accuracy, load and priority. It also considers network QoS such as bandwidth, delay, packet loss rate and jitter. Additional QoS aspects such as resource coverage, time synchronisation and resource mobility are also addressed. IoT QoS-based service selection and scheduling models are proposed in [96] and [82]. Both models employ QoS parameters that consider relative QoS metrics to IoT in addition to the traditional QoS performance-related ones. Those include cost, power consumption, utilisation time, load and reputation of IoT services. A wide range of IoT QoS approaches is investigated in [167]. The focus of the study is on QoS across different IoT layers, including physical, deployment, link network, application, middleware and Cloud layers. It also addresses several quality factors that impact QoS in IoT such as functional stability, performance, interoperability, usability, reliability, security, maintainability and portability. Further insights on QoS for IoT are discussed in [16].

2.6 Discussion

The related work presented in this chapter covers integrating Cloud and IoT, resource allocation in CoT, commiditisation of CoT resources and QoS in both Cloud and IoT. The review aims to identify the research problem of the limited shared access to CoT resources, investigate the existing solutions, identify the limitations of the existing approaches to solve it, discuss how the work presented in this thesis resolves the research problem and how it differs from existing approaches. This section is dedicated to discuss the limitations of the literature and how the proposed approach in this thesis improve shared access to CoT resources.

Service-oriented deployment approaches of CoT may enable restricted shared access to virtualised IoT resources but not to the actual physical IoT resources. The virtualisation techniques used to integrate virtualised IoT resources into the Cloud need further investigations about their impact on the energy of the virtualised resources. Many IoT nodes are battery-powered in which virtualisation techniques may accelerate their power depletion rate that minimises the lifetime of IoT resources [113]. Software-oriented approaches are also still emerging and subject to improve heterogeneity, scalability and dynamism aspects of IoT [142].

The review presented in Section 2.3 shows that Cloud approaches are more mature than the ones used for IoT. It can also be observed that IoT resource allocation approaches are still emerging and developed for IoT when integrated with other systems (e.g. Cloud, WSNs). Existing CoT resource allocation techniques lack partially or fully the appropriate support for QoS constraints and SLAs [33]. This means resource allocation in CoT can be achieved but without meeting the QoS requirements of the applications or by violating SLAs.

The vast majority of existing approaches are merely focused on commoditising IoT data-sets/streams but not on IoT physical devices. Many of the existing CoT trading mechanisms and architectures did not take into account several important aspects including IoT resources integration, resource sharing and interoperability [106]. Further limitations of existing approaches that consider improving shared access to CoT resources revolve around small-scale and hardware-specific support mechanisms [7, 8]. These solutions fail to address the scalability requirements of CoT.

Although there has been QoS-focused research as presented in Section 2.5, including proposed architectures, in both Cloud Computing and IoT, there is a paucity of studies in QoS support for CoT. QoS-aware resource allocation techniques have been studied for Cloud and IoT separately while they are still developing for the CoT [157]. Limited support to QoS will severely impact the performance of many CoT applications that require QoS support to maintain a certain level of service quality. Therefore, QoS support is still needed for many existing CoT proposals.

The work presented in this thesis is motivated as follows. CoT is complex with large-scale computing infrastructure and heterogeneous resources. This complexity leads to limited or missing shared access to CoT resources. This limitation hinders the reusability of CoT resources and creates expensive replications of CoT deployments as each CoT application requires a dedicated CoT infrastructure. This also makes investments in CoT infeasible to many prospective stakeholders. Cloud and IoT consequently cannot yet fully utilise each other's capabilities because IoT resources cannot be shared similarly as Cloud resources.

The work presented in this thesis aims to fill the gap created by the limitations of the existing approaches discussed earlier as follows. Improving shared access to CoT resources can be achieved by using market-based mechanisms. The approach presented in this thesis re-imagines CoT resources as commodities rather than organisational assets. This approach is inspired by similar market-based mechanisms used in similar large-scale computing infrastructures such as Cloud computing [54, 74, 76, 79], Grid computing [26, 35] and WSNs [92, 156].

The following requirements/considerations are taking into account when the proposed approach is constructed to achieve improved shared access to CoT using market-based mechanisms. CoT resources need to be described generically to enable quantifying their values and monetising them. A strategy/mechanism is required to enable/improve shared access to physical IoT resources with constrained power and computing capabilities. A trading model is required to match the consumers (applications) to providers (resources). A marketplace system architecture for trading CoT resources is required to implement the trading model and proof the concept and validate the feasibility of the proposed approach. The scalability and QoS requirements of CoT applications must be taken into account.

The requirements mentioned earlier are addressed as follows: the requirements for trading CoT resources and the justification of the proposed approach are discussed in Chapter 3, a generic description model for CoT resources and shared access and allocation strategy for CoT resources are proposed and presented in Chapter 4, the trading model for CoT resource and the marketplace architecture are developed and discussed in Chapter 5, Chapter 6 presents the multi-objective QoS model to address the QoS requirements while trading CoT resources. To the best of author's knowledge, this is the first dedicated approach to trading CoT resources.

Chapter 3

Trading Cloud of Things Resources

3.1 Introduction

Motivating trading CoT resources is essential to understand the drivers for developing the proposed approach and the requirements of the solution. The research presented in this chapter provides the background for these topics and presents the methodology of the research to design a solution based on the discussed requirements.

The remainder of this chapter is organised as follows: Section 3.2 discusses the motivations for trading CoT resources, Section 3.3 presents the requirements for a solution to the problem of limited shared access to CoT resources, motivations for the proposed solution are discussed in Section 3.4, Section 3.5 describes the methodology of the work presented in this thesis, conclusions of this chapter are drawn in Section 3.6.

3.2 Motivations for Trading Cloud of Things Resources

The rapid development of the IoT has led to a large number of providers of hardware and software platforms. The costs of building and deploying IoT applications is dropping dramatically to the point where generic commodity IoT deployments are feasible and motivated. In the future, commoditised resources can be greatly utilised in high-density areas (e.g. metropolitan areas, city centres) where CoT resources can be offered to many consumers. Providers will be able to deploy IoT nodes with a range of sensors, actuators, cameras and other resources, and make these devices available to clients to monitor and control the surrounding environments. The desirable situation in which IoT resources will be globally available to such clients requires the creation of an open market for commodity IoT resources in the same way that a market for Cloud Computing resources has emerged. Currently, managing IoT resources is still a challenge due to their heterogeneity and constrained capabilities [17]. For this to be viable, there needs to be both technical and commercial CoT integration support. The technical support involves the use of dynamic bridges, proxies and gateways to allow IoT applications development using established Cloud Computing platforms [10, 166]. The commercial support includes using market-based mechanisms as an approach to improve resources management in CoT, especially resource allocation [151]. The following attempts to solidify this by highlighting the important considerations in the argument for a market for commodity CoT resources.

- **Enabling interoperability:** Enabling interoperability is a well-known challenge for Cloud and IoT implementations due to the heterogeneity of both technologies. Commodity CoT resources will be used only if consumers are not restricted to a specific service provider and can switch between providers due to changes in consumers' requirements or providers' offerings. A market for trading CoT resources would encourage the development of standards and improve interoperability.

- **Creating new business values:** As the number of CoT deployments increase, the risk of a small number of providers controlling the market is high; such as is currently being observed in the Cloud Computing market. This increases the risk of single provider technical failures, as well as single vendor lock-in [125]. Technical failures such as bugs, misconfigurations and security breaches can have a substantial negative impact on the operations of many consumers simultaneously. A CoT marketplace will enable competitive and independent implementations of CoT protocols which will significantly reduce any monopoly-related risks. Consumers will also benefit from enjoying the freedom of choices from a multitude of providers.

CoT services also require joint efforts and cooperation between businesses to bring new services to the market. A market will enable businesses to go beyond the traditional known business models (e.g. Business to Business, Business to consumer) to new ones such as business to business to consumer (B2B2C) where the end service is traded by the adjacent industry partner who owns and manages the relationship with the end consumer [57].

Furthermore, the provision of IoT services usually requires significant investments which are not affordable by most small and medium enterprises. A marketplace of commodity CoT resources will enable SMEs to be involved in a larger community. This can also attract smaller consumers with specialised needs who are best served on a retail rather than a wholesale basis. Aggregations of small providers can also form offerings from multiple CoT resource sets.

- **Improving service level agreements:** Essential to the success of commodity CoT resources is the development of well-defined service level agreements [41]. SLAs are currently negotiated between each provider and consumer in Cloud Computing. A market has a standard SLA which defines the minimum terms of contracts that will cover both providers and consumers. Those terms are based on the characteristics of a service rather than a provider or a consumer-based agreement. Both providers and consumers can negotiate further terms and conditions to be included in their own SLAs without breaking the basic market SLA. A standard SLA has some benefits including better legal protection for consumers and providers, better pricing policies and improved standards for market entry.

- **Enabling innovations:** A market for commodity CoT resources will add a large number of players to the current market. This will promote innovation in the required infrastructure, including IoT and Cloud technologies. This should allow infrastructure vendors to produce, market and support a wide range of differentiated services. It may also motivate the emergence of new infrastructure suppliers, and motivate innovative design and adoption of mobile sensor networks that can support mobile CoT applications. It is also expected to enable hardware and software innovations when a large number of software developers and hardware makers respond to the requirements of the CoT market.

Although there is no standard for building CoT applications yet, this creates a unique solution for every deployment. Service providers also restrain innovations by locking-in their consumers and restricting application development to the providers' infrastructure. A market will support development by facilitating the emergence of standard interfaces.

These motivations show the many advantages of providing support for the commoditisation of CoT resources. A market will enable technical innovation through interoperability between types of CoT resources and applications as well as business innovations and improved SLAs. To support these goals, there needs to be a standard way of describing CoT resources and services. An architecture for trading these resources with efficient algorithms that match resources provided with potential consumers of those resources would also be needed. The following section describes the requirements for efficient trading of CoT resources supported by the ideas discussed in this section.

3.3 Requirements for Trading Cloud of Things Resources

For efficient commoditisation of CoT resources, global on-demand access, efficient sharing, and optimal allocation of CoT resources have to be enabled using market-based mechanisms. Trading CoT resources is a multifaceted process. It involves describing heterogeneous CoT resources, mapping resources to applications, optimising the proposed maps, performing resource allocation and scheduling of the optimal map. To shape this approach, the following considerations are taken into account:

1. CoT resources and applications are heterogeneous. To decompose this complexity and improve interoperability, resource deployment and application development are considered distinct from each other. This lowers the investment required for infrastructure deployments, accelerate application prototyping and enable efficient trading.

2. CoT systems are large-scale and their commoditisation mechanisms, therefore, have to be scalable. In particular, having the ability to handle significant large numbers of resources and application requests simultaneously.

3. CoT resources are naturally constrained in terms of computing and power capabilities, IoT physical nodes in particular. Due to such challenge, concurrent shared access to those resources may not be possible, but has to be resolved.

Based on the above considerations, the following requirements of the proposed approach are identified:

- **A Description Model for CoT Resources:** The challenges of CoT heterogeneity and complexity lead to challenges in defining CoT resources and quantifying their values. A market-based mechanism to trade CoT resources should consider using a generic mean of describing such heterogeneous resources and provide an efficient way of quantifying their value.

- **Shared Access Strategy for Constrained CoT Resources:** Resource sharing mechanisms in Cloud Computing differs significantly from the ones available to IoT resources. The primary difference between the two is the capabilities of each technology. Cloud resources are usually hosted in powerful large-scale data-centres to provide virtually unlimited, elastic and on-demand computing resources. In contrast, IoT resources are

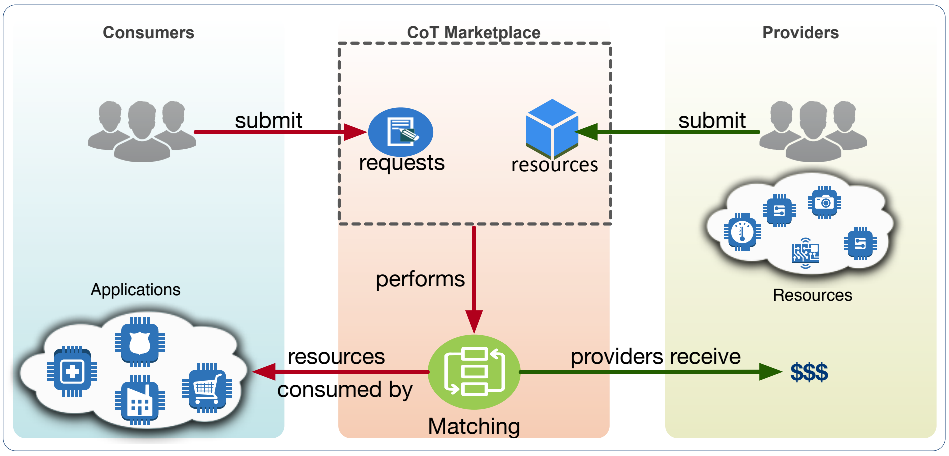


Figure 3.1: CoT marketplace concept.

widely distributed across the application area with constrained computational and power resources where they cannot be demanded or shared elastically as Cloud resources. There should be an efficient mechanism to improve shared access to CoT resources.

- **A Trading Model:** For the commodity IoT resources to be entirely accepted and integrated with current infrastructures, they must be publicly accessible. The access method appropriate for this is using the Cloud Computing service model where consumers purchase openly available resources or services and pay for the level of actual utilisation. The Cloud service model is preferable to users due to its speedy trading process and its job-oriented pricing model. Although this can be described as minimal integration CoT in the literature, it can be tailored to support other integration approaches.

- **A Marketplace System:** The concept of CoT marketplace revolves around the idea of having a marketplace where providers offer their deployed CoT resources and applications (consumers) request access to the offered resources. The marketplace maps requests to resources by forming a bundle of resources from multiple providers based on the application requirements. The concept of the proposed marketplace is visualised in Figure 3.1.

- **The Marketplace Participants:** For the marketplace to be fully functional, there should be at least two types of participants in the marketplace. Participants are providers of resources and consumers of resources. A third type that combines the roles of consumers and providers is also possible where it is called prosumers [13]. In CoT trading environment, providers are the owners or deployers of resources while the consumers are the application owners or developers.

- **Other Requirements:** Other requirements of the marketplace can be implemented based on its design and operational goals. These requirements can be justified as follows:
  - **QoS Support.** To support efficient trading of CoT resources, a generic and dynamic QoS support is needed. QoS is vital in CoT trading environment to measure the performance qualities of the resources that meet the requirements of the consumers. A market-based mechanism, therefore, has to take QoS into account.
  - **Scalability.** A marketplace for CoT resources is expected to handle a considerable number of consumer requests and provider resources simultaneously. This can impact the system performance significantly and consequently fail to trade CoT resources. The marketplace system should have the ability to handle various scales of consumer requests and provider resources.
  - **Security.** Security is vital for any marketplace architecture. Security vulnerabilities have to be identified and addressed by using various security measures to secure the operations of the marketplace.
  - **Multiple Business Models.** To satisfy the requirements of different CoT applications, the marketplace should support different business models. This includes consumer-to-consumer (C2C), business-to-consumer (B2C), business-to-business (B2B) and business-to-business-to-consumer (B2B2C). It may also consider implementing the system to support one or more of the following market structures: broker systems, monopoly markets, oligopoly markets, single-side auctions and double-side auctions [122].

- **Multiple Participant Objectives.** Marketplace participants have different requirements and goals. For instance, consumers are likely to bid for the lowest cost possible while providers aim to maximise their revenues. Consumers may have conflicting objectives at the same time (e.g. maximising resource coverage while minimising the response time) [100]. The marketplace system should provide a mechanism to maintain the balance among conflict objectives.

3.4 Motivations for The Proposed Approach

The following can be observed from the motivations and requirements discussed in the preceding sections, as well as the literature review presented in the preceding chapter. Despite many advantages to providing support for the commoditisation of CoT resources, there needs to be an efficient way of trading CoT resources that takes into account the heterogeneity, complexity and scalability of CoT. The proposed approach presented in this thesis re-imagines the CoT trading environment to involve optimisation algorithms instead of traditional dedicated auctioneers that match resources provided with potential consumers. Using optimisation-based strategies provide the flexibility and dynamism needed in CoT systems. In contrast to the traditional auctioneer-based approaches, the optimisation approaches require minimal changes either by using different objectives or different optimisation strategies. This approach reduces the time needed to find an optimal map of consumers to providers for real-time CoT applications.

This approach is inspired and motivated by the successful use of optimisation-based approaches in large-scale computing systems similar to CoT. Motivating examples are discussed as follows. An optimisation-based task scheduling model is proposed in [59] to minimise the computational cost of transferring data and processing applications in Cloud Computing. A meta-heuristic application for scheduling workflows tasks in the Cloud is presented in [103] to optimise the cost of resources. A market model based on combinatorial double auction resource allocation is introduced in [143] to improve the trading fairness for consumers and providers of Cloud services. A model for service allocation is proposed in [36] to optimise the computational cost and resource availability in fog computing environments. An optimisation algorithm is used in [56] to optimise the profit of clients in a cloud system based on SLAs. A multiobjective model for optimising response time, resource cost and brokers' profit in Cloud environments is developed in [81]. A scheduling framework for optimising the cost and execution time of high-performance computing applications in the Cloud is presented in [150]. An evaluation of optimisation-based approach is performed in [114] to optimise the performance and cost of IoT applications on Cloud. To optimise the energy of Cloud radio access network for IoT, an optimisation-based resource allocation scheme is proposed in [165]. A further optimisation approaches and objectives for cloud computing are surveyed in [60, 140] and for WSNs in [51].

3.5 Methodology

The project is developed in two phases; first, the preliminary design and evaluation, followed by the final design and evaluation. The first phase aims to investigate the research problem of limited shared access to CoT resources, and to preliminary design a solution with adequate experimental evaluation. This phase includes the research proposal that thoroughly reviews the related work in areas of Cloud Computing, IoT, WSNs and Grid Computing. The analysis of the research gap confirms that there is a need for a potential solution using market-based mechanisms as in other large-scale computing environments, but the focus of previous studies is on improving shared access to CoT data only while improving shared access to physical IoT resources is still missing. Upon the completion of the gap analysis, the following research questions are formed:

- Can market-based mechanisms improve the shared access to CoT resources?
- Are optimisation algorithms effective and efficient in trading CoT resources to improve their shared access?

Forming the research questions is followed by proposing the aims and objectives as discussed earlier. The preliminary design and evaluation in this phase focus on developing the following outcomes: 1) resource description model for CoT resources, 2) shared access and resource allocation strategy, 3) the optimisation model used to map requests to resources and 4) the prototype marketplace system. Prototyping the proposed system is performed through the cycle of requirements analysis, design, development and intensive testing. The experimental evaluation at this stage aims to validate the feasibility and effectiveness of the proposed approach.

Phase two is dedicated to improving the proposed models developed during phase one by tuning the design and performing the final experimental evaluation. In addition to improving the components developed in phase one, a QoS model for optimising QoS trading objectives is proposed and evaluated during this phase. The same software development cycle is used in this phase as in phase one. The experimental evaluation in phase two tests the performance and scalability of the proposed marketplace system. Both phases are visualised in Figure 3.2 to illustrate the research methodology.

Setting up a real-world CoT environment for this research is complex and very expensive. To commoditise CoT resources in reality, a large number of heterogeneous resources has to be involved including various types of IoT nodes with different sensors, actuators and computing components. This complexity is required to justify the research approach and the results when commoditising CoT resources. Therefore, the approach taken in this research is to simulate the data generation of resources and requests. Existing public data sets such as Citypulse [124] and SocialIoT [158] do not provide metadata for IoT physical resources. Data sets evaluated in this thesis are generated based on surveying the most common properties available for IoT resources in large vendors such as Amazon, Google and IBM.

3.6 Discussion

This chapter discussed motivations and requirements for the solution of limited or missing reusability of CoT resources. It aimed to justify the need for a practical solution based on the requirements identified. The chapter also described the methodology of the research presented in this thesis.

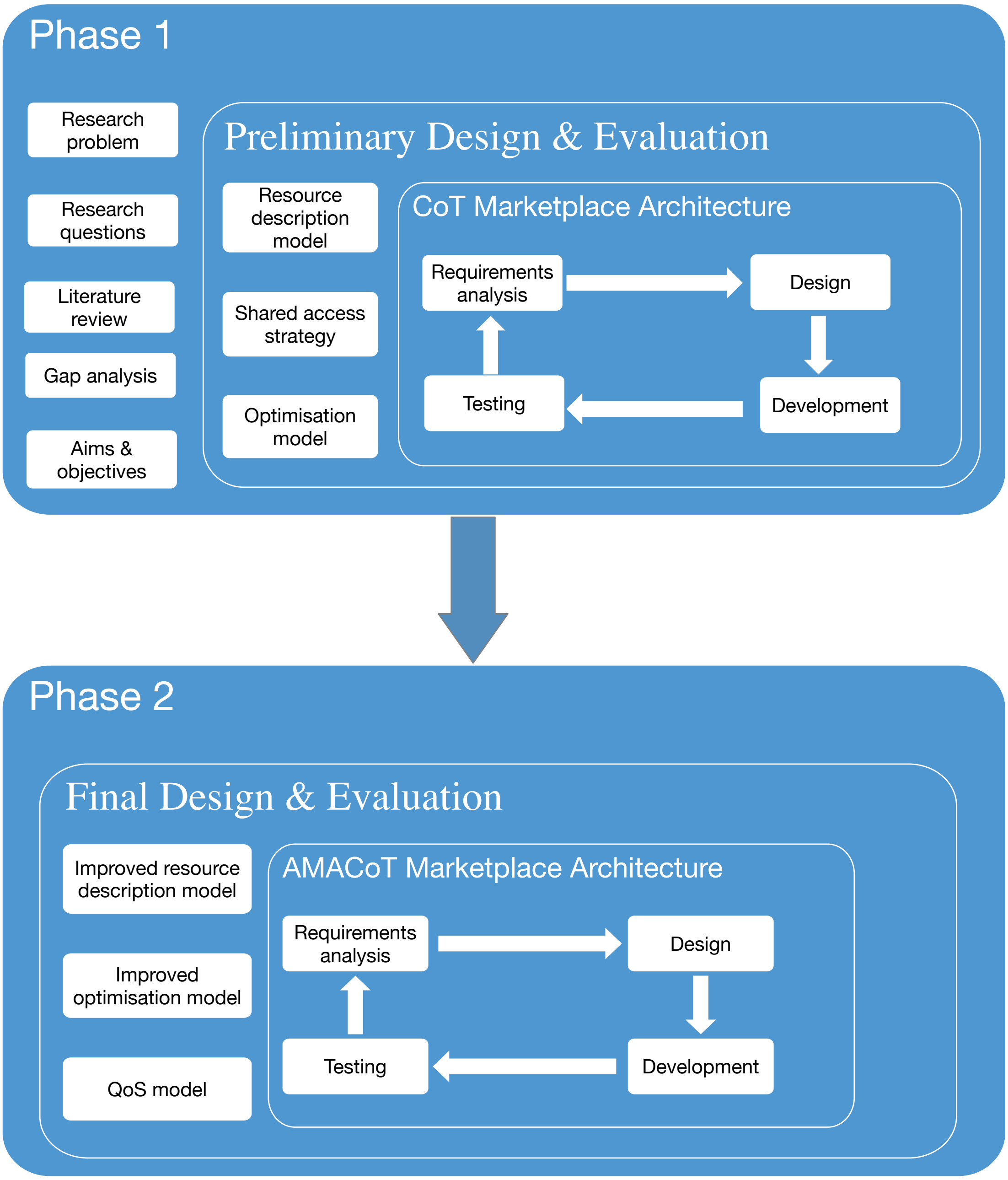


Figure 3.2: Summary of the methodology for this 2-phase research project.

Technical and business benefits presented in Section 3.2 motivated supporting commodity CoT resources. Technical aspects discussed imply that improving shared access to CoT resources may resolve CoT interoperability issues among different standards and enable CoT software and hardware innovations. Based on the survey of existing research that attempts to address the problem of shared access to CoT resources, Section 3.3 identified and discussed the requirements of the solution. This includes a standard way of describing CoT resources, a mechanism to improve shared access during the resource allocation process, a trading model for matching the supply and demand, a marketplace system and several non-functional requirements. It can be understood from the requirements identified that traditional auctioneer-based approaches may not be feasible for CoT dynamism, performance and scalability.

In Section 3.4, motivations for the proposed solution described many proposed optimisation-based solutions for Cloud, IoT and WSNs that inspired the proposed approach presented in this thesis. It is understood from the discussion that optimisation-based approaches were able to improve shared access to resources in other large-scale computing environments with similar complexity and scalability aspects of CoT. This would enable CoT to utilise optimisation strategies in addressing its flexibility and dynamism requirements.

Section 3.5 presented the methodology of the research. Based on the requirements identified earlier, the research is divided into two phases. The first phase focuses on the initial design of the solution and the preliminary experiments to validate the concept. The improved design in phase two is to be evaluated by further experiments to test the system performance and scalability. As illustrated in Figure 3.2, the proposed methodology features a systematic theoretical review of the research problem and conceptualisation of the solution as well as the design and implementation of the proposed system architecture.

Chapter 4

Proposed Optimisation-Based Approach for Trading Cloud of Things Resources

4.1 Introduction

Cloud of Things is increasingly viewed as a paradigm that can satisfy the diverse requirements of emerging IoT applications. The potential of CoT is not yet realised due to challenges in sharing and reusing IoT physical resources across multiple applications. Existing approaches provide small-scale and hardware-dependent shared access to IoT resources. The research presented in this thesis considers using market-based mechanisms to commoditise CoT resources as the approach to enable shared access to CoT resources and to improve their reusability. In order to achieve these goals, this chapter describes the proposed approach of trading CoT resources using optimisation strategies to match the demand of CoT applications to the supply of CoT resources based on the requirements for trading CoT resources.

The remainder of this chapter is structured as follows: Section 4.2 introduces and discusses the the main components of the proposed approach, Section 4.3 describes the proposed shared access mechanism for CoT resources, Section 4.4 provides a case study for the proposed approach applications, Preliminary experimental evaluation is presented in Section 4.5, Conclusions drawn from this chapter are discussed in Section 4.6.

4.2 Proposed Approach for Trading CoT Resources

Trading CoT resources is a multifaceted process. It describes the process of commoditising CoT resources by mapping resources from multiple providers to applications from multiple consumers based on the application requirements, optimising the proposed maps, performing resource allocation and scheduling of the optimal map. To shape this approach based on the considerations and requirements for trading CoT resources discussed earlier in Chapter 3, the following components are designed:

4.2.1 The Marketplace System

For efficient commoditisation of CoT resources, global on-demand access, efficient sharing, and optimal allocation of CoT resources have to be enabled. In order to achieve this goal, a marketplace architecture for trading CoT resources is needed. CoT marketplace concept was introduced and depicted earlier in Figure 3.1. The idea is to have a marketplace - denoted by M - where providers offer their deployed CoT resources and applications request access to the offered resources. The marketplace matches requests to resources by forming a bundle of resources from multiple providers based on the application requirements.

In this section, an initial design of CoT marketplace is proposed and presented in Figure 4.1. The proposed architecture and the process of trading CoT resources described in this section will be significantly improved in the next chapter to reflect the development of the research presented in this thesis. The aim of this preliminary design and evaluation of the marketplace architecture is to provide a proof of concept that experimentally validates the feasibility of the proposed approach in trading CoT and improving shared access to CoT resources. The trading process is described as follows.

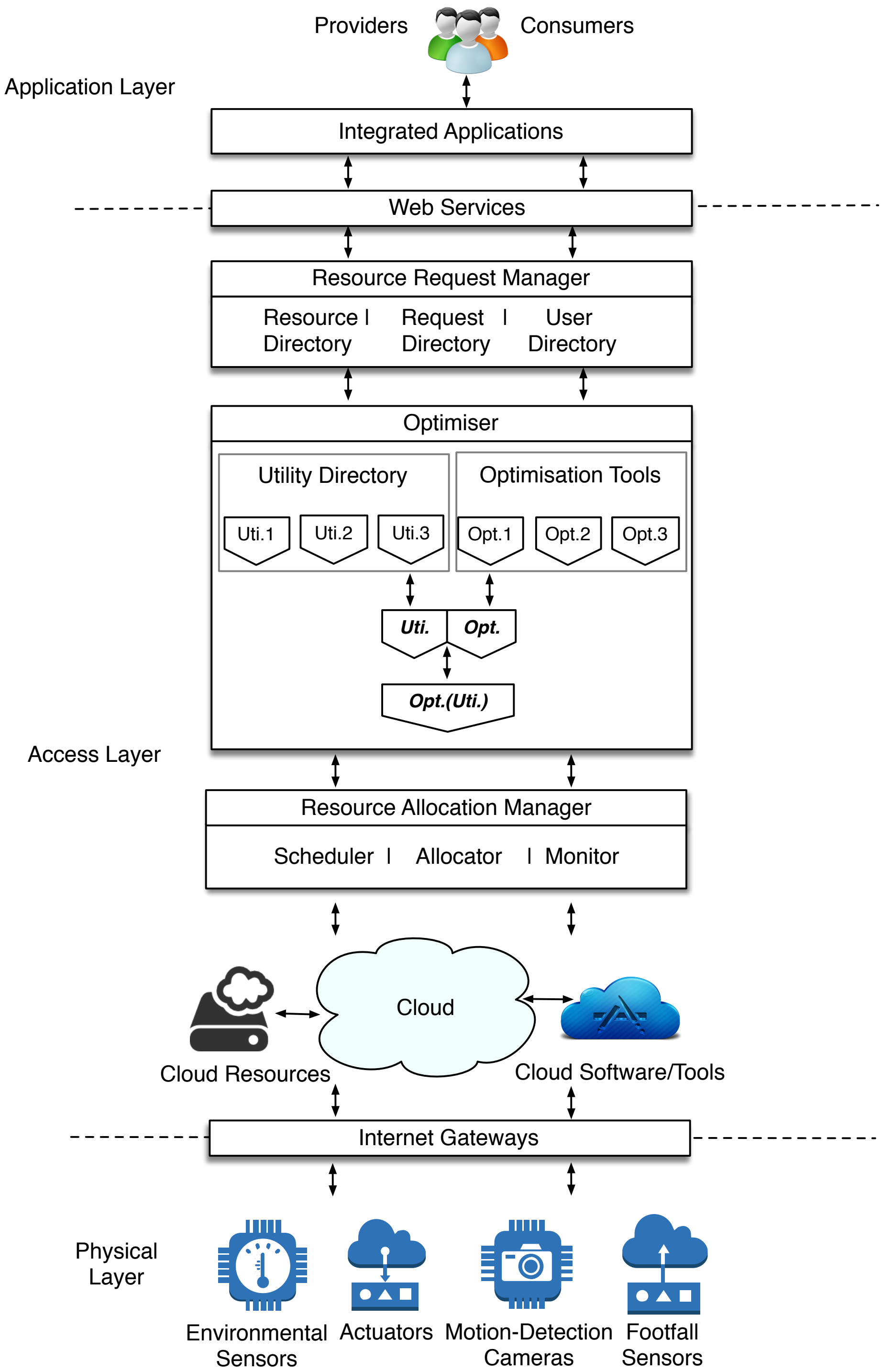


Figure 4.1: Preliminary design of CoT marketplace architecture.

Providers submit their resources and consumers submit their requests to the marketplace using their integrated applications via web services. The resource request manager filters both requests and resources to ensure they meet the marketplace standards. The resource directory tries extracting the metadata of the resources that meet the marketplace standards while rejecting the ones that do not meet the marketplace standards, or their metadata could not be extracted. Upon successful extraction of the resources' properties from their metadata, a standard description of the resources is stored in the resource directory. Requests that met the marketplace standards are stored directly into the request directly. The users who submit their resources or request are profiled and stored into the user directory. Once resources and requests are ready for trading, the resource request manager sends the available resources and requests to the optimiser to start the trading process. The optimiser consists of two components, namely utility directory and optimisation tool. The utility directory maintains all utilities defined for trading proposes (e.g. cost-based objectives, time-based objectives, performance-based objectives). The optimisation tool stores and implements the most suitable optimisation techniques to optimise matching requests to resources efficiently. Based on the optimisation process, the optimiser produces an optimal solution often called an optimal map or optimal assignment. The optimal assignment is submitted by the optimiser to the resource allocation manager to allocate the requested resources accordingly. The scheduler maintains the resource schedule, controls the lease-time of resources and manages the assignments of tasks in the Cloud. The allocator orchestrates mechanisms of joining and dis-joining resources based on scheduler plan. The monitor tool monitors the ongoing consumption of the resources by CoT applications and the availability of the resources.

This architecture is designed with the consideration of flexibility and dynamism required in CoT where the optimiser is the heart of the system. The optimiser employs optimisation algorithms that require no or minimal changes when changes occur to resources, requests or the objectives of trading. Changes can be addressed by either using different/improved utility functions or different optimisation techniques. It reduces the time required to find a better assignment of resource allocation, or increases to the number of candidate solutions and provides significant support to the scalability requirements of CoT. These benefits are challenging to achieve by existing approaches reviewed in Chapters 2 and 3.

4.2.2 The Marketplace Participants

The marketplace consists of two categories of participants, namely providers and consumers. Providers can be CoT infrastructure owners or deployers and are denoted by $P = (p_1, \ldots, p_m)$. p_m represents an individual, an organisation or a broker who manages resources on behalf of others. P submit their resources $R = (r_1, \ldots, r_j)$ to the marketplace.

Consumers are application owners or developers and are set to $C = (c_1, \ldots, c_n)$. c_n represents an individual, organisation or a broker who manages applications on behalf of others. C submit requests $RQ = (rq_1, \ldots, rq_i)$ to the marketplace for their CoT applications $A = (a_1, \ldots, a_z)$ to access and utilise a set of R.

4.2.3 A Multi-attribute Description Model for CoT Resource

There is a wide range of CoT heterogeneous physical resources. For instance, sensors, actuators, smart meters, cameras, mobile phones and fitness trackers. The heterogeneity of CoT resources poses challenges for two reasons. First, it is challenging to describe what a "resource" is in a generic way that can be used to describe all existing and potential CoT resources. It is important for the description model to be generic enough to describe any resources regardless of their vendor, type, hardware and software properties. Second, it is challenging to quantify the value of such heterogeneous resources to enable them to be commoditised. Thus, there is a need for a description model that takes into the account the complexity of describing heterogeneous CoT resources by; 1) defining CoT resources, 2) providing a generic description of the resources' properties and 3) quantifying the value of CoT resources based on their described properties.

There are several description models for IoT resources such as the Sensor Model Language (SensorML) [105], IoT Ontologies [61] and a unified IoT ontology [6]. However, these models have the following shortcomings. They provide either a very limited support to a number of IoT resources (e.g. sensors only or sensors and actuators only), or they do not consider a resource description that enables quantifying the resource value. Additionally, some of the existing models suffer from a heavyweight encoding that is not suitable for constrained IoT resources [99].

To ease these challenges, CoT resources can be defined generically based on their main physical components and functionalities. A CoT resource, node, device or thing can be defined as a device that is powered by an energy source with one or more basic computing functionalities (I/O, Processing, Storage) that interacts by monitoring, sensing and/or actuating of certain events in the surrounding environment (using Sensor(s), Camera(s), Actuator(s)) and communicates with other entities of a network (using Communication unit(s)). Figure 4.2 illustrates the main components of a CoT node.

To overcome the complexity of CoT nodes heterogeneity, a set of generic attributes is proposed to describe the properties and functionalities of the resource. Each resource r_j has a set of attributes that contributes to its value when traded as a commodity. This includes multi-attributes of physical components (e.g. processing, actuating, sensing, power) and non-physical functions or features (e.g. security, location, redundancy). Each property or feature can be expanded into a multilevel sub-attributes to improve the presence of the commodity resource in the marketplace. For instance, the sensing capability of the resource can be described in terms of its sensing type (e.g. environmental, footfall), sensing range (e.g. limited, average, long), maximum transmission power and the number of sensors available in the resource. The reset of the resource properties can be described in the same way, forming a multi-attribute generic model of describing CoT resources. The proposed model describes CoT resources autonomously as follows. Providers submit their resources to the resource manager where it checks the submitted resources against several marketplace requirements such as pricing, billing, QoS and SLAs. Resources that fall below the marketplace standards are rejected and

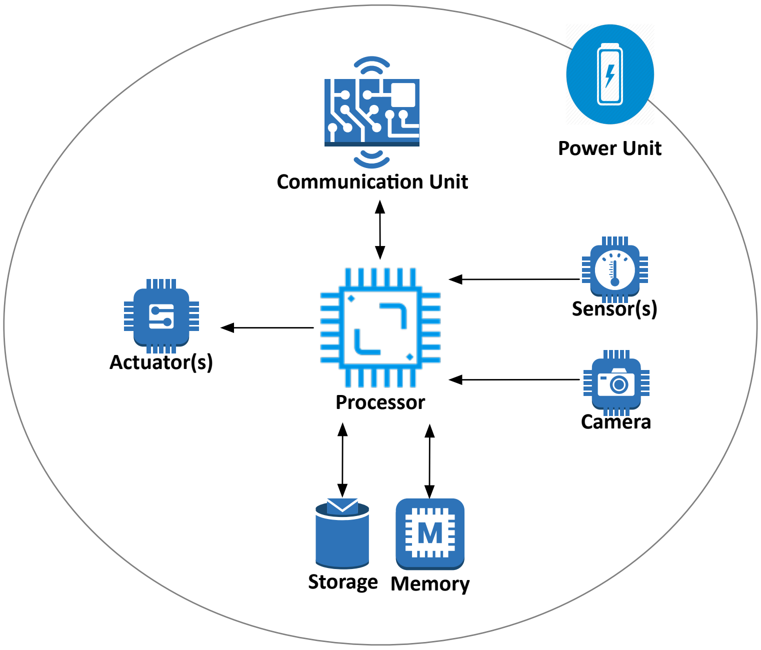


Figure 4.2: Main components of IoT node.

returned to their providers. Accepted resources flow to the next stage filtration, where the resource manager attempts to extract the values of the resource properties. Resources with unreadable values or missing the majority of required values are rejected and returned to their providers for revision. Readable metadata of resources get extracted and stored into the resource directory. The final step of describing the resources is quantifying their values. The description model is illustrated in Figure 4.3.

The value of the multi-attributes presented in the previous step can then be quantified by assigning corresponding numerical values. This is a vital step to monetise heterogeneous resources generically, to enable them to be traded and as a result, to improve shared access to the resources. This can be achieved by assigning lower numerical values to attributes representing low resource specification and vice versa. Zero is a corresponding value for a missing resource component while a positive number corresponds to an attribute. This can be formulated as follows:

$$ra = \begin{cases} > 0, & \text{if } \; ra_j \text{ can be described from } \; r_j \\ 0, & \text{otherwise} \end{cases} \tag{4.1}$$

For instance, a power capacity attribute of r_j can be quantified as either $[Power = [Permanent, 2]]$ or $[Power = [Battery, 1]]$. In this case, a resource with permanent power supply is assigned a higher value than the one operates by a battery and is likely to have a better value when commoditised. This offers the flexibility required in quantifying the value of heterogeneous CoT resources before the trading process starts. A snapshot of a single resource description is provided in Table 4.1.

Once the resource is properly described, the request can be formed accordingly. Requests can be submitted by a consumer who consumes resources only, or by a provider of resources who provides resources for consumption or re-providing them to a third party. It is assumed that the request is for a homogeneous set of resources for an application. Four examples of requests are explained as part of case studies in Section 4.4.

4.2.4 The Optimisation Model

After CoT resources being described and their values quantified, resources can be matched with the application requests. The work presented in this thesis takes a unique approach to perform the matching by using optimisation algorithms as a mapper to match requests and resources. This approach provides the following advantages that justify the use of optimisation algorithms.

1. Using optimisation algorithms provides improved architectural flexibility for CoT systems over the conventional designed components (e.g. auctioneer, mapper). This means there is minimal or no need to modify the optimisation algorithm to support any changes in other marketplace system components or CoT resources and requests.

2. Optimisation algorithms are known to find optimal solutions to very complex problems that may have a very large number of candidate

Table 4.1: Snapshot of CoT resource description.

Components	**Attributes**	
	Properties	**Example Values**
Processor	Clock speed	320MHz
Memory	RAM Flash	256KB 1MB
Sensor(s)	Type(s) Sensing Range Max. Transmission Power Num. of sensors accuracy	Footfall, light 20 meter 13.5dBm 4 +/- 0.2 meter
Actuator(s)	Type(s) Num. of Actuator	Light 1
Camera	Type(s) Num. of cameras	Motion detection 1
Communication	Type(s) Protocols Bandwidth	WiFi IEEE 802.11b/g/n 16Mbps
Power	Mode capacity	Permanent 5V
Security	Not available	0

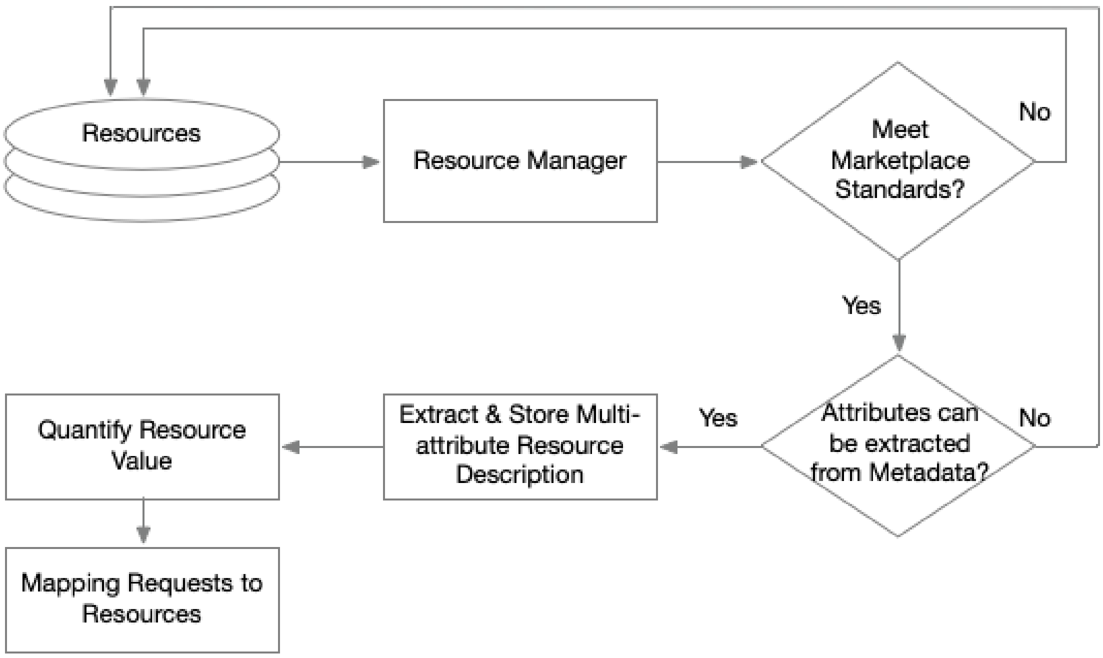


Figure 4.3: Overview of the CoT resource description model.

solutions. These problems are similar in complexity and scalability to the problem of trading CoT resources presented in this thesis [30, 130].

3. The speed of many optimisation algorithms addresses the requirements of the CoT marketplace in finding an optimal solution promptly.

4. The proposed approach presented in this thesis relies on the use of gradient-free optimisation algorithms (a.k.a. derivative-free algorithms). Using this type of optimisation algorithms does not require the calculation of the gradient or derivative to find the optimal solution. The computation of the gradient can be impractical or computationally costly for large-scale optimisation problems such as trading CoT resources.

The optimisation layer represents the operational tier of the system. Applications can discover resources that are already stored in the resource request manager. Bids flow from resource request manager to the optimiser where they are mapped to form bundles. Resource bundles represent a set of resources from multiple providers that can potentially be utilised by multiple applications. Whilst being forwarded to the optimiser, resource bundles and application requests are filtered. Pluggable filters include a wide range of filtering criteria such as location, resource coverage, computing and energy requirements whose aim is to reduce the search space of the problem and generate potential optimal maps only. This is achieved by evaluating each map based on the objective and its compliance with search constraints discarding maps that are either extreme (e.g. very expensive resource) or violate the constraints (e.g. below certain energy level). Filtered resource bundles and application requests are forwarded to the optimiser. The optimiser performs a two-stage process as follows: 1) construct optimal maps that consist of resource bundles and application requests ready for allocation, 2) evaluate the optimal maps based on the participants' goals using utility functions presented in Section 4.2.5. One optimal map is forwarded to the resource allocation manager for resources to be allocated to the applications. The optimisation model is presented in Figure 4.4.

4.2.5 Trading objectives

Trading objectives represent the goals of providers P and consumers C from participating in trading CoT resources. These goals are formulated as objective functions to provide significant flexibility for the optimisation model. Using this approach would minimise the re-development effort of the system components that may be required in case of resource changes. Changes can be implemented as a new objective function without or with minimal changes to the system. The stage of preliminary design and evaluation considers the following trading objectives:

Objective 1: Maximising Provider Profit. The providers always aim to maximise their profit. A utility is needed to achieve this objective. rc_j denotes the cost of a resource from provider j, and t_i denotes the requested lease time of a resource by consumer i. The cost of allocating a resource to a consumer can be calculated as $(rc_j.t_i)$. The utility for maximising the profit of providers can be represented as follows:

$$\underset{RQ,R}{\text{Maximise}} \quad PR = \sum_{i=1}^{n} \sum_{j=1}^{m} rc_j.t_i \tag{4.2}$$

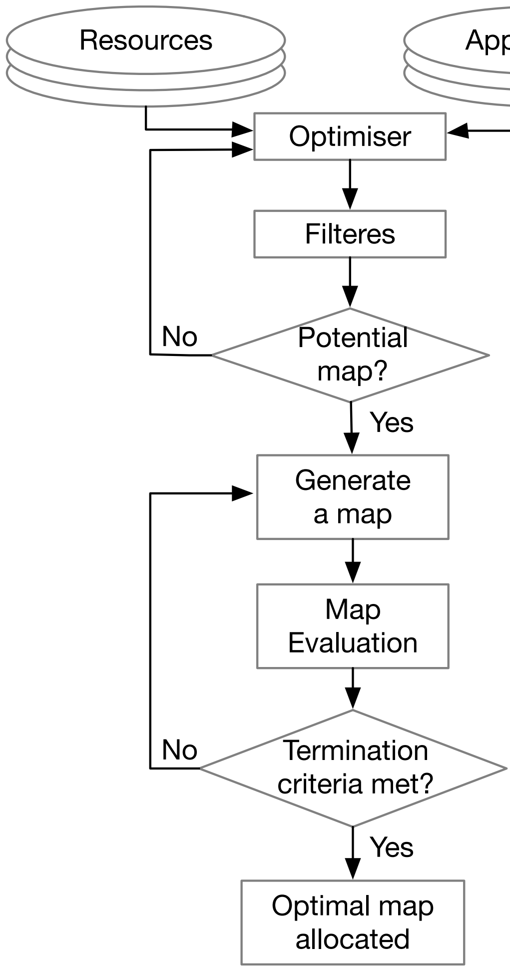


Figure 4.4: The optimisation model.

The pseudocode of maximising provider profit utility is shown in Algorithm 1.

Objective 2: Maximising Resource Coverage. Consumers are expected to look for resources that provide them with the maximum area coverage when utilising resources. This is challenging due to the fact that different measurements of power and area should be considered simultaneously where each of them have a different scale. To achieve this goal, the sensing range of a resource s_j and the maximum transmission power level Et_{max} can be used to measure how far a resource can reach ($s_j.Et_{max}$). The requested location of resources is equal for all consumers and formulated as $A = (x_i y_i)$ that represents rectangular grids of identical dimensions. All variables are then normalised to adjust their values on

Algorithm 1 Utility for maximising provider profit

Input: 1) list of consumers with their requests' attributes, 2) list of providers with their resources' attributes.
Output: assignment(consumers, providers) of best optimised profit
Function: Provider Profit Utility Function

Initialise cost and capacity counter to zero, assignments to empty
Loop over array of initial guess
Generate an assignment of (consumers, providers)
End Loop
Loop over list of generated assignments
If length(consumers) = length(assignments) AND capacity Counter < provider Capacity AND consumer Bid ≤ provider Cost AND energy Requested ≤ energy Offered
Then, calculate Provider Profit
Else, set provider Cost to minimum value
End Loop
increase capacity Counter
calculate profit of all participated Providers in the assignment
return profit of the assignment to optimiser

END Function

different scales to a numerical common scale. The objective of maximising the coverage is introduced as follows:

$$\underset{RQ,R}{\text{Maximise}} \quad Cv = \sum_{i=1}^{n}\sum_{j=1}^{m} \frac{s_j.Et_{max}}{A_i} + s_j \tag{4.3}$$

Algorithm 2 shows the pseudocode of a utility maximising area coverage.

Objective 3: Minimising Response Time. Response time is also considered one of the very important objectives to minimise in large-scale distributed systems. The latency between consumer i and provider j is denoted by $L_{ij} = t_{ack} - t_{start}$ which measures the elapsed time from submitting the request by consumer i to the time of receiving an acknowledgement from a provider j. Estimated queuing and transmitting delays t_{qd} are also considered here where they can be formulated as $t_{qd} = \frac{L_{ij}}{RQ_i}$. The objective to minimise

Algorithm 2 Utility for maximising area coverage

Input: 1) list of consumers with their resource requests' attributes, 2) list of providers with their resources' attributes, 3) Area of requested resources
Output: assignment(consumers, providers) of best optimised coverage
Function: Coverage Utility Function
1: Initialise capacity counter to zero, assignments to empty
2: Loop over array of initial guess
3: Generate an assignment of (consumers, providers)
4: End Loop
5: Loop over list of generated assignments
6: If length(consumers) = length(assignments) AND capacity Counter $<$ provider Capacity AND consumer Bid $\leq$ provider Cost AND energy Requested $\leq$ energy Offered
7: Then, calculate coverage of the requested resource
8: Else, set coverage to minimum value
9: End Loop
10: increase capacity Counter
11: calculate the coverage of all participated resources in the assignment
12: **return** the coverage of the assignment to optimiser
END Function

response time R_t is proposed as follows:

$$\text{Minimise} \quad R_t = \sum_{i=1}^{n}\sum_{j=1}^{m} L_{ij} + t_{qd} \tag{4.4}$$

The pseudocode of the utility minimising the response time is shown in Algorithm 3.

Objective 4: Minimising Energy Consumption. Another important objective is to minimise the power consumption of matched resources while being utilised by consumers. It can be measured by the difference between the initial power supply of the resource and the estimated power consumption requested by the consumer($Ep_j - Er_i$). The objective of power consumption can be presented as follows:

$$\text{Minimise} \quad E = \sum_{i=1}^{n}\sum_{j=1}^{m} (Ep_j - Er_i) \tag{4.5}$$

Algorithm 4 presents the pseudocode of the utility minimising the energy consumption.

Algorithm 3 Utility for minimising response time

Input: 1) list of consumers with their resource requests' attributes, 2) list of providers with their resources' attributes
Output: assignment(consumers, providers) of best optimised response time
Function: Response Time Utility Function
1: Initialise capacity counter to zero, assignments to empty
2: Loop over array of initial guess
3: Generate an assignment of (consumers, providers)
4: End Loop
5: Loop over list of generated assignments
6: If length(consumers) = length(assignments) AND capacity Counter < provider Capacity AND consumer Bid ≤ provider Cost AND energy Requested ≤ energy Offered
7: Then, calculate response time of requested resource
8: Else, set response time to maximum value
9: End Loop
10: increase capacity Counter
11: calculate response time of all participated resources in the assignment
12: **return** the response time of the assignment to optimiser
END Function

Objective 5: Maximising Marketplace Profit. In case the marketplace is non-volunteering or not a community-based, there will be fees for trading CoT resources called a marketplace commission that is denoted by cm. The marketplace will aim to maximise its profit at each successful round of resource allocation. b_i is set as a bid of consumer i, rc_j denoted the cost of a resource from provider j and t_i denotes the requested lease time of a resource by consumer i. The commission of the market can be presented as $cm = (b_i - rc_j).t_i$. The cost of a resource is presented as rc_j. The objective to maximise the profit of the marketplace M_g can be formulated as follows:

$$\text{Maximise} \quad M_g = \sum_{i=1}^{n} \sum_{j=1}^{m} cm_{ij} + rc_j \tag{4.6}$$

The pseudocode of the utility maximising M_g is provided in Algorithm 5.

Each resource provider has a limited capacity for offering its resources to consumers. The capacity of the provider has to be greater than or equal to the total capacity requested from consumers. A capacity constraint is introduced as

Algorithm 4 Utility for minimising energy consumption

Input: 1) list of consumers with their resource requests' attributes, 2) list of providers with their resources' attributes
Output: assignment(consumers, providers) of best optimised energy consumption
Function: Energy Consumption Utility Function

1: Initialise capacity counter to zero, assignments to empty
2: Loop over array of initial guess
3: Generate an assignment of (consumers, providers)
4: End Loop
5: Loop over list of generated assignments
6: If length(consumers) = length(assignments) AND capacity Counter < provider Capacity AND consumer Bid ≤ provider Cost AND energy Requested ≤ energy Offered
7: Then, calculate energy consumption of participated resource
8: Else, set energy consumption to maximum value
9: End Loop
10: Increase capacity counter
11: Calculate the total energy consumption of resources in the assignment
12: **return** The total energy consumption of the assignment to optimiser

END Function

follows:

$$\sum_{i=1}^{n} rq_i \leq cp_j, \ where \ j = 1, ..., m \tag{4.7}$$

rq_i in constraint (4.7) denotes the number of requests from consumers while cp_j is set to total capacity of provider j.

Constraint (4.8) shows the cost of a resource rc_j and the bid from consumer b_i have to be positive and b_i has to be greater than or equal rc_j.

$$0 < rc_j \leq b_i \tag{4.8}$$

Constraint (4.9) ensures the initial power Ep_j of a resource and the estimated power consumption of the consumer Er_i are positive values and Er_i is less than Ep_j. The three constraints are applied together to all utility functions used in

Algorithm 5 Utility for maximising marketplace profit

Input: 1) list of consumers with their resource requests' attributes, 2) list of providers with their resources' attributes
Output: assignment(consumers, providers) of best optimised marketplace profit
Function: Marketplace Profit Utility Function

1: Initialise capacity counter to zero, assignments to empty
2: Loop over array of initial guess
3: Generate an assignment of (consumers, providers)
4: End Loop
5: Loop over list of generated assignments
6: If length(consumers) = length(assignments) AND capacity Counter < provider Capacity AND consumer Bid ≤ provider Cost AND energy Requested ≤ energy Offered
7: Then, calculate marketplace profit
8: Else, set marketplace profit to minimum value
9: End Loop
10: Increase capacity counter
11: Calculate the total marketplace profit from the assignment
12: **return** The total marketplace profit to optimiser

END Function

this chapter.

$$0 < Er_i \leq Ep_j \tag{4.9}$$

4.3 Enabling Shared Access to CoT Resources

As discussed earlier, resource sharing mechanisms in Cloud Computing matured over time while approaches to sharing IoT resources are still emerging. One of the major differences between the two types of resources is their computing and energy capabilities. Cloud resources are usually hosted in powerful large-scale data-centres to provide virtually unlimited, elastic and on-demand computing resources supplied by permanent sources of energy. Conversely, IoT resources are widely distributed across the application area with constrained computational and power resources. Therefore, there is a need for the shared access methods to enable shared access to such constrained resources.

The solution presented in this chapter is described as follows. A marketplace system receives requests from consumers and resources from providers. Market-based notations are used to quantify the value of IoT physical resources and requests. Based on the goal of the marketplace, an optimisation strategy is used to perform two tasks as follows: 1) Map the requests to resources that satisfy them, and 2) Evaluate the mapped assignments of requests and resources to propose an optimal assignment. The optimal assignment is scheduled as presented in the following sections.

To support shared access to CoT resources, the concept of Exclusive Shared Access (ESA) is introduced. The concept describes the process of scheduling CoT physical resources to be accessed and utilised by a single consumer at a given time and by multiple consumers over the length of the schedule. The concept is two-fold: 1) Exclusive access by each consumer to the desired resources at the required time, and 2) Shared access for multiple consumers to the same resources throughout the schedule. When the utilisation time of a consumer elapses, the resources are released and assigned to the next consumer in the schedule. When the schedule completes, the assigned resources are totally released back to the proposed system for a new round of mapping to different consumers.

The advantages of this approach include the following:

1. **Improving Interoperability:** IoT physical resources are truly utilised when consumers are not restricted to specific infrastructure and can move their applications to different providers due to changes in requirements or market offerings. The proposed approach is implemented by a marketplace where heterogeneous vendor-independent and platform-independent resources can be utilised by various CoT applications.

2. **Reducing Costs:** It is a cost-effective approach that separates between CoT application development and CoT infrastructure deployment. Infrastructure deployers can deploy their IoT resources independently without considering application-specific requirements. Similarly, application developers can develop their applications without usual concerns about infrastructure complexity and costs. The cost is reduced for application developers as they do not require a dedicated IoT infrastructure and any maintenance or specialised personnel to deploy it.

Infrastructure owners reduce their application development costs and increase their revenue from the trading which may justify the return on investment (ROI) of IoT infrastructure that can be very costly and infeasible for many emerging applications. This will likely reduce the overall costs and motivate new services and applications.

3. **Providing Flexibility:** The proposed approach provides significant flexibility to various CoT applications. For instance, time-sensitive applications including law enforcement and emergency agencies can gain high priority access to various CoT resources to monitor and respond to incidents as needed in real-time. More case uses are described in Section 4.4.

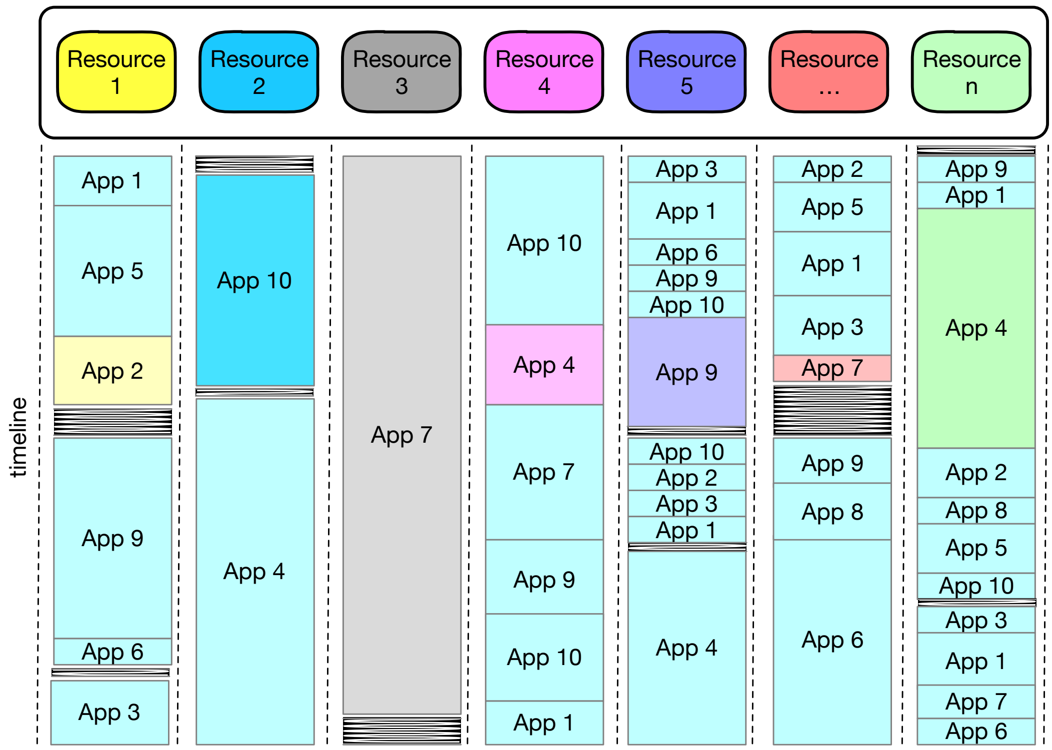
To the best of the author's knowledge, this work is the first to coin the concept of Exclusive Shared Access (ESA) to CoT resources. It is also the first to implement the concept in trading CoT setup and evaluate it using different optimisation strategies. Figure 4.5a and Figure 4.5b provide a visual illustration of ESA of an optimised map of CoT applications and resources.

4.4 Case Study

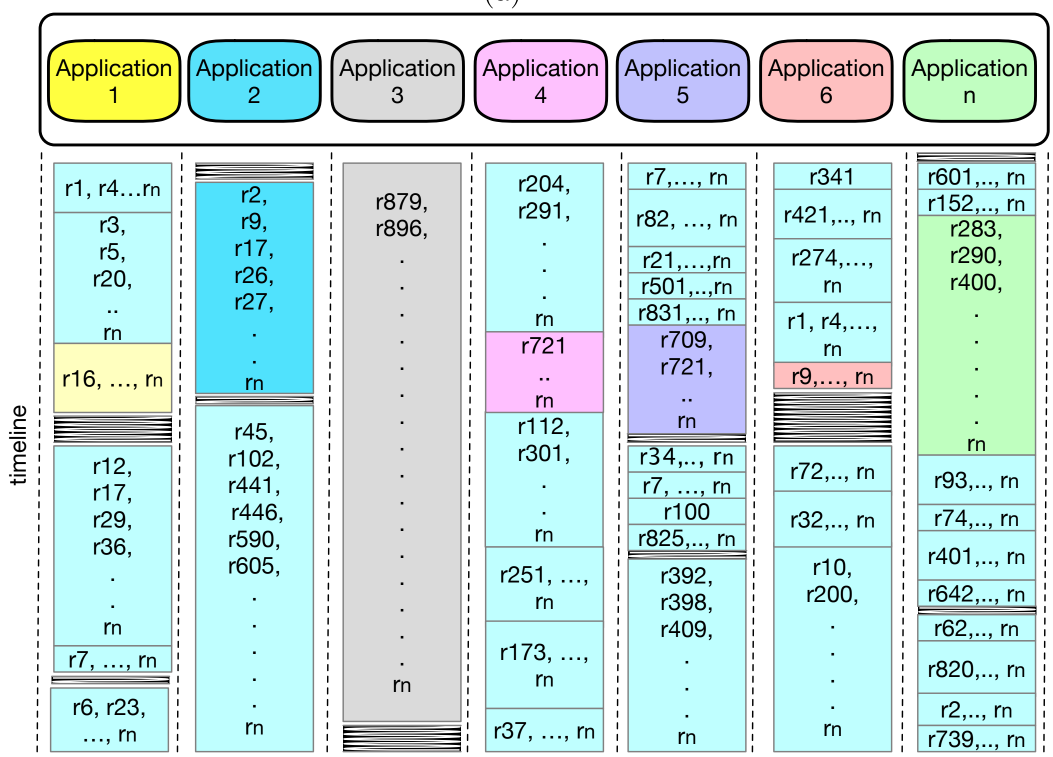
In this section, the following case study is discussed. The area around a high-traffic street of a metropolitan city is considered a desirable location for multiple enterprises and public organisations to implement their IoT applications. To elaborate, CoT application scenarios are illustrated in Figure 4.6.

4.4.1 Resource Providers

In this case study, four providers deploy their networks of IoT resources across the considered area. Each network of a provider consists of multiple homogeneous nodes. Nodes of all providers become heterogeneous when compared with each others'. Each node consists of constrained computing capabilities that may differ from one to another. This may include a microprocessor, memory, a power supply, storage, sensor, actuator and network



(a)



(b)

Figure 4.5: Two views of an optimised map of CoT applications and resources as a result of ESA (a) resource view (b) application view.

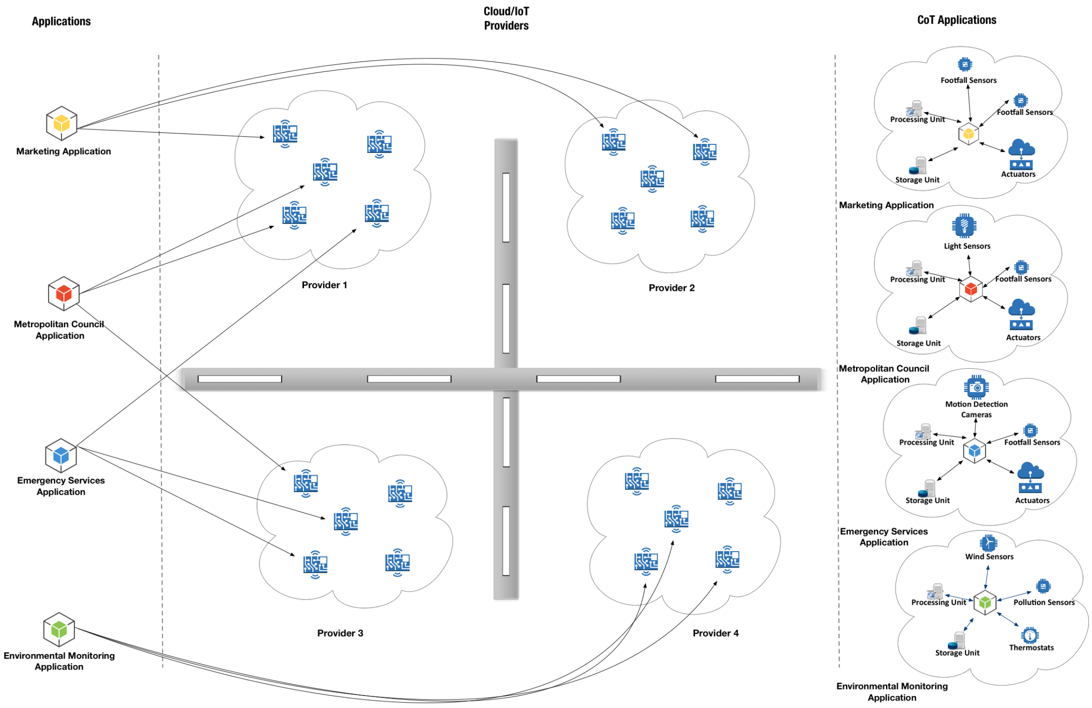


Figure 4.6: CoT application.

chip. IoT nodes are connected via their providers' area-wide wireless networks. The usage of different node types is discussed in the following section.

4.4.2 Resource Consumers

IoT resources can be consumed by a wide range of applications. Upon successful allocation of required resources, a consumer can send a software component (e.g. Java applet or Python script) to configure and utilise the acquired resources based on the application requirements. In this case study, four applications are considered as IoT resource consumers including one business and three public organisations. The four presented applications support the vision of a smart city.

Marketing Application. A marketing agency owns electronic billboards around the area wants to develop an advertising application that uses statistics of pedestrians footfall across sidewalks. The agency can use footfall statistics along with other data sources to dynamically tailor selling of the electronic billboard spaces to clients. In this case, the agency would request a resource bundle of multiple footfall sensors, specifying a location (e.g. 400m × 10m), undefined node processing power, constant energy source, undefined storage capacity, and network access, and certain security level.

Metropolitan Council Application. A metropolitan council has increasing responsibilities towards the metropolitan area of the city. The council plans to build an IoT application that can help in making better-informed decisions. Pedestrian footfall is a good indicator of human activities within the area. It can be used to plan maintenance of sidewalks and pavements as well as building new ones. Maintenance projects within the area may require an installation of temporary traffic lights to control pedestrian activities and car traffic. Footfall sensors, traffic sensors and actuators play an important role in optimising the traffic within the maintained area, especially during peak times. The sensors can measure pedestrian activities and density of the traffic while the actuators take control of traffic lights based on sensors readings. Light sensors can also be used to switch on/off street and sidewalks lights at the right time avoiding earlier or late switch on/off optimising the energy consumption and the operational costs of the lights.

For long-term planning, the request would be for a bundle of any footfall sensors within the area, minimal storage and processing capacities, minimal network connectivity and basic security features. For day-to-day tasks, the request would be for a bundle of good light sensors, footfall sensors and actuators within 500m × 500m area. The power of the resources should be consistent, with adequate storage and processing units, responsive network access and good security characteristics.

Emergency Services Application. Metropolitan emergency services including police, ambulance and fire brigade want to build an IoT application that helps their teams accelerate their response to incidents. For instance, footfall can be used for crowd tracking and analysis during public events. It also allows to plan and aid evacuation procedures during incidents. Motion detection can be employed to early discover breaches of controlled zones. Using this application, emergency services can gain high priority access to a bundle of resources for short periods of time. For planning and prediction, the resource request would be for footfall sensors and motion detection cameras in a general location, with limited power, network, access and security characteristics. For a live emergency event, the request would be for the maximum number of resources around the incident location with the maximum reliability possible.

Environmental Monitoring Application. An environment agency aims to build an application for environmental impact analysis. The application is useful for monitoring and analysing various environmental indicators (e.g. pollution, temperature, pressure, wind). These indicators help public decision-makers to control pollutions and promote environment-friendly lifestyles in the metropolitan area. The agency would request a bundle of distributed environmental sensors across the area. Footfall sensors can also help to gain a detailed picture of the environmental impact of activities in the area. As these applications are usually financially constrained, the bundle request would be submitted with minimal resources properties at the lowest price possible.

4.5 Preliminary Evaluation

This section provides two sets of preliminary experimental evaluation of the trading model proposed and the ESA approach. Evaluation is presented as follows:

4.5.1 Evaluating The Proposed Trading Approach

This section presents a proof of concept evaluation of trading CoT resources. A 3-tier marketplace system architecture is proposed to perform a set of simulations. Simulations have the following aims: 1) evaluate the feasibility of using market-based mechanisms to allocate CoT resources efficiently, 2) test various utility functions to propose candidate assignments of consumers and providers or requests and resources, and 3) evaluate the use of three optimisation techniques in CoT trading setup.

4.5.1.1 Experimental Setup

Resource allocation in CoT is formulated as an optimisation problem where different optimisation algorithms are applied including Particle Swarm

Optimisation (PSO) [80], Differential Evolution (DE) [153] and Basin Hopping (BH) [164]. These algorithms are selected for two reasons. They are gradient-free, and they are well known to solve problems similar to the problem of trading CoT resources in complexity and scalability. The optimisation problem formulated in this chapter is considered as a single-objective problem and the implementation is performed accordingly.

The marketplace is assumed to find the optimal assignment of providers to consumers based on the five utilities introduced earlier. The scenario used in all simulations in this section is presented as follows. A number of 100 providers submit their resources to the marketplace to match them with requests of 50 consumers.

Three optimisation techniques are used to find optimal solutions. The three techniques implemented without modification or improvement using Python programming language. A maximum number of 200 iterations is allowed for all techniques and swarm size of PSO is set to 100. Simulations are performed on a computer with the following hardware specifications: Processor: 2.6 GHz Intel Core i7, Memory: 16 GB 1600 MHz DDR3.

4.5.1.2 Experimental Results

This section is dedicated to discuss the results of simulations performed. Results presented in Figure 4.7 to Figure 4.11 compare optimal solutions found at the end of certain iterations.

Figure 4.7 shows the provider profit utility. It is clear that DE considerably outperforms PSO and BH respectively in maximising the profit of the provider. DE and PSO maintain a steady increase in optimised profit overtime while BH experiences a sharp increase between iteration 1 and 75 before it maintains reasonable increases to the last iteration.

Figure 4.8 illustrates the utility to minimise response time. It shows a competition between PSO and BH to minimise the response time while DE is clearly falling behind. BH takes more iterations than PSO to converge but both algorithms find the same optimal response time.

In Figure 4.9, the utility of minimising energy consumption is illustrated.

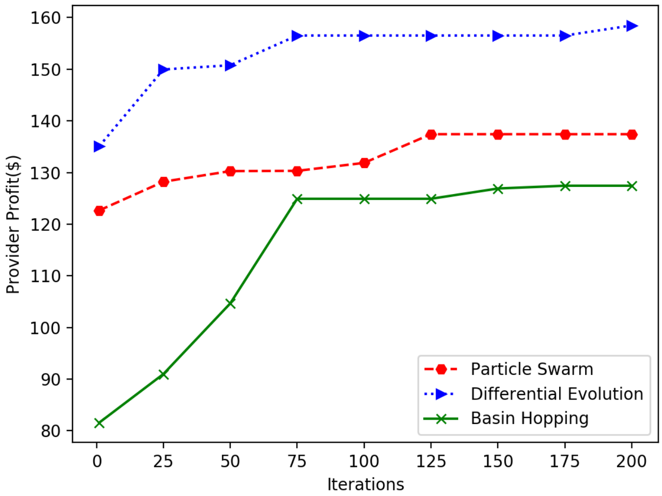


Figure 4.7: Optimisation of provider profit.

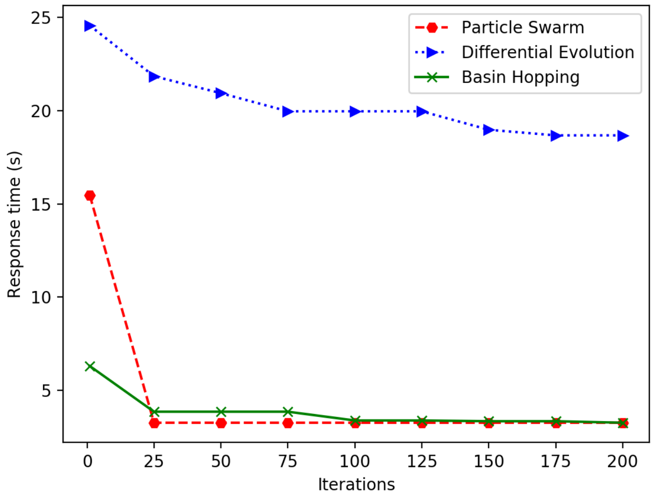


Figure 4.8: Minimising the response time.

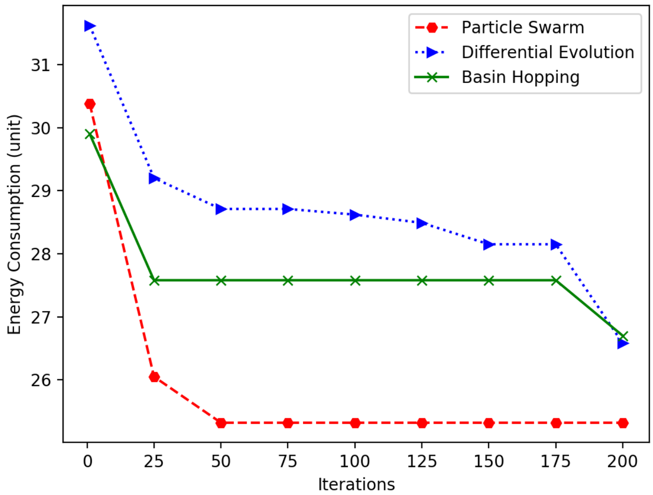


Figure 4.9: Optimising the resource energy consumption.

PSO is notably better than DE and BH. PSO minimises energy consumption and converged in a fewer number of iterations than DE and BH. It is also observed all algorithms experience sharp drops between iteration 1 and iteration 25 before starting to maintain steady decreases.

Figure 4.10 shows the utility for maximising the coverage of requested resources. PSO outperforms the others while differential Evolution falls behind again. The three algorithms have sharp increases between iteration 1 and iteration 25 before maintaining steady increases. DE seems to be trapped by a local coverage optimal value.

The utility for maximising the profit of the marketplace is shown in Figure 4.11. PSO and BH significantly maximise the profit of the marketplace than DE, but PSO outperforms the others and converges to the optimal marketplace profit. Table 4.2 summarises the utility values in terms of minimum, average and maximum values at the end of the last iteration.

Implementation issues are summarised as follows: 1) BH algorithm requires setting more parameters (e.g. temperature, step size, interval) than PSO and DE. It requires careful tuning of parameters to obtain better results. It is more complex than other algorithms applied and a bit slower in convergence, and 2)

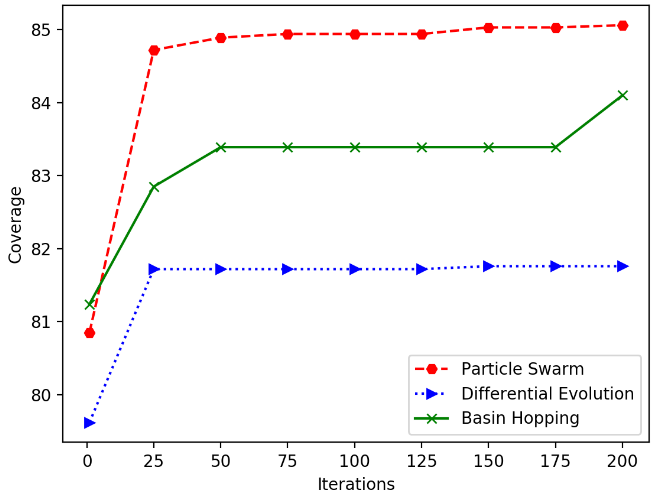


Figure 4.10: Maximising resource area coverage.

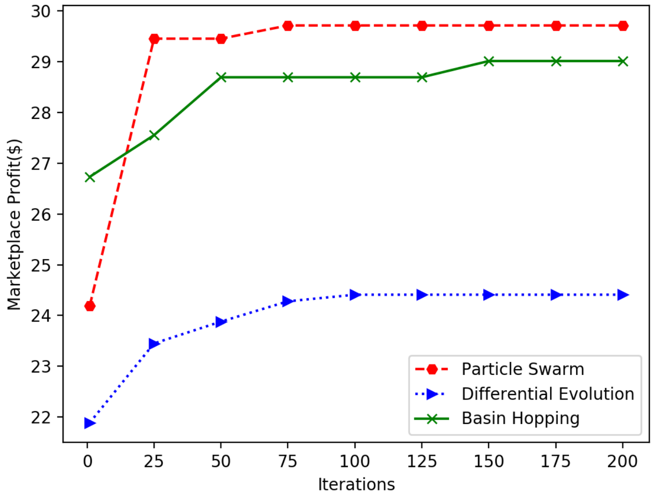


Figure 4.11: Maximising profit of the marketplace.

Table 4.2: Comparison of simulation results.

Algorithm(Utility)	**Min**	**Avg**	**Max**
PSO(P_g)	44.54	100.67	137.42
DE(P_g)	87.72	124.44	158.47
BH(P_g)	81.55	117.12	127.44
PSO(CV)	77.66	83.17	85.06
DE(CV)	73.20	77.55	81.76
BH(CV)	81.24	82.11	84.10
PSO(R_t)	3.27	4.08	20.20
DE(R_t)	18.67	25.86	32.42
BH(R_t)	3.27	4.37	6.49
PSO(E)	25.32	28.92	36.28
DE(E)	26.58	33.55	40.59
BH(E)	26.70	28.79	31.43
PSO(M_g)	21.30	29.18	29.71
DE(M_g)	18.30	21.63	24.41
BH(M_g)	26.41	27.36	29.01

Falling into local optima (minima and maxima) may not be avoidable in some situations by all optimisation techniques used in the research presented in this chapter.

4.5.2 Evaluating The Exclusive Shared Access Strategy

The section presents the experimental setup and results of evaluating the ESA strategy. The evaluation aims to validate the feasibility of the proposed ESA in optimisation-based resource allocation in CoT system.

4.5.2.1 Experimental Setup

Trading of CoT resources is formulated as an optimisation problem where different optimisation algorithms are applied including Non-dominated Sorting Genetic Algorithm II (NSGA2) [39], Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [63] and The Third Evolution Step of Generalised Differential Evolution Algorithm (GDE3) [89].

The simulated system is assumed to have 100 consumers with 100 requests each and 200 providers offering 200 resources each. The locations of all resources are randomly generated within a 100-meter radius of a busy street in the city centre of Nottingham, UK. The locations are exact Latitude and Longitude (x_j, y_j). It is assumed that each consumer requests homogeneous resources while each provider offers heterogeneous resources. The total number of requests is 10000, whereas the total number of resources is 40000. The system uses the three optimisation strategies mentioned earlier to minimise the consumer cost and maximise the coverage of the resources. The three techniques implemented without modification or improvement using Python programming language. Both objective functions are evaluated individually as a single objective function. Simulations are configured up to 250 iterations and population size of 50. Simulations are performed in a computer with the following hardware specifications: Processor: 2.6 GHz Intel Core i7, Memory: 16 GB 1600 MHz DDR3.

4.5.2.2 Experimental Results

This section discusses the simulation results obtained. Figure 4.12 and Figure 4.13 show the best results of each objective function at specific iterations. The results show that CMS-ES contributes to the optimality of consumer cost and the resource coverage better than NSGA2 and GDE3. Despite the NSGA2 complexity, it converges faster than CMA-ES but falls into the local optima in both scenarios. This can be improved by using different parameters and operators. GDE3 also requires further parameters improvement as it is the lowest contributor in both scenario.

The results assert the feasibility of the proposed approach by using various optimisation strategies as a market mechanism for trading and sharing access to CoT resources. The proposed architecture demonstrates the flexibility and scalability of the approach in optimising objectives for that require mapping of a large number of requests and resources. The use of objective functions along with proposed notations shows their flexibility and effectiveness in quantifying the value of heterogeneous CoT resources.

Simulation limitations are summarised as follows: 1) Working with optimisation approaches may require trying different values of parameters (e.g. iteration, population size, mutation rate) to obtain satisfactory results. This

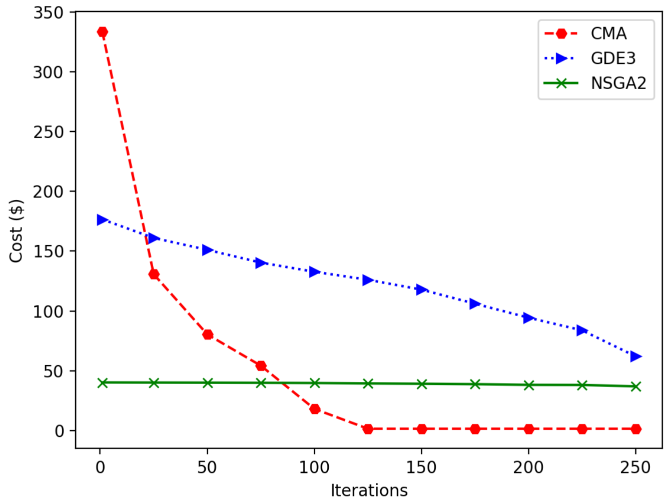


Figure 4.12: Minimisation of consumer cost.

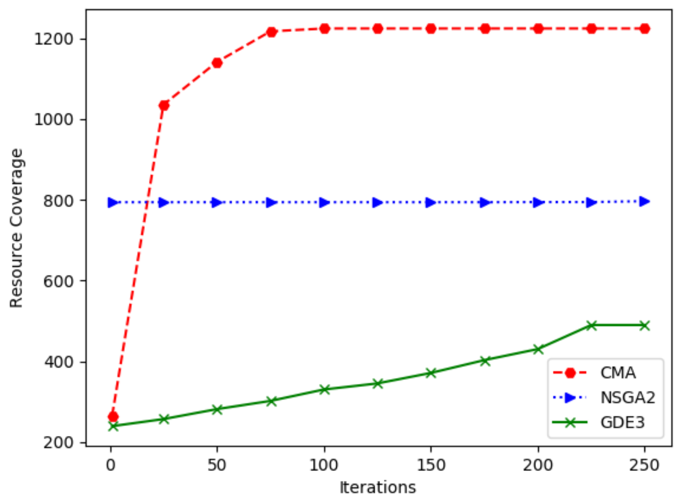


Figure 4.13: Maximisation of resource coverage.

can be computationally expensive and time-consuming, and 2) Falling into the local optima (minima or maxima) may not be preventable in some scenarios by all optimisation techniques used.

4.6 Discussion

The research presented in this chapter describes the proposed approach for trading CoT resources and enabling shared access to CoT resources in details. Section 4.2.1 presents a preliminary design of a marketplace for trading CoT resources. The marketplace design addresses the requirements for a CoT marketplace that have been surveyed in the previous chapter. A multi-attribute description model for CoT resources is introduced in Section 4.2.3 to describe heterogeneous CoT resources generically and to quantify the value of CoT resources based on their described attributes. The marketplace system employs the proposed model to define, describe CoT resources and quantify their values autonomously without intervention. Several trading objectives are proposed in Section 4.2.5 to describe the goals of the resource providers and consumers. Objectives include minimising the resource costs, minimising the response time, minimising the energy consumption, maximising the provider's profit, maximising the resource coverage and maximising the marketplace profit. Section 4.3 introduces the concept of ESA in which constrained resources can be accessed and utilised by a single application at a given time while accessed and utilised collectively by multiple applications over the time of the proposed schedule. Section 4.4 provides a real-world case study for the proposed approach including several examples for provider, consumers and CoT applications. All presented scenarios support the vision of a smart city.

Preliminary experiments are performed in Section 4.5 to validate the feasibility of the trading approach and the ESA strategy. The problem of resource allocation in CoT is presented as a single-objective trading optimisation problem. The simulation results show that the approach used in this study is promising and have several benefits. The results show the feasibility of using various optimisation algorithms as a market-based mechanism for trading CoT resources. Results also show at least one optimisation technique is able to find an optimum solution in all utilities proposed.

The approach described in this thesis also demonstrates that the proposed marketplace architecture can decrease the architectural complexity in CoT. The use of utility functions along with vocabularies proposed shows their effectiveness in quantifying the value of various CoT resources. This implies potential higher satisfaction for the requirements of CoT consumers and providers and higher utilisation of CoT resources.

Chapter 5

AMACoT: A Marketplace Architecture for Trading Cloud of Things Resources

5.1 Introduction

This chapter presents a marketplace architecture which can provide efficient resource allocation and deal with the complex issues present in the CoT. The solution presented in this thesis re-imagines CoT resources as commodities rather than as organisational assets. It considers the business model of a marketplace whereby consumers request access (lease) to providers' resources. A marketplace that potentially can automate the trading between CoT resources and CoT applications. The research presented in this chapter proposes improved marketplace architecture for trading CoT resources called AMACoT.

There are various use cases for AMACoT. For instance, an event management agency manages event facilities in a metropolitan area where it aims to improve its operational efficiency. The agency wants to develop an application that performs the following tasks: 1) find the least congested routes to exhibition centre leavers, 2) crowd monitoring of fans attending games in a nearby stadium for better incidents response, and 3) waste monitoring to efficiently automate the waste collection after organised events if needed.

Having a dedicated CoT infrastructure for this application may require a significant upfront investment. In AMACoT, the agency would request a bundle of CoT resources to perform the tasks. For instance, footfall sensors and motion detection cameras around event facilities can help organisers in guiding people to the least congested tracks. The resource bundle may include sensors, actuators, cameras and other resources. The application consumes the required resources for a specific time and then releases them back to the marketplace when lease-time elapses. In this case, the application utilises the required resources without considerable investment nor dedicated infrastructure. Similarly, providers deploy their CoT resources without being tied-up to particular applications.

The remainder of this chapter is organised as follows: trading objectives evaluated in this chapter are discussed in Section 5.2, the proposed marketplace architecture is presented in Section 5.3, Section 5.4 analyses the potential security threats to AMACoT, the experimental setup and evaluation are presented in Section 5.5, Section 5.6 concludes this chapter and describes the planned future work.

5.2 Trading Objectives

Trading objectives represent the goals of providers P and consumers C from participating in trading CoT resources. These goals are formulated as objective functions to provide significant flexibility for the trading model. Using this approach minimises the re-development efforts and costs of the system components that may be required in case of resource changes. Changes can be implemented as a new objective function without or with minimal changes in the system side.

The trading of CoT resources is presented as a multi-objective optimisation problem as follows:

$$\text{Minimise} \quad CS = \sum_{i=1}^{n}\sum_{j=1}^{m} rc_j \cdot (t_i + tq_{ij}) \cdot rp_j \tag{5.1}$$

$$\text{Maximise} \quad RU = \sum_{i=1}^{n}\sum_{j=1}^{m} pu_i \cdot (ac_j - uc_i) \tag{5.2}$$

$$\text{Minimise} \quad P_{lock} = \sum_{j=1}^{m} py_j + pt_j \tag{5.3}$$

$$\text{Maximise} \quad PR = \sum_{i=1}^{n}\sum_{j=1}^{m} rc_j \cdot (t_i + tq_{ij}) - mc_j \tag{5.4}$$

$$\text{subject to} \quad 0 < cs_j \leq b_i \tag{5.5}$$

$$0 < Er_i \leq Ep_j \tag{5.6}$$

$$se_i \geq se_j \tag{5.7}$$

$$rp_i \geq rp_j \tag{5.8}$$

$$ra_i \geq ra_j \tag{5.9}$$

where $i = 1, ..., n; j = 1, ..., m$ for Constraints 5.5, 5.6, 5.7, 5.8 and 5.9. Descriptions of the objectives and constraints are provided below:

Objective 1: Resource Cost. Minimising the resource cost is one of the usual motivations for the consumers. Consumers are likely to bid for minimal cost resources. The cost objective function is presented in Objective 5.1. The following contributors to the total cost CS are considered when minimising the cost of requested resources. Let b_i be the bid from a consumer and cs_j the provider's cost. The initial cost rc_j can be calculated as $rc_j = (b_i - cs_j)$. The requested utilisation time of a resource is set to t_i while TQ_{ij} denotes the estimated transmission and delay time. Provider reputation rp_j is set based on the credibility measures of the marketplace to determines the trustworthiness of the provider. The reputation is assumed as part of the resource cost for two reasons. 1) It enables the marketplace to use any feedback mechanism that allows consumers to rate their providers' trustworthiness. 2) The reputation of the provider has an indirect impact on the cost of resources. A provider with a higher reputation is enabled to offer its resources with better cost than a low-rated provider.

Objective 2: Resource Utilisation. CoT applications are assumed to monopolise access to a set of resources for a given time. This can cause low utilisation of the allocated resources due to the light requirements of some applications. For trading CoT resources to be efficient, resource utilisation has to be optimised. Resource utilisation objective function is presented in Equation 5.2. The objective considers the requested resource utilisation time pu_i, the available resource components ac_j and the actually utilised components of a resource uc_i. Since pu_i has a different measurement unit than ac_j and uc_i, the three variables are re-scaled to the same numerical range.

Objective 3: Provider Lock-in. Objective 5.3 presents the objective of minimising provider lock-in. Vendor lock-in is a common challenge for commoditised computing services. It describes the situation where consumers can not migrate their data or applications to different providers due to various reasons. This objective aims to minimise the lock-in by considering the provider policy py_j that enables consumers to migrate and the proprietary technologies of the provider pt_j. Both factors are rated from $1 - 5$, where 1 is the most flexible policy towards consumer migration and lowest proprietary technologies that may hinder consumers from migrating to different providers.

Objective 4: Provider Profit. Providers always aim to maximise their profit PR. Equation 5.4 presents the profit objective function. This can be achieved by maximising the cost of resources rc_j and their utilisation time t_i while considering the marketplace charges mc_j as expenses.

In addition to the objectives, constraints are used to identify feasible solutions to the resource trading problem. This significantly minimises the search space of such scalable and complex set of candidate solutions. Constraint 5.5 illustrates that costs and bids have to be positive, and bids are always greater than or equal resource costs. The energy constraint presented in Constraint 5.6 ensures the required energy Er_i to perform application tasks does not exceed the available resource energy Ep_j. Constraint 5.7 specifies the security requirements of the application se_i to be satisfied by the security capabilities of the resource se_j. Constraint 5.8 provides credibility insurance to the marketplace participants based on their performance. Providers have to maintain a certain reputation level rp_j in the marketplace while consumers specify their providers' credibility requirements rp_i. Constraint 5.9 enables participants to specify participant-specific requirements or limitations in responding to some applications or resources attributes. This constraint is part of this approach genuineness and flexibility to handle heterogeneous CoT resources and applications.

Table 5.1: Snapshot of CoT application requirements.

Requirements	**Example Value**
Processing	$\geq$ 1GHz
Memory	$\geq$ 1GB
Storage	Any
Network	Heterogeneous
Energy	Battery/Permanent
Sensing	Environmental
Actuator	0
Security	$\geq$ Basic
Location	[52.95610793607633, -1.1453494058431906]
Provider's Rating	$\geq$ 3/5
Budget	$\leq$ $10 per hour

5.3 The Marketplace Architecture

The final design of AMACoT is illustrated in Figure 5.1. The architecture is structured into four functional layers as follows. Submission layer represents the marketplace entry point where participants are authenticated and granted authorised access to trade.

The mapping layer consists of resource and request managers. Resource manager provides interfaces that enable resource providers to submit, update and remove their resource specifications. Resources are described and quantified based on the description model discussed in Chapter 4. Resource descriptions include connectivity options and resources are assumed to be connected already to the Internet via IoT gateways. Similarly, the request manager's interfaces receive application requirements from consumers. Application requirements are high-level descriptions of the computing and budget needs as illustrated in

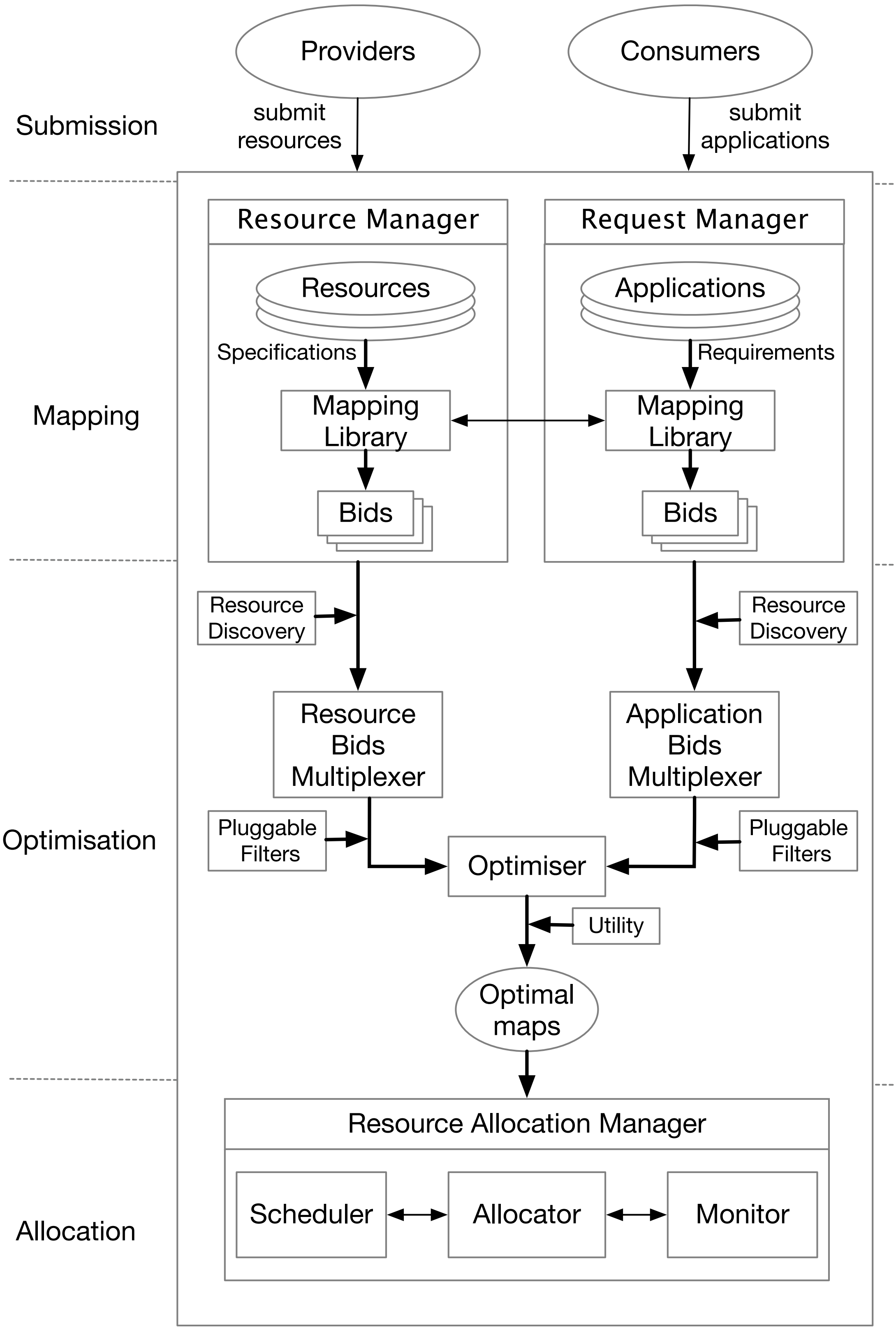


Figure 5.1: AMACoT marketplace architecture.

Table 5.1. Consumers can also update and remove their applications using the request manager. Mapping libraries of both resource and request managers provide early local coordination to turn resource specifications and application requirements into bids.

Optimisation layer represents the operational tier of the system. Applications can discover resources that are already stored in the resource manager. Bids flow from resource and request managers to multiplexers where they are selected to form bundles. Resource bundles represent a set of resources from multiple providers that can potentially be utilised by multiple applications. Whilst being forwarded to the optimiser, resource bundles and application requests are filtered. Pluggable filters include a wide range of filtering criteria such as location, resource coverage, computing and energy requirements. Filtered resource bundles and application requests are forwarded to the optimiser. The optimiser performs a two-stage process as follows: 1) construct optimal maps that consist of resource bundles and application requests ready for allocation, 2) evaluate the optimal maps based on the participants' goals using utility functions presented in Section 5.2. One optimal map is forwarded to the resource allocation manager for resources to be allocated to the applications.

Allocation layer consists mainly of the resource allocation manager. The scheduler manages the utilisation time of the resources based on the application requirements. It also coordinates with the allocator to enable resources joining the application network and dis-joining when the lease-time elapses. The monitor captures resource allocation events in real-time and provides interfaces where consumers and providers oversee their transactions.

5.4 Threat Analysis

The marketplace system should enforce different security measures to secure its operations. Security threats are analysed using the STRIDE model [64] to help the design of the architecture by identifying potential threats. The STRIDE model is used due to its maturity among other threat modelling techniques and due to its simplicity. Table 5.2 illustrates the STRIDE threats, security propriety violated and the impacted layers of the proposed architecture.

As Table 5.2 shows, each layer of AMACoT components may be impacted by one or more type of threats. At submission layer, an attacker can illegally gain access and use a consumer's or a provider's credentials to access the marketplace. AMACoT can mitigate this threat by using authentication protocols that do not require a password or use signed certificates to verify the authenticity of consumers and providers. The attacker can also tamper at the submission layer by maliciously modify a consumer's requests or a provider's offerings. These types of threats may occur using bit-flipping or injection attacks. AMACoT can mitigate these attacks by integrating adequate users' input and output validation tools for proper data integrity validation. Submission layer is also susceptible to Denial of Service (DoS) attacks where the attacker aims to interrupt the marketplace making it unavailable or unstable to providers and consumers. This can occur when the system is flooded with a large number of concurrent requests. The security manager can alleviate DoS attacks by employing requests and offers limiter to maintain the number of submissions at an acceptable level.

The mapping layer can be vulnerable to the threats of information disclosure and elevation of privileges. Information disclosure threats the confidentiality of marketplace users when the attacker maliciously gets hold of the users' sensitive data stored in the resource manager and/or the request manager. AMACoT can use a common practice to mitigate this threat by encrypting users' sensitive data. Elevation of privileges also poses a considerable risk at the mapping layer. An attacker can attempt to gain some privileges that enable him to perform some actions that he cannot achieve. This may include manipulating bids at either resource or requests manager or both. The system should implement robust authorisation techniques and operate the components at the mapping layer using

Table 5.2: STRIDE model of AMACoT

	Threat	Property Violated	Impacted Layer			
			Submission	Mapping	Optimisation	Allocation
S	Spoofing identity	Authentication	•			
T	Tampering with data	Integrity	•			•
R	Repudiation	Nonrepudiation				
I	Information disclosure	Confidentiality		•	•	
D	Denial of service	Avaiability	•		•	
E	Elevation of privilege	Authorisation		•	•	

non-root users. The optimisation layer can also be vulnerable to both threats, in addition to the DoS attacks. AMACoT can prevent such attacks using the same mitigating mechanisms discussed earlier for the submission and mapping layers.

The resource allocation manager is vulnerable to data tampering. This may occur when a user tries to manipulate a schedule before resources are allocated to take advantage of other users. Mitigation may include validating users' input/output to detect and prevent data tampering.

Although the security aspects are crucial to the marketplace architecture, the focus of the work presented in this chapter is on demonstrating the feasibility and performance of AMACoT in CoT resource allocation using optimisation algorithms. For deployment of this architecture if it is important to take security as the aim of the study, it would be necessary to take standard IoT security precautions such as those identified in [176]. Those precautions and any deployment of specific security mechanisms, therefore, fall out of this thesis's scope.

5.5 Experimental Evaluation

This section presents the experimental evaluation of the final design of AMACoT architecture. After the description of the experimental setup in Section 5.5.1, Section 5.5.2 provides system verification, aiming to evaluate the system footprint and Section 5.5.3 presents algorithmic evaluation of the proposed AMACoT approach using different optimisation algorithms.

5.5.1 Experimental Setup

The architecture is developed using Python in a computer with 2.3 GHz Xeon processor, a 125GB memory and Linux OS. In order to simulate the behaviour of distributed systems, Python Remote Object (Pyro4) is used to connect the main components of the system as well as consumers (applications) and providers (resources).

The optimisation component of AMACoT integrates optimisation algorithms to map requests to resources and to evaluate the optimal resource allocation. The following optimisation algorithms are implemented as follows:

NSGAII. Non-dominated Sorting Genetic Algorithm II [39] is an improved genetic algorithm that is widely used in real-world multi-objective optimisation applications. The population size is set to 200 with a maximum number of 200 iterations for all experiments.

NSGAIII. This algorithm is an extension of the NSGAII that uses reference points to diversify the Pareto points during the search [38]. Besides the same settings used for NSGAII, the number of divisions is set to 12.

SPEA2. Strength Pareto Evolutionary Algorithm 2 [181] is designed and used to optimise combinatorial problems. The population size is set to 200 with a maximum number of 200 iterations for all experiments. The three algorithms are chosen due to their capabilities in optimising similar problems to the trading CoT resources, their scalability in optimising very large number of candidate solutions and their low computational cost as they are derivative-free.

5.5.2 Implementation Verification

Stress tests are performed to evaluate the footprint of the system components when they interact with each other as well as interacting with providers and consumers. Three experiments are performed using three different scale factors as shown in Table 5.3. The scale factors aim to evaluate the scalability of AMACoT system and measure the performance overheads generated. In these three experiments, the SPEA2 algorithm is used to minimise the resource cost while maximising the provider profit. This evaluation measures the following system footprints; 1) CPU usage, 2) memory usage, 3) latency that is measured

Table 5.3: Simulated marketplace participants.

Parameter	**Experiment 1** (scale factor 1)	**Experiment 2** (scale factor 2)	**Experiment 3** (scale factor 3)
Number of Requests	10K	20K	30K
Number of Resources	200K	400K	600k
Number of Consumers	100	200	300
Number of Providers	100	200	300

from the time of request submission to the time of resource allocation confirmation, and 4) throughput to measure the number of requests and resources handled by AMACoT over the trading time.

Experiment 1. 10,000 requests and 200,000 resources are submitted to AMACoT by 100 consumers and 100 providers respectively. Experiment 1 requires 10% of CPU, 3GB of memory and 57 second to produce an optimal resource allocation. AMACoT handles 175 requests and over 3500 resources per second.

Experiment 2. 20,000 requests and 400,000 resources are submitted to AMACoT by 200 consumers and 200 providers respectively. Experiment 2 consumes 11% of CPU, 7GB of memory and 119 sec. to produce an optimal resource allocation. The maximum throughput of this experiment is 168 requests and 3361 resources per second.

Experiment 3. 30,000 requests and 600,000 resources are submitted to AMACoT by 300 consumers and 300 providers respectively. The peak CPU load of Experiment 3 is 13% while 11GB of memory used. Producing an optimal resource allocation requires 185 sec. for Experiment 3. AMACoT processed 162 requests and 3243 resources per second.

The results of the verification tests are summarised in Table 5.4. Results show that CPU usage increases from 10% in Experiment 1 to 11% in Experiment 2 when experiment 2 scales up by 100%. The CPU load also increases from 11% in Experiment 2 to 13% in Experiment 3 that scales up by 18%. This implies a reasonable CPU usage when marketplace participants increase significantly. Memory usage is also measured for the three experiments as follows: Experiment 1 requires $3GB$ of memory, $7GB$ for Experiment 2 and $11GB$ for Experiment 3. Memory consumption increases from $3GB$ in Experiment 1 to $7GB$ in Experiment 2, when the marketplace participants rise by 100%. In Experiment 3, the memory consumption increases up to $11GB$ when the experiment scales up by further 100%. Results indicate a fair memory usage across the three experiments when different scale factors are considered.

Latency relies significantly on three aspects as follows: 1) the optimisation algorithm used, 2) the complexity of optimised objectives and 3) the number of optimised objectives. The latency results presented are obtained from SPEA2 algorithm optimising two objectives. The latency of experiment 2 is about 2 times the latency of experiment 1 while the latency of experiment 3 is about 1.5 times the latency of experiment 2. This implies that the latency is doubled as the experiment scales up by 100%. The throughput results show that request throughput decreases 4% only when the marketplace participants rise by 100% in experiment 2 from experiment 1. The request throughput declines 3.5% further in experiment 3 when compared to experiment 2. The resource throughput in experiment 2 shows 4% reduction in comparison to experiment 1 while it decreases 3.5% in experiment 3 when compared to experiment 2. The overall evaluation of throughput shows sensible throughput variations across the three experiments considering the three scale factors.

5.5.3 Algorithmic Evaluation

In order to evaluate the performance of the optimisation algorithms used and the quality of their optimal solutions, a set of ten experiments are performed using the same scale factor of Experiment 3. The following experiments optimise a single objective and compare the results of the three algorithms used.

Experiment 4. This experiment minimises the resource cost as presented in Objective 5.1. Figure 5.2 shows the comparative evaluation results for resource cost optimality. All algorithms compete towards optimal solutions but NSGAIII and SPEA2 find better cost than NSGAII.

Experiment 5. This aims to minimise the possibility of provider lock-in as presented in Objective 5.3. Figure 5.3 illustrates that NSGAII and NSGAIII algorithms converged into an optimal solution that is approximately 24% lower than the solution of SPEA2.

Table 5.4: AMACoT performance comparison.

Parameter	Experiment 1	Experiment 2	Experiment 3
Peak CPU(%)	10	11	13
Peak memory(GB)	3	7	11
Latency(sec)	57	119	185
Throughput(Request/sec)	175	168	162
Throughput(Resource/sec)	3508	3361	3243

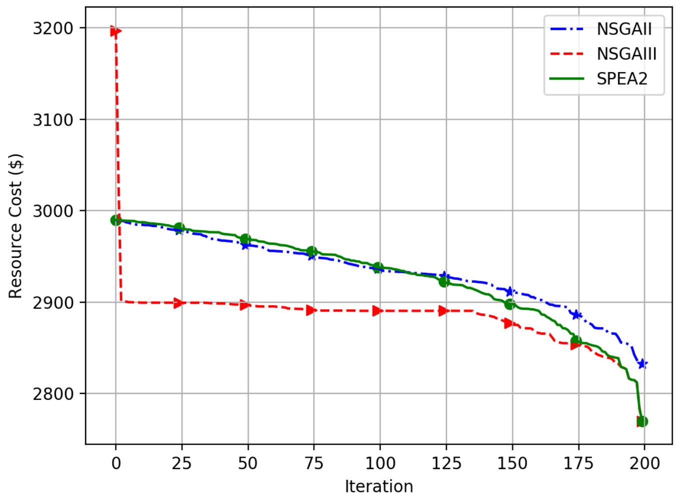


Figure 5.2: Optimising the resource cost at the end of each iteration.

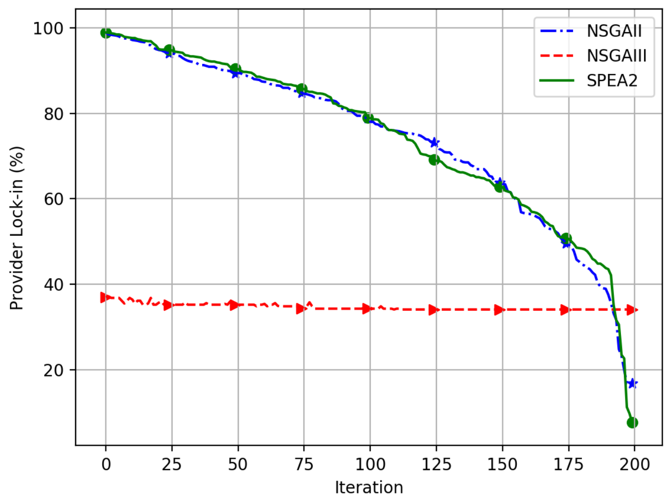


Figure 5.3: Optimising the provider lock-in utility at the end of each iteration. The provider lock-in rate is minimised, so consumers avoid being locked-in using resources from a single or very few providers.

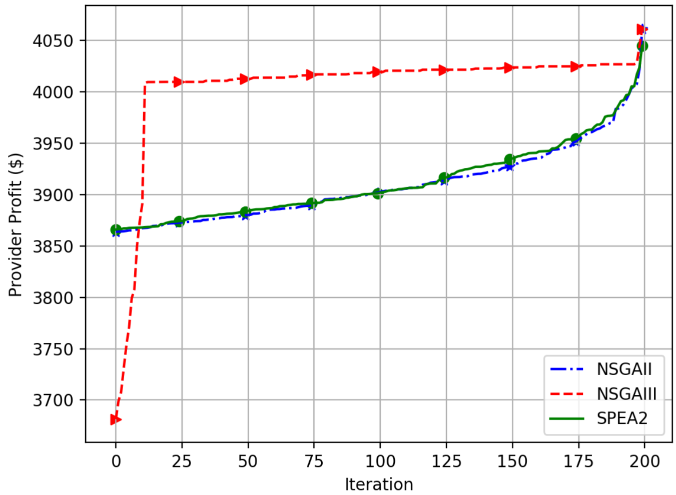


Figure 5.4: Optimising the provider profit at the end of each iteration.

Experiment 6. This experiment is intended to maximise the provider profit as described in Objective 5.4. Figure 5.4 demonstrates the competition between NSGAII and SPEA2 in which both algorithms take approximately the same direction to the optimal profit. In contrast, NSGAIII improves its solutions significantly during early iterations and maintain steady improvements towards the last iteration. NSGAII and NSGAIII provide slightly better profit for providers than SPEA2.

Experiment 7. Resource optimisation is performed in this experiment to maximise the resource utilisation by consumers as presented in Objective 5.2. Figure 5.5 illustrates NSGAII outperforms other algorithms in maximising the resource utilisation. In contrast to the other algorithms, NSGAIII shows insignificant changes throughout the process.

Experiments 4-7 perform the standard optimisation of a single objective that may not be practical for many real-world CoT applications. CoT applications often involve multiple objectives and therefore require multi-objective optimisation to find optimal solutions for two or more objectives including conflicting ones (e.g. minimising resource cost while maximising resource utilisation). This conflict is commonly addressed by using the Pareto

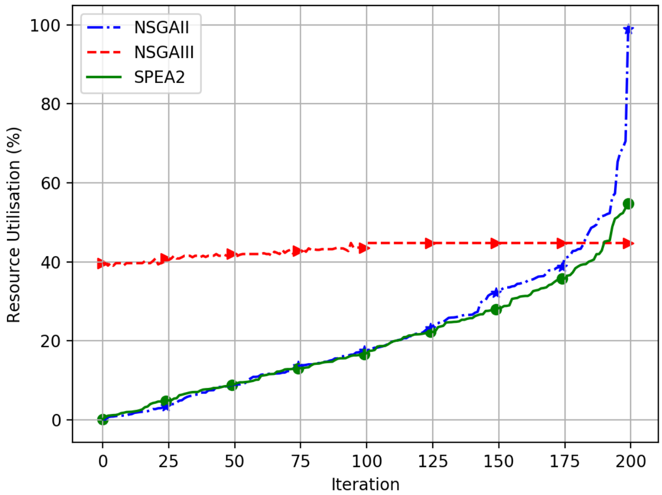


Figure 5.5: Optimising the resource utilisation at the end of each iteration.

approach [163] to evaluate a set of trade-off solutions. The following experiments show the progression of optimising multiple objectives using different approaches.

Experiment 8. This experiment optimises resource cost and provider profit as presented in Objectives 5.1 and 5.4, respectively. A way of performing this is to aggregate both objectives into a single one using weight factors as follows:

$$\text{Minimise} \quad AU = w_1 CS - w_2 PR \tag{5.10}$$

$$\text{subject to} \quad 5.5, 5.6, 5.7, 5.8, 5.9 \tag{5.11}$$

where w_1 and w_2 are the weights for resource cost and provider profit respectively. Weighting factors are used to prioritise objectives in weighted sum optimisation approaches. To maintain the balance between the two objectives, the values of w_1 and w_2 are set equally to 0.5 where $w_1 + w_2 = 1$. Figure 5.6 shows that NSGAII and SPEA2 outperforms NSGAIII despite their starting points.

It is worth noting that prioritising objectives for CoT applications is challenging as it is for many applications for the following reasons. First, it requires prior knowledge of the problem to assign appropriate weights. This

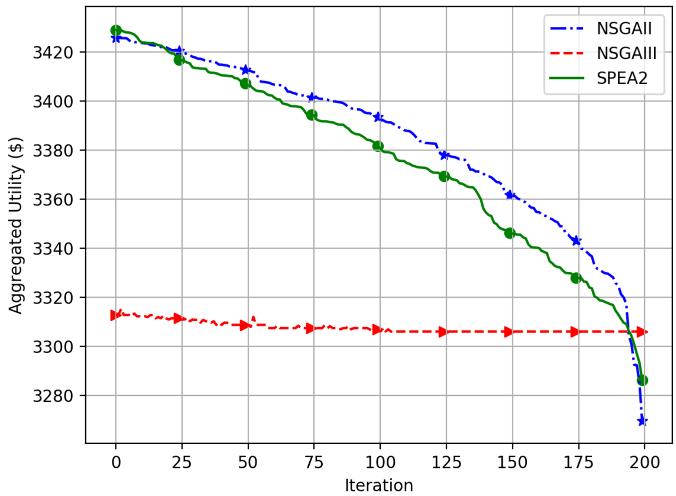


Figure 5.6: Optimising the aggregated utility of resource cost and provider profit.

prior knowledge may not always be available to CoT applications. Second, optimisation objectives are application-specific and therefore need re-prioritisation more frequently using weighting factors based on the requirements of the application. Third, this approach yields one optimal solution only, which gives the decision-maker no other solutions to the problem. Although this method may benefit specific applications with prior knowledge about the problem, the following experiments consider using Pareto approach [163] to evaluate a set of optimal solutions rather than one solution only. Experiments 9-12 evaluate conflicting bi-objectives that reflect real-world business requirements. Using the Pareto approach produces a set of optimal solutions for both objectives where an optimal solution of an objective does not worsen the solution of the other objective. Using this approach aims to maintain the balance among conflicting objectives of consumers and providers.

The challenge is how to measure the quality of Pareto-generated solutions of different optimisation algorithms. To overcome this shortcoming, each set of optimal solutions produced in the following experiments is evaluated using the Hyper-volume Indicator (HV) [180]. HV measures the size of the covered space by the generated set of Pareto solutions from a reference point. A higher value of HV indicates a better distribution of the Pareto solutions and approximately closer to the optimality.

Experiment 9. This experiment optimises resource cost and provider profit as presented in Objectives 5.1 and 5.4, respectively. This experiment aims to provide a fair optimal solution for both consumers and providers where the marketplace is not biased in favour of any of the participants in the trade. Figure 5.7 shows there is an insignificant difference among the optimal solutions of the three algorithms. The HV values of NSGAII, NSGAIII and SPEA2 are $0.57, 0.56$ and 0.62, respectively as shown in Figure 5.12. This implies that SPEA2 generates slightly better optimal solutions for minimising resource cost and maximising provider profit.

Experiment 10. Another business requirement for CoT applications is to optimise resource cost and resource utilisation benefiting the resource consumers. Experiment 10 is intended to minimise the resource cost and maximise the resource utilisation as described in Objective 5.1 and 5.2, respectively. As illustrated in Figure 5.8, all algorithms provide multiple solutions that maintain a balance between minimising the cost and maximising the resource utilisation. It can also be noted that the resource cost increases as the resource utilisation increases. This implies there is a trad-off between resource cost and utilisation, as resource providers may enable higher resource utilisation with higher cost. Figure 5.12 shows an overall high HV indicator for the three algorithms with slight differences among them.

Experiment 11. This experiment optimises provider profit and provider lock-in as presented in Objectives 5.4 and 5.3, respectively. It aims to benefit both providers and consumers by maximising the provider profit while minimising the chance of consumers being locked in one or very few providers' infrastructures. Figure 5.9 shows that NSGAII and SPEA2 provide numbers of solutions that are approximately twice as NSGAIII does. NSGAIII, however, generates a similar distribution of Pareto-generated solutions. Figure 5.9a and 5.9c, respectively, illustrate that NSGAII and SPEA2 find over 50% of the solutions with provider lock-in rate of 40% or more. This may indicate providers' preference of locking consumers to maximise the profit. Figure 5.12 shows that NSGAII outperforms NSGAIII by 7% and SPEA2 by approximately 10%.

Experiment 12. This experiment addresses the requirement of CoT applications to optimise resource utilisation and provider lock-in as presented in Objective 5.2 and 5.3. Figure 5.10 shows that all algorithms produce over 60% of their solutions with provider lock-in rate of 30% or more. This may imply that resources with high utilisation rates are associated with high chances of provider lock-in. Figure 5.12 illustrates that all algorithms attain similar HV values. This suggests a similar performance of the algorithms in finding the optimality of resource utilisation and provider lock-in utility.

Experiment 13. This experiment aims to optimise resource cost, resource utilisation, provider lock-in and provider profit and as described in Objectives 5.1, 5.2, 5.3 and 5.4. It explores the potential optimality of multiple conflicting objectives as well as the performance of the optimisation algorithms. Visualising the Pareto fronts of large-dimensional multi-objective optimisation problems is known to be a challenge [86]. One of the ways to visualise the results of this experiment is to use the scatter plot matrix as illustrated in Figure 5.11.

In this experiment, Pareto fronts can be identified as shown in Figure 5.11d, 5.11e and 5.11f. Figures 5.11a, 5.11b and 5.11c show the solutions scattered across the solution space while Pareto fronts are not typically formed yet. This can be clearly seen in Figure 5.11b as a typical front should be formed towards the left side of both axes when both objectives are minimised. This may imply the following: 1) generating Pareto optimal solutions is still possible in the case of a high-dimensional optimisation problem, and 2) the optimiser parameters may need to be improved to address the increase in the number of objectives. The number of iterations and the population size have not been changed in this experiment to be consistent with other experiments performed with the same parameters.

Experiments presented in this section demonstrate the progression of optimising various objectives. Experiments 4-7 optimise a single objective, Experiment 8 optimises multiple objectives using the weighted sum method while Experiments 9-12 optimise multiple objectives using the Pareto approach. Experiment 13 optimises all Objectives presented earlier using Pareto approach. Experiments 4-8 generate one optimal solution each while Experiments 9-13 provide a set of optimal solutions each. The evaluation of Experiments' results using HV indicator suggests the following. Experiment 8 (weighted sum approach) produces the lowest HV score. This is likely because the approach produces one optimal solution only that cannot contribute to the volume calculation. In Experiments 9-13, HV values indicate that NSGAII performs better than the other algorithms on three experiments while SPEA2 outperforms others on two experiments.

5.6 Discussion

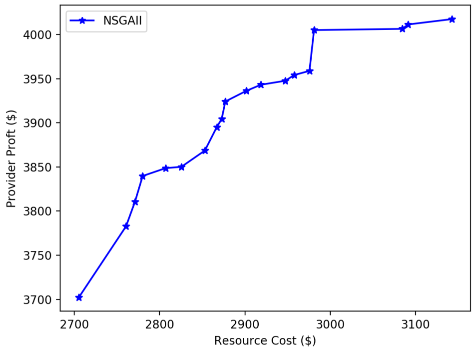
This chapter presents the design of AMACoT marketplace architecture which is built on the trading approach for CoT resources. This consists of a multi-attribute description model for CoT resources, the trading objectives and the preliminary design of the marketplace architecture presented earlier in Chapter 4. In contrast to other approaches, the proposed approach separates between CoT application development and hardware deployment considering CoT resources as commodities. Experimental evaluation validates the system and algorithmic performance. AMACoT generates optimal solutions using single optimisation, weighted sum and Pareto fronts approaches. The optimality of resource cost, provider lock-in, resource utilisation and provider profit is evaluated. The HV indicator is used to measure the performance of the optimisation algorithms and assess the quality of the optimal solutions they produced.

The performance results presented in Section 5.5.2 show reasonable system overheads and demonstrate good scalability. AMACoT incurs insignificant CPU and memory overheads when marketplace participants are doubled in Experiment 2 and tripled in Experiment 3. AMACoT also maintains a good level of throughput with a minimal reduction below 5% across all the three experiments. The overall stress results imply the advantage of reducing the architectural complexity in CoT by using an optimisation algorithm as the core of the trading manager rather than a specific-purpose system component.

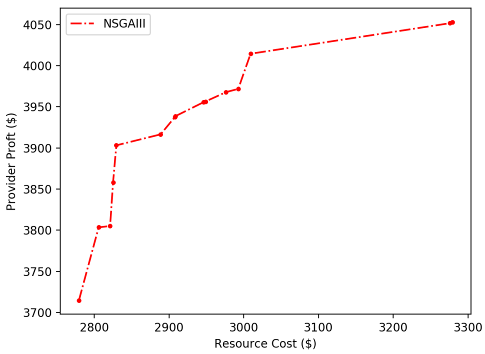
Latency becomes a limitation for AMACoT performance with respect to scalability. The latency increases by approximately two-fold when resources and requests are doubled. This may imply the dependency of optimisation algorithms on the hardware setup. Performing the same experiments in higher hardware specifications may reduce the latency of the system significantly. Running the optimiser on multi-processing setup can also provide further improvement to the system latency.

The evaluation of optimisation algorithms presented in Section 5.5.3 provide comparative results of the algorithms performance and the optimality of their solutions. The evaluation validates the use of objective functions in quantifying the value of CoT resources. This implies the heterogeneity of CoT resources and the dynamic requirements of CoT consumers can be formulated as objective functions that are likely to be optimised. It can be concluded that using optimisation-based approaches as market mechanisms for CoT resource is feasible and promising.

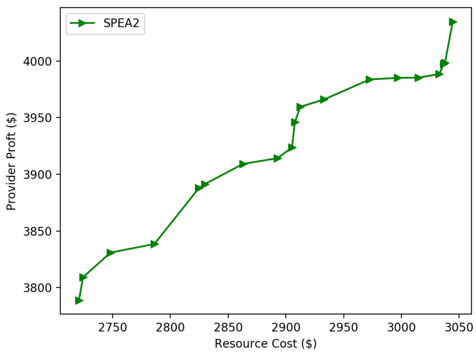
In order to address the limitations of the proposed marketplace system, the following future work is planned. The adaptivity requirements of trading CoT resources will be investigated. Improving the system latency by potentially performing the optimisation on multiprocessor setup and exploring further trading objectives to address more consumer and provider requirements (e.g. QoS attributes).



(a)

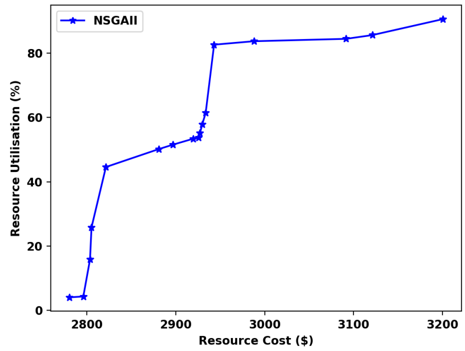


(b)

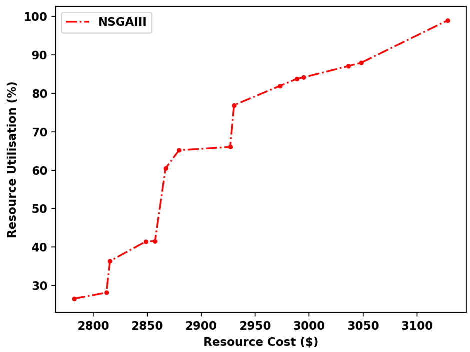


(c)

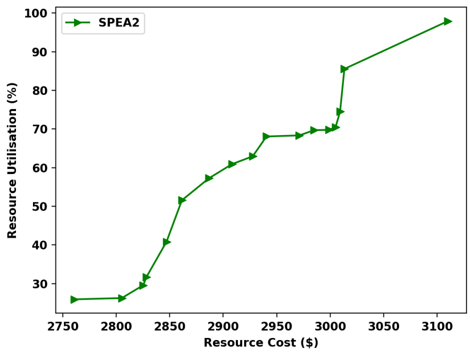
Figure 5.7: Optimising the resource cost and the provider profit. NSGAII produces the largest set of solutions, the lowest resource cost and the lowest provider profit. NSGAIII provides lowest set of solutions, the highest resource cost and the highest provider profit. SPEA2 yields various optimal solutions that maintain the balance when compared to the bi-objective optimality of the other algorithms.



NSGAII

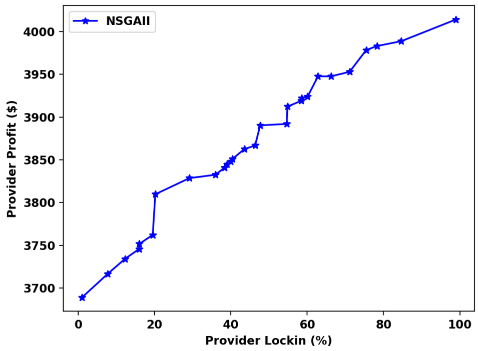


NSGAIII

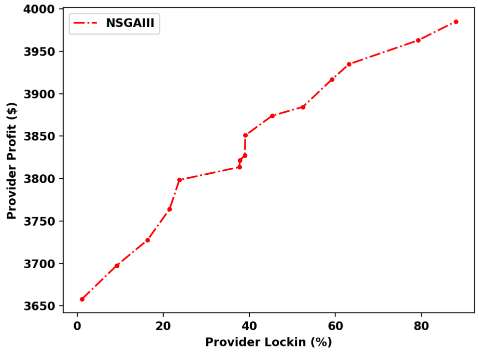


SPEA2

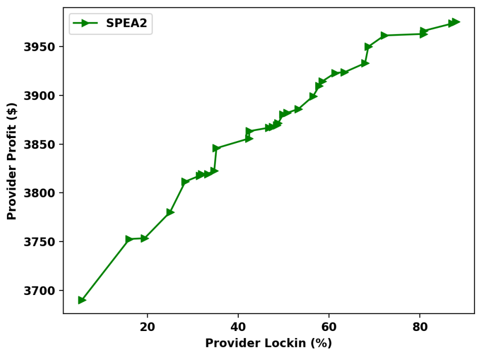
Figure 5.8: Optimising resource cost and resource utilisation. NSGAII generates the lowest resource utilisation and the most expensive resource cost. NSGAIII produces the maximum resource utilisation while SPEA2 yields the minimum resource cost.



NSGAII

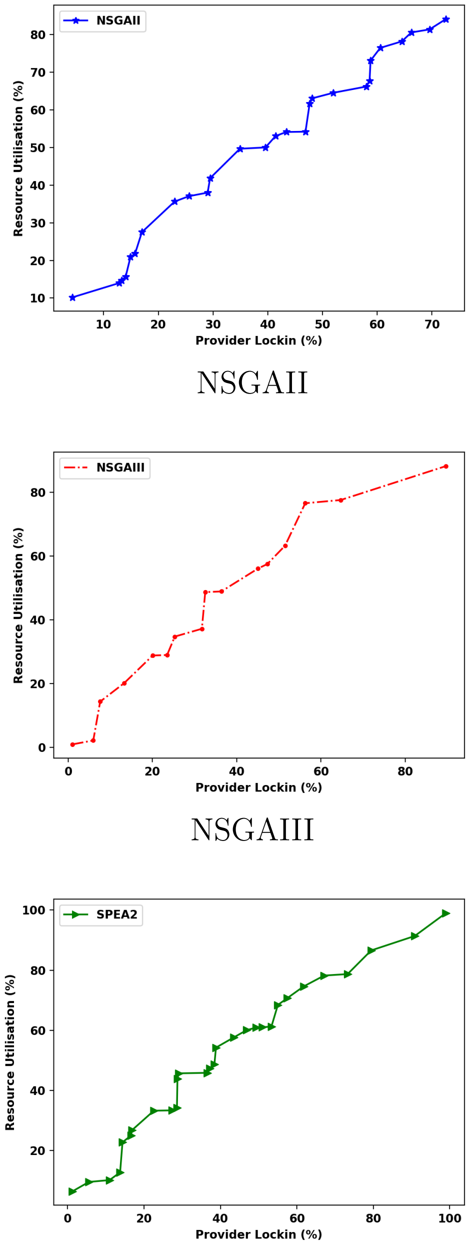


NSGAIII



SPEA2

Figure 5.9: Optimising provider profit and provider lock-in. NSGAII provides the largest set of solutions, the the maximum provider profit and the highest provider lock-in. NSGAIII produces the smallest set of solutions and the minimum provider lock-in. SPEA2 maintains the balance between the number of generated solutions, the minimum provider lock-in and the maximum provider profit.



NSGAII

NSGAIII

SPEA2

Figure 5.10: Optimising resource utilisation and provider lock-in. NSGAIII provides the smallest set of solutions, the lowest resource utilisation and the minimum provider lock-in. SPEA2 produces the largest set of solutions, the maximum resource utilisation and the highest rate of provider lock-in.

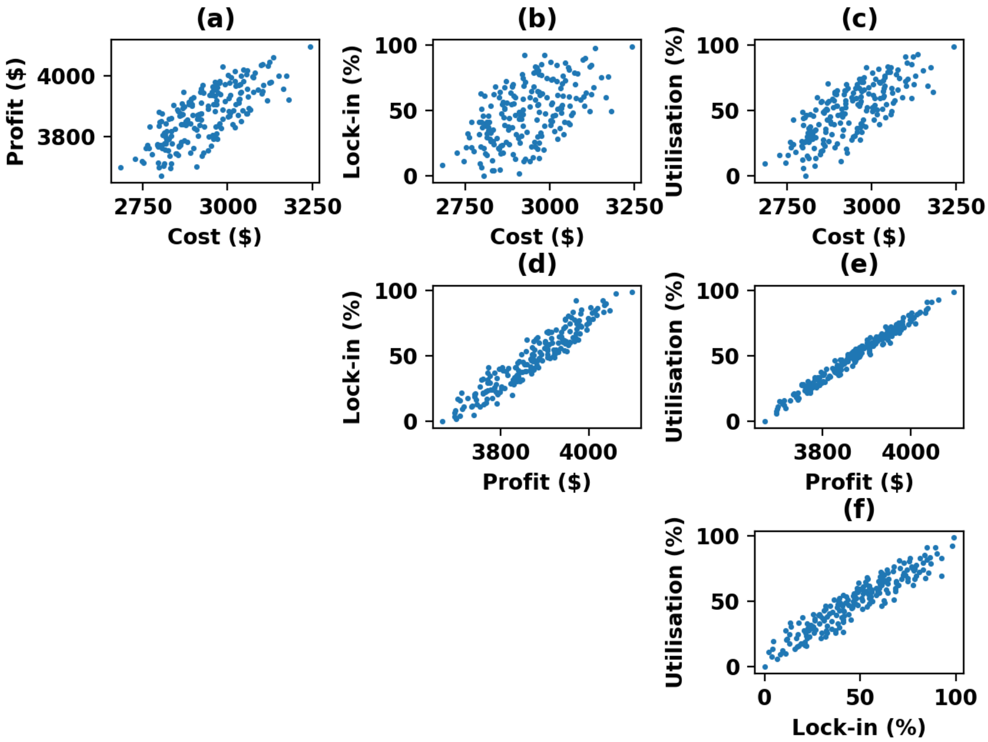


Figure 5.11: Scatter plot matrix showing Pareto solutions of all bi-objective combinations of Experiment 13.

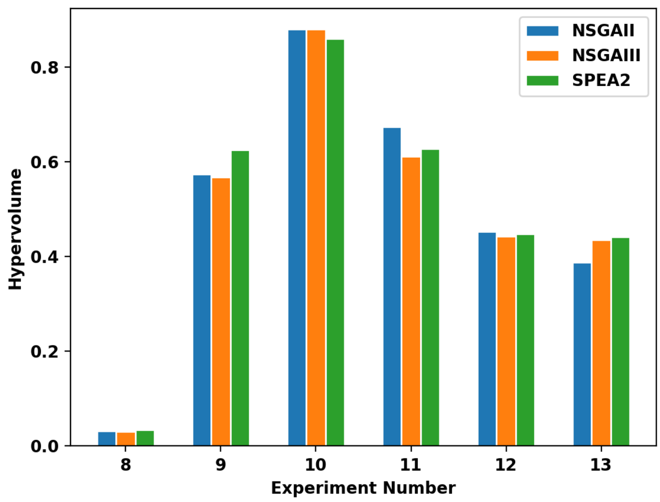


Figure 5.12: Evaluating the Pareto-generated solutions using HV indicator.

Chapter 6

A Multiobjective QoS Model for Trading Cloud of Things Resources

6.1 Introduction

Despite the interest in integrating Cloud Computing and IoT to support the emerging CoT paradigm, there are still many open challenges [42]. One of these is in supporting QoS for CoT applications. All CoT applications focus on particular QoS attributes, either explicitly or implicitly in the application aims. For example, latency-sensitive applications (e.g. military, emergency services) benefit from the larger number of IoT sensing nodes. Less time-sensitive applications (e.g. marketing, planning) utilise the scalability and reliability of the Cloud to process big data generated from distributed IoT resources and make decisions accordingly. Supporting QoS for these applications means enabling these attributes to be prioritised.

Supporting QoS in CoT applications is particularly challenging in scenarios where there are many resource providers and consumers such as in smart cities. Using market-based mechanisms to commodify resources is an approach used in similar large-scale computing infrastructures such as Grids and federated Clouds. The commoditisation of CoT deployments will prevent the slow down in the rate of IoT adoption [132] caused by the considerable investment in hardware, software and maintenance. In a CoT marketplace, resources can be traded as commodities rather than as physical products and priced using Cloud pay-per-use pricing model. The commoditisation of CoT resources will reduce overall costs, enable sharing and reusing of IoT resources, and motivate for new services and applications. In this scenario, the use of resources will be very dynamic and will require efficient market-based mechanisms to support QoS in CoT.

The work presented in this thesis aims to support QoS in the scenarios of integrating Cloud and IoT and the emerging CoT. This is achieved by proposing an optimisation-based approach for managing QoS in trading CoT resources. The contributions of the research presented in this chapter are 1) Investigating the problem of managing QoS in CoT by considering several QoS objectives including resource cost, response time, resource energy consumption, fault tolerance and resource coverage, 2) Proposing a new QoS model to optimise the QoS objectives for either a single-objective,bi-objective and multi-objective optimisation problems, 3) Performing rigorous simulations to evaluate the proposed model using three optimisation algorithms.

The research presented in this chapter intends to evaluate the use of optimisation algorithms when managing QoS in CoT environments. The approach of using the optimisation algorithms to solve this trading problem is justified due to their capabilities in finding optimal solutions to similar problems in complexity and scalability. In this case, the complexity resides here due to the heterogeneity of Cloud and IoT resources that results in difficulties when quantifying their values and leading to the involvement of multifaceted variables and decisions.

The remainder of this chapter is organised as follows: Section 6.2 describes the proposed QoS model and defines the problem of supporting QoS whilst trading resources in CoT, evaluation results are discussed in Section 6.3, and finally conclusions are presented in Section 6.4.

6.2 QoS-Based Resource Allocation Model for CoT Applications

To support an efficient resource allocation for the emerging CoT applications, a generic and dynamic QoS model is needed. The QoS model is proposed here with the following assumptions and considerations.

1. CoT resources are allocated to the applications based on QoS attributes as part of a trading process where QoS is vital to address the requirements of CoT applications that are independent of each other.

2. The CoT application can simultaneously utilise multiple physical resources from different providers while maintaining the required QoS level collectively.

3. The CoT application should maintain a certain QoS level to fulfil consumers' requirements even in a case of conflicting QoS objectives at the same time (e.g. minimising resource energy consumption, maximising resource coverage).

6.2.1 QoS Attributes for CoT Application

The complex nature of CoT applications requires a generic QoS model to allocate the required resources optimally. The complexity resides here for two reasons. The heterogeneity of CoT resources makes it challenging to build a unified QoS model with a broad scope of QoS attributes that can satisfy the QoS requirements of different applications. CoT applications have diverse QoS requirements that make it challenging to maintain the required QoS levels, particularly in a case of conflicting QoS requirements.

To overcome the above-mentioned obstacles, using optimisation strategies is considered to trade CoT resources while satisfying the QoS requirements. This approach supports a dynamic selection of QoS attributes based on the application requirements. Thus, allowing a better measurability of individual QoS attributes as discussed in Section 6.2.6 or collectively as presented in Section 6.2.7. The main QoS attributes considered by this model are resource cost, resource coverage,

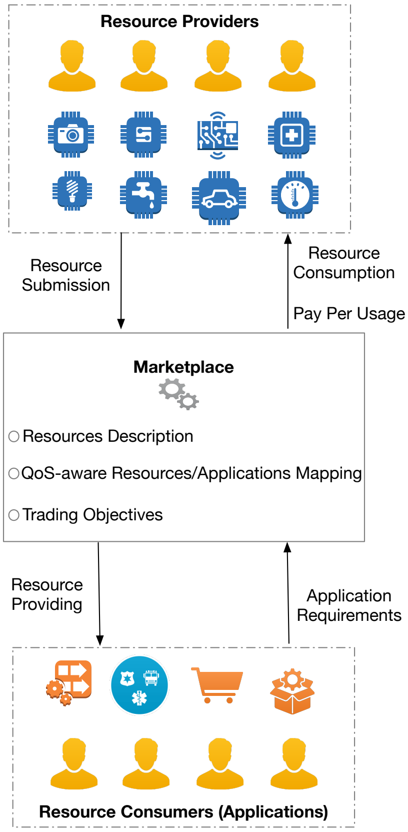


Figure 6.1: CoT Trading Model consists of a marketplace, resource providers and consumers.

response time, energy consumption and fault tolerance. A detailed description of each attribute is presented in Section 6.2.6.

6.2.2 Problem Formulation

The QoS model assumes a CoT marketplace system M, as illustrated in the trading model in Figure 6.1 with multiple consumers $C = (c_1, ..., c_d)$ who request multiple set of resources $R = (r_1, ..., r_j)$ from multiple providers $P = (p_1, ..., p_m)$ to develop multiple concurrent applications $A = (a_1, ..., a_z)$. The marketplace system has to find the optimal match between consumer requests and provider resources. This mapping process considers QoS requirements of the consumers taking into account that each application has different QoS requirements. The decision variables in this context are mainly derived from the resources j whose values will be manipulated by the optimisation algorithm in the search to find the optimal solutions.

The proposed model aims to optimally allocate resources to various applications while satisfying their QoS requirements. The resource allocation is considered optimal when it satisfies two conditions as follows:

1. The allocated resources to each application are sufficient to fulfil the minimal QoS requirements of the application.

2. The overall QoS objective for all participating applications in the trading is maximised.

This process can be demonstrated by the binary variables as illustrated in Equation 6.1. 1 represents a successful resource allocation while 0 indicates otherwise.

$$a_{ij} = \begin{cases} 1, & \text{if } r_j \text{ is allocated to } a_i \\ 0, & \text{otherwise} \end{cases} \tag{6.1}$$

6.2.3 Problem Complexity

The resource allocation in large-scale computing infrastructures is described in the literature as NP-hard or NP-complete problem [91]. The complexity of allocating CoT resources with QoS constraints is described below.

The research space for the optimisation problem can be formed by considering the total number of requests RQ, the number of available resources to match these requests R and the number of resources that violates the QoS constraints V. This can be formulated as $RQ^R - V$. To illustrate, if we consider $RQ = 10$ and $R = 10$ without any constraints, the search space is formed of 10 billion possible solutions represented as 10^{10}.

There are two conflicting considerations that should be taken into account.

1. Violations of constraints are expected to exist which limit the search space and consequently the problem complexity.

2. The violations of the QoS constraints are expected to reduce neither the size of the search space nor the problem complexity due to the heterogeneity of CoT resources and the scalability of the problem.

A CoT marketplace is expected to host a large number of heterogeneous resources that increases the search space exponentially. To relax this challenge, the QoS attributes are considered as utility functions to be optimised individually as a single objective problem or collectively as a bi-objective or a multi-objective problem. The following sections discuss each QoS objective in details.

6.2.4 Marketplace System Architecture

For efficient resource allocation with QoS support in CoT, efficient commiditisation of CoT resources has to be enabled. To achieve this goal, a marketplace system architecture is depicted in Figure 6.2. It is worth noting that the marketplace architecture presented in this chapter is a high-level abstraction of the AMACoT architecture proposed in Chapter 5. The system architecture and the process of finding an optimal QoS-aware resource allocation solution are described below.

Consumers submit their application requests and providers submit their resource offerings to the marketplace. Requests and resources are stored in different directories where the mapper can generate candidate maps of mapped resources to applications. The mapper transfers candidate maps to the optimiser for QoS evaluation. In the optimiser, the evaluator assesses candidate maps based on the QoS constraints available for each round of the optimisation cycle. The evaluator terminates its cycle when the optimal map is found. The resource allocator is responsible for the overall resource allocation process. The scheduler maintains the resources and applications schedules where it controls the lease-time of resources and manages the allocated resources in the Cloud. The allocator also orchestrates the process of joining and dis-joining resources based on the proposed schedules. The monitor component communicates the resource allocation events with the system, consumers and providers.

The use of the optimisation component provides significant flexibility to this approach. It can be implemented as a core of system architecture or as complementary to other market-based mechanisms. When used as part of system architecture, it can be adopted as a substitute for the core component in one of the following market structures.

- broker system
- monopoly market
- oligopoly market
- single-side auction
- double-side auction

The marketplace system proposed and discussed in this thesis satisfies the double-sided auction market structure.

A typical deployment scenario of the marketplace architecture requires each component of the system to be deployed on a separate computing node. This consists of one primary node dedicated to the optimiser and two secondary nodes for the mapper and the resource allocator. Both secondary nodes are used to balance the overall workload of the system. The secondary node of the mapper acts as the access point to the system for all consumers and providers. It is where all incoming requests and the metadata of IoT resources offered by providers are stored and filtered. Due to the processing-intensive nature of optimisation algorithms, the primary node is always assumed to have adequate and better processing capacity than the other nodes of the system. The node of the resource allocator shares a considerable amount of the system workload by scheduling and monitoring all optimal maps of allocated resources.

6.2.5 Illustrative Scenario

To elaborate, the following scenario presents a use case of QoS-driven resource allocation using the marketplace system. A high-density metropolitan area is considered a desirable location for multiple public, private and academic organisations to implement their IoT environmental monitoring applications. Applications monitor various indicators including light, pollution, temperature, pressure, humidity and wind. Considering the existing IoT practice, each organisation is required to deploy its infrastructure (e.g. various sensors,

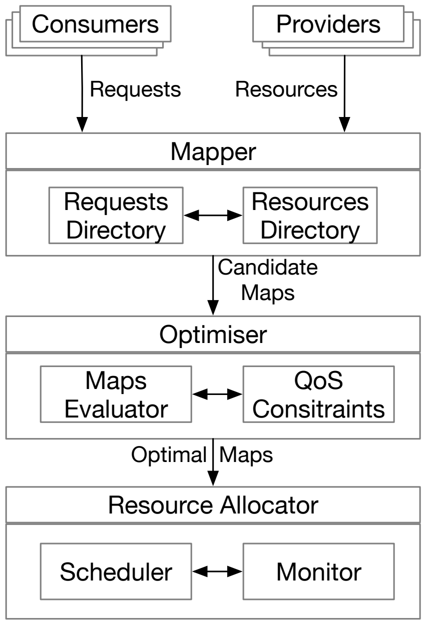


Figure 6.2: High-level marketplace architecture.

dedicated network or gateways to the Internet, other computing nodes) and develop its application. This may not be feasible for all interested parties or expensive replication is created otherwise.

The proposed optimisation-based approach separates infrastructure deployment and application development. Providers can deploy their resources across the metropolitan area and submit their offerings to the marketplace. Consumers also submit their applications requirements to the marketplace for the required resources. The mapping process is based on the QoS requirements of applications. As these applications are financially constrained, public and academic organisations can prioritise their requests with minimised cost and energy consumption while private organisations can prioritise their requests with maximised area coverage and fault tolerance. Upon successful resource allocation, each application can send a software component (e.g. Java applet or Python script) to configure and utilise the acquired resources based on their application and QoS requirements.

6.2.6 Single Objective Optimisation Problem

Objective 1: Minimising Cost. Consumers usually aim to have a cost-efficient resource allocation. The cost of resources is an important aspect to be considered while optimising QoS levels. The importance comes from the balance enforced by the cost when other QoS constraints exist. To elaborate, an application requires a certain level of response time, energy consumption and fault tolerance within a limited budget constraint of the consumer. Without considering the cost as a constraint, there would be more resource allocation options for the application where many of them are not feasible.

To minimise the cost of allocated resources, let cs_j be the resource cost whereas the consumer bid is set to be b_i. t_i denotes the requested lease time of a resource that is specified by consumers. TQ_{ij} denotes the estimated transmission and delay time that can impact the total lease time. TQ_{ij} consists of T_{ij} which is the latency between resource and application while dl_ij is the distance between a requested location of a resource and its actual location. Considering the location of resources is assumed to have a direct impact on latency as some resources will require additional network hops based on their location. This can increase transmission time and latency, impacting the lease time as a sequence. TQ_{ij} is measured by $TQ_{ij} = T_{ij} \times dl_{ij}$. Let rp_j denotes the reputation of the provider based on the credibility measures of the marketplace. rp_j is assumed to determine the trust level of a provider at providing high-quality resources. The higher the reputation, the better the quality of the resources. To optimise the cost utility, the following objective is formulated.

$$\text{Minimise} \quad CS = \sum_{i=1}^{n}\sum_{j=1}^{m}(b_i - cs_j \times rp_j) \times (t_i + TQ_{ij}) \tag{6.2}$$

$$\text{subject to} \quad \sum_{i=1}^{n} rq_i \leq cp_j, where \quad j = 1, ..., m \tag{6.3}$$

$$0 < cs_j \leq b_i \tag{6.4}$$

$$0 < Er_i \leq Ep_j \tag{6.5}$$

$$se_i \leq se_j \tag{6.6}$$

$$dl_i \leq Cv_j \tag{6.7}$$

$$rp_i \leq rp_j \tag{6.8}$$

$$ra_i \leq ra_j \tag{6.9}$$

where i=1,...,n and j = 1,...,m for constraints 6.4, 6.5, 6.6, 6.7, 6.8, 6.9.

Optimisation constraints provide significant support to the proposed model where additional measures can be formulated to enforce the QoS requirements. Constraint 6.3 limits the resource allocation to the capacity of providers and ensures the fair distribution of resources from multiple providers. rq_i is set to the number of requests from consumers, whereas cp_j denotes the capacity of a provider. Thus, the number of requests does not exceed the capacity limit. Constraint 6.4 indicates whether both the cost of a resource cs_j and the bid from a consumer b_i are always positive and b_i has to be always greater than cs_j.

Constraint 6.5 presents an energy consumption constraint in which the required energy Er_i for an application does not exceed the available resource energy Ep_j. Zero or negative values of Ep_j indicates the unavailability of the resource due to power lifetime. Constraint 6.6 ensures the security requirements of the application se_i can be satisfied by the security capabilities of the resource se_j. Constraint 6.7 illustrates a constraint to ensure the maximum acceptable distance between the required coverage area of an application dl_i is within the boundaries of the allocated resource coverage Cv_j.

To address the challenges of provider credibility, Constraint 6.8 ensures that each provider maintains the minimal credibility requirements to formulate a reputation rate rp_j in the marketplace. The constraint also assures the minimal required reputation level rp_i of a consumer is met. Constraint 6.9 specifies the bounds for any additional resource attributes. ra denotes some resource attributes that are not standard or common. ra_i represents those attributes requested by consumers while ra_j is denoted to resource attributes offered by providers. It is introduced to accommodate uncommon resource properties in some IoT devices due to the heterogeneity of IoT resources. This aims to identify the hardware properties of the physical CoT resource that impact QoS directly or indirectly. This includes specifications of the processing, storage, memory, actuating and sensing components of the CoT nodes. Each property can expand into a multilevel sub-properties to improve the optimality of the resource allocation. For instance, the sensing component(s) of a resource described by its properties (*sensorType = [footfall, environmental, light], sensingRange = [0: poor, 1: good, 2: very good, 3: excellent], sensorAccuracy= 0: poor, 1: good, 2: very good, 3: excellent*) and so on. The resource attribute constraint offers the flexibility required for trading heterogeneous resources where QoS would significantly vary without a genuine approach of defining the QoS requirements or levels.

Objective 2: Minimising Response Time. Response time is an important QoS consideration, especially in large-scale distributed systems. CoT can be very widely distributed across a large geographical area where the response time is vital for application QoS. Latency is one contributor to response time. Variable L_{ij} corresponds to the latency between a consumer and a provider and it is measured by $L_{ij} = t_{ack} - t_{start}$. This measures the elapsed time from submitting the request by consumer t_{start} to the time of receiving an acknowledgement from a provider t_{ack}. The R_t utility also consider the estimated queuing and transmitting delays t_{qd} that is expected to be at its minimal for many time-sensitive applications. It is calculated as $t_{qd} = \frac{L_{ij}}{dl_{ij}}$ where dl_{ij} is the distance between the consumer and the provider. R_t utility can be optimised as follows:

$$\text{Minimise} \quad R_t = \sum_{i=1}^{n} \sum_{j=1}^{m} L_{ij} + t_{qd} \tag{6.10}$$

$$\text{subject to} \quad 6.3, 6.4, 6.5, 6.6, 6.7, 6.8, 6.9 \tag{6.11}$$

Objective 3: Minimising Energy Consumption. The energy efficiency is a critical measurement for QoS in CoT application. Many IoT physical resources are power-constrained in which their performance are limited. The energy consumption utility E aims to minimise the power consumption of allocated resources while being utilised by consumers. This can be presented by the difference between the initial power supply of the resource Ep_j and the estimated power consumption requested by the consumer Er_i. This can be optimised as follows:

$$\text{Minimise} \quad E = \sum_{i=1}^{n} \sum_{j=1}^{m} Ep_j - Er_i \tag{6.12}$$

$$\text{subject to} \quad 6.3, 6.4, 6.5, 6.6, 6.7, 6.8, 6.9 \tag{6.13}$$

Objective 4: Maximising Fault Tolerance. Fault tolerance in this context describes the ability of a set of allocated resources to continue providing an acceptable service level in case of a failure. The proposed QoS model in this study considers both soft and hard faults for IoT resources.

The use of concurrent communication interfaces in a resource is denoted by mu_j. This enables allocated resources to reconfigure a different interface for the same application in which resources were assigned to. In case of unavailability of multiple interfaces in a resource $mu_j = 0$, the providers may already have deployed a redundant or standby resources rr_j nearby with the similar QoS attributes of the failing resource. Another important aspect that may impact the recovery of a resource from failures is the difference in response time of that resource during or after a failure. The variable ΔRt denotes the difference between the current response time after failure βRt and the average Rt where $\Delta Rt = \beta Rt - avg(Rt)$. Due to the difference in scales of values, the fault tolerance variables are normalised by re-scaling them into a single numerical range. In order to optimise the fault tolerance utility, the following objective function is presented.

$$\text{Maximise} \quad F_t = \sum_{i=1}^{n}\sum_{j=1}^{m} mu_j + rr_j - \Delta Rt \tag{6.14}$$

$$\text{subject to} \quad 0 \leq mu_i \leq mu_j \tag{6.15}$$

$$cr_i \leq cr_j \tag{6.16}$$

$$rr_i \leq rr_j \tag{6.17}$$

$$6.3, 6.4, 6.5, 6.6, 6.7, 6.8, 6.9 \tag{6.18}$$

where i=1,...,n and j = 1,...,m for constraints 6.16, 6.17 and 6.18.

Due to the vitality of fault tolerance for QoS requirements, the following constraints are enforced. Constraint 6.16 indicates whether a resource supports concurrent interfaces or not where $mu_j = 0$ means the resource has one interface only. The constraint also assures that the minimal number of requested interfaces mu_i is satisfied. Constraint 6.17 is set to minimise the impact of communication reliability during failures. Let cr_i the required level of communication reliability for an application while cr_i is the actual communication reliability of the allocated resource to that application. Constraint 6.18 ensures the required level of redundancy by an application rr_i can be satisfied by the correspondent level of the provider rr_j.

Objective 5: Maximising Resource Coverage. Many CoT applications require a specific area coverage, especially for sensing capabilities. Without certain coverage level, CoT applications may not achieve their reachability goals. The proposed QoS model considers the resource coverage as an integral QoS utility for CoT applications. The resource coverage can be calculated using the sensing range s_j of the resource and the maximum transmission power Et_{max} available. The distance dl_i between requested location and the actual location of the resource is also considered. To optimise the resource coverage, the following objective is formulated.

$$\text{Maximise} \quad Cv = \sum_{i=1}^{n}\sum_{j=1}^{m} \frac{s_j \times Et_{max}}{dl_i} \tag{6.19}$$

$$\text{subject to} \quad 6.3, 6.4, 6.5, 6.6, 6.7, 6.8, 6.9 \tag{6.20}$$

6.2.7 Multiobjective Optimisation Problem

In Section 6.2.6, the QoS attributes are presented as individual objectives. In a marketplace environment, consumers are expected to have a multi-attribute QoS for their CoT applications. This adds considerable complexity to the problem due to the following reason. QoS attributes may conflict with one another in which trade-offs between conflicting attributes has to be taken into account. For instance, an application requires a set of resources with minimum cost, response time and the maximum possible area coverage. To overcome this challenge, the proposed QoS utilities are re-defined as a multi-objective optimisation problem as follows:

6.2.7.1 The Weighted Sum Method

The five QoS utilities are aggregated into a single-objective optimisation problem and denoted by S_o. The problem is formulated as follows:

$$\begin{aligned}&\text{Minimise}\\ &\quad S_o = (w_1 \times CS) + (w_2 \times R_t) \qquad (6.21)\\ &\qquad + (w_3 \times E) - (w_4 \times C_v)\\ &\qquad + (w_5 \times F_t)\\ &\text{subject to} \quad 6.3, 6.4, 6.5, 6.6, 6.7, 6.8, 6.9, 6.16, 6.17, 6.18 \qquad (6.22)\end{aligned}$$

where each w_n is a weighting factor that determines the priority of each objective. The sum of w_n is set to one ($w_1 + w_2 + w_3 + w_4 + w_5 = 1$). Prioritising QoS objectives is application-specific and it is very challenging to address in CoT trading environment.

Prioritising QoS objectives in CoT environments is challenging due to the following reasons. QoS parameters are application-specific and cannot be generalised for a wide range of CoT applications. This means priorities will significantly vary across applications. Prioritising QoS objectives using this method requires some prior knowledge about the problem which may not always be available. Even with prior knowledge, this method yields one solution only at a time. To verify all possible weights, the optimisation algorithm has to be run many times to evaluate all possible weights. This is not feasible and it is impossible for many high-dimensional problems. Due to these challenges, all weights used for the evaluation in this study are equal to 0.2 to maintain the balance among all objectives without prioritising one objective over another.

6.2.7.2 Multiobjective Optimisation

Although the weighted sum method benefits specific applications with prior knowledge about the problem, the multi-objective optimisation problem is presented using a different approach to address the complex requirements of the applications where limited or no prior knowledge is available.

$$\text{Minimise} \quad CS = \sum_{i=1}^{n}\sum_{j=1}^{m}(b_i - cs_j \times rp_j) \times (t_i + TQ_{ij}) \tag{6.23}$$

$$\text{Minimise} \quad R_t = \sum_{i=1}^{n}\sum_{j=1}^{m} L_{ij} + t_{qd} \tag{6.24}$$

$$\text{Minimise} \quad E = \sum_{i=1}^{n}\sum_{j=1}^{m} Ep_j - Er_i \tag{6.25}$$

$$\text{Maximise} \quad Cv = \sum_{i=1}^{n}\sum_{j=1}^{m} \frac{s_j \times Et_{max}}{dl_i} \tag{6.26}$$

$$\text{Maximise} \quad F_t = \sum_{i=1}^{n}\sum_{j=1}^{m} mu_j + cr_j + rr_j - \Delta Rt \tag{6.27}$$

$$\text{subject to} \quad 6.3, 6.4, 6.5, 6.6, 6.7, 6.8, 6.9, 6.16, 6.17, 6.18 \tag{6.28}$$

To solve the problem of resource allocation with QoS constraints, the following optimisation algorithms are used. The improved Strength Pareto

Evolutionary Approach (SPEA2) [181], A Multiobjective Evolutionary Algorithm Based on Decomposition (MOEA/D) [173] and Multi-Objective Indicator-Based Evolutionary Algorithm (IBEA) [179]. These algorithms are chosen for the following three reasons. First, they are gradient-free algorithms which means derivatives calculation is not required and therefore the computational cost is low. Second, these algorithms are known to solve problems similar to the trading CoT resources problem in complexity and scalability. Third, SPEA2 and MOEA/D are reported to produce high quality solutions [65, 66].

6.3 Evaluation

This section presents the experimental setup, analyses and discusses the results of resource allocation with five different QoS utilities.

6.3.1 Experimental Setup

The simulated marketplace system is assumed to use different optimisation strategies to map the optimal resources that satisfy the QoS requirements of multiple CoT applications. The participants of this simulation are summarised in Table 6.1 and described as follows; 10 consumers submit a total number of 10K requests to the marketplace where a number of 20 providers offers 200K heterogeneous resources deployed in a circle area of 2000 meter radius. Each consumer is assumed to request a homogeneous type of resources to be allocated for one application. Experiments presented in this section has the following aims. First, to assess the feasibility and practicability of the proposed QoS model for CoT applications. Second, to evaluate the performance of different optimisation strategies when optimising QoS-based utilities.

Experiments using a synthetic data-set in this study is justified as follows: First, it is technically challenging and financially unfeasible to build a real test-bed for this problem with similar scalability to a real-world scenario; Second, to the best of our knowledge, there is no available public meta-data of IoT physical resources that can be used to implement the proposed QoS model. To overcome both challenges, a large set of meta-data for 200k resources is generated based on the properties of IoT nodes surveyed from several IoT vendors, including Amazon, Microsoft and Google.

The experimental environment is Python 3.6 for 64-bit Mac OS with a 2.6 GHz Intel Core i7 processor and a 16 GB RAM. The common parameters are the maximum number of 250 iterations with a population size of 250. The algorithm-specific parameters are described in Table 6.2.

6.3.2 Experimental Results

As discussed earlier in Section 6.2.2, the problem of resource allocation with QoS constraints is defined as a single objective optimisation problem where the QoS utility functions are optimised individually and also defined as a multiobjective optimisation problem where the QoS utility functions are optimised collectively. In this section, two categories of results are presented as follows:

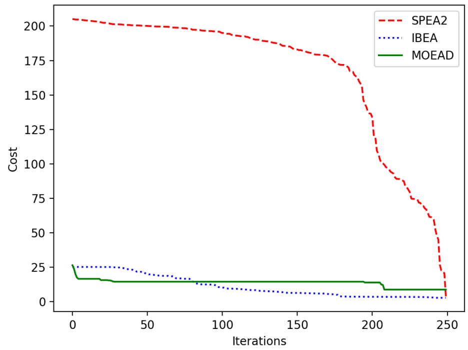
6.3.2.1 Single Objective Problem

To evaluate the proposed QoS objectives, each algorithm is run to optimise each QoS utility individually. Figure 6.3a, 6.3b and 6.3c illustrate the optimal resource allocation solutions for the cost-utility, energy consumption and the response time at the end of each iteration, respectively. The results show that MOEAD outperforms SPEA2 and IBEA in optimising energy consumption and response time while all algorithms find similar optimal solutions for the cost objective.

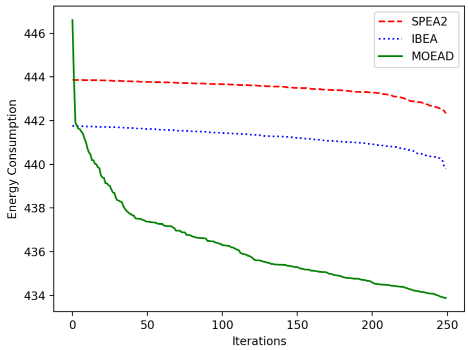
Figure 6.4a, 6.4b present illustrative comparisons of the algorithms when maximising the fault tolerance and the resource coverage utilities, respectively.

Table 6.1: Simulation parameters.

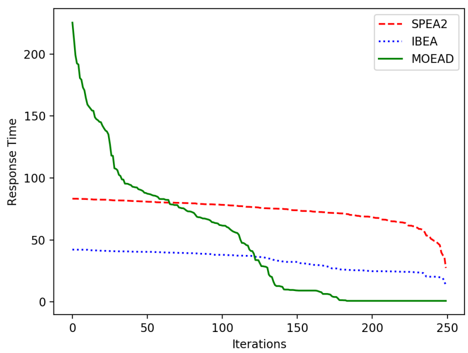
Parameter	**Value**
Simulated Area Radius	2 Km
Number of Requests	10K
Number of Resources	200K
Number of Consumers	10
Number of Providers	20
Number of Applications	10



(a)



(b)



(c)

Figure 6.3: Results of minimising different utilities (a) Cost of resources (b) Energy Consumption (c) Response time.

Figure 6.4a shows all algorithms converge to an optimal solution while IBEA outperforms the others significantly. Figure 6.4b compares between the optimisers when maximising the resource coverage utility. It is clear that the performance of MOEAD and IBEA is better than SPEA2 that may require further iterations to converge.

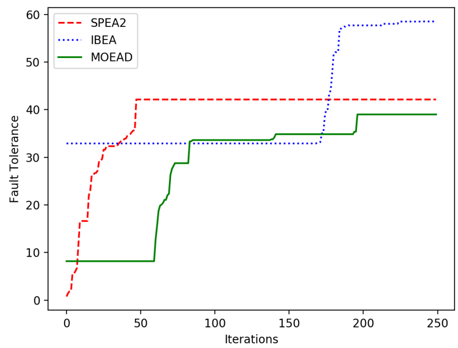
From results compared in the above-mentioned figures, the following can be observed. There are at least two optimal solutions for each QoS utility. MOEAD contributes to the optimality of energy consumption and response time more than SPEA2 and IBEA while IBEA contributes more to the rest of the objectives. It is worth noting that the iterations are stopped at 250 though there are still some changes in the solutions axis (e.g. see Fig 6.3b). Based on the considerable performed experiments, the maximum practical iteration is around 250 considering the trade-off between the solution produced and the computational time required. The comparison made earlier, therefore, is based on the experimental results obtained using algorithms' parameters stated in Table 6.2 without taking into account any potential out-performance of the used algorithms beyond iteration 250.

6.3.2.2 Multi-Objective Problem

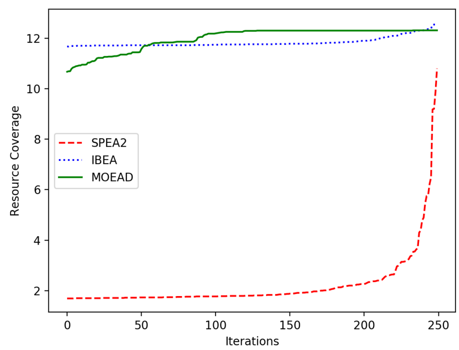
As discussed earlier, performing a multiobjective optimisation is necessary to address the QoS requirements of applications when trading CoT resources. The first approach used in multiobjective optimisation is the weighted sum approach. The five objectives are aggregated into a single objective to optimise

Table 6.2: Algorithm-specific parameters.

Algorithm	**Parameter**
SPEA2	Indicator value $K = 1$, initial population randomly generated between 1 and RQ_n
IBEA	Initial population randomly generated between 1 and RQ_n
MOEAD	Neighbourhood size = 10, initial population randomly generated between 1 and RQ_n, wights randomly generated, decomposition = Tchebycheff, $\delta = 0.8$, $\eta = 1$



(a)



(b)

Figure 6.4: Results of maximising different utilities (a) Fault tolerance (b) Resource coverage.

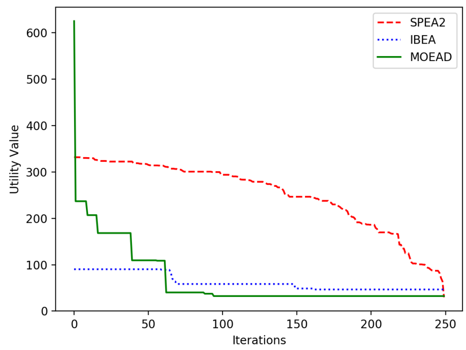


Figure 6.5: Results of optimising all objectives using the weighted sum approach.

the overall QoS utility. Aggregated functions rely heavily on weight values which are challenging to assign. In this case, each weight value is set to $w_n = 0.2$ in order to maintain a balance among the five utilities without prioritising one over another. The following can be observed in Figure 6.5. Every algorithm starts from a utility value that is significantly different from the others. It can also be noted that MOEAD and IBEA converge to an optimal solution while SPEA2 is showing a trend of changing. SPEA2 performance here is similar to its performance in most individual objectives. This may imply its inefficiency for global search in this CoT experimental setup.

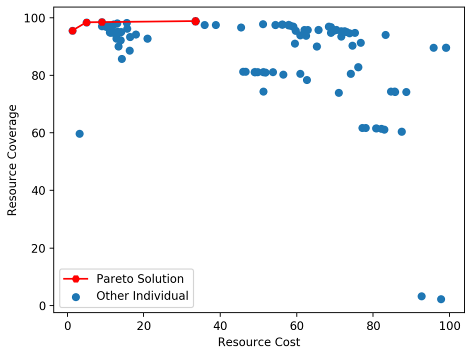
The other approach in optimising multiobjective is where multiple objectives are optimised collectively by the optimiser to yield different optimal solutions rather than a single solution. The optimal solutions are called a Pareto Front, and a decision has to be made to select the best solution. In CoT marketplace, it is assumed that the decision is made autonomously by the marketplace system based on predefined preferences of a consumer.

The results presented in Figures 6.6, 6.8, 6.10, 6.12, 6.14, 6.16 show bi-objective optimisation of the QoS objectives. This includes minimising the cost while maximising the resource coverage, minimising the cost while maximising the fault tolerance, minimising the cost and the response time, minimising the energy consumption while maximising the resource coverage, minimising the energy and response time and maximising fault tolerance and resource coverage, respectively. Additional results presented in Figure 6.7, 6.9, 6.11, 6.13, 6.15, 6.17 illustrate optimising the other three objectives of each bi-objective experiment. Figure 6.6 illustrates the various optimal resource allocation maps that minimise the costs and maximise the resource coverage. Figure 6.6a, 6.6b and 6.6c show that all algorithms produce Pareto fronts. IBEA and MOEAD produce less but better solutions when compared to SPEA2.

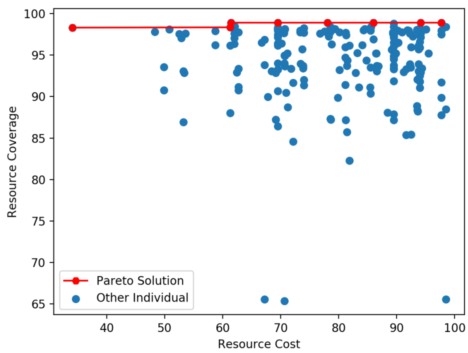
Figure 6.7 presents optimising energy consumption, response time and fault tolerance. SPEA algorithm shown in Figure 6.7b yields the largest set of Pareto fronts for all objectives. All algorithms compete to produce very similar response time but vary when it comes to energy consumption and fault tolerance. For instance, MOEAD algorithm shown in Figure 6.7c demonstrates very similar response time to the other two algorithms and competes with IBEA towards similar fault tolerance but with more high energy consumption. SPEA2 illustrated in Figure 6.7b show some solutions that are approximately %50 better than the fault tolerance produced by the other two algorithms. When compared with Figure 6.6, the following can be observed. SPEA2 algorithm as can be seen in Figure 6.6b, produces the largest set of Pareto fronts as well in this case. Considering the five objectives collectively, IBEA algorithm contributes most to the optimality of the resource coverage and the response time. SPEA2 contributes most to the fault tolerance and fairly to the resource coverage. MOEAD contributes most to the resource cost and fairly to the response time. IBEA and SPEA2 have similar energy consumption Pareto fronts.

The Pareto fronts of minimising the cost while maximising the fault tolerance are presented in Figure 6.8. All algorithms produce a similar number of Pareto solutions while IBEA produces the best in terms of resource cost and fault tolerance level.

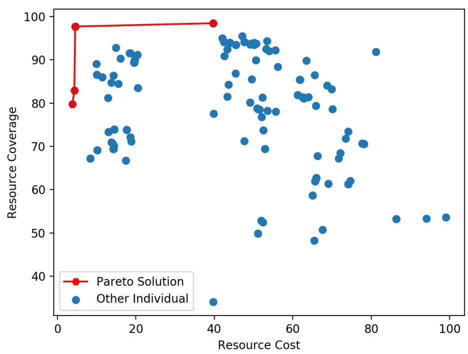
In Figure 6.9, the results of optimising energy consumption, response time and resource coverage are depicted. Figure 6.9a and 6.9b show that both IBEA and SPEA2 produce very similar response time in terms of the quantity and quality. It can be observed from Figure 6.9c that MOEAD does not form a typical Pareto front considering the energy consumption and the response time but provides significantly better resource coverage than IBEA and SPEA2. Considering the



(a)

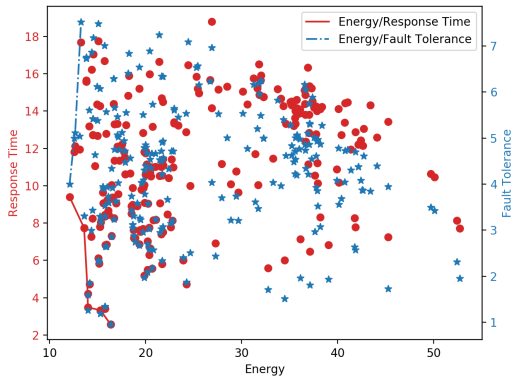


(b)

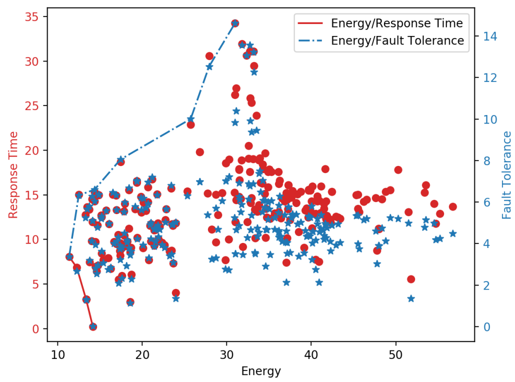


(c)

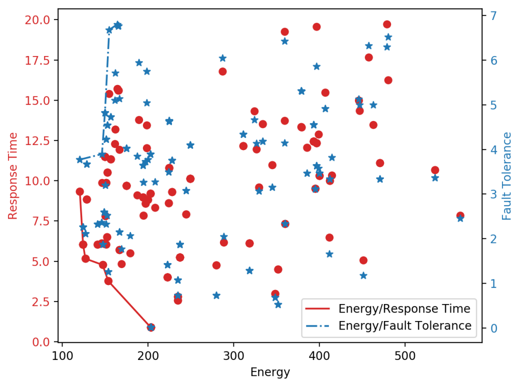
Figure 6.6: Pareto optimal results minimising the cost while maximising the resource coverage (a) IBEA algorithm (b) SPEA2 algorithm (c) MOEAD algorithm.



(a)

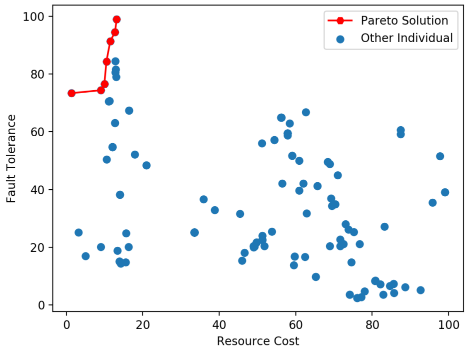


(b)

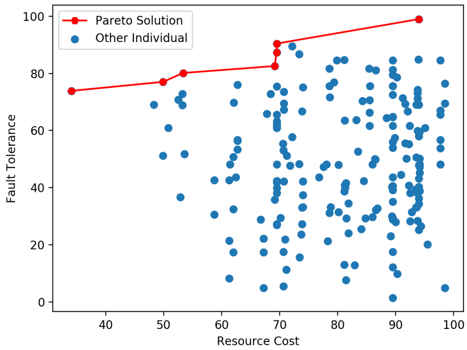


(c)

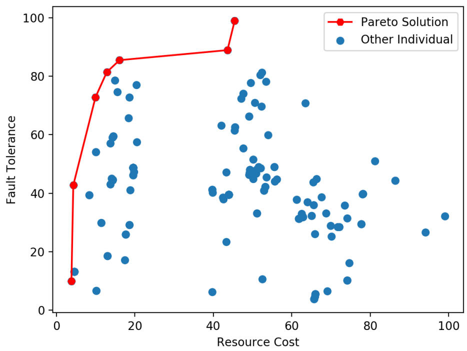
Figure 6.7: Pareto optimal results minimising energy consumption and response time while maximising fault tolerance (a) IBEA algorithm (b) SPEA2 algorithm (c) MOEAD algorithm.



(a)

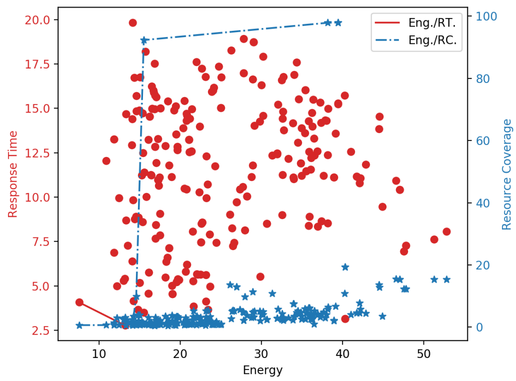


(b)

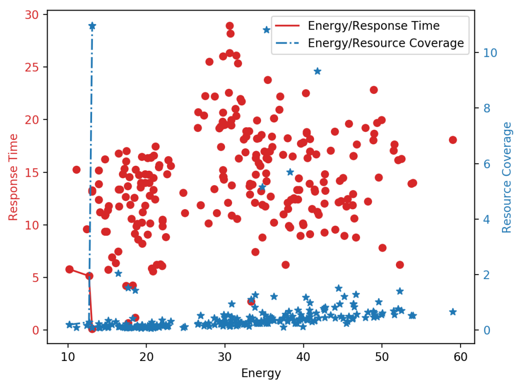


(c)

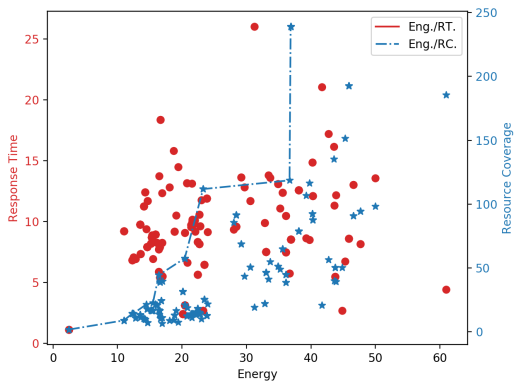
Figure 6.8: Pareto fronts of minimising the cost while maximising the fault tolerance (a) IBEA algorithm (b) SPEA2 algorithm (c) MOEAD algorithm.



(a)



(b)



(c)

Figure 6.9: Pareto optimal results minimising the energy consumption and the response time while maximising the resource coverage (a) IBEA algorithm (b) SPEA2 algorithm (c) MOEAD algorithm.

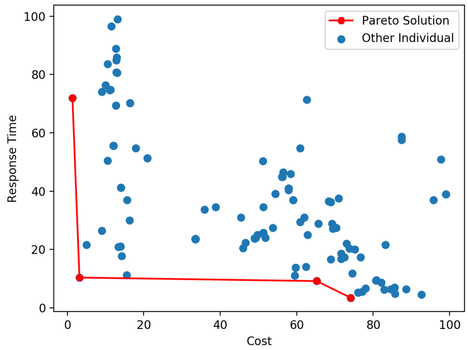
five objectives in Figure 6.8 and Figure 6.9, all algorithms produce a similar number of Pareto fronts but vary in the quality of the solutions. As can be seen, IBEA contributes most to the optimality of the cost, fault tolerance and to a greater extent of response time and energy consumption. SPEA2 provides the most optimal response time and considerably high fault tolerance. MOEAD yields significantly higher resource coverage than the other two algorithms and generates a set of low-cost Pareto solutions.

IBEA and MOEAD algorithms compete to minimise the cost and response time as demonstrated in Figure 6.10. SPEA2 produces one optimal solution that optimises the response time well but provides an unbalanced cost to response time fronts which may not be attractive for consumers, especially with time-sensitive applications.

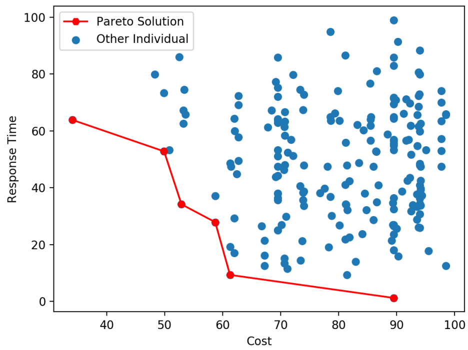
Figure 6.11 shows the results of optimising energy consumption, fault tolerance and resource coverage. Although all algorithms produce a large number of solutions, SPEA2 produces the largest set as can be seen in Figure 6.11b. It can be observed from Figure 6.11a, Figure 6.11b and Figure 6.11c that the energy consumption increases as the resource coverage increases. SPEA2 achieves the highest resource coverage with lower energy consumption in comparison to the other two algorithms. IBEA produce the most optimal fault tolerance considering correspondent energy consumption and resource coverage. Considering the five objectives presented in Figure 6.10 and Figure 6.11, the following are observed. SPEA2 produces the largest set of solutions in both cases. IBEA contributes most to the optimality of fault tolerance. Although IBEA produces similar resource coverage to SPEA2, it achieves that with higher energy consumption. SPEA2 contributes most to resource coverage and energy consumption. MOEAD contributes better to the response time and the resource cost than the other two algorithms.

In Figure 6.12, Pareto solutions for minimising the energy consumption and maximising the resource coverage are presented. IBEA produces the largest and best set of optimal solutions. SPEA2 and MOEAD yield very similar fronts.

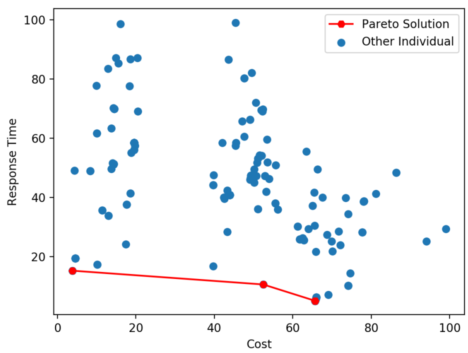
The results of optimising the cost, the response time and fault tolerance are depicted in Figure 6.13. SPEA2 produces the largest set of Pareto fronts. IBEA produces fairly high fault tolerance with low cost and response time as presented



(a)

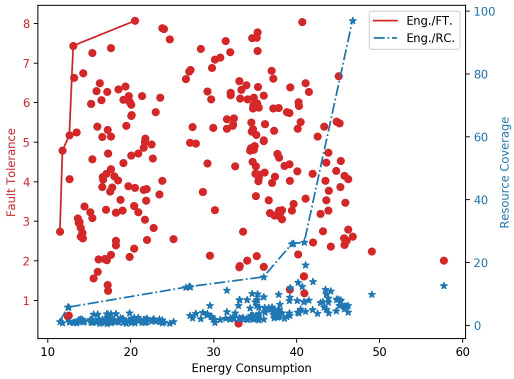


(b)

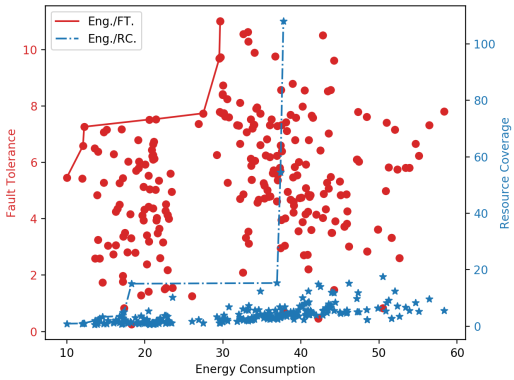


(c)

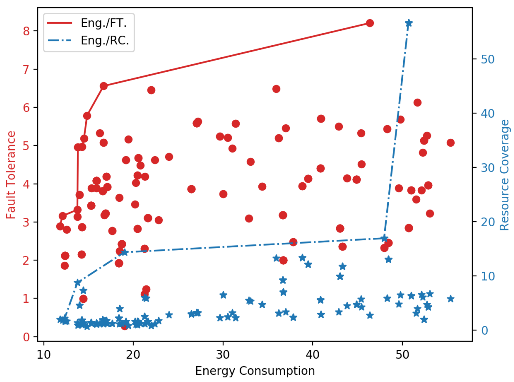
Figure 6.10: Pareto optimal results minimising the cost and the response time (a) IBEA algorithm (b) SPEA2 algorithm (c) MOEAD algorithm.



(a)

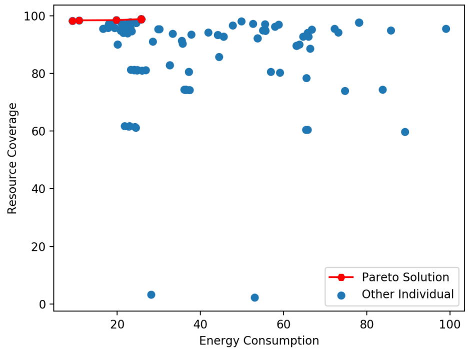


(b)

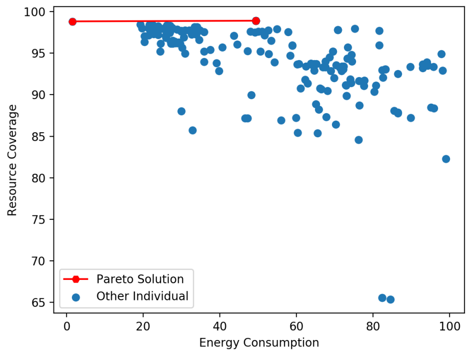


(c)

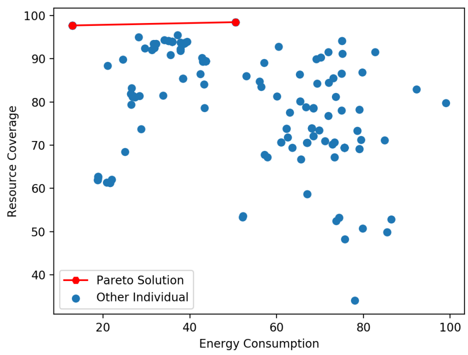
Figure 6.11: Pareto optimal results minimising the energy consumption while maximising the fault tolerance and resource coverage (a) IBEA algorithm (b) SPEA2 algorithm (c) MOEAD algorithm.



(a)

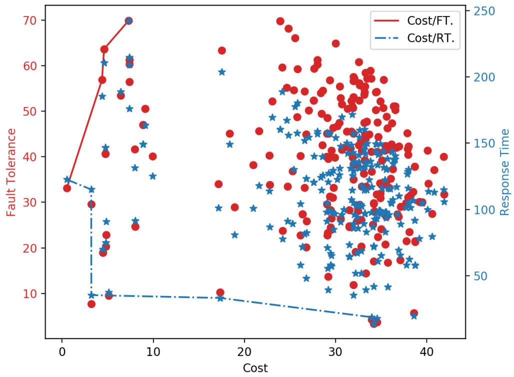


(b)

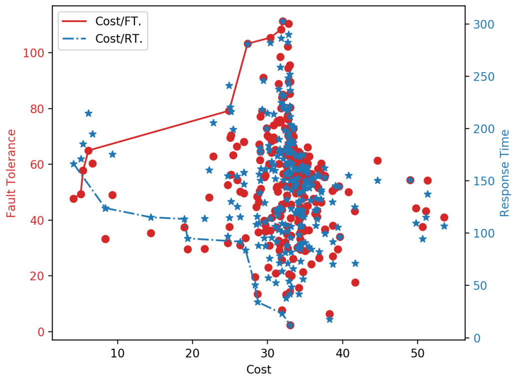


(c)

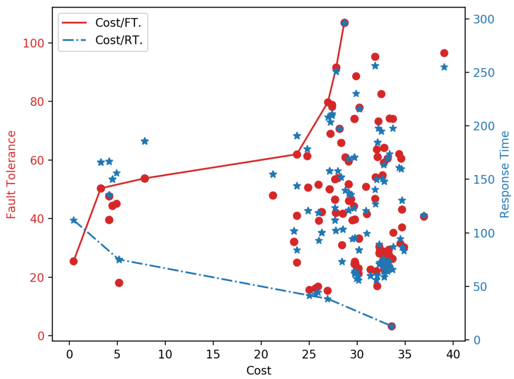
Figure 6.12: Pareto optimal results minimising the energy consumption and maximising the resource coverage (a) IBEA algorithm (b) SPEA2 algorithm (c) MOEAD algorithm.



(a)



(b)



(c)

Figure 6.13: Pareto optimal fronts minimising the cost and response time while maximising the fault tolerance (a) IBEA algorithm (b) SPEA2 algorithm (c) MOEAD algorithm.

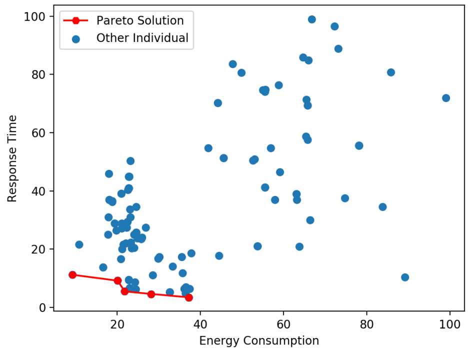
in Figure 6.13a. SPEA2 yields similar and better fault tolerance than IBEA but with higher cost and response time as can be seen in Figure 6.13b. Figure 6.13c shows MOEAD producing the lowest response time but with a similar cost to IBEA and SPEA2. Comparing Figure 6.12 and Figure 6.13, the following can be observed. IBEA contributes most to resource coverage and energy consumption. The three algorithms have similar results for the resource cost. SPEA2 and MOEAD have similar fault tolerance levels.

Figure 6.14 corresponds to applications that require minimising energy and response time. All algorithms presented compete well and minimise their fronts to the near-optimal solutions. Figure 6.14b shows SPEA2 with only one front that represents a solution near zero for both axes. Two near-optimal solutions are presented in Figure 6.14c where response time and energy consumption do not exceed 20. IBEA algorithm produces the largest set of Pareto fronts in this scenario as shown in Figure 6.14a. All fronts have a response time less than 20 with reasonable energy consumption.

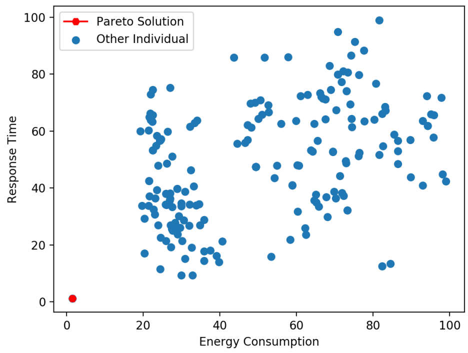
Figure 6.15 presents the results of optimising the resource cost, fault tolerance and resource coverage. MOEAD produces the largest number of Pareto fronts. The results of the IBEA algorithm shown in Figure 6.15a and MOEAD algorithm shown in Figure 6.15c are similar for the cost and resource coverage while MOEAD generates slightly better fault tolerance than IBEA. Although SPEA2 yields similar results of fault tolerance and cost, it produces lower resource coverage in comparison to MOEAD. Considering the five objectives of Figure 6.14 and Figure 6.15 collectively, the following is observed. IBEA contributes most to the response time and resource coverage. MOEAD contributes most to energy consumption, the cost and the fault tolerance.

In Figure 6.16, Pareto optimal results maximising fault tolerance and maximising resource are illustrated. All algorithms produce at least one or more optimal front near 100 for the resource coverage and fault tolerance alike.

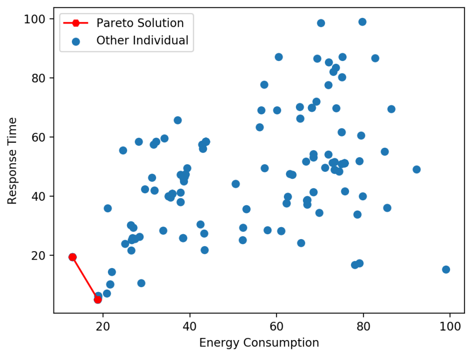
The results of optimising the cost, the response time and energy consumption are illustrated in Figure 6.17. SPEA2 as can be seen in Figure 6.17b produces the largest number of Pareto fronts. IBEA results presented in Figure 6.17a show significantly low response time and cost but with higher energy consumption. SPEA2 results show insignificant higher response time



(a)

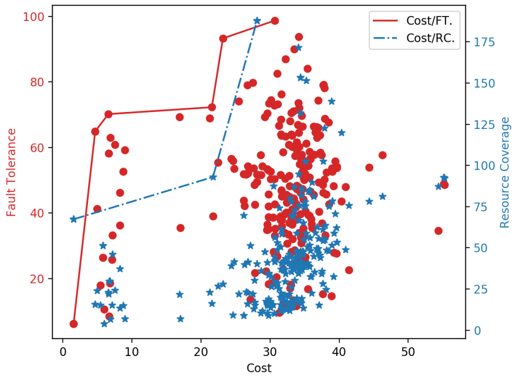


(b)

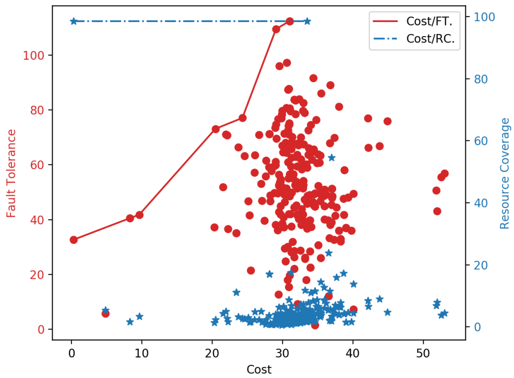


(c)

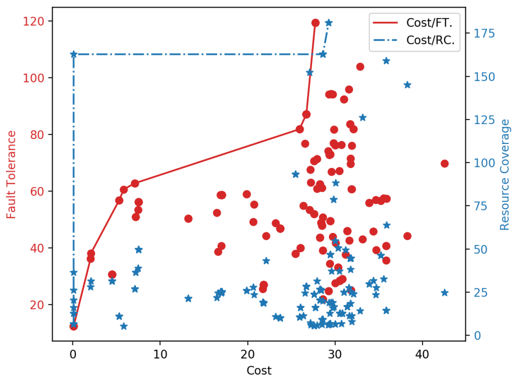
Figure 6.14: Pareto optimal fronts minimising the energy consumption and the response time (a) IBEA algorithm (b) SPEA2 algorithm (c) MOEAD algorithm.



(a)

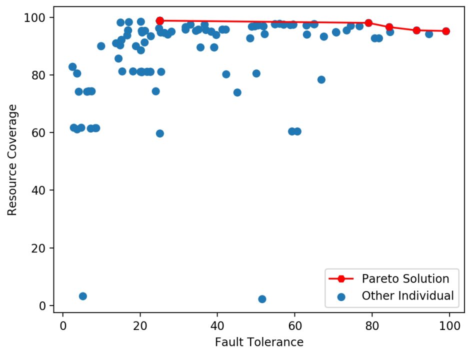


(b)

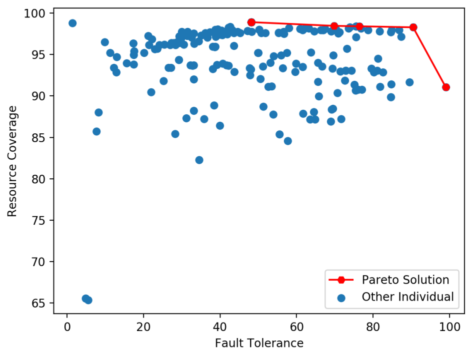


(c)

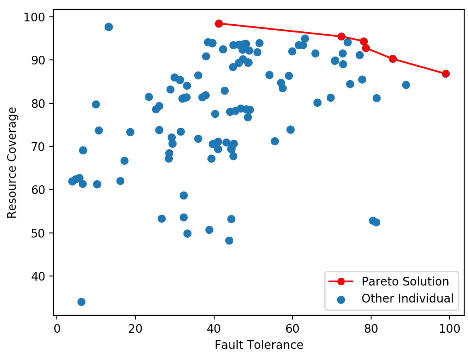
Figure 6.15: Pareto optimal results minimising the resource cost while maximising the fault tolerance and resource coverage (a) IBEA algorithm (b) SPEA2 algorithm (c) MOEAD algorithm.



(a)

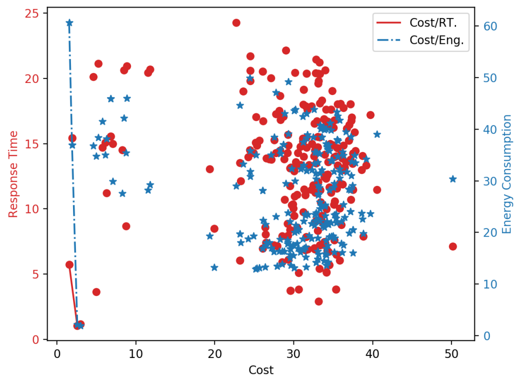


(b)

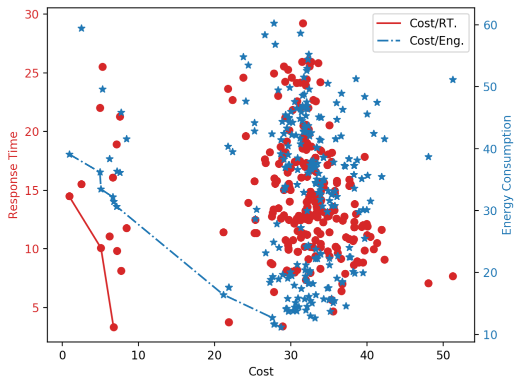


(c)

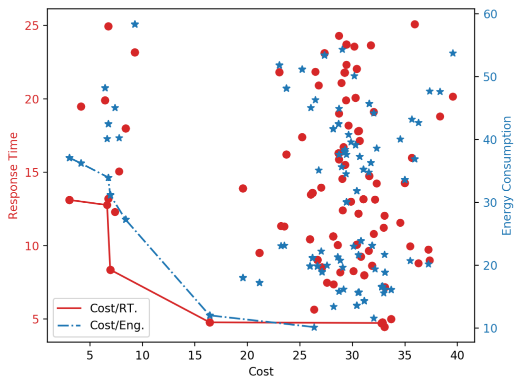
Figure 6.16: Pareto optimal results maximising fault tolerance and resource coverage (a) IBEA algorithm (b) SPEA2 algorithm (c) MOEAD algorithm.



(a)



(b)



(c)

Figure 6.17: Pareto optimal results minimising resource cost, response time and energy consumption (a) IBEA algorithm (b) SPEA2 algorithm (c) MOEAD algorithm.

than IBEA but with similar cost and better energy consumption than IBEA. MOEAD produces similar energy consumption to SPEA2 but with higher resource cost. The following are observations from considering the five objectives presented in Figure 6.16 and Figure 6.17 collectively. SPEA2 produces the largest set of Pareto solutions for the five objectives. IBEA contributes most to the optimality of the response time and the cost. SPEA2 contributes the most to resource coverage and fault tolerance. MOEAD contributes to energy consumption.

Visualising Pareto optimal solutions of multiobjective problems is known to be challenging. To overcome this challenge, Figure 6.18 is a scatter plot matrix that shows the Pareto solutions of the five objectives using IBEA algorithm. It can be observed that IBEA produces a variety of optimal solutions except in three cases. This may imply either the algorithm requires more time to produce the Pareto fronts or Pareto solutions cannot be generated in this complex formulation for all the five objectives.

6.4 Discussion

Managing QoS in CoT environments is challenging due to the complexity and uncertainty in CoT applications. This challenge is relaxed by defining the problem of resource allocation in the CoT trading setup as a single objective and multi-objective optimisation problem to satisfy several QoS requirements. Using different optimisation algorithms as a market-based mechanism is the approach considered to evaluate the proposed QoS model. Three optimisation strategies are applied to optimise QoS utilities including consumer cost, response time, energy consumption, area coverage and fault tolerance.

The simulation results show that the approach investigated in this chapter is feasible for allocating resources to applications with QoS requirements in most cases. Results also show the ability of optimisers to produce at least one optimal solution for each objective tested and multiple solutions for the bi-objective and the multiobjective formulations. Results from SPEA2 demonstrate the ability of the algorithm to produce a larger set of Pareto solutions than the other algorithms. This provides the decision-maker with flexibility when selecting

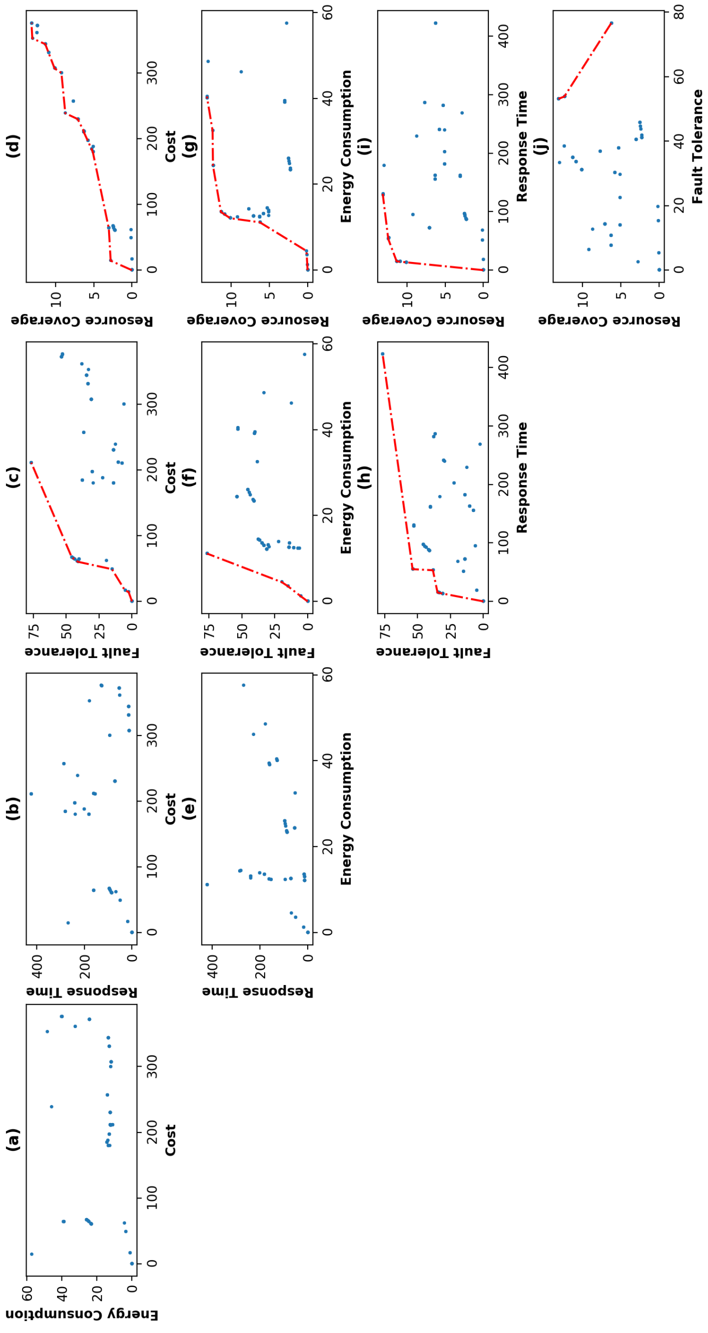


Figure 6.18: Scatter plot matrix showing Pareto solutions of all bi-objective combinations of all objectives using IBEA algorithm

from a range of available solutions. This may also imply that optimisation strategies can be used as a market-based mechanism for trading CoT resources instead of using traditional auctioneers or other dedicated mapping solutions.

The QoS model presented in this chapter is hardware and software independent and can be implemented by any marketplace system. It can also be implemented as a complementary trading mechanism to support other trading mechanisms. This supports separating the development of CoT applications from the deployment of physical resources, making it easy to add any QoS objectives. Utility functions used with vocabularies proposed show their effectiveness in quantifying the value of various CoT resources. This implies potential higher satisfaction for the QoS requirements. Implementation challenges are summarised as follows:

1. Visualising the Pareto solutions for the proposed multiobjective formulation is challenging.
2. High CPU utilisation is observed during the run of experiments but did not have any impact on algorithms performance or the results obtained.

Future work will take into account the following: First, assessing the scalability of this approach by optimising larger sets of resources, Second, optimising more QoS utilities to address application-specific requirements. Third, implementing this approach using different optimisation strategies.

Chapter 7

Conclusions and Future Work

7.1 Summary

The work presented in this thesis is a novel attempt to answer the research questions formed in Chapter 1 using the methodology presented in Chapter 3 theoretically and practically. The experimental evaluation performed in this thesis concluded that using market-based mechanisms can improve shared access to constrained CoT resources. The proposed approach employs optimisation strategies to map the demand of CoT application requirements to the supply of CoT resources. The proposed approach also enables heterogeneous CoT resources to be described generically, and their values quantified accordingly. The QoS support for trading CoT resources was also provided to meet the QoS requirements of CoT applications.

The research aimed to improve shared access to CoT resources efficiently. The research presented in this thesis achieved that by investigating and implementing optimisation-based market mechanisms for commoditising CoT resources with the focus on trading physical CoT resources. The research was conducted to support the integration of Cloud Computing and IoT that were challenged by the limited shared access to IoT resources, to improve the emerging paradigm by improving its resources reusability.

Experiments performed in this study were simulation-based because setting up a real-world CoT environment was very complex and expensive. To commoditise

CoT resources in reality, a large number of heterogeneous resources had to be involved including various types of IoT nodes with different sensors, actuators and computing components. This complexity was required to justify the research approach for commoditising CoT resources. Therefore, the approach taken in this research was to simulate resources and requests data generation and the trading process. To ensure that the simulations performed are most realistic, data sets generated based on surveying the most common properties available for IoT resources in large vendors such as Amazon, Google and IBM.

In conclusion, the work presented in this thesis generated original knowledge in CoT, CoT resource allocation, CoT resource trading and QoS in CoT. The remainder of this chapter is dedicated to providing critical concluding remarks as well as important directions for future work.

7.2 Concluding Remarks

This thesis tackles the problem of missing or limited shared access to CoT resources by using optimised market-based mechanisms to trade those resources and improve their reusability. The remainder of this section presents some concluding remarks regarding different aspects of the work presented in this thesis.

7.2.1 Integrating Cloud Computing and IoT

This research argued the need for dynamic and efficient support to integrate Cloud Computing and IoT into a new paradigm called Cloud of Things. It highlights the technical benefits of integrating both technologies such as expanding the Cloud scope and improving the computing capabilities of IoT as well as the business benefits such as reducing the cost of CoT deployments and motivating new business models. The proposed approach in this thesis was experimentally evaluated and proved its efficiency in addressing some of the integration challenges such as the complexity, heterogeneity and the interoperability of CoT resources.

7.2.2 Shared Access to CoT Resources

The problem of a missing or limited shared access to CoT resources is investigated rigorously in this thesis, particularly IoT physical resources. The surveyed approaches were proven to be infeasible for CoT paradigm due to their hardware-specific design or their inability to address the scalability requirements of CoT applications. The need for a new approach to accessing and reusing CoT resources was motivated. The Exclusive Shared Access (ESA) strategy was introduced in this thesis to enable multiple applications utilising the same resource throughout the scheduled time, but exclusive access is granted to a single application at a given time due to the constrained power and computing capabilities of the resources. The proposed ESA was integrated with market-based mechanisms to allocate the required resources in a trading environment that collectively improve shared access to the CoT resources. The results obtained from the experimental evaluation concluded that ESA could efficiently be employed by marketplace systems to improve the reusability of CoT resources.

7.2.3 Optimisation-based Market Mechanisms for Trading CoT Resources

The work presented in this thesis was inspired by optimisation market approaches used to improve shared access to resources in large-scale computing environments such as Cloud Computing, Grid Computing and WSNs. This research considered CoT resources as commodities rather than organisational assets. This enabled the creation of generic description of CoT resources which was challenging. The evaluation of the proposed description model showed that CoT resources could be described and their value quantified in a standard way that takes into account the heterogeneity of CoT resources and applications.

The optimisation strategies applied in this thesis contributed primarily to the efficiency and dynamism of the proposed trading approach. The use of optimisation-based approaches was justified due to their capabilities in solving similar NP-hard problems in a reasonable time. Optimisation also reduced the architectural complexity of CoT systems where infrastructure deployments were considered independent from application development. Furthermore, using optimisation reduced the development costs and efforts because changes in providers' offerings or consumers' requirements can be reflected in the trading objectives without any changes on the system side.

Experiments performed to evaluate the proposed approach showed the feasibility and efficiency of trading CoT resources. These were proven by optimising a range of objectives including minimising the resource cost, response time, energy consumption and the provider lock-in, and maximising the resource utilisation, fault tolerance, resource coverage, provider profit and the marketplace profit. The preliminary design and experiments provided invaluable insights for the final design and experiments. They resulted in improving the resource description model, the optimisation-based trading model, improved functional prototype of the marketplace architecture (AMACoT) and a proposal of a new multiobjective QoS model for trading CoT resources.

7.2.4 A Marketplace System for Trading CoT Resources

This thesis presented an evaluation of optimisation-based approaches in CoT resource trading through the design and implementation of AMACoT architecture in a computer that simulates the distributed system environment using Python Remote Objects. The first set of experiments tested the performance of the proposed system to validate its usability while the second set provided a comprehensive algorithmic evaluation. The performance evaluation measured the system footprints while the system scaled up twice. The algorithmic experiments optimised the trading of CoT resources for the resource cost, resource utilisation, provider lock-in and provider profit. The optimisation constraints enabled the AMACoT system to efficiently reduce the search space by generating only the potential optimal maps of requests and resources.

The main findings of the work demonstrated in this chapter are summarised as follows:

1. The performance evaluation showed that AMACoT system scaled well using three different scale factors with reasonable CPU, memory and throughput. The only limitation observed from the obtained results was

the latency of the system where it was doubled when the system scaled up by %100. This drawback may be caused by the hardware specification of the computer used during experimentation and should be improved as the hardware setup improve. Other potential causes discussed were the optimisation algorithms used, the complexity of optimised objective functions and the number of optimised objectives. This finding provided an invaluable base for addressing the scalability and dynamism requirements of CoT system design and development where optimisation approaches can be used to replace static system components or dedicated trading components such as auctioneers in marketplace systems.

2. The algorithmic experimentation involved optimising single objective and bio-objective formulations using three algorithms. The Pareto-generated optimal solutions were evaluated using HV indicator to assess the quality of their optimality. This quality assessment of the produced solutions implemented by the marketplace to provide business assurance to the marketplace participants and to evaluate the compliance with SLAs. This would support market-based mechanisms in CoT trading environments where critical applications require advanced QoS or SLAs such as in emergency and security scenarios.

3. Heterogeneous attributes of CoT resources and the dynamic requirements of CoT applications were formulated as objective functions that were optimised. Objective functions could be re-formulated to address any changes in consumers' requirements or providers' offering without or with minimal re-development of the system components. This provided the dynamism and adaptivity required by CoT applications and would enable any business requirements to be formulated as an optimisation problem that can be addressed.

4. The trading of CoT resources experimented using AMACoT marketplace system involved multifaceted technical and business aspects such as describing resources, quantifying resource value, mapping requests to resources, auctioning, resource allocation and scheduling. Therefore, the

terminology of the trading was not limited to matching demand and supply only. This provided the theoretical and practical support to existing related work that described trading in a similar way this thesis did.

7.2.5 QoS for Trading CoT Resources

The research presented in this thesis investigated the need for QoS support for CoT applications. Addressing the requirements for many consumers and providers were proven to be challenging, especially when QoS requirements were conflicting (e.g. minimising the response time while maximising the fault tolerance). The proposed QoS model considered several QoS parameters in CoT including resource cost, response time, resource energy consumption, fault tolerance and resource coverage.

Experimental evaluation optimised QoS objectives in single-objective, bi-objective and multi-objective optimisation problems. Results obtained from experiments confirmed the feasibility of the proposed model in optimising all QoS objectives presented. This model provided the decision-maker with multiple optimal solutions to choose from, based on the priority of the QoS objectives required. Supporting QoS in CoT would likely to improve the providers' commitments to SLAs as well as the satisfaction of consumers. Although the proposed model was intended to work with AMACoT system, the model is hardware and software independent where it can be implemented with any other CoT system or marketplace to support QoS resource allocation.

7.3 Directions for Future Work

Based on the work presented in this thesis, the following aspects are identified as potential directions for future work:

1. The next potential direction to extend the work of this thesis is to study the adaptivity of trading CoT resources. CoT consists of heterogeneous Cloud and IoT resources where CoT increasingly requires adaptive run-time management due to the CoT dynamism, environmental

uncertainties and unpredictable changes in IoT resources. Adapting to these changes benefits particularly trading CoT resources where the adaptability of traded resources and applications remains a challenge. Run-time changes in CoT trading environments can impact vital aspects including resource allocation, resource utilisation and application performance. This topic needs a rigorous investigation to support adaptations when trading CoT resources. The author of this thesis has started this direction of study in [11].

2. A further investigation is needed to validate the proposed approaches of trading CoT resources presented in this thesis for many objectives (e.g. optimising more than five objectives). CoT is complex with heterogeneous resources that may require optimising many objectives simultaneously. Assessing the impact of an increased number of objectives is a complex task that needs further research.

3. As discussed throughout this thesis, trading CoT resources is a multifaceted process that involves many aspects. This complexity would likely produce more trading requirements that need to be addressed. These requirements have to be formulated as trading objectives for optimisation. Therefore, further exploration of trading and QoS requirements is needed.

4. Although this thesis presented an experimental evaluation of various optimisation algorithms, there is a wide range of optimisation algorithms that could be evaluated. The behaviour of optimisation algorithms vary significantly from one another and therefore deserve the focus of future work to obtain better results and make the trading process more efficient.

5. It would be interesting to integrate machine learning with optimisation-based approaches to improve the optimality of proposed trading objectives. Machine learning can aid the performance of optimisation algorithms by testing the ongoing optimisation results and either guide the optimiser toward a better direction or trigger the termination criteria promptly.

6. Although the experiments performed in this thesis attempted to simulate the distributed system behaviour, implementing the proposed work in this thesis on a real distributed environment would be very valuable. It is likely to be challenging and costly but worth the attempt to advance real-world systems for future computing paradigms such as CoT.

References

[1] Mohammad Aazam and Eui-Nam Huh. Fog computing and smart gateway based communication for cloud of things. In *2014 International Conference on Future Internet of Things and Cloud*, pages 464–470. IEEE, 2014. 4

[2] Mohammad Aazam, Imran Khan, Aymen Abdullah Alsaffar, and Eui-Nam Huh. Cloud of things: Integrating internet of things and cloud computing and the issues involved. In *Applied Sciences and Technology (IBCAST), 2014 11th International Bhurban Conference on*, pages 414–419. IEEE, 2014. 11, 15

[3] Sherif Abdelwahab, Bechir Hamdaoui, Mohsen Guizani, and Ammar Rayes. Enabling smart cloud services through remote sensing: An internet of everything enabler. *IEEE Internet of Things Journal*, **1**[3]:276–288, 2014. 16

[4] Sherif Abdelwahab, Bechir Hamdaoui, Mohsen Guizani, and Taieb Znati. Cloud of things for sensing as a service: sensing resource discovery and virtualization. In *Global Communications Conference (GLOBECOM), 2015 IEEE*, pages 1–7. IEEE, 2015. 16

[5] Sherif Abdelwahab, Bechir Hamdaoui, Mohsen Guizani, and Taieb Znati. Cloud of things for sensing-as-a-service: Architecture, algorithms, and use case. *IEEE Internet of Things Journal*, **3**[6]:1099–1112, 2016. 11, 15

[6] Rachit Agarwal, David Gomez Fernandez, Tarek Elsaleh, Amelie Gyrard, Jorge Lanza, Luis Sanchez, Nikolaos

Georgantas, and Valerie Issarny. Unified iot ontology to enable interoperability and federation of testbeds. In *2016 IEEE 3rd World Forum on Internet of Things (WF-IoT)*, pages 70–75. IEEE, 2016. 42

[7] Sven Akkermans, Wilfried Daniels, Gowri Sankar R, Bruno Crispo, and Danny Hughes. Cerberos: a resource-secure os for sharing iot devices. In *Proceedings of the 2017 International Conference on Embedded Wireless Systems and Networks*, pages 96–107. Junction Publishing, 2017. 1, 6, 13, 15, 23

[8] Sven Akkermans, Nicolas Small, Wouter Joosen, and Danny Hughes. Niflheim: End-to-end middleware for applications across all tiers of the iot. In *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems*, page 41. ACM, 2017. 1, 6, 23

[9] Ala Al-Fuqaha, Mohsen Guizani, Mehdi Mohammadi, Mohammed Aledhari, and Moussa Ayyash. Internet of things: A survey on enabling technologies, protocols, and applications. *IEEE Communications Surveys & Tutorials*, **17**[4]:2347–2376, 2015. 4, 11, 15

[10] Najla Al-Nabhan, Nadia Al-Aboody, and ABM Alim Al Islam. A hybrid iot-based approach for emergency evacuation. *Computer Networks*, **155**:87–97, 2019. 26

[11] Ahmed Salim Al Rawahi, Kevin Lee, and Ahmad Lotfi. Adaptive trading of cloud of things resources. In *2019 IEEE 7th International Conference on Future Internet of Things and Cloud (FiCloud)*. IEEE, 2019. 139

[12] Benoit Lheureux Alfonso Velosa, Roy Schulte. *Hype Cycle for the Internet of Things, 2016*, 2016 (accessed June 28, 2018). `https://www.gartner.com/doc/3371743/hype-cycle-internet-things-`. 3

[13] Gianluca Aloi, Giuseppe Caliciuri, Giancarlo Fortino, Raffaele Gravina, Pasquale Pace, Wilma Russo, and Claudio Savaglio. Enabling iot interoperability through opportunistic smartphone-based mobile gateways. *Journal of Network and Computer Applications*, **81**:74–84, 2017. 31

[14] Danilo Ardagna, Giuliano Casale, Michele Ciavotta, Juan F Pérez, and Weikun Wang. Quality-of-service in cloud computing: modeling techniques and their applications. *Journal of Internet Services and Applications*, **5**[1]:11, 2014. 21

[15] Michael Armbrust, Armando Fox, Rean Griffith, Anthony D Joseph, Randy Katz, Andy Konwinski, Gunho Lee, David Patterson, Ariel Rabkin, Ion Stoica, et al. A view of cloud computing. *Communications of the ACM*, **53**[4]:50–58, 2010. 2

[16] Irfan Awan and Muhammad Younas. Towards qos in internet of things for delay sensitive information. In *International Conference on Mobile Web and Information Systems*, pages 86–94. Springer, 2013. 22

[17] Thar Baker, Muhammad Asim, Hissam Tawfik, Bandar Aldawsari, and Rajkumar Buyya. An energy-aware service composition algorithm for multiple cloud-based iot applications. *Journal of Network and Computer Applications*, **89**:96–108, 2017. 26

[18] Javier Barbarán, Manuel Díaz, and Bartolomé Rubio. A virtual channel-based framework for the integration of wireless sensor networks in the cloud. In *Future Internet of Things and Cloud (FiCloud), 2014 International Conference on*, pages 334–339. IEEE, 2014. 15

[19] Yazid Benazzouz, Christophe Munilla, Ozan Gunalp, Mathieu Gallissot, and Levent Gurgen. Sharing user iot devices in the cloud. In *Internet of Things (WF-IoT), 2014 IEEE World Forum on*, pages 373–374. IEEE, 2014. 5

[20] Benjamin Billet and Valérie Issarny. From task graphs to concrete actions: a new task mapping algorithm for the future internet of things. In *Mobile Ad Hoc and Sensor Systems (MASS), 2014 IEEE 11th International Conference on*, pages 470–478. IEEE, 2014. 16

[21] Abdur Rahim Biswas and Raffaele Giaffreda. Iot and cloud convergence: Opportunities and challenges. In *Internet of Things (WF-IoT), 2014 IEEE World Forum on*, pages 375–376. IEEE, 2014. 2

[22] Alberto Blázquez Rodríguez. Security and aaa architectures in an iot marketplace, 2014. 19

[23] Alessio Botta, Walter De Donato, Valerio Persico, and Antonio Pescapé. On the integration of cloud computing and internet of things. In *Future Internet of Things and Cloud (FiCloud), 2014 International Conference on*, pages 23–30. IEEE, 2014. 14, 15

[24] Alessio Botta, Walter de Donato, Valerio Persico, and Antonio Pescapé. Integration of cloud computing and internet of things: a survey. *Future Generation Computer Systems*, **56**:684–700, 2016. 3, 12, 13

[25] Arne Bröring, Stefan Schmid, Corina-Kim Schindhelm, Abdelmajid Khelil, Sebastian Kabisch, Denis Kramer, Danh Le Phuoc, Jelena Mitic, Darko Anicic, and Ernest Teniente López. Enabling iot ecosystems through platform interoperability. *IEEE software*, **34**[1]:54–61, 2017. 19

[26] Massimiliano Caramia and Stefano Giordani. Resource allocation in grid computing: an economic model. *WSEAS Transactions on Computer Research*, **3**:19–27, 2008. 23

[27] Sean Carlin and Kevin Curran. Cloud computing security. In *Pervasive and Ubiquitous Technology Innovations for Ambient Intelligence Environments*, pages 12–17. IGI Global, 2013. 5

[28] Everton Cavalcante, Jorge Pereira, Marcelo Pitanga Alves, Pedro Maia, Roniceli Moura, Thais Batista, Flavia C Delicato, and Paulo F Pires. On the interplay of internet of things and cloud computing: A systematic mapping study. *Computer Communications*, **89**:17–33, 2016. 3, 14

[29] Antonio Celesti, Maria Fazio, Maurizio Giacobbe, Antonio Puliafito, and Massimo Villari. Characterizing cloud federation in iot. In *2016 30th International Conference on Advanced Information Networking and Applications Workshops (WAINA)*, pages 93–98. IEEE, 2016. 4

[30] Sivadon Chaisiri, Bu-Sung Lee, and Dusit Niyato. Optimization of resource provisioning cost in cloud computing. *IEEE Transactions on Services Computing*, **5**[2]:164–177, 2012. 46

[31] Chii Chang, Satish Narayana Srirama, and Rajkumar Buyya. Mobile cloud business process management system for the internet of things: a survey. *ACM Computing Surveys (CSUR)*, **49**[4]:70, 2017. 14

[32] Victor Charpenay, Hoan Nguyen, Mohamad Ibrahim, Achille Zappa, and Arne Bröring. Matching offerings and queries on an internet of things marketplace. In *European Semantic Web Conference*, pages 66–71. Springer, 2018. 18

[33] Yeongho Choi and Yujin Lim. Optimization approach for resource allocation on cloud computing for iot. *International Journal of Distributed Sensor Networks*, **12**[3]:3479247, 2016. 18, 22

[34] Katie Costello. *Gartner Forecasts Worldwide Public Cloud Revenue to Grow 17.5 Percent in 2019*, 2019 (accessed September 8, 2019). . 3

[35] Costas Courcoubetis, Manos Dramitinos, Thierry Rayna, Sergios Soursos, and George D Stamoulis. Market mechanisms for trading grid resources. In *International Workshop on Grid Economics and Business Models*, pages 58–72. Springer, 2008. 23

[36] Nader Daneshfar, Nikolaos Pappas, Valentin Polishchuk, and Vangelis Angelakis. Service allocation in a mobile fog infrastructure under availability and qos constraints. In *2018 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6. IEEE, 2018. 33

[37] Francisco de la Vega, Javier Soriano, Miguel Jimenez, and David Lizcano. A peer-to-peer architecture for distributed data monetization in fog computing scenarios. *Wireless Communications and Mobile Computing*, **2018**, 2018. 20

[38] Kalyanmoy Deb and Himanshu Jain. An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting

approach, part i: solving problems with box constraints. *IEEE Transactions on Evolutionary Computation*, **18**[4]:577–601, 2014. 78

[39] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation*, **6**[2]:182–197, 2002. 64, 78

[40] Flávia C Delicato, Paulo F Pires, Thais Batista, et al. *Resource management for Internet of Things*. Springer, 2017. 16

[41] Lubna Luxmi Dhirani, Thomas Newe, Elfed Lewis, and Shahzad Nizamani. Cloud computing and internet of things fusion: Cost issues. In *2017 Eleventh International Conference on Sensing Technology (ICST)*, pages 1–6. IEEE, 2017. 27

[42] Manuel Díaz, Cristian Martín, and Bartolomé Rubio. State-of-the-art, challenges, and open issues in the integration of internet of things and cloud computing. *Journal of Network and Computer Applications*, **67**:99–117, 2016. 4, 11, 12, 94

[43] Salvatore Distefano, Giovanni Merlino, and Antonio Puliafito. Enabling the cloud of things. In *2012 Sixth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*, pages 858–863. IEEE, 2012. 11

[44] Salvatore Distefano, Giovanni Merlino, and Antonio Puliafito. Towards the cloud of things sensing and actuation as a service, a key enabler for a new cloud paradigm. In *P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC), 2013 Eighth International Conference on*, pages 60–67. IEEE, 2013. 11, 15

[45] Salvatore Distefano, Giovanni Merlino, and Antonio Puliafito. A utility paradigm for iot: The sensing cloud. *Pervasive and mobile computing*, **20**:127–144, 2015. 17

[46] Charalampos Doukas and Ilias Maglogiannis. Bringing iot and cloud computing towards pervasive healthcare. In *Innovative Mobile and*

Internet Services in Ubiquitous Computing (IMIS), 2012 Sixth International Conference on, pages 922–926. IEEE, 2012. 14

[47] Ren Duan, Xiaojiang Chen, and Tianzhang Xing. A qos architecture for iot. In *Internet of Things (iThings/CPSCom), 2011 International Conference on and 4th International Conference on Cyber, Physical and Social Computing*, pages 717–720. IEEE, 2011. 21

[48] Yucong Duan, Guohua Fu, Nianjun Zhou, Xiaobing Sun, Nanjangud C Narendra, and Bo Hu. Everything as a service (xaas) on the cloud: origins, current and future trends. In *2015 IEEE 8th International Conference on Cloud Computing*, pages 621–628. IEEE, 2015. 4

[49] Ivan Farris, Leonardo Militano, Michele Nitti, Luigi Atzori, and Antonio Iera. Mifaas: A mobile-iot-federation-as-a-service model for dynamic cooperation of iot cloud providers. *Future Generation Computer Systems*, **70**:126–137, 2017. 18

[50] Maria Fazio, Antonio Celesti, Antonio Puliafito, and Massimo Villari. An integrated system for advanced multi-risk management based on cloud for iot. In *Advances onto the Internet of Things*, pages 253–269. Springer, 2014. 15

[51] Zesong Fei, Bin Li, Shaoshi Yang, Chengwen Xing, Hongbin Chen, and Lajos Hanzo. A survey of multi-objective optimization in wireless sensor networks: Metrics, algorithms, and open problems. *IEEE Communications Surveys & Tutorials*, **19**[1]:550–586, 2017. 33

[52] Paula Fraga-Lamas, Tiago M Fernández-Caramés, Manuel Suárez-Albela, Luis Castedo, and Miguel González-López. A review on internet of things for defense and public safety. *Sensors*, **16**[10]:1644, 2016. 12

[53] José Antonio Galache, Takuro Yonezawa, Levent Gurgen, Daniele Pavia, Marco Grella, and Hiroyuki Maeomichi. Clout: Leveraging cloud computing techniques for improving management of

massive iot data. In *2014 IEEE 7th International Conference on Service-Oriented Computing and Applications*, pages 324–327. IEEE, 2014. 15

[54] Saurabh Kumar Garg, Christian Vecchiola, and Rajkumar Buyya. Mandi: a market exchange for trading utility and cloud computing services. *The Journal of Supercomputing*, **64**[3]:1153–1174, 2013. 6, 23

[55] Ammar Gharaibeh, Abdallah Khreishah, Mehdi Mohammadi, Ala Al-Fuqaha, Issa Khalil, and Ammar Rayes. Online auction of cloud resources in support of the internet of things. *IEEE Internet of Things Journal*, **4**[5]:1583–1596, 2017. 18

[56] Hadi Goudarzi and Massoud Pedram. Multi-dimensional sla-based resource allocation for multi-tier cloud computing systems. In *Cloud Computing (CLOUD), 2011 IEEE International Conference on*, pages 324–331. IEEE, 2011. 33

[57] GSMA. Understanding the internet of things (iot), 2014. 27

[58] Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami. Internet of things (iot): A vision, architectural elements, and future directions. *Future generation computer systems*, **29**[7]:1645–1660, 2013. 14

[59] Lizheng Guo, Shuguang Zhao, Shigen Shen, and Changyuan Jiang. Task scheduling optimization in cloud computing based on heuristic algorithm. *JNW*, **7**[3]:547–553, 2012. 32

[60] Mateusz Guzek, Pascal Bouvry, and El-Ghazali Talbi. A survey of evolutionary computation for resource management of processing in cloud computing. *IEEE Computational Intelligence Magazine*, **10**[2]:53–67, 2015. 15, 33

[61] Sara Hachem, Thiago Teixeira, and Valérie Issarny. Ontologies for the internet of things. In *Proceedings of the 8th middleware doctoral symposium*, page 3. ACM, 2011. 42

[62] Sarra Hammoudi, Zibouda Aliouat, and Saad Harous. A new infrastructure as a service for iot-cloud. In *2018 14th International Wireless Communications & Mobile Computing Conference (IWCMC)*, pages 786–792. IEEE, 2018. 5

[63] Nikolaus Hansen, Sibylle D Müller, and Petros Koumoutsakos. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (cma-es). *Evolutionary computation*, **11**[1]:1–18, 2003. 64

[64] Shawn Hernan, Scott Lambert, Tomasz Ostwald, and Adam Shostack. Threat modeling-uncover security design flaws using the stride approach. *MSDN Magazine-Louisville*, pages 68–75, 2006. 75

[65] Vinh Ho-Huu, Sander Hartjes, Hendrikus G Visser, and Richard Curran. An efficient application of the moea/d algorithm for designing noise abatement departure trajectories. *Aerospace*, **4**[4]:54, 2017. 109

[66] Ilham Huseyinov and Ali Bayrakdar. Performance evaluation of nsga-iii and spea2 in solving a multi-objective single-period multi-item inventory problem. In *2019 4th International Conference on Computer Science and Engineering (UBMK)*, pages 531–535. IEEE, 2019. 109

[67] Exosite Inc. Build your smart-product empire with Exchange, an IoT marketplace. `https://exosite.com/iot-marketplace/`, 2019. [Online; accessed 12-September-2019]. 20

[68] Google Inc. Google Cloud IoT. `https://cloud.google.com/solutions/iot/`, 2019. [Online; accessed 12-September-2019]. 14

[69] PTC Inc. PTC Marketplace. `https://www.ptc.com/marketplace/about`, 2019. [Online; accessed 12-September-2019]. 20

[70] SensorCloud Inc. Solutions. `https://sensorcloud.com/solutions`, 2019. [Online; accessed 12-September-2019]. 14

[71] Terbine Inc. The Global Exchange for IoT Data. `https://terbine.com/`, 2019. [Online; accessed 12-September-2019]. 20

[72] The MathWorks Inc. ThingSpeak for IoT Projects. https://thingspeak.com/, 2019. [Online; accessed 12-September-2019]. 14

[73] European IoT Platforms Initiative. IoT Marketplace. https://market.big-iot.org/, 2018. [Online; accessed 15-November-2018]. 18

[74] Shrenik Jain, Suresh Purini, and Puduru V Reddy. A multi-cloud marketplace model with multiple brokers for iaas layer and generalized stable matching. In *2018 IEEE/ACM 11th International Conference on Utility and Cloud Computing (UCC)*, pages 257–266. IEEE, 2018. 23

[75] Peter Bisson Jonathan Woetzel Richard Dobbs Jacques Bughin James Manyika, Michael Chui and Dan Aharon. *Unlocking the potential of the Internet of Things*, 2016. . 3

[76] Barkha Javed, Peter Bloodsworth, Raihan Ur Rasool, Kamran Munir, and Omer Rana. Cloud market maker: An automated dynamic pricing marketplace for cloud users. *Future Generation Computer Systems*, **54**:52–67, 2016. 23

[77] Adrián Juan-Verdejo and Bholanathsingh Surajbali. Xaas multi-cloud marketplace architecture enacting the industry 4.0 concepts. In *Doctoral Conference on Computing, Electrical and Industrial Systems*, pages 11–23. Springer, 2016. 19

[78] Burak Kantarci and Hussein T Mouftah. Trustworthy sensing for public safety in cloud-centric internet of things. *IEEE Internet of Things Journal*, **1**[4]:360–368, 2014. 18

[79] James Kempf, Sambit Nayak, Remi Robert, Jim Feng, Kunal Rajan Deshmukh, Anshu Shukla, Aleksandra Obeso Duque, Nanjangud Narendra, and Johan Sjöberg. The nubo virtual services marketplace. *arXiv preprint arXiv:1909.04934*, 2019. 23

[80] James Kennedy. Particle swarm optimization. In *Encyclopedia of machine learning*, pages 760–766. Springer, 2011. 60

[81] Yacine Kessaci, Nouredine Melab, and El-Ghazali Talbi. A pareto-based genetic algorithm for optimized assignment of vm requests on a cloud brokering environment. In *Evolutionary Computation (CEC), 2013 IEEE Congress on*, pages 2496–2503. IEEE, 2013. 33

[82] Mohamed Essaid Khanouche, Yacine Amirat, Abdelghani Chibani, Moussa Kerkar, and Ali Yachir. Energy-centered and qos-aware services selection for internet of things. *IEEE Transactions on Automation Science and Engineering*, **13**[3]:1256–1269, 2016. 21

[83] Hourieh Khodkari, Saeed Ghazi-Maghrebi, and Abbas Asosheh. Assurance of qos in the integration of cloud services and internet of things. In *2017 International Symposium on Networks, Computers and Communications (ISNCC)*, pages 1–5. IEEE, 2017. 21

[84] Seong Hoon Kim and Daeyoung Kim. Multi-tenancy support with organization management in the cloud of things. In *Services Computing (SCC), 2013 IEEE International Conference on*, pages 232–239. IEEE, 2013. 15

[85] Sungwook Kim. Nested game-based computation offloading scheme for mobile cloud iot systems. *EURASIP Journal on Wireless Communications and Networking*, **2015**[1]:229, 2015. 18

[86] Roozbeh Haghnazar Koochaksaraei, Ivan Reinaldo Meneghini, Vitor Nazário Coelho, and Frederico Gadelha Guimarães. A new visualization method in many-objective optimization with chord diagram and angular mapping. *Knowledge-Based Systems*, **138**:134–154, 2017. 86

[87] Matthias Kovatsch, Martin Lanter, and Zach Shelby. Californium: Scalable cloud services for the internet of things with coap. In *Internet of Things (IOT), 2014 International Conference on the*, pages 1–6. IEEE, 2014. 12

[88] Bhaskar Krishnamachari, Jerry Power, Seon Ho Kim, and Cyrus Shahabi. I3: An iot marketplace for smart communities.

In *Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services*, pages 498–499. ACM, 2018. 19

[89] Saku Kukkonen and Jouni Lampinen. Gde3: The third evolution step of generalized differential evolution. In *Evolutionary Computation, 2005. The 2005 IEEE Congress on*, **1**, pages 443–450. IEEE, 2005. 64

[90] Mrityunjay Kumar and Ram Bhushan Agnihotri. Sharing unused user iot devices in the cloud. In *2019 3rd International conference on Electronics, Communication and Aerospace Technology (ICECA)*, pages 400–403. IEEE, 2019. 5

[91] Teerawat Kumrai, Kaoru Ota, Mianxiong Dong, Jay Kishigami, and Dan Keun Sung. Multiobjective optimization in cloud brokering systems for connected internet of things. *IEEE Internet of Things Journal*, **4**[2]:404–413, 2017. 17, 98

[92] Kevin Lee, Ahmed Salim Alrawahi, and Danny Toohey. Enabling commodity environmental sensor networks using multi-attribute combinatorial marketplaces. In *Communications (APCC), 2013 19th Asia-Pacific Conference on*, pages 115–120. IEEE, 2013. 23

[93] Kevin Lee, Georg Buss, and Daniel Veit. A heuristic approach for the allocation of resources in large-scale computing infrastructures. *Concurrency and Computation: Practice and Experience*, **28**[5]:1527–1547, 2016. 6, 17

[94] Kevin Lee and Danny Hughes. System architecture directions for tangible cloud computing. In *Cryptography and Network Security, Data Mining and Knowledge Discovery, E-Commerce & Its Applications and Embedded Systems (CDEE), 2010 First ACIS International Symposium on*, pages 258–262. IEEE, 2010. 4, 11

[95] Kevin Lee, David Murray, Danny Hughes, and Wouter Joosen. Extending sensor networks into the cloud using amazon web services. In *Networked Embedded Systems for Enterprise Applications (NESEA), 2010 IEEE International Conference on*, pages 1–7. IEEE, 2010. 4, 10, 12, 16

[96] Ling Li, Shancang Li, and Shanshan Zhao. Qos-aware scheduling of services-oriented internet of things. *IEEE Transactions on Industrial Informatics*, **10**[2]:1497–1505, 2014. 21

[97] Shancang Li, Li Da Xu, and Shanshan Zhao. 5g internet of things: A survey. *Journal of Industrial Information Integration*, **10**:1–9, 2018. 11

[98] Wei Li, Igor Santos, Flavia C Delicato, Paulo F Pires, Luci Pirmez, Wei Wei, Houbing Song, Albert Zomaya, and Samee Khan. System modelling and performance evaluation of a three-tier cloud of things. *Future Generation Computer Systems*, **70**:104–125, 2017. 11, 15, 16

[99] Yujin Lim and Jaesung Park. Sensor resource sharing approaches in sensor-cloud infrastructure. *International Journal of Distributed Sensor Networks*, **10**[4]:476090, 2014. 42

[100] Chun-Cheng Lin, Der-Jiunn Deng, and Liang-Yi Lu. Many-objective sensor selection in iot systems. *IEEE wireless communications*, **24**[3]:40–47, 2017. 32

[101] Jie Lin, Wei Yu, Nan Zhang, Xinyu Yang, Hanlin Zhang, and Wei Zhao. A survey on internet of things: Architecture, enabling technologies, security and privacy, and applications. *IEEE Internet of Things Journal*, **4**[5]:1125–1142, 2017. 4, 15

[102] Paul Manuel. A trust model of cloud computing based on quality of service. *Annals of Operations Research*, **233**[1]:281–292, 2015. 21

[103] Shikha Mehta and Parmeet Kaur. Scheduling data intensive scientific workflows in cloud environment using nature inspired algorithms. In *Nature-Inspired Algorithms for Big Data Frameworks*, pages 196–217. IGI Global, 2019. 32

[104] Peter Mell, Tim Grance, et al. The nist definition of cloud computing. 2011. 2, 11

[105] Alexandre Robin Mike Botts. Opengis sensor model language (sensorml) implementation specification, 2007. 42

[106] Julien Mineraud, Oleksiy Mazhelis, Xiang Su, and Sasu Tarkoma. A gap analysis of internet-of-things platforms. *Computer Communications*, **89**:5–16, 2016. 23

[107] Sudip Misra, Subarna Chatterjee, and Mohammad S Obaidat. On theoretical modeling of sensor cloud: A paradigm shift from wireless sensor network. *IEEE Systems journal*, **11**[2]:1084–1093, 2017. 16

[108] Sudip Misra, Anuj Singh, Subarna Chatterjee, and Mohammad S Obaidat. Mils-cloud: A sensor-cloud-based architecture for the integration of military tri-services operations and decision making. *IEEE Systems Journal*, **10**[2]:628–636, 2016. 3, 11, 17

[109] Paolo Missier, Shaimaa Bajoudah, Angelo Capossele, Andrea Gaglione, and Michele Nati. Mind my value: a decentralized infrastructure for fair and trusted iot data trading. In *Proceedings of the Seventh International Conference on the Internet of Things*, page 15. ACM, 2017. 20

[110] Krešimir Mišura and Mario Žagar. Data marketplace for internet of things. In *Smart Systems and Technologies (SST), International Conference on*, pages 255–260. IEEE, 2016. 19

[111] Nathalie Mitton, Symeon Papavassiliou, Antonio Puliafito, and Kishor S Trivedi. Combining cloud and sensors in a smart city environment, 2012. 15

[112] Assaad Moawad, Thomas Hartmann, François Fouquet, Grégory Nain, Jacques Klein, and Yves Le Traon. Beyond discrete modeling: A continuous and efficient model for iot. In *Model Driven Engineering Languages and Systems (MODELS), 2015 ACM/IEEE 18th International Conference on*, pages 90–99. IEEE, 2015. 11

[113] Roberto Morabito. Virtualization on internet of things edge devices with container technologies: a performance evaluation. *IEEE Access*, **5**:8835–8850, 2017. 22

[114] Ioannis A Moschakis and Helen D Karatza. Towards scheduling for internet-of-things applications on clouds: a simulated annealing approach. *Concurrency and Computation: Practice and Experience*, **27**[8]:1886–1899, 2015. 33

[115] Husnu S Narman, Md Shohrab Hossain, Mohammed Atiquzzaman, and Haiying Shen. Scheduling internet of things applications in cloud computing. *Annals of Telecommunications*, **72**[1-2]:79–93, 2017. 17

[116] Stefan Nastic, Hong-Linh Truong, and Schahram Dustdar. A middleware infrastructure for utility-based provisioning of iot cloud systems. In *Edge Computing (SEC), IEEE/ACM Symposium on*, pages 28–40. IEEE, 2016. 4, 15

[117] Falak Nawaz, Omar Hussain, Farookh Khadeer Hussain, Naeem Khalid Janjua, Morteza Saberi, and Elizabeth Chang. Proactive management of sla violations by capturing relevant external events in a cloud of things environment. *Future Generation Computer Systems*, **95**:26–44, 2019. 21

[118] Anne H Ngu, Mario Gutierrez, Vangelis Metsis, Surya Nepal, and Quan Z Sheng. Iot middleware: A survey on issues and enabling technologies. *IEEE Internet of Things Journal*, **4**[1]:1–20, 2016. 13

[119] Trung-Viet Nguyen, Lam-Son Lê, Hong-Linh Truong, Khuong Nguyen-An, and Phuong Hoai Ha. Handling service level agreements in iot. In *2018 IEEE 22nd International Enterprise Distributed Object Computing Conference (EDOC)*, pages 145–153. IEEE, 2018. 19

[120] Matthew Niekerk and Roderik Veer. Global market for local data. *Whitepaper*, 2016. 20

[121] Zhaolong Ning, Xiangjie Kong, Feng Xia, Weigang Hou, and Xiaojie Wang. Green and sustainable cloud of things: Enabling collaborative edge computing. *IEEE Communications Magazine*, **57**[1]:72–78, 2019. 11

[122] Dusit Niyato, Xiao Lu, Ping Wang, Dong In Kim, and Zhu Han. Economics of internet of things: An information market approach. *IEEE Wireless Communications*, **23**[4]:136–145, 2016. 18, 31

[123] Mahda Noura, Mohammed Atiquzzaman, and Martin Gaedke. Interoperability in internet of things: Taxonomies and open challenges. *Mobile Networks and Applications*, **24**[3]:796–809, Jun 2019. 13

[124] University of Surrey. CityPulse Dataset Collection. `http://iot.ee.surrey.ac.uk:8080/datasets.html`, 2019. [Online; accessed 30-September-2019]. 34

[125] Justice Opara-Martins. *A decision framework to mitigate vendor lock-in risks in cloud (SaaS category) migration.* PhD thesis, Bournemouth University, 2017. 26

[126] Justice Opara-Martins, M Sahandi, and Feng Tian. A holistic decision framework to avoid vendor lock-in for cloud saas migration. *Computer and Information Science*, **10**[3], 2017. 21

[127] Kazim Rifat Özyilmaz, Mehmet Doğan, and Arda Yurdakul. Idmob: Iot data marketplace on blockchain. In *2018 Crypto Valley Conference on Blockchain Technology (CVCBT)*, pages 11–19. IEEE, 2018. 20

[128] Ji-Sun Park, Taek-Young Youn, Hye-Bin Kim, Kyung-Hyune Rhee, and Sang-Uk Shin. Smart contract-based review system for an iot data marketplace. *Sensors*, **18**[10]:3577, 2018. 18

[129] Pritee Parwekar. From internet of things towards cloud of things. In *Computer and Communication Technology (ICCCT), 2011 2nd International Conference on*, pages 329–333. IEEE, 2011. 11

[130] Norman Paton, Marcelo AT De Aragão, Kevin Lee, Alvaro AA Fernandes, and Rizos Sakellariou. Optimizing utility in cloud computing through autonomic workload execution. *Bulletin of the Technical Committee on Data Engineering*, **32**[1]:51–58, 2009. 46

[131] Santosh Pattar, Rajkumar Buyya, Kuppanna Rajuk Venugopal, SS Iyengar, and LM Patnaik. Searching for the iot resources: Fundamentals, requirements, comprehensive review, and future directions. *IEEE Communications Surveys & Tutorials*, **20**[3]:2101–2132, 2018. 13

[132] Charith Perera and Arkady Zaslavsky. Improve the sustainability of internet of things through trading-based value creation. In *Internet of Things (WF-IoT), 2014 IEEE World Forum on*, pages 135–140. IEEE, 2014. 6, 13, 17, 95

[133] Charith Perera, Arkady Zaslavsky, Peter Christen, and Dimitrios Georgakopoulos. Sensing as a service model for smart cities supported by internet of things. *Transactions on Emerging Telecommunications Technologies*, **25**[1]:81–93, 2014. 15

[134] Virginia Pilloni and Luigi Atzori. Consensus-based resource allocation among objects in the internet of things. *Annals of Telecommunications*, **72**[7-8]:415–429, 2017. 16

[135] Pawani Porambage, Jude Okwuibe, Madhusanka Liyanage, Mika Ylianttila, and Tarik Taleb. Survey on multi-access edge computing for internet of things realization. *IEEE Communications Surveys & Tutorials*, **20**[4]:2961–2991, 2018. 13

[136] Gowri Sankar Ramachandran, Rahul Radhakrishnan, and Bhaskar Krishnamachari. Towards a decentralized data marketplace for smart cities. In *4th IEEE Annual International Smart Cities Conference*. IEEE, 2018. 20

[137] BB Prahlada Rao, Paval Saluia, Neetu Sharma, Ankit Mittal, and Shivay Veer Sharma. Cloud computing for internet of things & sensing based applications. In *Sensing Technology (ICST), 2012 Sixth International Conference on*, pages 374–380. IEEE, 2012. 13

[138] Bushra Rashid and Mubashir Husain Rehmani. Applications of wireless sensor networks for urban areas: A survey. *Journal of network and computer applications*, **60**:192–219, 2016. 3

[139] M Mazhar Rathore, Anand Paul, Awais Ahmad, and Gwanggil Jeon. Iot-based big data: from smart city towards next generation super city planning. *International Journal on Semantic Web and Information Systems (IJSWIS)*, **13**[1]:28–47, 2017. 3

[140] Maria Alejandra Rodriguez and Rajkumar Buyya. A taxonomy and survey on scheduling algorithms for scientific workflows in iaas cloud computing environments. *Concurrency and Computation: Practice and Experience*, **29**[8], 2017. 33

[141] Maria Barbara Safianowska, Robert Gdowski, and ChingYao Huang. Revenue maximizing auction for perishable iot resources. In *Information and Communication Technology Convergence (ICTC), 2016 International Conference on*, pages 417–422. IEEE, 2016. 18

[142] Mayra Samaniego and Ralph Deters. Hosting virtual iot resources on edge-hosts with blockchain. In *2016 IEEE International Conference on Computer and Information Technology (CIT)*, pages 116–119. IEEE, 2016. 22

[143] Parnia Samimi, Youness Teimouri, and Muriati Mukhtar. A combinatorial double auction resource allocation model in cloud computing. *Information Sciences*, **357**:201–216, 2016. 32

[144] Igor L Santos, Luci Pirmez, Flavia C Delicato, Samee U Khan, and Albert Y Zomaya. Olympus: The cloud of sensors. *IEEE Cloud Computing*, **2**[2]:48–56, 2015. 12

[145] Luca Sciullo, Cristiano Aguzzi, Marco Di Felice, and Tullio Salmon Cinotti. Wot store: Enabling things and applications discovery for the w3c web of things. In *2019 16th IEEE Annual Consumer Communications & Networking Conference (CCNC)*, pages 1–8. IEEE, 2019. 4

[146] Damián Serrano, Sara Bouchenak, Yousri Kouki, Frederico Alvares de Oliveira Jr, Thomas Ledoux, Jonathan Lejeune, Julien Sopena, Luciana Arantes, and Pierre Sens.

Sla guarantees for cloud services. *Future Generation Computer Systems*, **54**:233–246, 2016. 21

[147] Damián Serrano, Sara Bouchenak, Yousri Kouki, Thomas Ledoux, Jonathan Lejeune, Julien Sopena, Luciana Arantes, and Pierre Sens. Towards qos-oriented sla guarantees for online cloud services. In *Cluster, Cloud and Grid Computing (CCGrid), 2013 13th IEEE/ACM International Symposium on*, pages 50–57. IEEE, 2013. 21

[148] Sugam Sharma, Victor Chang, U Sunday Tim, Johnny Wong, and Shashi Gadia. Cloud and iot-based emerging services systems. *Cluster Computing*, **22**[1]:71–91, 2019. 13

[149] Nicolas Small, Sven Akkermans, Wouter Joosen, and Danny Hughes. Niflheim: An end-to-end middleware for applications on a multi-tier iot infrastructure. In *Network Computing and Applications (NCA), 2017 IEEE 16th International Symposium on*, pages 1–8. IEEE, 2017. 15

[150] Thamarai Selvi Somasundaram and Kannan Govindarajan. Cloudrb: A framework for scheduling and managing high-performance computing (hpc) applications in science cloud. *Future Generation Computer Systems*, **34**:47–65, 2014. 33

[151] Christos Stergiou and Kostas E Psannis. Recent advances delivered by mobile cloud computing and internet of things for big data applications: a survey. *International Journal of Network Management*, **27**[3]:e1930, 2017. 26

[152] Christos Stergiou, Kostas E Psannis, Byung-Gyu Kim, and Brij Gupta. Secure integration of iot and cloud computing. *Future Generation Computer Systems*, **78**:964–975, 2018. 12, 13, 15

[153] Rainer Storn and Kenneth Price. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization*, **11**[4]:341–359, 1997. 60

[154] Ahmed Suliman, Zainab Husain, Menatallah Abououf, Mansoor Alblooshi, and Khaled Salah. Monetization of iot data using smart contracts. *IET Networks*, 2018. 19

[155] Dawex Systems. The data marketplace for sharing, acquiring and monetizing data. https://www.dawex.com/en/, 2019. [Online; accessed 12-September-2019]. 20

[156] Boleslaw Szymanski, S Yousaf Shah, Sahin Cem Geyik, Sanmay Das, Meenal Chhabra, and Petros Zerfos. Market mechanisms for value of information driven resource allocation in sensor networks. In *2011 IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops)*, pages 20–25. IEEE, 2011. 23

[157] Giacomo Tanganelli, Carlo Vallati, and Enzo Mingozzi. Energy-efficient qos-aware service allocation for the cloud of things. In *2014 IEEE 6th International Conference on Cloud Computing Technology and Science (CloudCom)*, pages 787–792. IEEE, 2014. 11, 12, 23

[158] The SIoT team. IoT Network Dataset. http://www.social-iot.org/index.php?p=downloads, 2019. [Online; accessed 30-September-2019]. 34

[159] Kenji Tei and Levent Gurgen. Clout: Cloud of things for empowering the citizen clout in smart cities. In *Internet of Things (WF-IoT), 2014 IEEE World Forum on*, pages 369–370. IEEE, 2014. 11, 14

[160] Aparna Saisree Thuluva, Arne Bröring, Ganindu P Medagoda, Hettige Don, Darko Anicic, and Jan Seeger. Recipes for iot applications. In *Proceedings of the Seventh International Conference on the Internet of Things*, page 10. ACM, 2017. 18

[161] Hong-Linh Truong and Schahram Dustdar. Principles for engineering iot cloud systems. *IEEE Cloud Computing*, **2**[2]:68–76, 2015. 14

[162] Emmanuel Tuyishimire, Ismail Adiel, Slim Rekhis, B Antoine Bagula, and Noureddine Boudriga. Internet

of things in motion: A cooperative data muling model under revisit constraints. In *Ubiquitous Intelligence & Computing (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld), 2016 Intl IEEE Conferences*, pages 1123–1130. IEEE, 2016. 17

[163] David A Van Veldhuizen and Gary B Lamont. Evolutionary computation and convergence to a pareto front. In *Late breaking papers at the genetic programming 1998 conference*, pages 221–228, 1998. 83, 84

[164] David J Wales and Jonathan PK Doye. Global optimization by basin-hopping and the lowest energy structures of lennard-jones clusters containing up to 110 atoms. *The Journal of Physical Chemistry A*, **101**[28]:5111–5116, 1997. 60

[165] Kaiwei Wang, Wuyang Zhou, and Shiwen Mao. On joint bbu/rrh resource allocation in heterogeneous cloud-rans. *IEEE Internet of Things Journal*, **4**[3]:749–759, 2017. 33

[166] Wei Wang, Kevin Lee, and David Murray. Integrating sensors with the cloud using dynamic proxies. In *Personal Indoor and Mobile Radio Communications (PIMRC), 2012 IEEE 23rd International Symposium on*, pages 1466–1471. IEEE, 2012. 16, 26

[167] Gary White, Vivek Nallur, and Siobhán Clarke. Quality of service approaches in iot: A systematic mapping. *Journal of Systems and Software*, **132**:186–203, 2017. 21

[168] Y Yoon, D Ban, S Han, D An, and E Heo. Device/cloud collaboration framework for intelligence applications. In *Internet of Things*, pages 49–60. Elsevier, 2016. 16

[169] Rong Yu, Yan Zhang, Stein Gjessing, Wenlong Xia, and Kun Yang. Toward cloud-based vehicular networks with efficient resource management. *IEEE Network*, **27**[5]:48–55, 2013. 13, 17

[170] Zhifeng Yuan, Guanghui Yu, Weimin Li, Yifei Yuan, Xinhui Wang, and Jun Xu. Multi-user shared access for internet of things.

In *2016 IEEE 83rd Vehicular Technology Conference (VTC Spring)*, pages 1–5. IEEE, 2016. 5

[171] Andrea Zanella, Nicola Bui, Angelo Castellani, Lorenzo Vangelista, and Michele Zorzi. Internet of things for smart cities. *IEEE Internet of Things journal*, **1**[1]:22–32, 2014. 11

[172] Ivana Podnar Žarko, Krešimir Pripužić, Martin Serrano, and Manfred Hauswirth. Iot data management methods and optimisation algorithms for mobile publish/subscribe services in cloud environments. In *Networks and Communications (EuCNC), 2014 European Conference on*, pages 1–5. IEEE, 2014. 15

[173] Qingfu Zhang and Hui Li. Moead: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on evolutionary computation*, **11**[6]:712–731, 2007. 109

[174] Yang Zhang, Zehui Xiong, Dusit Niyato, Ping Wang, and Zhu Han. Market-oriented information trading in internet of things (iot) for smart cities. *arXiv preprint arXiv:1806.05583*, 2018. 18

[175] Yang Zhang, Zehui Xiong, Dusit Niyato, Ping Wang, and Jiangming Jin. Joint optimization of information trading in internet of things (iot) market with externalities. In *Wireless Communications and Networking Conference (WCNC), 2018 IEEE*, pages 1–6. IEEE, 2018. 19

[176] Kai Zhao and Lina Ge. A survey on the internet of things security. In *2013 Ninth international conference on computational intelligence and security*, pages 663–667. IEEE, 2013. 77

[177] Sébastien Ziegler, Cedric Crettaz, and Ian Thomas. Ipv6 as a global addressing scheme and integrator for the internet of things and the cloud. In *Advanced Information Networking and Applications Workshops (WAINA), 2014 28th International Conference On*, pages 797–802. IEEE, 2014. 14

[178] Paul Zikopoulos, Chris Eaton, et al. *Understanding big data: Analytics for enterprise class hadoop and streaming data*. McGraw-Hill Osborne Media, 2011. 13

[179] Eckart Zitzler and Simon Künzli. Indicator-based selection in multiobjective search. In *International Conference on Parallel Problem Solving from Nature*, pages 832–842. Springer, 2004. 109

[180] Eckart Zitzler, Lothar Thiele, Marco Laumanns, Carlos M Fonseca, and Viviane Da Fonseca Grunert. Performance assessment of multiobjective optimizers: An analysis and review. *TIK-Report*, **139**, 2002. 84

[181] Eckart Ziztler, Marco Laumanns, and Lothar Thiele. Spea2: Improving the strength pareto evolutionary algorithm for multiobjective optimization. *Evolutionary Methods for Design, Optimization, and Control*, pages 95–100, 2002. 78, 109