



**EFFICIENT TASK ALLOCATION IN INTERNET OF THINGS BASED
SYSTEMS**

**A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
GAZI UNIVERSITY**

BY

Enan Ameen Khalil SAFFAR

**IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF DOCTOR OF PHILOSOPHY
IN
COMPUTER ENGINEERING**

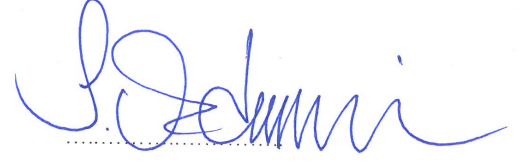
FEBRUARY 2019

The thesis study titled "EFFICIENT TASK ALLOCATION IN INTERNET OF THINGS BASED SYSTEMS" is submitted by Enan Ameen Khalil SAFFAR in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Computer Engineering, Gazi University by the following committee.

Supervisor: Prof. Dr. Suat ÖZDEMİR

Computer Engineering Department, Gazi University

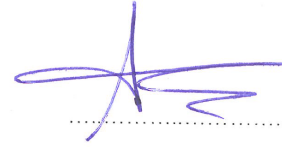
I certify that this thesis is a graduate thesis in terms of quality and content



Chair: Prof. Dr. Ebru A. SEZER

Computer Engineering Department, Hacettepe University

I certify that this thesis is a graduate thesis in terms of quality and content



Member: Prof. Dr. M. Ali AKCAYOL

Computer Engineering Department, Gazi University

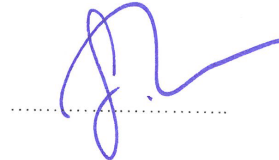
I certify that this thesis is a graduate thesis in terms of quality and content



Member: Assoc. Prof. Dr. Süleyman TOSUN

Computer Engineering Department, Hacettepe University

I certify that this thesis is a graduate thesis in terms of quality and content



Member: Asst. Prof. Dr. Mehmet DEMİRCİ

Computer Engineering Department, Gazi University

I certify that this thesis is a graduate thesis in terms of quality and content



Date: 04/02/2019

I certify that this thesis, accepted by the committee, meets the requirements for being a Doctor of Philosophy Thesis.

.....
Prof. Dr. Sena YAŞYERLİ

Dean of Graduate School of Natural and Applied Sciences

ETHICAL STATEMENT

I hereby declare that in this thesis study I prepared in accordance with thesis writing rules of Gazi University Graduate School of Natural and Applied Sciences;

- All data, information and documents presented in this thesis have been obtained within the scope of academic rules and ethical conduct,
 - All information, documents, assessments and results have been presented in accordance with scientific ethical conduct and moral rules,
 - All material used in this thesis that are not original to this work have been fully cited and referenced,
 - No change has been made in the data used,
 - The work presented in this thesis is original,
- or else, I admit all loss of rights to be incurred against me.



Enan Ameen Khalil SAFFAR

04/02/2019

EFFICIENT TASK ALLOCATION IN INTERNET OF THINGS BASED SYSTEMS

(Ph. D. Thesis)

Enan Ameen Khalil SAFFAR

GAZI UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

February 2019

ABSTRACT

Internet of Things is a fast growing technology for information collection, communications and processing. Successful applications of Internet of Things aim to interconnect objects with varied capabilities within the same heterogeneous network. The goal is to allow the network entities to cooperate and make their resources available in order to perform the demanded task. However for variety of Internet of Things objects minimizing the energy to be spent for task allocation purposes will be one of the primary constraints. Currently, almost all existing studies employ heuristic optimizations to cope with different aspects of task allocation problems without taking into consideration the heterogeneous nature of IoT objects in terms of their computation and energy levels. In this thesis, the problem of task allocation in Internet of Things is addressed. The problem is modeled using task groups and virtual objects concept by adopting evolutionary based methods. Considering different design requirements of different applications, we have proposed seven novel protocols to meet the requirements of different application scenarios. The proposed protocols are tailored to meet different goals of energy efficiency, extended operational and stability periods, and maximized computation power. To the best of our knowledge, our work is among the first works to propose task groups and virtual objects based framework to solve the task allocation problem in IoT, and the first work that adopts meta-heuristic methods for this purpose. To evaluate the proposed protocols several evaluation metrics are considered, such as: energy efficiency, number of operational rounds, length of stability periods, average available energy in virtual objects, computation power, computation time, and the quality of the proposed evolutionary algorithm. The performance of the protocols are analyzed and benchmarked by comparing them with the most relative work in the literature through extensive simulations. The results have proved the superiority of the proposed protocols.

Science Code : 92407

Key Words : Task allocation, resource management, energy efficiency, stability period, computation power, evolutionary algorithms, multi-objective optimization, Internet of Things (IoT)

Page Number : 176

Supervisor : Prof. Dr. Suat Özdemir

IoT TABANLI SİSTEMLERDE ETKİN GÖREV DAĞILIMI
(Doktora Tezi)

Enan Ameen Khalil SAFFAR

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
Şubat 2019

ÖZET

Nesnelerin İnterneti (Nİ), bilgi toplamak, iletişim ve bilgi işlemek için kullanılan gelişmekte olan bir teknolojidir. Nİ'nin başarılı uygulamaları, farklı kabiliyetlere sahip olan nesneleri bir heterojen ağ çatısı altında birbirine bağlı bir şekilde tutar. Amaç ağda bulunan nesnelerin kaynaklarını bulundurup ve dinamik bir şekilde işbirliği yaparak istenilen bir görevi yerine getirmektir. Ancak Nİ'nin birçok cihaz için görev dağılımında harcanacak enerjiyi minimize etmek hayati bir önem taşımaktadır. Günümüzde, bu konu ile ilgili mevcut çalışmaların hemen hemen tümü nesnelerin farklı enerji seviyelerini ve çeşitli işletim güçlerini dikkate almadan sezgisel en iyileme yöntemlerini kullanarak görev dağılımının farklı sorunlarını çözmeye uğraşmıştır. Bu tezde nesnelerin İnternetin'de görev dağılımı problemi ele alınmıştır. Problem görev grupları ve sanal nesneler kavramını kullanarak evrimsel yöntemler çatısı altında modellenmiştir. Ayrıca, farklı uygulamaların farklı tasarım özelliklerini düşünerek, farklı uygulama senaryoların gereksinimlerini karşılayabilen yedi yeni protokol geliştirilmiştir. Geliştirilen protokoller, enerji verimliliği, uzatılmış operasyonel ve stabil süreler ve işleme gücü artırmak gibi hedeflere uyacak şekilde tasarlanmıştır. Bilgimiz dahilinde çalışmamız Nİ'de görev dağılımı problemi için görev grupları ve sanal nesneler kavramı tabanlı bir çözüm çerçevesi öneren ilk çalışmalar arasındadır. Ayrıca, meta sezgisel yöntemleri kullanarak bu problemi çözen ilk çalışmadır. Geliştirilen protokollerin performansını değerlendirmek için, enerji verimliliği, operasyonel tur sayısı, stabil sürelerin uzunluğu, sanal nesnelerde ortalama enerji miktarı, hesaplama gücü, gereken hesaplama zamanı ve önerilen evrimsel algoritmanın kalitesi gibi çeşitli değerlendirme metrikleri kullanılmıştır. Protokollerin performansı analiz edilmiş ve literatürdeki en ilgili çalışmayla karşılaştırılmıştır. Sonuçlar geliştirilen protokollerin performans üstünlüğünü kanıtladı.

Bilim Kodu : 92407
Anahtar : Görev dağılımı, kaynak yönetimi, enerji verimliliği, stabil süre, işleme
Kelimeler gücü, evrimsel algoritmalar, çok amaçlı optimizasyon, Nesnelerin
İnterneti (Nİ)
Sayfa Adedi : 176
Danışman : Prof. Dr. Suat Özdemir

ACKNOWLEDGEMENTS

First of all, I would like to express my sincere gratitude to the Almighty ALLAH the Most Gracious for His grace and sustenance upon my life for the successful completion of this thesis. This thesis would not have been possible without the guidance, the advice and the support of several individuals who contributed in the completion of this work. First and foremost, I would like to express my appreciation to my supervisor Dr. Suat Özdemir, Professor at the Department of Computer Engineering, Gazi University for his directions, constructive comments, intellectual guidance, advice, and constant encouragements especially when I was confused. Heartfelt thanks to him, for not only supporting me as a supervisor, but also as a big brother. Also, I would like to express my special appreciation to Dr. Süleyman Tosun, whose valuable comments enriched my research. Moreover, I would like to thank Dr. Bara'a A. Attea, Professor at the Department of Computer science, University of Baghdad who was the first to teach me the scientific research methods and I will always remember and appreciate her helps. Finally and most of all, I am very grateful to my family especially my parents, they taught me the value of knowledge and the importance of being honest.

TABLE OF CONTENTS

	Page
ABSTRACT	iv
ÖZET	v
ACKNOWLEDGEMENTS	vi
TABLE OF CONTENTS	vii
LIST OF FIGURES	xi
LIST OF TABLES	xiv
LIST OF ALGORITHMS	xvii
LIST OF ABBREVIATIONS AND SYMBOLS	xviii
1. INTRODUCTION	1
2. TASK ALLOCATION PROBLEM IN IoT	7
2.1. Background.....	7
2.2. The Concept of IoT.....	10
2.3. Elements of IoT	11
2.3.1. Radio frequency identification (RFID)	12
2.3.2. Wireless sensor networks (WSNs).....	12
2.3.3. Embedded systems and nanotechnology.....	12
2.4. Technologies of the IoT	13
2.4.1. Communication.....	13
2.4.2. Sensing	15
2.4.3. Computation.....	15
2.4.4. Identification and addressing	16
2.4.5. Semantics	16
2.5. The main services of IoT	17

	Page
2.6. Architecture of IoT	18
2.7. Platforms of IoT	21
2.8. Applications of IoT.....	23
2.9. Challenges of IoT	26
2.10. Task Allocation in IoT.....	31
2.10.1. Major challenges in task allocation in IoT	31
2.10.2. Literature review	33
2.11. The Concept of Task Groups and Virtual Objects (The Clustering).....	37
2.12. Consensus Based Approach for Task Allocation (CBATA).....	40
2.12.1. Virtual objects selection	41
2.12.2. Consensus-based negotiation and task allocation.....	42
2.13. Problem complexity.....	43
3. METAHEURISTICS AND EVOLUTIONARY OPTIMIZATION.....	45
3.1. Metaheuristics.....	45
3.2. Evolutionary Algorithms (EAs)	48
3.2.1. General algorithmic framework of EA	49
3.2.2. Problem related aspects.....	51
3.2.3. Parameter control in EAs	53
3.3. Description of some important issues in EAs.....	54
3.3.1. Solution infeasibility	54
3.3.2. Exploration and exploitation.....	54
3.4. Evolutionary Multi-objective Optimization	55
3.4.1. Multi-objective optimization problems (MOPs).....	56
3.4.2. Dominance concept and Pareto optimality	57

	Page
3.4.3. Elitist non-dominated sorting genetic algorithm II (NSGA-II)	59
4. THE PROPOSED TASK ALLOCATION PROTOCOLS WITH METAHEURISTIC METHODOLOGY	63
4.1. Problem Definition and System Model	65
4.2. Overview of the Proposed Protocols in Context of Scenario #1	66
4.2.1. Virtual objects election phase	72
4.2.2. Association phase.....	85
4.2.3. Toward stability aware protocols.....	85
4.3. Overview of the Proposed Protocols in Context of Scenario #2	88
4.3.1. Modified-CBATA (M-CBATA).....	89
4.3.2. Steady task allocation protocol (STAP).....	92
4.3.3. Multi-objective task allocation protocol (MOTAP).....	99
5. SIMULATION RESULTS AND DISCUSSION	109
5.1. Performance Analysis of Protocols in Context of Scenario #1	110
5.1.1. Simulation environment.....	110
5.1.2. Network energy model.....	112
5.1.3. Parameters and rules settings in the evolutionary task allocation algorithms	112
5.1.4. Results of ETAP1 and ETAP2.....	113
5.1.5. Results of SETAP1 and SETAP2 (The effect of energy aware heuristic on ETAP1 and ETAP2)	124
5.2. Performance Analysis of Protocols in Context of Scenario #2	128
5.2.1. Simulation environment.....	128
5.2.2. Parameters and rules settings for network energy model and the evolutionary task allocation algorithms	129
5.2.3. Results of simulation (by MATLAB simulator).....	130

	Page
5.2.4. Results of simulation (by OMNeT++ simulator).....	149
6. CONCLUSIONS AND FUTURE WORK.....	153
6.1. Conclusions	153
6.2. Summary of Thesis Contributions.....	154
6.3. Summary of the Simulation Results	156
6.4. Future Research	159
REFERENCES.....	161
CURRICULUM VITAE	175

LIST OF FIGURES

Figure	Page
Figure 2.1. The raped development of the internet from 1995	8
Figure 2.2. Internet of Things	11
Figure 2.3. The taxonomy of Internet of Things.....	18
Figure 2.4. Architecture of IoT (a): Three layered model (b): Middle-ware based model (c): Five layered model	19
Figure 2.5. Applications domains of IoT and relevant major scenarios	24
Figure 2.6. The concept of task groups and virtual objects	40
Figure 2.7. A reference scenario for CBATA.....	41
Figure 3.1. Detailed overview of EA	52
Figure 3.2. Multi-objective optimization problem based on domination concept while maximizing two objectives	59
Figure 4.1. First-order radio model.....	66
Figure 4.2. The concept of task groups and virtual objects in scenario #1	67
Figure 4.3. The conceptual visualization of the proposed evolutionary task allocation protocols in scenario #1	70
Figure 4.4. Examples for genotype (a) and phenotype (b) individuals of evolutionary task allocation protocols in scenario #1	74
Figure 4.5. Two-point cut and cross fill.....	82
Figure 4.6. An example of mutation	83
Figure 4.7. The concept of task groups and virtual objects in scenario #2.....	89
Figure 4.8. Illustrative example for CBATA and M-CBATA (a): A network with two task groups (b): Object i joined to the network (c): Object j joined to the network (d): Object k joined to the network	91
Figure 4.9. Population visualization in STAP	95
Figure 4.10. The General layout of MOTAP	101

Figure 4.11. Examples for genotype (a) and phenotype (b) individuals of MOTAP	103
Figure 5.1. Average dissipated energy in 5 test instances for different object density settings	116
Figure 5.2. Average dissipated energy in 5 test instances for different number of tasks.....	117
Figure 5.3. Average dissipated energy in each task group for 5 test instances with network dimension = 200×200 , object density = 250 and number of tasks = 6	118
Figure 5.4. Average number of rounds until each task group becomes non-operational for Objects density = 250.....	121
Figure 5.5. Average number of rounds until each task group becomes non-operational for Objects density = 500.....	122
Figure 5.6. Convergence of ETAP1 toward optimal solution for 50 generations + initialization phase.....	123
Figure 5.7. Convergence of ETAP2 toward optimal solution for 50 generations + initialization phase.	123
Figure 5.8. Number of rounds until each task group becomes unstable for network with Objects density = 250	125
Figure 5.9. Number of rounds until each task group becomes unstable for network with Objects density = 500.....	126
Figure 5.10. Average dissipated energy in 5 test instances for different network dimensions	133
Figure 5.11. Average dissipated energy in 5 test instances for different number of objects	134
Figure 5.12. Average dissipated energy in 5 test instances for different number of tasks	134
Figure 5.13. Average dissipated energy in each task group	135
Figure 5.14. Average dissipated energy of virtual objects for different network dimensions	137
Figure 5.15. Average dissipated energy of virtual objects for different number of objects	137

Figure 5.16. Average dissipated energy of virtual objects for different number of tasks.....	138
Figure 5.17. Average energy of virtual objects in each task group for number of objects = 200, network dimensions = 200×200 unit and number of tasks = 6	139
Figure 5.18. Average number of rounds until each task group becomes non-operational	141
Figure 5.19. Average number of rounds until each task group becomes unstable	142
Figure 5.20. Average processing power of virtual objects in each task group for number of tasks = 6	144
Figure 5.21. Convergence of evolutionary algorithm in STAP toward optimal solution for 100 generations + initialization phase	145
Figure 5.22. Convergence of evolutionary algorithm in MOTAP toward optimal solution for 100 generations + initialization phase (a): Convergence of first objective (processing power) (b): Convergence of second objective (energy efficiency)	146
Figure 5.23. Non-dominated solutions of MOTAP for one network.....	146
Figure 5.24. Number of rounds until each task group becomes non-operational	150
Figure 5.25. Number of rounds until each task group becomes unstable	151

LIST OF TABLES

Table	Page
Table 2.1. The definitions and standards of IoT According to different research groups	10
Table 2.2. Technical specifications of some LPWAN platforms	14
Table 2.3. The most commonly used operating systems in IoT	16
Table 2.4. Characteristics of some software platforms used in IoT framework	23
Table 5.1. Average dissipated energy (in joules) in 5 test instances for 200 × 200 network scale, 250 object and 4 task groups and different packet error rates	111
Table 5.2. Parameters and rules used in the evolutionary algorithms of scenario #1	113
Table 5.3. Average dissipated energy (in joules) in 5 test instances for different network scales	114
Table 5.4. Average dissipated energy (in joules) in 5 test instances for different object density settings	114
Table 5.5. Average dissipated energy (in joules) for 5 test instances for different number of tasks	115
Table 5.6. Average dissipated energy in each task group for 5 test instances with network dimension = 200 × 200, object density = 250 and number of tasks = 6	118
Table 5.7. Average dissipated energy in each task group for 5 test instances with object density = 250, number of tasks = 4 and different network dimensions	119
Table 5.8. Average dissipated energy in each task group for 5 test instances with network dimension = 200 × 200, number of tasks = 4 and different object density	120
Table 5.9. The mean value of virtual objects energies (in joules) for 8 task groups	124
Table 5.10. Average dissipated energy (in joules) in 5 test instances for different network scales in a single round	127
Table 5.11. Average dissipated energy (in joules) in 5 test instances for different number of objects in a single round	127

	Page
Table 5.12. Average dissipated energy (in joules) in 5 test instances for different number of tasks in a single round.....	127
Table 5.13. Network energy model parameters settings.....	129
Table 5.14. MOTAP parameters and rules	130
Table 5.15. Average dissipated energy (in joules) in 5 test instances for different network scales	131
Table 5.16. Average dissipated energy (in joules) in 5 test instances for different object density settings	132
Table 5.17. Average dissipated energy (in joules) for 5 test instances for different number of tasks	132
Table 5.18. Average dissipated energy (in joules) for 5 test instances in each task group.....	135
Table 5.19. Average dissipated energy of virtual objects (in joules) for different network scales	136
Table 5.20. Average dissipated energy of virtual objects (in joules) for different object density settings	136
Table 5.21. Average dissipated energy of virtual objects (in joules) for different number of tasks	136
Table 5.22. Average energy of virtual objects (in joules) in each task group for number of objects = 200, network dimensions = 200×200 unit and number of tasks = 6.....	140
Table 5.23. Average number of rounds until each task group becomes non-operational for different object density	141
Table 5.24. Average number of rounds until each task group becomes unstable for different object density	142
Table 5.25. Average processing power of virtual objects (in MHz) in each task group for number of tasks = 6	143
Table 5.26. Time duration (in seconds) for each protocol for different network scales	147
Table 5.27. Time duration (in seconds) for each protocol for different number of objects.....	148

	Page
Table 5.28. Time duration (in seconds) for each protocol for different number of tasks	148
Table 5.29. STAP and MOTAP objective functions' values for different number of generations	149
Table 5.30. Number of rounds until each task group becomes non-operational for 5 random generated networks	150
Table 5.31. Number of rounds until each task group becomes unstable for 5 random generated networks	152
Table 6.1. Evaluation summary for the energy efficiency of scenario #1 protocols	157
Table 6.2. Evaluation summary for the energy efficiency of scenario #2 protocols	157
Table 6.3. Evaluation summary for the operational period of ETAP1, ETAP2 and CBATA within the context of scenario #1	157
Table 6.4. Evaluation summary for the operational period of scenario #2 protocols	157
Table 6.5. Evaluation summary for the stability period of scenario #1 protocols	158
Table 6.6. Evaluation summary for the stability period of scenario #2 protocols	158
Table 6.7. Evaluation summary for the computation power of scenario #2 protocols ...	158

LIST OF ALGORITHMS

Algorithm	Page
Algorithm 3.1. General algorithmic framework of Evolutionary Algorithm (EA)	51
Algorithm 3.2. Algorithmic Framework of NSGA-II.....	62
Algorithm 4.1. Outline of the evolutionary algorithm trajectory.....	69
Algorithm 4.2. Initial population generation algorithm for protocols in scenario #1	75
Algorithm 4.3. Population initialization algorithm in SETAP1 and SETAP2	86



LIST OF ABBREVIATIONS AND SYMBOLS

The symbols and abbreviations used in this study are presented below with their meanings.

Symbol	Meaning
P_{opt}	Optimal election probability
p_c	Probability of recombination
m_r	Mutation rate
E_{DA}	Energy for data aggregation
E_{Rx}	Reception energy
E_{Tx}	Transmission energy
E_{elec}	Energy dissipation per bit for the transceiver circuit
K_{opt}	Desired percentage of virtual objects
p_m	Probability of mutation
ε_{amp}	Energy amplifier
DV	Crowding distance value
MHz	Mega Hertz
Ω	Search space/search space
C	Cluster
N	Population size
T	Tasks of the network
n	Number of objects/Length of decision variables
t	A task
A	Network area
AP	Access point
Φ	Evaluation function

Abbreviation	Meaning
ADS	Application Deployment Server
BFM	Bit Flop Mutation
BTS	Binary Tournament Selection

Abbreviation	Meaning
CA	Certificate Authority
CBATA	Consensus Based Approach for Task Allocation
CCS	Central Control Station
CI	Computational Intelligence
EA	Evolutionary Algorithm
EMO	Evolutionary Multi-objective Optimization
ETAP1	Evolutionary Task Allocation Protocol-1
ETAP2	Evolutionary Task Allocation Protocol-2
EVOP	EVolutionary Operator
ICT	Information and Communications Technology
ID	IDentification
IM	Indication Message
IoT	Internet of Thing
ITU	International Telecommunications Union
LPWAN	Low Power Wide Area Network
LTE	Long-Term Evolution
M2M	Machine to Machine
M-CBATA	Modified-CBATA
MIT	Massachusetts Institute of Technology
MOEA	Multi-Objective Evolutionary Algorithm
MOP	Multi-objective Optimization Problem
MOTAP	Multi-Objective Task Allocation Protocol
NFC	Near Field Communication
NSGA-II	elitist Non-dominated Sorting Genetic Algorithm II
OS	Operating System
RFID	Radio-Frequency Identification
RTOS	Real-Time Operating System
SETAP1	Stable Evolutionary Task Allocation Protocol-1
SETAP2	Stable Evolutionary Task Allocation Protocol-2
SNR	Signal-to-Noise Ratio
SOP	Single-objective Optimization Problem
STAP	Steady Task Allocation Protocol

Abbreviation	Meaning
TG	Task Group
TID	Task IDentification
UWB	Ultra-Wide Bandwidth
VO	Virtual Object
WSN	Wireless Sensor Network



1. INTRODUCTION

Motivation

Recent advances in information, communication, and internet technologies impose a new form of life that is open to emergence of new breakthroughs. Internet of Things (IoT) is an emerging and fast growing technology, which links digital and physical entities together to enable a whole new class of applications and services by means of appropriate information and communication technologies [1]. For a wide variety of application domains, including agriculture, manufacturing, transportation, healthcare, environment surveillance, smart buildings, and many others, IoT is a significant approach to improve the flow of information across their organizational structures [2–4].

IoT concept aims to interconnect objects with different capabilities within the same heterogeneous network. Most of these entities suffer from lack of sufficient resources such as “energy”. For example, in wireless sensor networks, sensor nodes are often battery powered, and therefore have strictly limited energy reserve [5]. More extensive and continuous use of wireless network services will only aggravate this problem. Thus, for most of IoT systems, reducing the energy consumption (e.g., for communication or performing a given task) is a primary constraint. However, even today, research is still focused on performance and (low power) circuit design. There has been substantial research in the hardware aspects of mobile communications energy-efficiency, such as low-power electronics, power-down modes, and energy efficient modulation. However, due to fundamental physical limitations, progress towards further energy-efficiency will become mostly an architectural and software-level issue. Due to these limitations, there is a lot of focus on finding the best energy efficient at all layers of the networking protocol stack. For example, at the network layer, it is highly desirable to find methods for energy-efficient task allocation, route discovery and relaying of data from the objects to an external control station so that the lifetime of the network is maximized [6].

One way to minimize the effect of this resource constraint is enabling network entities to cooperate and make their resources available to perform the given tasks. In other words, the entities executing the same application should cooperate to reach the optimal allocation

of tasks among themselves. However, task allocation problem with the goal of increasing the lifetime of the network as well as identifying which tasks to be assigned to which objects is not a trivial task. Generally, when an object is in transmit mode, the transceiver drains much more energy from the battery than the microprocessor in active state or the sensors and the memory chip. The ratio between the energy needed for transmitting and for processing a bit of information is usually assumed to be much larger than one (more than one hundred or one thousand in most commercial platforms). For this reason, the communication protocols need to be designed according to paradigms of energy efficiency, while this constraint is less restrictive for processing tasks [7]. Then, the design of energy efficient task allocation protocols that consider the communication overhead is a very peculiar issue of IoT.

It is important to consider some key features that characterize many of IoT objects:

- i) Available resources (energy, processing, memory, object ability of perform a given task) are often limited. This is the case, for example, of battery powered nodes, which have limited energy amounts.
- ii) For a variety of IoT entities, minimizing the energy to be spent for communication/ computing purposes will be a primary constraint. Thereby the need to devise solutions that tend to optimize energy usage (even at the expenses of performance) will become more and more attractive.
- iii) Objects may provide redundant information that is not unique but can be generated by set of different objects which are capable of performing the same task.
- iv) The number of objects in the IoT is quickly overcoming the number of hosts in the traditional networks and most of these have a low reliability due mostly to the mobility and energy. This entails for a new paradigm of communication according to which objects coordinate with the other objects in groups and provide a unified service to the application that requires the service.
- v) The heterogeneity of the network in terms of capabilities and characteristics of the objects coupled with the need of an optimal creation of communications for energy efficient optimization.

Task allocation is a procedure of choosing, subdividing, coordinating and assigning the correct tasks to the correct entities. It is usually performed considering different aspects

such as network topology, energy constraints and processing capabilities of the network entities. However, most of the existing methods have generic and limited scope in extending resources regarding the application assigned to the network [8]. Moreover, although the problem of task allocation is extensively studied in the field of Wireless Sensor Networks (WSNs), the task allocation problem in IoT networks is an open research issue. IoT introduces much more dynamic and heterogeneous scenarios when compared to WSNs. In WSNs the nodes are managed by the same system and have similar characteristics whereas the IoT objects are grouped opportunistically as they provide cooperative services and then they have to find the way to act in a coordinated way [9]. In addition, the size and heterogeneity of the network in terms of capabilities and characteristics of the objects coupled with the need of an optimal creation of communications for energy efficient optimization make the problem of task allocation a very challenging procedure.

In its abstract level, we can consider the problem of task allocation as a special clustering problem. Accordingly, the optimal selection of task allocators with a high residual energy that is scattered in the area can be seen as an NP-hard problem [10]. Computational Intelligence (CI) is the study of adaptive mechanisms that enable or facilitate intelligent behavior in complex and changing environments. These mechanisms include paradigms that exhibit an ability to learn or adapt to new situations, generalize, abstract, discover, and associate. Different approaches of CI, including Evolutionary Algorithms (EAs), swarm intelligence, and more recently, harmony search, have been used by different researchers for energy-aware cluster-based routing. These algorithms are examples of population-based meta-heuristic algorithms, which have two important components: intensification and diversification [11]. For the algorithm to be efficient and effective, it must be able to generate a diverse range of solutions, including the potentially optimal solutions, so as to explore the whole search space effectively, while it intensifies its search around the neighborhood for an optimal or a nearly optimal solution. For EAs, genetic operators, such as crossover or reproductive recombination, mutation, and selection are used to evolve population's solutions based on their fitness (i.e., objective) function. With this in mind, the researchers must draw attention to consider the EA based methods in task allocation in IoT.

For all of the aforementioned reasons and to complete the lack of academic studies it is necessary to give a comprehensive and a systematic framework for task allocation problem in IoT. Thus a common middleware can be designed in order to ensure interoperability among different devices.

Thesis contributions

In this thesis we addressed the problem of task allocation in IoT. The thesis begins by presenting a comprehensive survey of concept, characteristics, technologies, applications and research challenges for IoT which have been developed and became popular in recent years. Then, the thesis tackles the problem of task allocation in IoT based applications considering special features and design characteristics of these applications. In this context, we considered a realistic and two basic scenarios and adopted meta-heuristic methods to solve this problem.

In the first scenario we assume that each object is capable of performing only a single task. Then, to cope with different requirements of the applications, we propose two sets of protocols: protocols with the goal of maximizing the network lifetime and protocols with the main goal of extending the stability period of the network. For typical applications (e.g., smart environments) we modeled a task allocation problem as a single objective optimization problem with the main goal to minimize energy consumption. We proposed two protocols with different objective models that can have a positive impact on the overall performance of the network. The objective models are tailored to meet the goal of minimizing the energy expenditures for communication in the process of tasks allocation to ensure maximal network longevity. On the other hand, for crucial applications (e.g., environmental monitoring, factory automation), we are motivated by the fact that most IoT based networks are heterogeneous especially in terms of energy levels. Hence, we proposed two more protocols with the goal of minimizing energy consumption as well as maximizing the stability periods to ensure stable and balanced operation of whole network. To validate the proposed protocols, we applied an extensive MATLAB based analysis and used more than 105 various test instances in the simulations. We evaluated the protocols by using different measurements and benchmarking parameters and compare them with the most relevant algorithm in the literature.

In the second scenario a more realistic and complex scenario is considered where objects are capable of performing a variety of tasks and can be members of more than one group. In this context we developed two novel protocols. The first protocol formats the problem of task allocation as a meta-heuristic optimization problem with the goal of increasing stability and operational periods of the network by developing a novel single objective optimization algorithm. This algorithm designed with an elegant objective function and heterogeneity aware heuristics that can cooperate to achieve the optimal goal of the algorithm. This thesis presents the first attempt to utilize EAs in this direction. The key idea of the proposed protocol is to hypothesize a possible energy-based heuristics for the individual solution's initialization, fitness evaluation, and mutation to properly maintain longer stability periods. The second protocol jointly formulates the computational power utilization and energy efficiency problems in task allocation of IoT as a novel Multi-Objective Optimization Problem (MOP). To the best of our knowledge, this is the first work attempting to address computational power utilization and energy efficiency problems in task allocation of IoT as a MOP. Also in this study, we modified the most relative method to the stated problem, namely CBATA [8, 12], and redirect its goal toward energy efficiency by developing M-CBATA algorithm. To evaluate these protocols, we performed extensive MATLAB based analysis and application layer simulations based on OMNeT++ using several benchmarking metrics.

Thesis outline

- Chapter 2: presents a general view of IoT, explaining their main characteristics and issues, introducing the applications in which the IoT are used. Later, the chapter explains the major challenges in IoT including the task allocation problem as one of the main problems in IoT. Finally, the chapter ends stating CBATA protocol as it the most relevant algorithm in the literature.
- Chapter 3: presents the concept of meta-heuristics and evolutionary algorithms defining both single objective and multi-objective optimization problems. Also this chapter presents the major components and characteristics of these problems. Finally, this chapter introduces the conceptual algorithmic framework for EAs.
- Chapter 4: presents the proposed task allocation protocols in the context of IoT. The chapter begins by defining the problem and describing the system model. Then, the

general layout of each of the proposed protocols is given. The components and evolutionary operators are, then, explained in details in both formal and informal ways.

- Chapter 5: contains simulation results and discussions. It begins with introducing the required test-bed settings for the playgrounds and the protocols parameters. The results of the proposed protocols are, then, compared with the results of the most related work in the literature. In this chapter, the results of protocols of the first scenario are presented first. Then, the results of the protocols of the second and the more complex scenario are provided supported by the application layer simulation results that are calculated by the network simulator OMNeT++. Energy efficiency, duration of the stability and operational periods, computational power optimization, computational time and evolutionary algorithm quality are used as evaluation metrics.
- Chapter 6: presents conclusion remarks of the whole work of this thesis, summarizing the results and giving some candidate future research directions.

2. TASK ALLOCATION PROBLEM IN IoT

The International Telecommunications Union (ITU) suggested that the “Internet of Things will connect the world's objects in both a sensory and intelligent manner” [13]. IoT is an environment that contains different embedded devices interacting with each other to perform tasks related to information collection, communications and processing. Successful applications of IoT aim to interconnect objects with various capabilities within the same heterogeneous network. The goal is to allow network entities to cooperate and make their resources available in order to perform the demanded task. However, assigning tasks to group of heterogeneous objects that are equipped with limited resources poses a challenging task. The most limited and valuable resource for variety of IoT objects is battery power. Therefore, improving the *energy efficiency* in task allocation process is one of the primary objectives.

This chapter provides an exposition of the fundamental aspects of IoT, and presents some of their applications as they are identified as one of the most important technologies in the recent years. Afterwards, the main issues and characteristics of this technology will be highlighted. Later, the major problems and challenges encountered while implementing IoT systems are explained. The main body of this chapter is elaborated in the final section where the details of the task allocation problem in IoT are explained and recapitulated with a main focus on the task groups and virtual object concept as it are realized to succeed to provide energy-efficient solutions.

2.1. Background

Nowadays, around four billions people around the world use the Internet for browsing the Web, sending and receiving emails, accessing multimedia content and services, playing games, using social networking applications and many other tasks. According to a report prepared by information and communication technologies agency for the United Nations, 50.1% of the world population is active internet user since September 2016 (See Figure 2.1*) [14].

* Data are calculated in December of each year except for 2016.

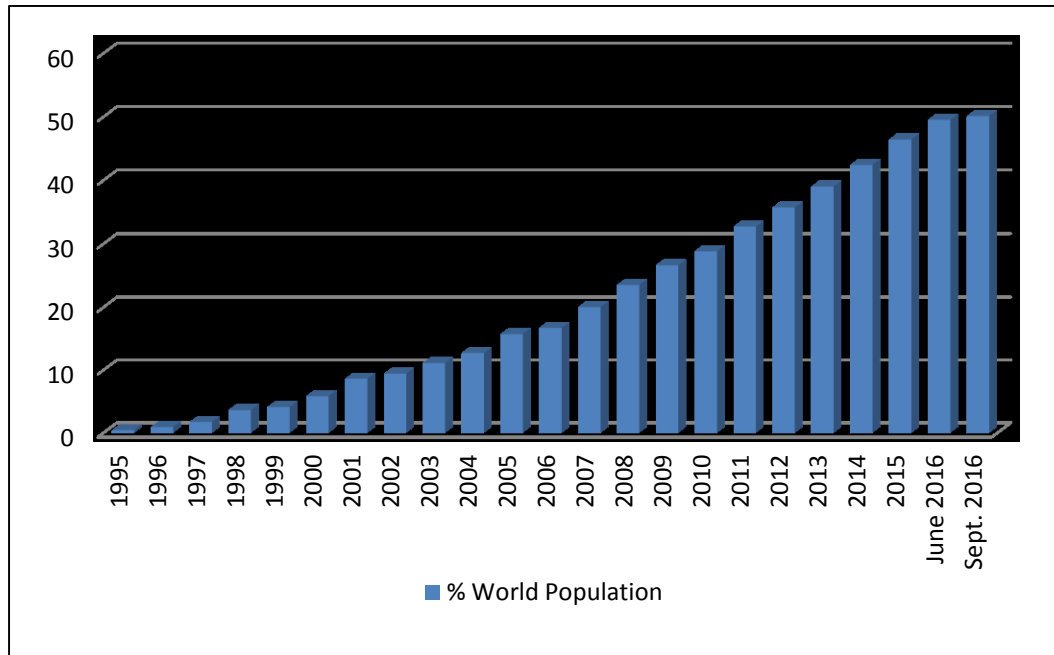


Figure 2.1. The rapid development of the internet from 1995

In parallel with the developing of Internet and communication technologies, smart and connectable objects have started to take more places in our daily lives. The capability of these objects to access to the Internet and connect and exchange information with each other over the Internet at an unprecedented rate has realized the idea of the Internet of Things (IoT). The IoT smart objects are expected to reach over 500 billion entities deployed globally by the end of 2020 [15]. The data that will be collected from all the IoT objects are expected to be very huge and diverse. Very different types of data can be obtained from many different applications, such as the number of vehicles at an intersection from the smart traffic lights, the density of people in a region from the mobile phones, the average indoor temperature from the air conditioners in the houses, or the rainfall intensity of an area from the vehicle wipers. Similarly, IoT can be used in the renewable energy market as well as in providing a better life for people in terms of food, clothing, housing, transportation, education and entertainment [16].

IoT refers to networks of objects that communicate with other objects and with computers through the Internet and exchange information about their status and/or the surrounding environment. “Things” may include virtually any object for which remote communication, data collection, or control might be useful, such as vehicles, appliances, medical devices, electric grids, transportation infrastructure, manufacturing equipment, or building systems

[17]. The concept of IoT originated at the Auto-ID Center at the Massachusetts Institute of Technology (MIT) by Kevin Ashton in 1999 [18]. In the early stages, radio frequency identification systems are used to connect devices and transmit information via radio frequency to the Internet to achieve intelligent identification and management. The importance of IoT quickly realized by some researchers; therefore, several organizations established research groups to discuss possible architectures and standards while trying to develop practicable IoT systems. As shown in Table 2.1, definitions of IoT can be found in different researches, such as United States, European, Japan, and China. Although definitions from different organizations are somehow different, the requirements for IoT are essentially the same, such as being able to integrate heterogeneous devices, ubiquitous data exchange, localization and tracking capabilities, and even being able to make the simple decision by themselves. To provide better services for the end user, the intelligence has become a vital issue of IoT [19].

In 2005, the International Telecommunication Union (ITU) formally identified the concept of the IoT at the world summit on the information society in Tunisia and released an ITU Internet report that provided an in-depth introduction to the IoT and its effects on businesses and individuals around the world [20]. The report contained information on key emerging technologies, market opportunities, and policy implications. In the report, the IoT is described as follows: connections will multiply and create an entirely new dynamic network of networks, namely, the IoT. This trend has led to a promising IoT concept that is an emerging field of study [3, 21-23]. Nowadays, the IoT vision provides a large set of opportunities to users, manufacturers and companies. In fact, IoT technologies have wide applicability in many productive sectors including, e.g., environmental monitoring, health-care, inventory and product management, workplace and home support, security and surveillance [23].

Table 2.1. The definitions and standards of IoT according to different research groups

Organization	Web link
MIT	http://www.autoidlabs.org/
EPCglobal	http://www.gs1.org/epcglobal
National Intelligence Council	http://www.fas.org/irp/nic/disruptive
European Commission	http://cordis.europa.eu/fp7/ict/enet/home_en.html
European Commission	http://www.smart-systems-integration.org/public/internet-of-things
Ubiquitous ID Center	http://www.uidcenter.org/
Internet of Things China	http://www.iotcn.org.cn/

2.2. The Concept of IoT

IoT is an emerging paradigm for information collection, communications and processing. In literature there are different definitions for the concept of IoT. Due to the multifaceted nature of IoT concept, definitions must be made on the basis of internet, things and their meanings [24]. In what follows some of the most used and prominent definitions are listed:

- A world-wide network of interconnected objects uniquely addressable based on standard communication protocols [2].
- A global infrastructure for information collection that provides advanced services by interconnected objects based on current and evolving information and communication technologies [25].
- A network of intelligent objects with the ability to act according to the environmental conditions, to share information, data and resources and to be organized automatically [26].

Radio-Frequency IDentification (RFID) tags are considered to be the basic idea behind IoT concept, thus, all of the objects that are equipped with radio tags may be able to communicate with other objects equally tagged through internet or any other protocols, to collaborate and to reach a common goal [27]. As could be seen in Figure 2.2, objects (or things) can be RFID tags, sensors, actuators, mobile phones, etc. which through unique addressing schemes, are able to interact with each other and cooperate with their

neighbors. In IoT the complex tasks are divided into several simple sub-tasks, each is assigned to an IoT object, in this way the performance of the entire system is improved [28]. From a conceptual standpoint, the IoT concept builds on three pillars, related to the ability of smart objects to be identifiable (anything identifies itself), communicable (anything communicates) and intractable (anything interacts).

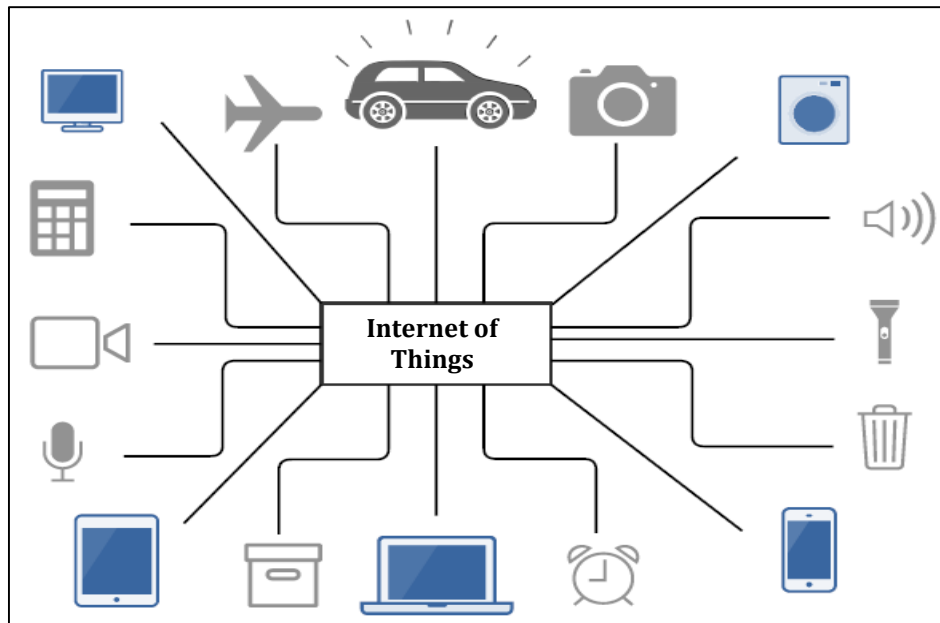


Figure 2.2. Internet of Things

From a system-level perspective, IoT can be looked at as a highly dynamic and radically distributed networked system, composed of a very large number of smart objects producing and consuming information. The ability to interface with the physical realm is achieved through the presence of devices able to sense physical phenomena and translate them into a stream of information data, as well as through the presence of devices able to trigger actions having an impact on the physical realm (through suitable actuators) [1].

2.3. Elements of IoT

The main motive of IoT is to make the things or objects in the world to be connected and able to share information automatically just like people sharing information [1]. To accomplish this motive, there are many technologies that help the things to communicate among them.

2.3.1. Radio frequency identification (RFID)

Radio Frequency Identification (RFID) is a wireless technology that is used for identification of objects [29]. Due to its reduced cost and increased abilities of tracking the location, status of objects and remote reading, it is more preferred than the usual bar code technology. It is the root cause factor for an object to be identified so that it can be connected to the internet. RFID uses radio waves to identify things and transfer its information to the RFID reader without physical contact [30].

2.3.2. Wireless sensor networks (WSNs)

Wireless Sensor Networks (WSNs) play a vital role in connecting the physical world to the information world [30]. These networks monitor the changes happening in the environment and report them so that corresponding responses can be taken. WSN contain many independent nodes that communicate among themselves with the help of wireless radio. The nodes contain a sensor (collecting data), microcontroller (computing data and controlling), memory (storing program and data), radio transceiver (for communication with other nodes) and battery (power supply) [31]. These sensors work together to collect data and send the results to the base station.

2.3.3. Embedded systems and nanotechnology

Embedded systems are intelligent and things with embedded intelligence become smart things. These make things perform certain actions automatically. For example a smart washing machine can wash and dry clothes automatically without human intervention. Nanotechnology is used to inject intelligence in things which are called smart devices (e.g., smart phones, smart watches, smart glasses, etc.). These smart devices are able to process information, self-configure and take independent decisions [32]. They are connected with the help of LAN, GPRS, WSN, Wi-Fi, 3G, etc.

2.4. Technologies of the IoT

2.4.1. Communication

Due to the heterogeneous nature of IoT objects, communication technologies with different characteristics can be used in the communication of these objects. Typically, most of the objects must operate using low power in noisy environments with lossy communication links. Examples of the communication protocols that are used in IoT include WiFi, IEEE 802.15.4, Bluetooth, Z-wave, LTE and LPWAN technologies.

The main communication technology used in IoT is WiFi that uses radio waves to exchange data among objects within 100 m range [33]. On the other hand, Bluetooth provides communication links to exchange data between devices over short distances using short-wavelength radio to minimize power consumption. In this context, the Bluetooth Special Interest Group (SIG) produced Bluetooth 4.1 that provides Bluetooth Low Energy as well as high-speed and IP connectivity to support IoT [3, 34]. Recently, this group has developed Bluetooth 4.2 and later Bluetooth 5.0 with more supportive features for IoT applications. Compared with older versions, Bluetooth 4.2 has provided lower energy consumption, more reliable connections and longer packet transmissions. Bluetooth 4.2 has also introduced the IPSP 6 (Internet Protocol Support Profile 6) protocol, which facilitates the connection of intelligent objects in smart homes. On June 16, 2016, Bluetooth 5.0 is announced during a media event in London. Bluetooth 5.0 has four times the communication range, two times the transmission rate, and eight times the data transmission capacity of older versions of Bluetooth [35]. All these advances are important to allow the intelligent objects to be connected throughout a smart home.

LTE (Long-Term Evolution) is a standard wireless communication protocol for high-speed data transfer (300 Mbit/s) between mobile phones based on GSM/EDGE and UMTS/HSPA network technologies [36]. LTE provides multicasting and broadcasting services for fast-travelling devices. LTE-A (LTE Advanced) is an improved version of LTE including bandwidth extension which supports up to 100 MHz, downlink and uplink spatial multiplexing, extended coverage, higher throughput and lower latencies[36, 37].

Finally, the LPWAN (Low Power Wide Area Network) is a long-range and cost-effective wireless communication technology that provides optimum power and resource management [38]. The main goal of LPWAN is to connect devices with low power and at a low bit rate over long distances using low bandwidth. In this way, this technology could be used in many M2M (Machine to Machine) and IoT applications that have limited budgets and energy problems. There are many standards development organizations and private industry alliances that make researches on this technology. These organizations have developed several platforms which support LPWAN, such as SIGFOX [39], LORAWAN [40], INGENU [41] and TELENDA [42]. These platforms have used a variety of features and techniques to achieve longer communication range, lower power consumption and higher scalability. Table 2.2 summarizes some of the technical characteristics of these platforms [43, 44].

Table 2.2. Technical specifications of some LPWAN platforms

Model	SIGFOX UNB DBPSK(UL), GFSK(DL)	LORAWAN CSS	INGENU RPMA-DSSS(UL), CDMA(DL)	TELENDA UNB 2-FSK
Band	868/915 MHz	433/868/780/915 MHz	2.4 GHz	SUB-GHZ Bands ₁
Data Rate	100 bps	50 kbps	19.5 kbps	346 Mbps
Coverage range (km)	10 (Urban), 50 (Rural)	5 (Urban), 15 (Rural)	15 (Urban)	1 (Urban)
Advanced error correction	×	✓	✓	✓
Topology	Star	Star of Stars	Star/Tree	Star
packet Size	12B	Up to 250 Byte	10KB	65KB
Roaming	✓	✓	✓	✓
Authentication and Encryption	No Encryption	AES 128b	16B hash, AES 256b	At the development stage

In addition to the aforementioned protocols, RFID, 6LoWPAN, Ultra-Wide Bandwidth (UWB) and Near Field Communication (NFC) are some of the special short-distance communication technologies used in IoT applications. RFID consist of a tag and a reader. RFID tag represents a simple chip or label to identify the objects. RFID reader uses radio waves to send query signals to the label and retransmits the signal from the label to a database connected to a processing center. The objects are then detected from the reflected signals [45]. 6LoWPAN is the abbreviation of IPv6 over low-power Wireless Personal Area Networks- WPAN (IPv6 over Low power WPAN). 6LoWPAN is a Mesh network of nodes with low resources (energy, processing and memory units). Each node in this

network has a unique IPv6 address and can connect directly to the Internet and the cloud using IEEE 802.15.4 and open IP standards [46, 47]. NFC is a wireless communication technology uses the ISO 18092 standard to provide communication with radio frequency over short distances. NFC operates in a high frequency band (at 13,56 MHz) and supports data rates up to 424 kbps with a range up to 10 cm [48]. UWB is designed to support the communications in small ranges using low energy and high bandwidth. The use of this technology has recently increased to connect sensor nodes [49].

2.4.2. Sensing

The IoT sensing means collecting of data from related objects within the network and sending it back to a data warehouse, database, or cloud. The gathered data is analyzed to perform certain actions based on the required tasks. The IoT sensors can be smart sensors, actuators or wearable sensing devices. To connect the sensor network to the Internet a standard based on the Internet Protocol could be used. For example, single board computers integrated with sensors and built-in TCP/IP and security functionalities are typically used in IoT products [50, 51].

2.4.3. Computation

In IoT computation, there are two components hardware and software. Processing units such as microcontrollers and microprocessors and software applications provide the computation capability for the IoT. Various hardware platforms with microcontrollers and microprocessors were developed to run IoT applications. Examples of these platforms are Arduino [52], Intel Galileo [53, 54], WiSense [3], Raspberry PI [55, 56], Gadgeteer [57], BeagleBone and BeagleBone Black [58-60], Cubieboard [61], UDOO [62], Z1 [63] and Mule [64].

On the other hand, many software platforms are utilized to provide IoT functionalities. Among these platforms, Operating Systems (OSs) are important since they run for the whole activation time of a device. There are several Real-Time Operating Systems (RTOS) that can provide supports for IoT applications. For instance, the Contiki RTOS [65] has been used widely in IoT scenarios. Contiki has a simulator called Cooja which allows researcher and developers to simulate IoT and WSN applications. TinyOS [66], LiteOS

[67] and Riot OS [68] are also examples of RTOS that are designed for IoT environments. Moreover, some auto industry leaders in partnership with Google established the Open Auto Alliance (OAA). OAA is planning to bring new features to Android platform that provide support for Internet of Vehicles (IoV) paradigm [69]. Some features of these operating systems are provided in Table 2.3.

Table 2.3. The most commonly used operating systems in IoT

Operating System	Language Support	Minimum Memory	Event Based Programming	Multi-threading	Dynamic memory
TinyOS [66]	nesC	1 KB	Yes	Partial	Yes
Contiki [65]	C	2 KB	Yes	Yes	Yes
LiteOS [67]	C	4 KB	Yes	Yes	Yes
Riot OS [68]	C/C++	1,5 KB	No	Yes	Yes
Android [70]	Java	-	Yes	Yes	Yes

Finally, cloud platforms play an important role in the computation in IoT. Smart objects can send the large data to the cloud for real-time processing. IoTCloud, OpenIoT, NimBits and Hadoop are examples of cloud platforms that support IoT [3].

2.4.4. Identification and addressing

Identification is used to provide an open identity to each object within the network. Identification is important to name and match services with their demand. Many identification methods are available for IoT. Examples of these methods include Electronic Product Codes (EPC) and Ubiquitous codes (uCode) [71].

Addressing the IoT objects is also important to distinguish between object ID and its address. Object ID refers to its name whereas object's address refers to its address within a communications network [72].

2.4.5. Semantics

Semantic in the IoT refers to the ability to extract knowledge smartly by different machines to provide the required services. Knowledge extraction includes discovering and using

resources, modeling information, and recognizing and analyzing data. Thus, semantic represents the brain of the IoT as it used to send demands to the right resource. The requirement of IoT applications of Semantic is solved using semantic web technologies such as the Resource Description Framework (RDF) and the Web Ontology Language (OWL) [73]. Recently, the World Wide Web consortium (W3C) has recommended the Efficient XML Interchange (EXI) format for a semantic service in the IoT [74]. EXI is important in the context of the IoT because it is designed to optimize XML applications for resource-constrained environments. In addition, EXI reduces the need for bandwidth without affecting resources such as battery life, code size, energy consumed for processing, and memory size. It converts XML messages to binary to reduce the required bandwidth and minimize the required storage size [3].

2.5. The main services of IoT

IoT has wide range of applications in many areas from home and office automation to production line and retail product tracking. To improve the development, optimization and speed of each IoT application one or more of IoT services are required. In general, IoT services can be categorized into four classes: Identity-related Services, Information Aggregation Services, Collaborative-Aware Services and Ubiquitous Services [3, 75, 76].

- **Identity-related Services:** The most basic and important services that are used in other types of services. Identity-related services are used whenever the identity information of the objects in the IoT application is needed. Every application that needs to bring real world objects to the virtual world has to identify those objects.
- **Information Aggregation Services:** These services are needed to collect and summarize raw sensory measurements that are needed to be processed and reported to the IoT application.
- **Collaborative-Aware Services:** These services are used together with the information aggregation services. The main goal of these services is to use the obtained data by the information aggregation services to make decisions and react accordingly.
- **Ubiquitous Services:** These services are used with the collaborative-aware services. They are intended to provide collaborative services to anyone, at any time, in anywhere.

The aforementioned concept of IoT and all of its related subjects and technologies are summarized in Figure 2.3.

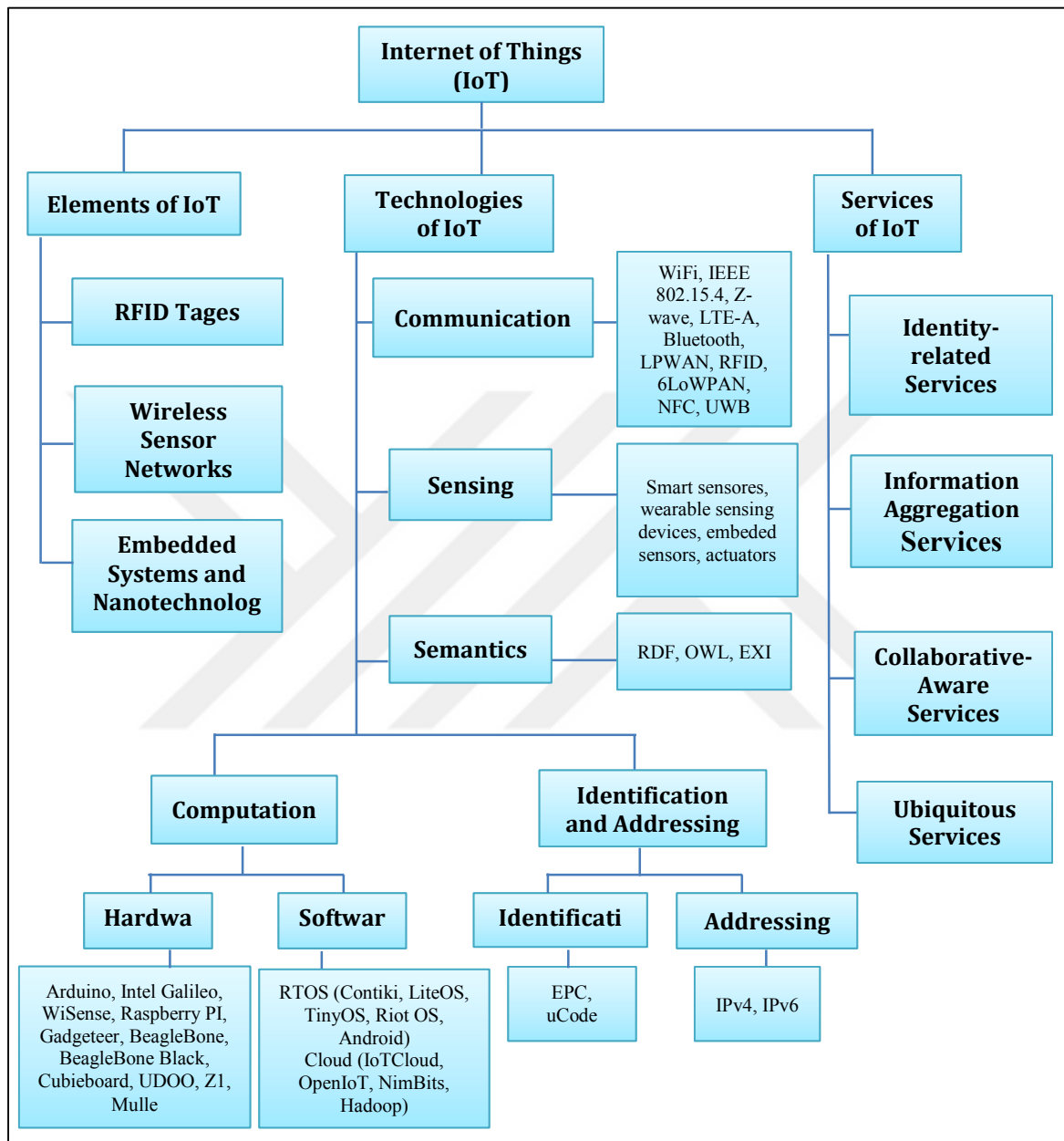


Figure 2.3. The taxonomy of Internet of Things

2.6. Architecture of IoT

The basic architecture of IoT consists of three layers. This architecture has been widely used to describe the IoT approach [19, 77-80]. As it can be seen in Figure 2.4 (a), the three layered architecture consists of the perception, network and application layers. The perception layer (also called the sensing layer or the technology layer) is the bottom layer.

This layer can be regarded as the hardware or physical layer and it is responsible of data collection process. The intermediate layer is the network layer. This layer is responsible for linking the perception layer and the application layer so that things can be passed from the perception layer to the application layer and systems, applications, services can be passed from the application layer to the perception layer. The application layer provides services and applications by analyzing and integrating the information received from the other two layers.

Although the three layered architecture is the basic model, other models that can bring more abstraction to the IoT architecture are also proposed in literature. Figures 2.4 (b) and (c) show the middle-ware based model and five layered model architectures, respectively.

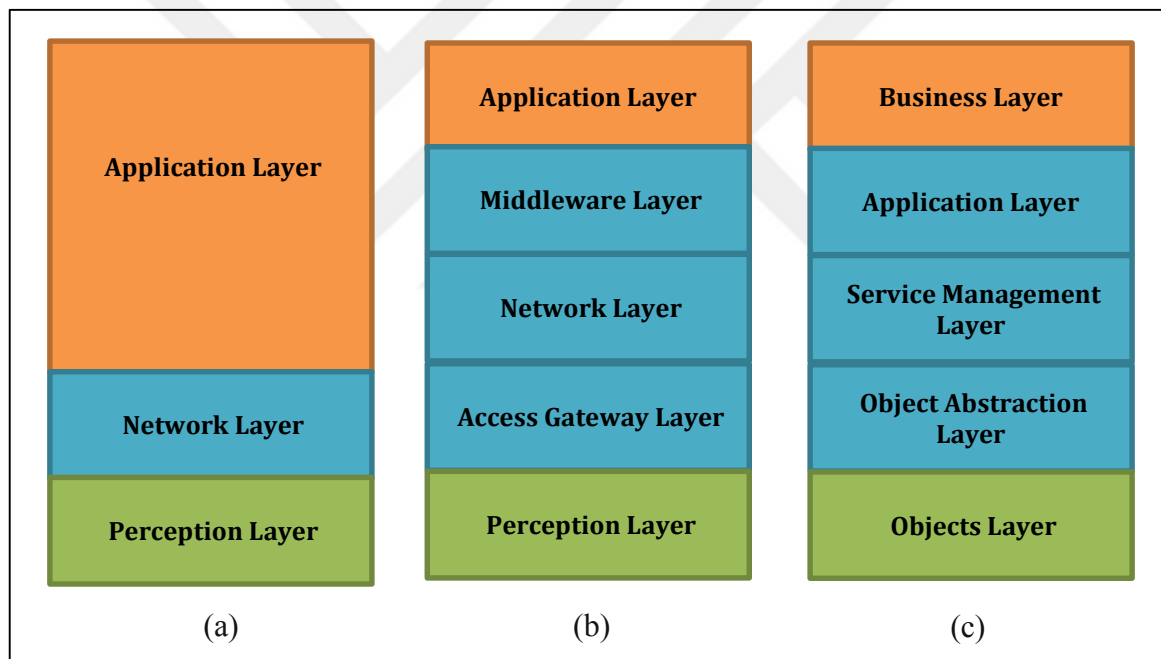


Figure 2.4. Architecture of IoT (a): Three layered model (b): Middle-ware based model (c): Five layered model

Middle-ware model adds the gateway and middleware layers to the basic three layered model of IoT [2], [81]. In middle-ware model the perception layer can be called the edge layer and it is used to provide a definition for data gathering. Unlike the three layered model which uses only one layer (i.e., the network layer) to connect the perception layer to the application layer, the middle-ware model uses the gateway layer together with the network layer to manage the communications in the IoT environment and to transmit messages between the objects and systems. The middleware layer is another layer added to

the middle-ware models structure. This layer is often used to provide a more flexible interface between hardware and applications. Finally, the top layer of the middle-ware model is the application layer. This layer has the same functionality and definition as the application layer in the three layered model.

The five layered model is another model used in the IoT architecture [82-84]. A brief discussion of the five layers of this model (not to be confused with the TCP/IP layers) is presented below:

- **Objects layer:** The first layer of the five layered model is the object layer (also called the perception layer). This layer represents the physical sensors of the IoT that perform information gathering and processing. In addition, this layer digitizes and transfers data to the object abstraction layer through secure channels. The big data created by the IoT are initiated at this layer.
- **Object abstraction Layer:** Exports the data generated at the objects layer to the service management layer using secure channels. The data transfer can be performed using various technologies such as RFID, 3G, GSM, UMTS, WiFi, Bluetooth, Infrared, ZigBee etc. Furthermore operations such as cloud computing and data management are handled in this layer.
- **Service management layer:** Also known as middleware layer. This layer matches requests with services based on addresses and names. Also, this layer processes the received data, makes decisions and provides the necessary services. Finally, the service management layer enables the IoT applications to work with heterogeneous objects without considering a specific hardware platform.
- **Application layer:** This layer is responsible of providing the services requested by customers.
- **Business layer:** This layer also known as management layer, it manages the overall IoT system activities and services. The responsibilities of this layer are building a business model, graphs, flowcharts, etc. based on the data received from the application layer. At the same time, this layer carries out designing, analyzing, implementation, evaluation, monitoring and development of components related to the IoT system.

2.7. Platforms of IoT

Section 2.4.3 has sight the light on some of the IoT platforms from the perspective of computation. These platforms can be divided into software and hardware platforms. This section focuses on the software platforms of IoT.

The purpose of any device in the IoT network is to communicate and exchange information with other devices and/or with cloud-based applications. The gap between the IoT devices and the data in the network is filled by the IoT platforms. An IoT platform provides links among objects and the data of the network. It also maintains background applications for understanding and analyzing the data. In this context, there are several IoT software platforms such as Appcelerator, AWS IoT, Ericsson Framework, IBM IoT Foundation Device Cloud and ThingWorx. Making a choice among these platforms is dependent on different features such as device management and integration support, information security, data gathering protocols, data analysis and visualization [85, 86]. These features are briefly summarized below:

- **Device Management and Integration Support:** Device management is one of the most important features expected from any IoT software platform. The platform should be able to monitor the status of objects and track their operation status, manage configuration and software updates, and provide device level error checking and error reporting techniques. At the end of the day, the platform must be able to provide users with device-level statistics. On the other hand, integration support is another important feature expected from the IoT software platform. APIs must provide access to essential operations and data. This access is usually accomplished through REST-APIs (REpresentational State Transfer APIs).
- **Information Security:** The information security that is required to operate an IoT software platform is much higher than the information security required for general applications and services. Generally, the network connection between the IoT objects and the IoT platform must be encrypted with a strong encryption mechanism to avoid potential eavesdropping. However, most of the low-cost and low-powered objects involved in IoT platforms cannot support an advanced access control measures. In this case, the IoT software platform itself should implement alternative measures to solve device level security issues. For example, separation of IoT traffic into private

networks, strong information security techniques at the cloud application level, requiring regular password updates, supporting software updates only by the way of authentication, and so on increase the security level of the IoT platform.

- **Data gathering protocols:** Another important issue which needs attention is the types of protocols used for data communication between the components of an IoT software platform. An IoT software platform may need to manage millions or even billions of objects. Lightweight communication and data gathering protocols should be used to ensure low energy consumption and low network bandwidth functionality.
- **Data Analysis:** The data collected from the objects of IoT must be analyzed in an intelligent way to obtain meaningful information. There are four data analysis methods that can be used within the framework of IoT: real time, batch analysis, predictive and interactive analysis [87]. Real-time data analysis performs on-the-fly analysis on the streaming data. An example operation includes the real-time streaming on the cloud applications. Batch analysis runs operations on an accumulated data set. Thus, operations take place at scheduled time intervals and may take several hours or days. Predictive analysis focuses on making estimations based on various statistical techniques and machine learning methods. Finally, interactive analysis performs multiple exploration analysis on both streaming data and accumulated data set.
- **Visualization:** The proper viewing of the information is another important requirement that is expected from the IoT software platform. Visualization is carried out together with the data analysis. Thus, the IoT software platform must provide the required tools and APIs to visualize the data gathered from the objects as well as the information obtained after the data analysis.

Table 2.4 summarizes the features of some IoT software platforms.

Table 2.4. Characteristics of some software platforms used in IoT framework

IoT Software Platform	Device Management	Integration Support	Security	Data Gathering Protocols	Data Analysis	Visualization
Appcelerator	×	REST API	SSL, IPsec, AES-256	MQTT, HTTP	Real Time	✓
AWS IoT	✓	REST API	TLS, SigV4, X.509	MQTT, HTTP _{1.1}	Real Time	✓
Ericsson Framework	✓	REST API	SSL/TSL, Authentication (SIM based)	CoAP	Real Time, Predictive Analysis	×
IBM IoT Foundation Device Cloud	✓	REST and Real Time APIs	TLS, IBM Cloud SSO, LDAP	MQTT, HTTPS	Real Time	✓
EVERYTHING - IoT	×	REST API	SSL	MQTT, CoAP, WebSockets	Real Time	✓
ThingWorx	✓	REST API	ISO 27001, LDAP	MQTT, AMQP, XMPP, CoAP, DDS, WebSockets	Real Time, Predictive Analysis	✓
2lemetry - IoT	✓	Salesforce, Heroku, ThingWorx APIs	SSL, ISO 27001, SAS70 Type II audit	MQTT, CoAP, STOMP, M3DA	Real Time	×

2.8. Applications of IoT

The Applications that are developed within the framework of IoT can be classified according to types of network availability, coverage, scalability, heterogeneity, repeatability, and user participation and impact [4, 88]. IoT can be seen as an important technology that improves the flow of information between the organizational structures of these applications. Figure 2.5 shows the application areas of IoT and some scenarios related to them.

IoT have applications in a wide variety of fields, including agriculture, manufacturing industry, transportation, healthcare services, environment monitoring, smart buildings, and many other areas [89]. Next, some areas of IoT application are presented:

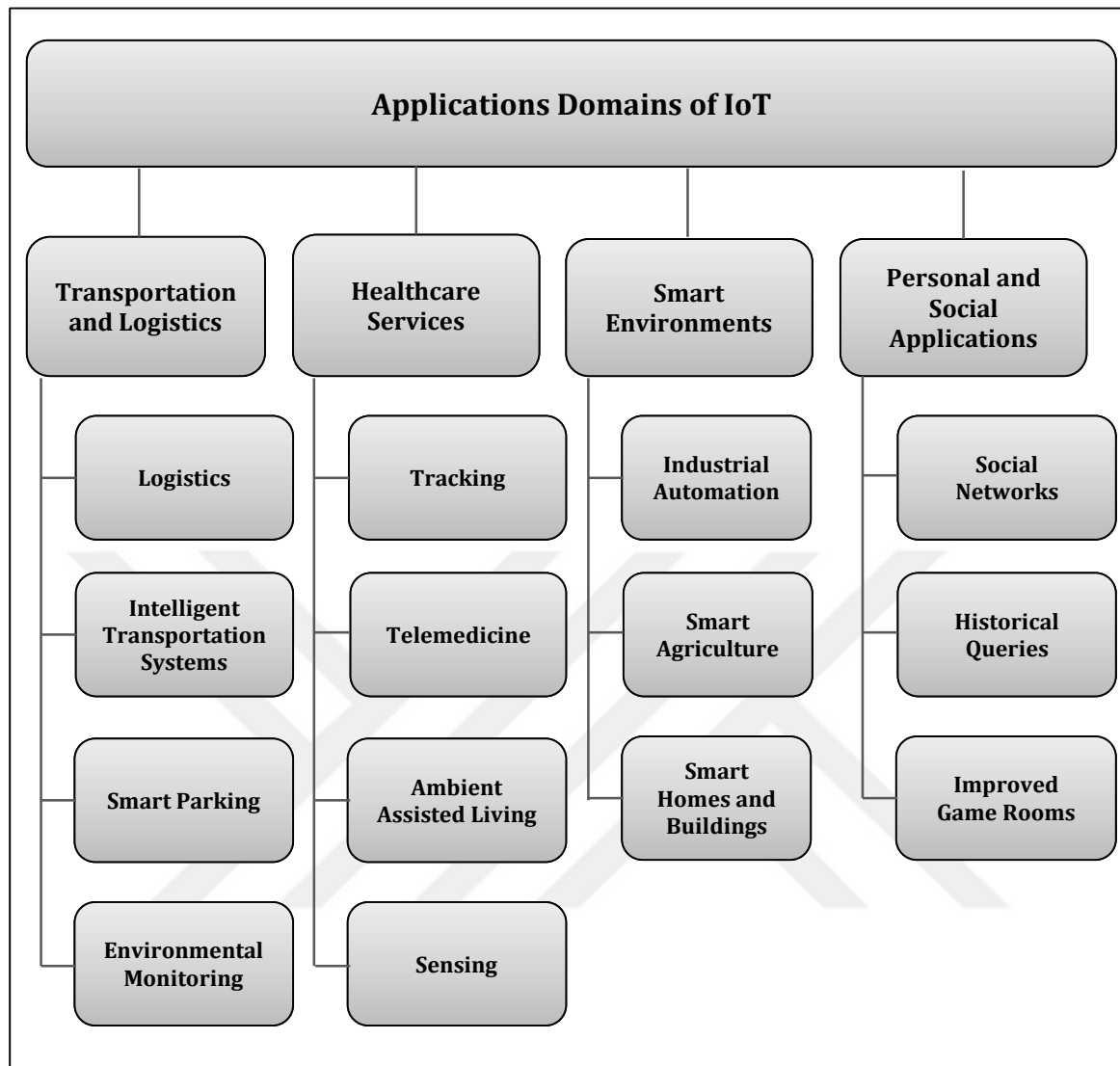


Figure 2.5. Applications domains of IoT and relevant major scenarios

- **Transportation:** IoT can be applied to produce intelligent transportation Systems. In intelligent transportation systems public and private transportations can interact, and choose the best paths to avoid delays and congestions. Another example of using IoT in transportation is smart parking. Finding a parking space in a busy city center can be time consuming and leads to increased traffic congestion. Installing a sensor that detects if there is a vehicle in each space can be used to provide drivers with information on whether there are empty parking spaces at any time.
- **Environment monitoring:** IoT technology can be successfully applied to environmental monitoring applications. In this case the most important role depends on the ability of distributed and self-managed sensors to detect natural phenomena (e.g., temperature,

wind, rainfall, river height etc.). Real-time information processing, coupled with the ability of a large number of devices to communicate among them, can provide a solid platform to detect and monitor abnormal events that may endanger human and animal life [90].

- **Healthcare Services:** IoT plays a crucial role in healthcare services. It can be used in many ways such as tracking the number of patients in a hospital, identifying the right patient for the right medicine and monitoring a patient's health conditions from a remote place which is known as Telemedicine [91]. Telemedicine includes remotely providing treatment, diagnosis and treatment. Ambient assisted living provides technical systems for elderly people who are alone at home and need to be monitored. The patient's health status is periodically sensed using RFID and sensors. The doctor from a remote location provides medical assistance based on the information received.
- **Industrial Manufacture:** In manufacture IoT can be used in industrial automation. Industrial automation ensures that goods being mass produced with lower costs and consistent quality. Moreover, IoT can affect factory operations in many different forms: from identifying of risky conditions to efficient logistics management [92], from task scheduling to machine status monitoring [93]. In such scenarios, sensors are ubiquitously placed along a networked production line to monitor operations. Data collected by the network is sent to a computer or a server to be processed. On the basis of decisions made after the processing, actuators can be triggered. In such a complex scenario, some messages such as scheduling of urgent tasks or warning messages of risky situations need more priority than others.
- **Agriculture:** In the field of intelligent agriculture, IoT can be widely used [94, 95]. In smart agriculture systems, IoT can be used to increase agricultural production and reducing environmental pollution caused by abusing agricultural chemicals. Moreover, IoT can be helpful in monitoring growth of plants. Plants are fitted with RFID tags and sensors. When there is a drastic or unexpected change in the growth of plant due to temperature / humidity, the sensors sense this and the RFID tags send information to the reader and are shared across the internet. Then, the farmer or scientist can access this information from a remote place and take necessary actions.
- **Smart homes and buildings:** Equipping buildings with advanced IoT technologies may help in both reducing the consumption of resources associated to buildings (such as electricity, water) as well as in improving the satisfaction level of people living in the

building. The effect is large both in economic terms (reduced operational expenditures) as well as environmental cleaning terms (reducing the carbon emissions associated to buildings). In this application, a key role is played by sensors, which are used to both monitor resource consumptions as well as to predict and identify the needs of current users [1].

- **Security and surveillance:** Security surveillance has become a necessity for enterprise buildings, shopping centers, factories, car parks and many other public places. IoT technologies can be used to greatly enhance the performance of existing solutions in this area [1]. IoT can provide cheaper and less invasive alternatives to the widespread deployment of cameras while at the same time preserving users' privacy. Ambient sensors can be used to detect the presence of dangerous chemicals. Sensors monitoring the behavior of people may be used to assess the presence of people acting in a suspicious way. Efficient early warning systems can therefore be built. Personal identification by means of RFID or similar technologies is also an option.
- **Personal and social applications:** The applications of this area are designed with the goal of the establishing and developing of social relations [2]. These applications are also used for creating social networks, tracking and questioning past behaviors and transactions, and prevention of losses and thefts. Sharing the current location and activity information, determining the location of lost items that attached with RFID tags and similar functions can also be achieved by this technology.

2.9. Challenges of IoT

The main concept of IoT is the extensive presence of interconnected objects. In IoT anything, conveniently tagged, may be able to communicate with other objects equally tagged through internet or any other protocols, to collaborate and to reach a common goal. Although many IoT based systems have been developed, there are many design challenges encountered by the researchers and developers. Given foreword the complexity of the IoT concept and the design challenges for its enabling technologies, a number of obstacles need to be overcome for IoT to achieve its vision. Bellow the main challenges for IoT are presented [96]:

- **Constrained resources:** The things composing the IoT are often resource constrained. Available resources on object such as electrical energy, memory, processing, and object

capability to perform a demanded task, are often limited. IoT objects need energy for sensing, communicating and processing information. If objects isolated from the electric grid they must rely on batteries (most of the object are battery powered). Replacement of these batteries can be impossible or a problem, even if energy consumption is highly efficient. That is especially the case for applications using large numbers of objects or they are placed at locations that are difficult to access. This is the case, for example, of wireless sensor nodes, which are often battery powered, and therefore have limited energy amounts. Another example is represented by the scarce processing capabilities of RFID tags. To maintain the constrained computational capacity in a most efficient manner, it needs to possess only computational capabilities for the task it has to perform.

- **Heterogeneity:** IoT concept relies on the implementation of network systems of cooperative intelligent objects with key interoperability capabilities. The devices taking part in the system are expected to have very different characteristics and capabilities in terms of computational and communication functionalities. In fact, one of the most important factors that make the IoT concept work efficiently and accurately is the integration of several technologies and communications solutions. However, such a high level of heterogeneity brings efficient management challenges at both the architectural and protocol levels.
- **Scalability:** The number of objects that are connected in an IoT system is much higher than the number of computers connected in the conventional Internet (may be in several times) [97]. As IoT concept implies that every object that is appropriately tagged can be a part of the IoT system, scalability issues arise at different levels, including:
 - Naming and addressing.
 - Data communication and networking.
 - Information and knowledge management.
 - Service provisioning and management.

There are two approaches to deal with these challenges:

- i) Reducing the number of messages and the amount of data transmitted throughout all the layers of the system [98].
- ii) Choosing a small cluster of objects that are able to execute a demanded task [99].

- **Identification:** Identifying of objects is one of the primary issues in IoT. In order to be able to address the billions of objects in the IoT, the system first need to be able to identify them with a unique ID. Currently, in the conventional Internet the IPv4 protocol identifies each node using a 4-byte address [100]. However, it is well known that the number of available IPv4 addresses is decreasing rapidly and will become insufficient in the very near future [101]. To this end, a much better choice is using IPv6 protocol which with its 128-bit addresses provides more suitable solution to address a larger number of devices [102]. However, more efficient and appropriate solutions need to be developed to meet the specific requirements of IoT environment such as bandwidth efficiency, energy-efficient and capabilities of working with limited hardware resources.
- **Searching and discovering:** In IoT, like in all distributed networks, a group of nodes want to cooperate to perform a given task. In order to perform this cooperation objects need to learn of the existence of each other. Therefore, searching and discovering services are fundamental of any distributed computing system [103]. Usually, in IoT context the exact location of the other objects and form of stored data are initially unknown to the requester. So an intermediate block can be used to extract this information.
- **Mobility:** A large part of IoT objects are not stationary, but have a certain degree of mobility. In IoT many services are expected to be established by mobile users or objects, therefore, mobility is taken place among the other challenges of IoT [3]. An example of this situation is a person who owns various devices and he is mobile during the daily activity of his life. This person may make data queries or request to activate other things, which will be probably also in movement. It's clear that mobility is not a negligible situation in the context of IoT since the fundamental concept of IoT is the ability of accessing and managing objects independently from where they are located. Therefore, IoT management and resource mobility schemes should be developed accordingly.
- **Security and privacy:** Due to the tight entanglement with the physical environment, security is critical to the widespread use of IoT applications. IoT systems should be designed with security and privacy-preserving in consideration. In this respect, three main challenges require innovative approaches and should be subject of researches. These challenges are data confidentiality, privacy and secrecy. Data confidentiality refers to protecting data from being accessed by unauthorized parties. In other words,

only the persons who are authorized should gain access to data and based on their authorization level only these persons should have the ability to modify the data. Privacy refers to the rules by which a person or group owning information control with whom to share information and under what conditions. In terms of Secrecy, it means that only the sender and intended receiver should be able to understand the contents of the transmitted message. Security should be considered a key system-level property, and be taken into account in the design of architectures and methods for IoT solutions. Solutions for security problems of IoT represent a key importance for acceptance of this technology by users. In order to make the IoT based objects and applications more reliable and attack resistant, the following issues need to be addressed:

- Data confidentiality: Data transmission, retrieval and processing are an integral part of any IoT application. Most of these data are personal data or information thus; such a sensitive data is needed to be protected using an encryption mechanism. Wherever the data is online secure Socket Layer Protocols (SSL) can be used to ensure that data is accessible only by authorized parties. However, the data must also be protected within the wireless protocol. These sensitive data is expected to be confidential and encrypted while being transmitted wirelessly. All of the suggested solutions for data privacy are based on encryption techniques. However, the traditional encryption techniques are inefficient in terms of resources consuming (consume a large amount of energy and bandwidth from both the sender and the receiver). Therefore these techniques cannot be applied to the resources limited IoT objects. There is requirement to develop new solutions to be implemented in the framework of IoT regardless of resource constraints.
- Authentication: Authenticating of data being received (or transmitted to other objects) in IoT applications is challenging and open research filed. In traditional networks, authentication can be applied using several techniques. Examples of these methods are password, pre-shared key, and public key cryptosystems. However, the heterogeneity and complexity of objects and networks in IoT applications make these methods unsuitable for IoT systems. Furthermore, the rapidly increasing number of objects can make key management difficult or in some cases an impossible procedure. For example, although public-key cryptosystems have advantage for constructing authentication schemes or authorization systems, the lack of a global root Certificate Authority (global root CA) hinders many theoretically feasible

schemes from actually being deployed. Without the global root CA, it becomes very challenging to design an authentication system for IoT. Furthermore, it may be infeasible to issue a certificate to each object in IoT since the total number of objects is often huge [104]. Therefore, the authentication methods for IoT should take into account the different design features and characteristics of IoT applications.

- **Device management:** Management of objects is of paramount importance for the development of the IoT applications. Objects should not only be able to connect and communicate over a wide number of communication technologies, but also these objects should be remote or self-managed. The device management process is complex, and includes a lot of different actions, such as switching on/off the device, configuring the device/network, updating firmware/software, recovering from errors, monitoring the device/network and gathering data and connectivity statistics [105]. Solutions for an efficient device management should take into account the heterogeneity of objects and the limitations of their existing resources. Furthermore, a large amount of efforts is required to distribute and configure objects in many IoT applications; this is the case for example in WSNs. This problem is an obstacle to the adaptation and long-term sustainability of large-scale applications. One of the most important solutions to increase the applicability of large-scale IoT applications is to use configurable middleware. This middleware layer should provide intelligent device management that supports self-configuration, self-optimization and self-improvement and maintenance. For example, gateway devices should be able to detect objects, configure detected objects, identify faulty nodes, and make decisions to eliminate errors [106]. In this context, many of the standardized device management solutions such as TR-069 [107], SNMP [108] and NETCONF [109] are not suitable for the management of resource restricted objects. Such solutions are generally used for the management of resource-rich devices such as routers, switches, and smartphones. Devices management solutions for IoT applications should take into account the design characteristics of IoT, such as scalability and limited resources.

In addition to the previously mentioned challenges, other difficulties encountered in IoT applications can be listed as follows:

- ✓ Availability of internet at everywhere and at no or low cost.
- ✓ Low-cost smart sensing system development.

- ✓ Fault Tolerance.
- ✓ QoS management.
- ✓ Acceptability among the society.

2.10. Task Allocation in IoT

Task allocation is a procedure of choosing, subdividing, coordinating and assigning the correct tasks to the correct entities [110]. In IoT task allocation with the goal of satisfying the specific design features of IoT applications as well as identifying which tasks to be assigned to which objects is not a trivial process. Given the IoT paradigm and the requirements of IoT applications, the objects involved in the execution of the same application should be cooperates and coordinated to reach the optimal allocation of tasks among them. The objects should execute tasks to reach the global application target and to satisfy the relevant requirements while optimizing the network performance in terms of resources used. This issue should be continuously addressed to dynamically adapt the system to changes in terms of application requirements and network topology.

2.10.1. Major challenges in task allocation in IoT

IoT poses various challenges to research community. This section briefly summarizes some of the major challenges faced while solving task allocation problem in IoT:

- Network deployment: Careful management of the network, where objects are randomly deployed in uniform or non-uniform distribution, is necessary in order to ensure entire area coverage and also to ensure that the energy consumption is also uniform across the network.
- Data aggregation: The network resources optimization is not only focused on the reduction of messages transmission power, but also includes convenient data processing that reduces the amount of data that are transmitted over the data sinks. This is the principle behind node clustering protocols, such as LEACH [111], in which cluster head nodes aggregate data and reduce transmitted data volume, which in turn reduces the overall transmission energy consumption of the network.
- Constrained resources: Most of IoT objects have limited amount of energy, memory, processing, and object capability to perform a given task. This is the case, for example,

in battery powered wireless sensors. Thus, for most of IoT systems, reducing the energy consumption (e.g., for communication or performing a given task) is a primary constraint. One way to minimize the effect of this resource constraint is enabling network entities to cooperate and make their resources available to perform the given tasks. In other words, the entities executing the same application should cooperate to reach the optimal allocation of tasks among themselves. However, task allocation problem with the goal of increasing the lifetime of the network as well as identifying which tasks to be assigned to which objects is not a trivial task.

- **Uniform energy consumption:** Task allocation schemes should ensure that energy dissipation across the IoT network should be balanced and the task allocator should be rotated in order to balance the network energy consumption.
- **Heterogeneous network:** Heterogeneity which regards both object capabilities and characteristic parameters makes resource allocation operation a challenging task. Heterogeneity not only entails heterogeneity among devices, but also heterogeneity among roles that the same device can assume. For example, a temperature sensing node could be used both to periodically send sensed data to a server for monitoring purposes.
- **Integrity:** The key issue is that for connecting and integrating all the objects into the IoT, there are many different technologies and protocol which introduce fragmentation in a scenario that should be rich of interoperability. IoT interoperability involves not only the ability of objects to exchange information but also includes the capability for interaction and joint execution of common tasks [112].
- **Scalability:** Given the size of a distributed heterogeneous system such as the IoT network, the optimal creation of communities and the task allocation within are not trivial issues. Furthermore, when an IoT network is deployed, in some cases, new nodes need to be added to the network in order to cover more area, prolong the life time of the current network or increase the accuracy of some tasks by achieving a certain level of redundancy. In all these cases the task allocation scheme should be able to adapt to changes in the topology of the network.
- **Data redundancy:** Objects may provide information that is not unique but can be generated by set of different nodes which for example are capable to sense the same physical measure of the same geographical. When the task allocation scheme assign tasks to a set of objects the level of data replication should be considered.

- **Dynamic nature:** Typical IoT networks are characterized by the dynamic behavior of their objects. In fact, emerging applications in smart environments such as smart cities and smart homes, where IoT is preponderant, are often based on opportunistic networks. In opportunistic networks, connections among nodes are created dynamically in an infrastructure-less way [113]. In such a dynamic context, with frequent and quick changes of scenario, task allocation approach should be adopted to such dynamic environment when assigning tasks or forwarding a messages.
- **Topology change:** IoT is strictly related to ubiquitous networking, which is characterized by a huge number of nodes deployed over an extensive area. The network is not only made of static or semi-static devices as it is in traditional networks, but topology changes quickly, so that it is impossible for objects to be able to know the whole network topology. As a consequence, challenges arise with respect to autonomous reconfiguration and interoperation of nodes when tasks are being allocated. For instance, it may happen that there are no nodes available to perform a given task at the desired geographical location and at a given time.

All these challenges portrait a very complex and dynamic network, where all objects need to collaborate in order to reason and allocate available resources among themselves with the aim of executing the applications assigned to the network.

2.10.2. Literature review

Task allocation is usually performed considering different aspects such as network topology, energy constraints and processing capabilities of the network entities. However, most of the existing methods have generic and limited scope in extending resources regarding the application assigned to the network [8]. Moreover, although the problem of task allocation is extensively studied in the field of WSNs, the task allocation problem in IoT networks is an open research issue.

In [114] the authors proposed an energy-balanced allocation protocol for real time WSN applications. They formulated the problem using both Integer Linear Programming (ILP) and a polynomial time 3-phase heuristic. In their model, they assumed that each node is equipped with discrete dynamic voltage scaling. Although their protocols succeed in

improving the lifetime of the network, they considered only homogeneous networks, which are not common in real life scenarios.

The same energy problem is studied in [115]. The authors provided an adaptive task allocation algorithm that aims at reducing the overall energy consumption by achieving a fair energy balance among the sensor nodes in the overall network. The proposed algorithm achieved a good level of adaptation to environmental changes and uncertain network conditions by using centralized and distributed message exchanged mechanisms between the nodes and task allocator. However, these mechanisms introduced considerable increments in packet overheads.

The authors of [116] developed a centralized solution for task allocation in WSNs. In this work, the application assigned to the network is divided into a sequence of distributed tasks to be assigned to each sensor device. Then, the energy consumed to perform each task is then considered to compute a cost function allowing the evaluation of each deployment solution. In this method the application assigned to the network can be performed in different approaches: gathered data can be immediately sent to a sink or it can be processed before being transmitted. In the second approach the size of data to be sent would be smaller, and therefore the transmission energy consumption would be lower than the second approach. However, this is achieved at the expense of consuming more energy for processing.

Another centralized algorithm is proposed in [117]. The algorithm is located at a gateway which is supposed to be equipped with sufficient power supplies. The algorithm adopted an adaptive intelligent task mapping scheme coupled with a scheduling mechanism based on a genetic algorithm to provide real-time guarantees. Finally, in order to extend the network lifetime a hybrid fitness function is embedded in the algorithm to ensure workload balancing among the collaborative nodes.

In [118], in an effort to reduce the overall energy consumption of the sensor network system while meeting the deadline requirement, the authors developed an energy-efficient tasks scheduling scheme that makes a trade-off between energy saving and QoS-guaranteeing. The proposed method aims to minimize the execution energy while allocating a set of real-time tasks with dependencies onto a heterogeneous sensor network.

Furthermore, in order to find an optimal allocation that meets user's deadline, the method adopts the divide-and-conquer. The method portions the tasks into tasks partitions and then optimally solves the scheduling problem in branches with several sequential tasks by modeling the branches as a Markov Decision Process. Then, sensors failure can be handled by rescheduling part of the tasks graph.

EBSEL, the Energy-Balancing task Scheduling and allocation heuristic whose main purpose is to Extend the network's Lifetime, through energy balancing is proposed in [119]. EBSEL has four phases in which tasks are combined into groups in order to minimize the communication energy but preserve possible parallelism. These phases are task grouping phase, node selection phase, threshold calculation phase, and task allocation phase. The goal of task grouping phase is to form groups of tasks in order to reduce the communication cost, without sacrificing parallelism. In node selection phase a number of sensor nodes, out of the total nodes is selected to be allocated the tasks. Threshold calculation phase attempts to extend the lifetime of the nodes. It sets an approximate energy threshold for achieving energy balancing, and tries to avoid severe energy depletion of the selected nodes. While allocating the groups, the nodes will not be depleted beyond this threshold. Finally, task allocation phase maps the task groups to the nodes while maintaining energy-balancing. It accomplish the energy balancing by allocating the task group with the highest computational energy to the node with the highest remaining energy

Recently, in [119] the authors suggested Logic Gate-based Evolutionary Algorithm (LGEA) to solve the problem of task allocation in WSNs. The problem is formulated as a binary multi-objective optimization problem with the goal of minimizing the number of active nodes and the computation and communication load distribution. The algorithm introduces an original logic gate mechanism as perturbation operator to search for the best task allocation scheme and satisfied the task workload and connectivity by considering them as the constraints of the problem.

To tackle the problem of dynamism of a distributed network, the authors of [121] developed a decentralized lifetime maximization algorithm that determines the distribution of tasks among the nodes in the network. The algorithm is formulated as distributed optimization algorithm based on a gossip communication scheme with the goal of

extending network lifetime. The algorithm uses iterative and asynchronous local optimizations of the task allocations among neighboring nodes.

Similar approaches are studied in [122] and [123]. In [122], a particle swarm optimization method based the distributed algorithm is proposed. The major drawback of these works is that they do not take into account the deadline of the applications assigned to the network. In [123], an adaptive decentralized task allocation algorithm for heterogeneous WSNs is proposed. This algorithm is based on non-cooperative game theory and focuses on reducing the overall energy consumption and task execution time.

In [124], a middleware algorithm, named SACHSEN, for resource allocation in heterogeneous WSNs that run multiple applications is presented. SACHSEN starts with a locality-aware service discovery and extracts important information in order to generate primary task-sensor assignment. Then, each application is decomposed into multiple tasks by applying a divide-and-conquer strategy and recursive decomposition is used in each category to divide a generic service into more specific services until all the decomposed services cannot be further divided. Finally, SACHSEN exploits resource heterogeneity to make effective task-sensor assignments explicitly taking performance requirements of application into account.

All the aforementioned works focus only on task allocation in WSNs. However, the focus of this work is different as it considers more complex IoT scenarios. IoT introduces much more heterogeneous scenarios than WSNs. In WSNs, the nodes are managed by the same system and have similar characteristics whereas the IoT objects are grouped opportunistically as they provide cooperative services. Therefore, IoT objects must find a way to coordinate the given job [9].

In [125], the authors dealt with the problem of task allocation in IoT from a different perspective. They assumed that IoT resources are not able to interact with each other directly to perform a given task. Instead, the interaction between resources can be done via service gateways which are deployed in an IoT environment along with the IoT resources. Then, they transformed the resource allocation problem into a variant of the degree constrained minimum spanning tree problem. In order to solve this problem the authors

proposed a service resource allocation approach which minimizes data transmissions between the devices in the network.

In [8] and [12] the authors developed a heuristic approach namely Consensus Based Approach for Task Allocation (CBATA). This approach has adopted the concept of task group and virtual objects to formulate the problem of task allocation in IoT with the main goal of fault tolerance. This method and the concept of task groups and virtual object will be explained in the next sections within this chapter.

Other studies for task allocation in IoT could be found in [126] and [127]. In these works, the aim is to find the available objects that can perform the needed task to enable service execution. However, none of these works focus on finding the best configuration for optimal resource allocation.

2.11. The Concept of Task Groups and Virtual Objects (The Clustering)

The concept of virtual objects is presented in an attempt to provide conceptualizations for the IoT domain. Several works such as, [128-130] missioned the notation of virtual object. However, all these works consider the virtual objects simply as the digital parallels of the physical objects and focus more on the middleware framework rather than the modeling of related information. As a result, the absence of a common format for virtual objects causes problems of interaction and communication, since there is no standardized ways to obtain the actions or services associated with a virtual object [112].

In the last years, several researchers were focused on proposing an architecture for the IoT. These works led to evolution of the definition of virtual object, and of its functionalities. Virtual objects are not anymore only digital interfaces to the real world but now provide a semantic enrichment of the data acquired, which makes easier the discovery of services [112]. For example, the CONVERGENCE project [131] used the Versatile Digital Item (VDI) as a common container for all kinds of digital contents that include one or more resources and metadata. This definition is similar to the one provided in [132] with a many-to-one association between real objects and VDI. However, the CONVERGENCE project provides a first attempt to implement the discovery of a particular VDI in the virtualization layer.

Another example is SENSEI [133]. SENSEI enables the integration of heterogeneous and distributed Sensor and Actuator Networks (SAN) into a homogenous framework for real world information and interactions. It provides an abstraction level of resources corresponding to the real world consisting of Entities of Interest (EoI). Resources may be associated with one or more EoIs for which they can either provide information or provide control over their surrounding environment, thus providing the same type of association of CONVERGENCE. In SENSEI, resources acquire the ability to enhance the data received by the sensors with environmental information.

Since then, several interesting definitions of virtual object are presented in literature. In [134], physical entities are represented in the digital world via virtual entities, which have two fundamental properties. Firstly, while ideally there is only one physical entity for each virtual entity, it is possible that the same physical entity can be associated to several virtual entities. Secondly, virtual entities are a synchronized representation of a given set of aspects of the physical entity. The association between virtual and physical entity is achieved by connecting one or more Information and Communications Technology (ICT) devices to the physical entity so as to provide the technological interface for interacting with the virtual world. The physical object is decomposed in its functionalities thus providing a one-to-many correspondence with the virtual entities. In COMPOSE [135], the focus was on objects service composition and for this reason they need to abstract the heterogeneity of physical objects in terms of computing power, protocols and communication mechanisms, by introducing the concept of Service Object. The Service Object then represents a standard internal digital representation that makes easier the creation of COMPOSE services and applications. Finally, in [136], a virtual object is a virtual representation of an ICT object that may be associated to one (or more) real-world objects. The term real-world object refers to any object that exists in the real/physical world and then can be classified both as ICT objects, such as, an email or a smartphone, and a non-ICT object, such as, a person or a fruit; an important trait of this project is that also a real-world object can be associated to one or more virtual objects. The virtualization layer, where all the virtual objects are located, acts as a management level that manages and provides interfaces for accessing virtual object to other components.

This thesis adopts the concept of virtuality coupled with the concept of task groups to formulate the problem of task allocation in IoT based applications. It frequently happens in most IoT applications that some objects perform the same task, such as measuring the humidity and/or the temperature, monitoring traffic congestions, tracking the movement of objects or persons, detecting the risky conditions in public facilities, and so on. However, not all objects have usually the same amount of resources to be dedicated to the same tasks and the set of objects that can cooperate in performing a given operations changes quickly as opportunistic behaviors make the scenario quite dynamic. Accordingly, a group of objects that are performing similar and replaceable tasks can be defined as a task group. It is obvious that in most IoT applications there may be several task groups and most of them intersect with each other. This can be attribute to the existence of different objects with diverse skills belong to several task groups. From each task group some objects (or an object) are selected to represent the task group. These objects are called Virtual Objects (VOs) [137]. VOs virtualize the physical objects connecting to them in the cyberspace and they are in charge of processing the requests to these physical objects. In other words, the IoT objects are partitioned into task groups each of them contains the objects performing the same task. Then, in each task group some objects are selected as VOs and the rest of the task group objects are clustered and connected to one virtual object. At this point, allocating proper resources to the required task is the duty of VOs. Given an example to illustrate the concept of task groups and virtual objects, assume a network that performs two tasks T1 and T2 (See Figure 2.6). Then, the network is partitioned into two task groups: TG1 and TG2. TG1 contains all the objects that capable of performing T1 whereas TG2 contains the T2 objects. TG1 and TG2 could intersect when some objects are capable of performing T1 and T2. As a result these objects belong to both task groups at the same time. Then, from each task group, some objects are selected as VOs to visualize the task group in the cyberspace and to assign the relevant task to the task group.

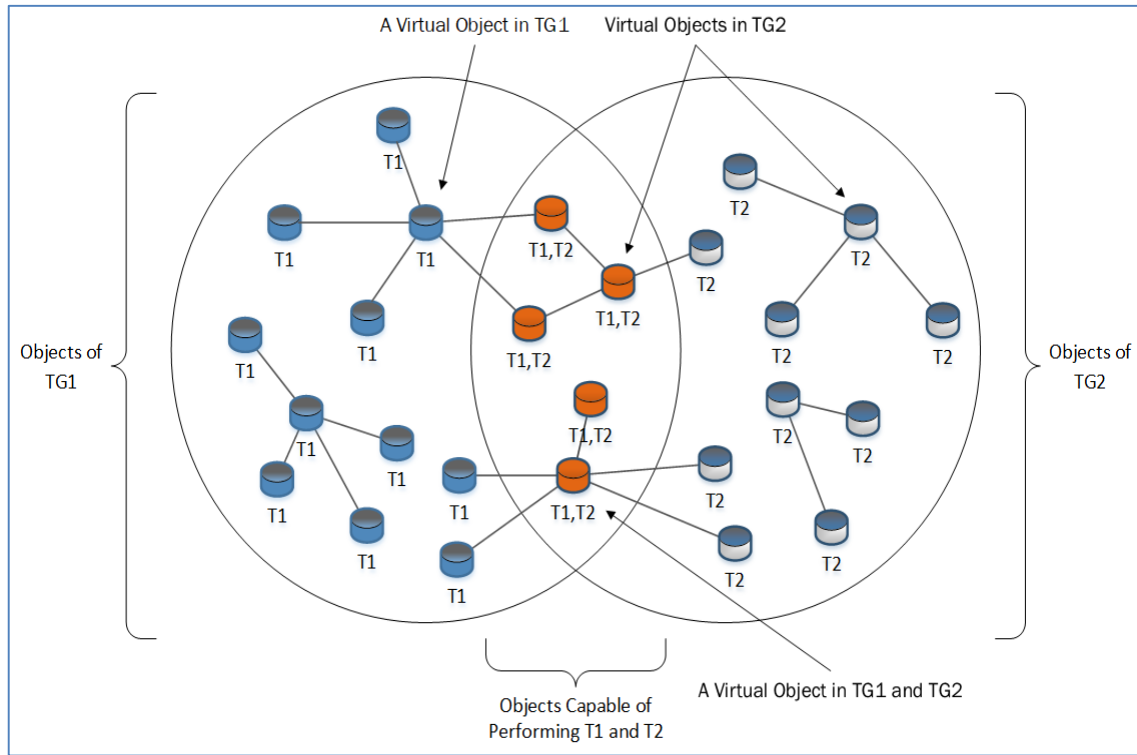


Figure 2.6. The concept of task groups and virtual objects

2.12. Consensus Based Approach for Task Allocation (CBATA)

One of the famous and attractive task allocation protocols in IoT is Consensus Based Approach for Task Allocation (CBATA) [8, 12]. CBATA is a heuristic method which designed to solve the problem of resource allocation and management in IoT heterogeneous networks with the main goal of fault tolerance. The algorithm uses the concept of task groups and virtual objects to formulate the problem. In this method a group of nodes that are capable of performing the same tasks are organized into the same task group. Given an example, suppose that the network is performing a temperature sensing in a specific area: only those nodes that are equipped with a temperature sensor and that are deployed within that area are included in the task group related to this task. These task groups are assigned with the relevant task by the Application Deployment Server (ADS), which could decide which exact node should perform each needed task. In CBATA the central server leaves the task groups to autonomously decide how to distribute the burden of tasks among them without the need for the central server to keep the role of single physical node controller. Accordingly, the IoT is made of VOs. These VOs are activated by the ADS. The VO role may be implemented by an object in the task group and is in

charge of processing the requests generated by the central server and forwarding configuration messages to the other physical nodes (note that the VO may coincide with the only single physical node that is capable of implementing the required task). At this point, allocating the proper resources to the required task is a duty of the objects in the task group. Figure 2.7 provides a sketch of the above described reference scenario. The central server, or a leader node, transmits the activation signal to the VO. Since the VO is responsible for keeping track of the physical nodes that belong to the same task group it leads, it knows which nodes the activation signal is addressed to. Therefore, it is able to forward the activation signal to the appropriate nodes, on the basis of their belonging to a determined task group.

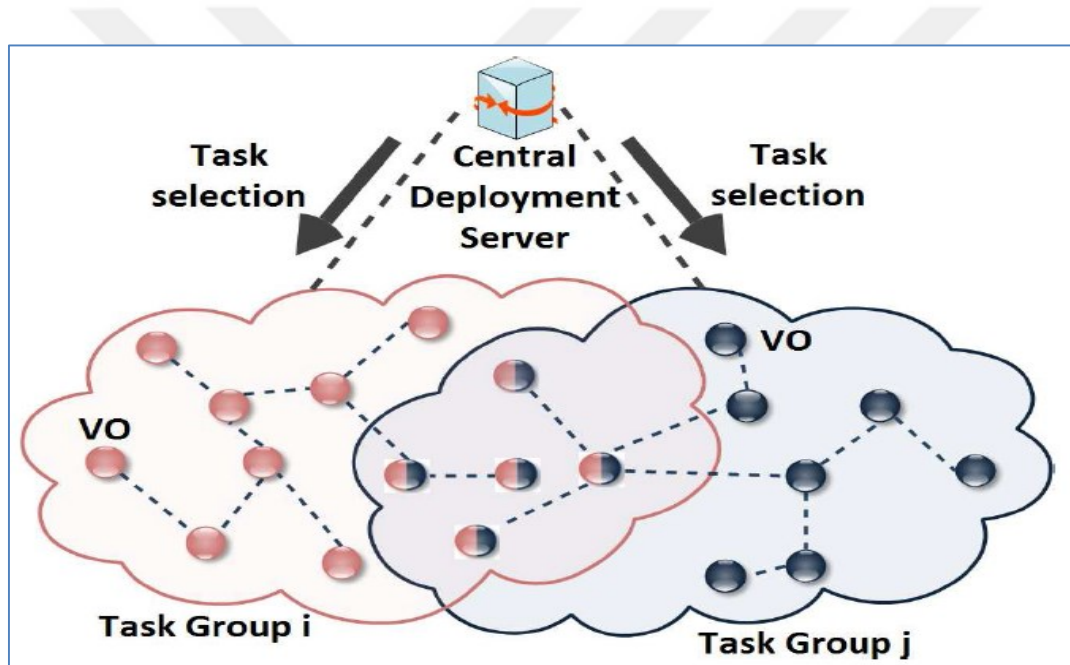


Figure 2.7. A reference scenario for CBATA

2.12.1. Virtual objects selection

The scenario proposed in CBATA is that of an opportunistic IoT, where nodes continuously join and leave the network. CBATA adopts the concept of virtual objects by choosing the first node in each task group as a VO and the next node in the list is selected as vice-virtual object (vice-VO). Then, the role of task allocator is carried out by the VOs. When a VO depletes its energy the vice-VO becomes a virtual object within the corresponding task group and the next node in the list becomes a vice-VO. To accomplish this procedure VOs and vice-VO periodically exchange Hello and Acknowledgment

messages. Although, this method provides robustness against links and nodes failures, it is obvious that it requires a heavy message exchange procedure which is very intensive with regard to communication, computation, storage, and energy overheads.

2.12.2. Consensus-based negotiation and task allocation

In order for nodes to start a negotiation, they need to have already joined the related task group. As soon as a node i joins the network, it broadcasts to its one-hop neighbors the information related to the tasks that it is able to perform. Accordingly, the VOs related to those specific tasks add node i to the list of nodes that belong to their task groups, and reply with an acknowledgement. If node i is the second node in that list, they designate it as the vice-VO, and the acknowledgement contains this information. If no VO is associated to one or more tasks yet, node i is designated as VO for those tasks. Then, node i notifies the ADS its designation as VO and/or vice-VO.

VOs periodically send Hello messages to their related vice-VOs. One of the following things might happen:

- The VO sends the Hello message and the vice-VO acknowledges the message. No further actions are performed.
- The VO sends the Hello message and the vice-VO does not acknowledge the message. In this case the VO assumes that the vice-VO is not reachable. If present, the VO designates the second node on its list as vice-VO and informs it, which in turns informs the ADS.
- The VO does not send the Hello message when it is supposed to do it. The vice-VO notices that the VO is not reachable. It broadcasts a request to know which nodes belong to its task group. The first one to reply is designated as vice-VO. Information about the failed VO and the new vice-VO is delivered to the ADS.

In order to avoid communication overhead, when a VO notices that it is about to leave the task group (e.g., for depletion of residual energy, or because it is moving) it notifies it to its vice-VO, sending the list of the nodes belonging to the task group. Then, the vice-VO becomes the VO, and the next node on the list becomes the vice-VO. Relevant information is delivered to the ADS.

When an application requires the execution of a given task, the ADS sends an activation signal to the appropriate VOs, which forward it to their list of nodes. Then, the negotiation algorithm is started. The negotiation algorithm includes a collection of laws that regulates the interaction and the exchange of information between nodes in a group which needs to reach a coordination to achieve common goals. If the ADS does not succeed in reaching a VO, it tries to contact its vice-VO. If even the vice-VO does not reply, the ADS assumes that no nodes are available anymore for the related task, and thus it cannot be activated. Once the VO has sent the activation signal to all the nodes in the group, then they start the consensus algorithm. What happens is that each node sends to all its neighbors (one-hop broadcast) the consensus messages for the reference task group, containing data that specifies the update values for the algorithm. Then they check whether the task group to which this message is related is of interest to them, i.e., they belong to this task group. If yes, they process the data further and then exploit this information in the next iteration of the algorithm.

2.13. Problem complexity

In its abstract level, we can consider the interplay of task groups and virtual objects as a special clustering problem as each task group is clustered based on the selection of virtual objects. The problem of clustering a set of n points into k clusters under some objective functions is known to be an NP-hard problem even when the points to be grouped are restricted to lie in two dimensional Euclidean space [138].

In context of IoT, clustering of each task group for keeping the total distance to a minimum, or in other words, the optimal selection of the virtual objects with high residual energy that scattered in the area can be seen as NP-hard problem. Consider a 100-object of each task group example, to perform an exhausted search of all possible solutions require [138]:

$$(C_{100}^1 + C_{100}^2 \cdots + C_{100}^{100}) \times T = (2^{100} - 1) \times T \quad (2.1)$$

different combination which is far too large to be handled by existing computer resources. In this equation, T is the number of task groups in the network (i.e., the number of tasks that the network is capable of performing).

The existing solutions in literature to this problem are based on heuristic approaches. A good clustering algorithm is the one that does not require frequent topology re-construction as this will lead to frequent information exchange among the nodes in the network. This will eventually lead to high computation overhead. Evolutionary algorithms typically intend to find a good solution in a reasonable amount of computing time. They have been successfully applied to many NP-hard problems such as multi-processor task scheduling, optimization, and traveling salesman problems [139].

3. METAHEURISTICS AND EVOLUTIONARY OPTIMIZATION

Metaheuristic algorithms are sophisticated heuristic algorithms aim to find a solution using higher level techniques by combining lower level techniques and tactics for exploration and exploitation of the large solution spaces. In those problem domains where the complexity makes the use of exact techniques unaffordable, employing of metaheuristics has steadily gained popularity and usage. Nowadays, these techniques exhibit a remarkable success record, and are considered cutting-edge methods for solving hard optimization problems. Thus, whenever new problem domains arise, metaheuristic is one of the primary weapons in our solving arsenal.

In this thesis the problem of task allocation in IoT is modeled by adopting the concept of task groups and virtual objects. Accordingly, the optimal selection of the virtual objects with a high residual energy that is scattered in the area can be seen as one of the NP-hard and combinatorial problems [10]. Solutions to NP-hard problems involve searches through vast spaces of possible solutions. Metaheuristic algorithms are applied successfully to a variety of NP-hard problems. One of the most promising and prominent approaches in metaheuristics is Evolutionary Algorithms (EAs). EAs are generic, population-based metaheuristic optimization algorithms that use bio-inspired mechanisms to maintain simultaneous search of multiple basins of attraction and to eliminate noise in evaluating solution quality.

The aim of this chapter is to briefly present the main concept and characteristics of metaheuristics with a focus on EAs. The presentation will be given in both informal and formal ways. Afterwards, the conceptual framework and the parameter control needed in the design of problem specific EAs are presented. Finally, the chapter ends with a formal overview of the Multi-Objective Evolutionary Algorithms (MOEAs) presenting the algorithmic framework of elitist Non-dominated Sorting Genetic Algorithm II (NSGA-II) as one of the prominent MOEAs published to date.

3.1. Metaheuristics

The term “heuristics” originates from the greek word “heuriskein” which means to “discover” or “find” or “search” by trials and errors. Heuristics are popularly known as

rules of thumb, educated guesses, intuitive judgments or simply common sense. In more precise terms, heuristics stand for strategies using readily accessible though loosely applicable information to control problem solving processes in human beings and machines. Heuristic algorithms typically attempt to find a good solution to an optimization problem in a reasonable amount of computing time. However, there is no guarantee to find the optimal solution, though it might find a better or improved solution than an educated guess. Broadly speaking, heuristic methods are local search methods because their searches focus on the local variations, and the optimal or best solution can be located outside this local region. However, a high-quality feasible solution in the local region of interest will usually be accepted as a good solution in many optimization problems in practice if time is the major constraint [140].

Two different types of heuristics are identified up to now [141]. *Constructive Heuristics* and *Search Heuristics (Heuristic Search Strategies)*. Constructive heuristics are mainly problem specific and try to construct one single solution with best possible quality by carefully selecting promising solution elements. Search heuristics implement a search in the solution space of a given problem during which they examine many different solutions in order to find the best possible one. Three problem-independent, basic principles of heuristic search can be identified: *repeated solution construction* where a new solution is obtained by constructing a new one from scratch, *repeated solution modification* where a new solution is obtained by modifying an existing one, and *repeated solution recombination* where a new solution is obtained by recombining two or more existing solutions.

The actual realization of the basic principle itself, however, is a problem specific issue, since it has to be defined how a solution may be constructed, modified or recombined. This distinction between problem independent and problem specific aspects of search heuristics reveals a major advantage: the search strategies and the problem specific parts can be implemented independently from each other. This means that it is possible to implement search strategies in a completely abstract way, like a framework. Later, when it comes to a concrete application, it is of course necessary to implement the problem specific parts (e.g., a solution modification mechanism). But once this is done, they can typically easily be “plugged” into an already existing (search) framework and it is possible to run the associated strategies. Hence abstraction allows the reuse components of search heuristics

and may avoid that the respective method has to be implemented from a scratch for each new problem. The abstract parts can be regarded as an upper or “meta” level of a search heuristic, leading to the term “metaheuristic” [141].

Metaheuristic algorithms are advanced heuristic algorithms. Because “meta” means “beyond” or “higher level”, metaheuristic literally means to find the solution using higher level techniques, though certain trial-and-error processes are still used. Broadly speaking, metaheuristic is an iterative master process that guides and modifies the operations of subordinate heuristics to efficiently produce high-quality solutions. It may manipulate a complete (or incomplete) single solution or a collection of solutions at each iteration. The subordinate heuristics may be high (or low) level procedures, or a simple local search, or just a construction method [142]. In recent years [140], the word metaheuristics refers to all modern higher-level algorithms, including Evolutionary Algorithms (EAs) Simulated Annealing (SA), Tabu Search (TS), Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO), Bee Algorithms (BA), Firefly Algorithm (FA), and Harmony Search Algorithm (HSA).

In general there are two important components in modern metaheuristics: *exploitation* (or intensification) and *exploration* (or diversification) [142]. For an algorithm to be efficient and effective, it must be able to generate a diverse range of solutions including the potentially optimal solutions so as to explore the whole search space effectively, while it intensifies its search around the neighborhood of an optimal or nearly optimal solution. In order to do so, every part of the search space must be accessible though not necessarily visited during the search. Diversification is often in the form of randomization with a random component attached to a deterministic component in order to explore the search space effectively and efficiently, while intensification is the exploitation of past solutions so as to select the potentially good solutions via elitism or use of memory or both. In other words; Intensification is to search carefully and intensively around good solutions found in the past search. Diversification, on the contrary, is to guide the search to unvisited regions. In general, metaheuristics can be classified into three categories according to the solution method [143, 141]: *metaheuristics based on solution construction*, *metaheuristics based on solution modification*, and *metaheuristics based on solution recombination*. Metaheuristics based on solution construction methods do not perform search, they construct a single solution in an iterative fashion by evaluating all remaining solution elements and according

to their performance add them to the partial solution. Elements are added as long as the solution is improved. If this is not the case anymore, the construction is stopped and the final solution is returned. Metaheuristics based on solution modification introduce the principle of repeatedly modifying solutions such as to, finally, obtain better ones. Finally, metaheuristics based on solution recombination introduce how search by recombination can be performed and that multiple solutions can be used to apply recombination. In the following section, the essence of Evolutionary Algorithms (EAs) is presented in some details as these are considered the most prominent, interesting, useful, easy-to-understand, and hot research topics in the field of metaheuristics.

3.2. Evolutionary Algorithms (EAs)

Evolutionary algorithms (EAs) are generic, population-based metaheuristic optimization algorithms that use biology-inspired mechanisms like selection and variation. The population approach allows simultaneous search of multiple basins of attraction and eliminates noise in evaluating solution quality. The selection operator nudges the search toward superior solutions, whereas the variation operators promote wider exploration [142, 144].

Classical EAs, include Genetic Algorithms (GAs), Evolution Strategy (ES), Evolutionary Programming (EP), and Genetic Programming (GP). All these methods are random based solution space searching metaheuristic algorithms. In the following, the basic EA structural terms and concepts are defined where the described terms' meanings are normally analogous to their genetic counterparts. A *structure* or *individual* is an encoded solution to some problem. Typically, an individual is represented as a string (or string of strings) corresponding to a biological *genotype*. A *genotype* describes the genetic composition of an individual as inherited from its parents. Genotypes provide a mechanism to store experiential evidence as gathered by parents. This genotype defines an individual organism when it is expressed (decoded) into a *phenotype*. A *phenotype* is the expressed behavioral traits of an individual in a specific environment. A genotype is composed of one or more *chromosomes*, where each chromosome is composed of separate *genes* which take on certain values (*alleles*) from some genetic alphabet. A *locus* identifies a gene's position within the chromosome. Thus, each individual decodes into a set of parameters used as input to the function under consideration. Finally, a given set of chromosomes is termed a

population, and a number of individual solutions are created to form an *initial population*. The following steps are then repeated iteratively until a solution has been found which satisfies a pre-defined *termination criterion*. Each individual is evaluated using a *fitness function* that is specific to the problem being solved. Based upon their fitness values, a number of individuals are chosen to be *parents*. New individuals, or *offspring*, are produced from those parents using *reproduction operators*. The fitness values of those offspring are determined. Finally, survivors may be selected from the old population and the offspring to form the new population of the next *generation* [144- 147].

The following subsections aim to enable readers to get a smooth touch with the concept of EAs. Essential definition with a focus on the generic meta-level of the EA is given next. Moreover, the relation to the problem-specific level is also considered important in designing any EA, and therefore treated explicitly in a separate subsection. Finally, one of the main difficulties that a user faces when applying an EA (or, as a matter of fact, any metaheuristic method) to solve a given problem is to decide on an appropriate set of parameter values. For example, before running the algorithm, the user typically has to specify values for a number of parameters, such as population size, selection rate, and operator probabilities. The third subsection will present briefly how to automate control of these parameters.

3.2.1. General algorithmic framework of EA

Just as in nature, Evolutionary Operators (EVOPs) operate on an EA's population, attempting to generate solutions with higher and higher fitness. The three major EVOPs associated with EAs are mutation, recombination, and selection. In the selection operation, the above average individuals with a proportion to their fitness values are selected (reproduced) to form a mating pool and become parents of the next generation's individuals more often than below average individuals. The selection EVOP effectively gives strings with higher fitness, a higher probability of contributing one or more children in the succeeding generation. Then the information contained in the good individual has more chances to be preserved and passed onto the next generation. Information exchange between two (or more) parents and small changes in the offspring promote the search for better individuals. Combining these two factors, the population will gradually increase in their fitness until the optimal or near optimal solution has been found [144, 148].

To formally define an EA, its general algorithm will be described in mathematical terms, allowing for exact specification of various EA instantiations. In this framework, each EA is associated with a non-empty set I called the EA's individual space. Each individual, $a \in I$, normally represents a candidate solution to the problem in interest that being solved by the EA. Individuals are often represented as a vector, where the vector's dimensions are analogous to a chromosome's genes. In the (generational) population transformations, the resulting collection of μ individuals is denoted via I^μ , and the population transformations are denoted by the relationship $T : I^\mu \rightarrow I^\lambda$, where $\mu, \lambda \in \mathbb{N}$ indicating succeeding populations may contain the same *or* different numbers of individuals. The framework also represents all population sizes, evolutionary operators, and parameters as sequences. This is due to the fact that different EAs use these factors in slightly different ways. The general algorithm thus recognizes and explicitly identifies this nuance. Having discussed the relevant background terminology, an EA is then defined as [144, 149]:

Definition (Evolutionary Algorithm): Let I be a non-empty set (the individual space), $\{\mu^{(i)}\}_{i \in \mathbb{N}}$ a sequence in \mathbb{Z}^+ (The parent population sizes), $\{\lambda^{(i)}\}_{i \in \mathbb{N}}$ a sequence in \mathbb{Z}^+ (the offspring population sizes), $\Phi : I \rightarrow \mathbb{R}$ a fitness function, $\iota : \cup_{i=1}^{\infty} (I^\mu)^{(i)} \rightarrow \{\text{true}, \text{false}\}$ (the termination criterion), $\chi \in \{\text{true}, \text{false}\}$, r a sequence $\{r^{(i)}\}$ of recombination operators $r^{(i)} : \mathbb{X}_r^{(i)} \rightarrow \Gamma(\Omega_r^{(i)}, \Gamma(I^{\mu^{(i)}}, I^{\lambda^{(i)}}))$, m a sequence $\{m^{(i)}\}$ of mutation operators $m^{(i)} : \mathbb{X}_m^{(i)} \rightarrow \Gamma(\Omega_m^{(i)}, \Gamma(I^{\mu^{(i)}}, I^{\lambda^{(i)}}))$, s a sequence $\{s^{(i)}\}$ of selection operators $s^{(i)} : \mathbb{X}_s^{(i)} \times \Gamma(I, \mathbb{R}) \rightarrow \Gamma(\Omega_s^{(i)}, \Gamma((I^{\lambda^{(i)} + \chi \mu}, I^{\mu^{(i+1)}}))$, $\Theta_r^{(i)} \in \mathbb{X}_r^{(i)}$ (the recombination parameters), $\Theta_m^{(i)} \in \mathbb{X}_m^{(i)}$ (the mutation parameters), and $\Theta_s^{(i)} \in \mathbb{X}_s^{(i)}$ (the selection parameters).

Accordingly, Algorithm 3.1 is called an algorithmic framework of Evolutionary Algorithm (EA).

Algorithm 3.1. General algorithmic framework of Evolutionary Algorithm (EA)

```

t := 0; // t is the generation number
Initialize  $P(0) := \{a_1(0), \dots, a_\mu(0)\} \in I^{\mu^{(0)}}$ ;
Evaluate  $P(0) := \{\Phi(a_1(0)), \dots, \Phi(a_\mu(0))\}$ ;
While ( $\{P(0), \dots, P(t)\} \neq \text{true}$ ) do
    Recombine:  $P'(t) := r_{\Theta_r^{(t)}}^{(t)}(P(t))$ ;
    Mutate:  $P''(t) := m_{\Theta_m^{(t)}}^{(t)}(P'(t))$ ;
    Evaluate  $P''(t) := \{\Phi(a_1(t)), \dots, \Phi(a_\lambda(t))\}$ ;
    select:
        if  $\chi$ 
            then  $(t+1) := S_{(\Theta_s^{(t)}, \Phi)}^{(t)}(P''(t))$ ;
            else  $P(t+1) := S_{(\Theta_s^{(t)}, \Phi)}^{(t)}(P''(t) \cup P(t))$ ;
        fi
    t := t + 1;
od

```

3.2.2. Problem related aspects

All elements of the general algorithmic framework of EA are more or less problem dependent. However, the mutual exclusion among these elements can help to visualize the dependency to the problem. Figure 3.1 emphasizes problem dependent, partly problem dependent and problem independent components of the general solution processing scheme [141, 144].

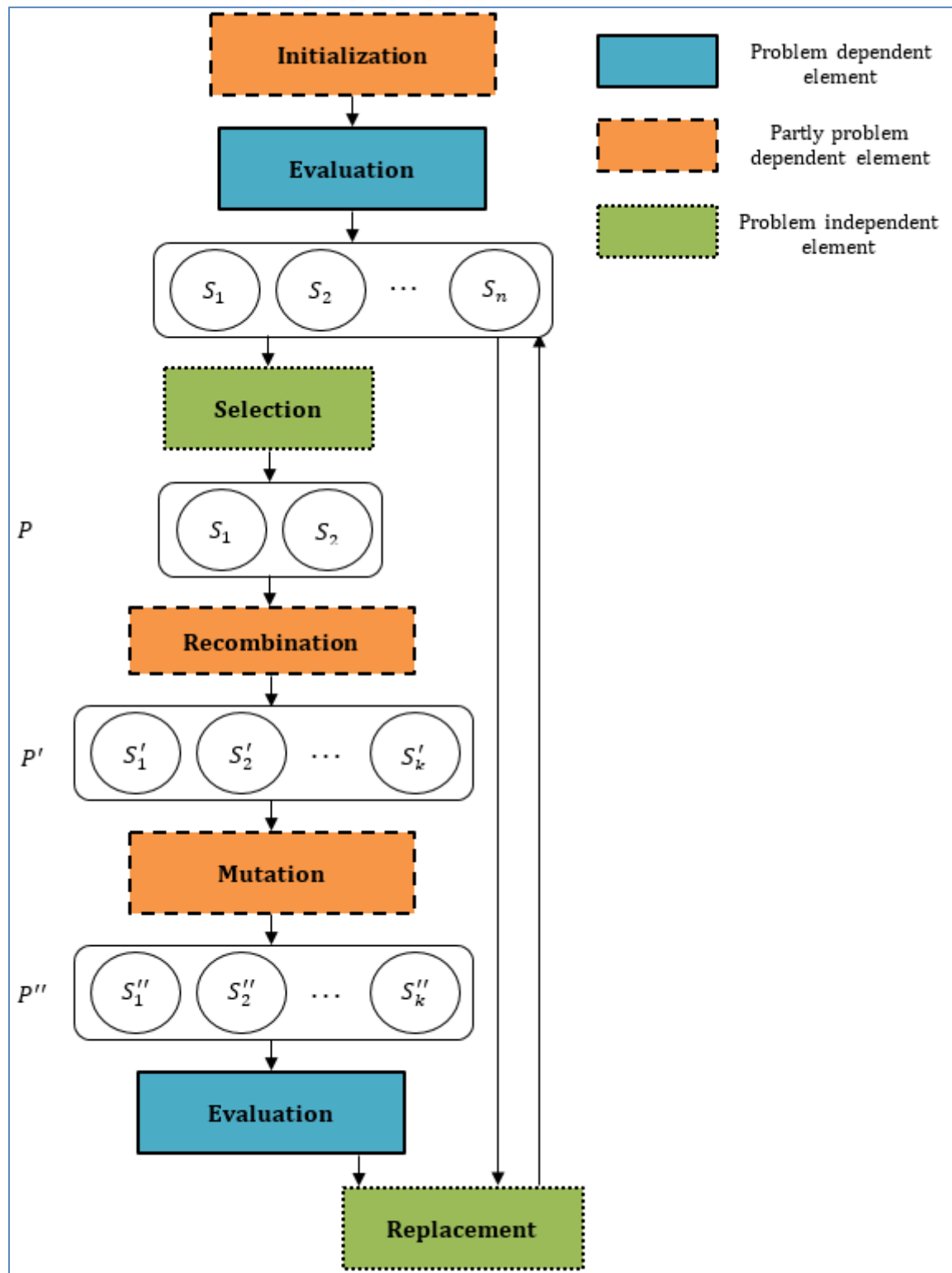


Figure 3.1. Detailed overview of EA

Obviously evaluation is fully problem dependent. Interestingly, the following components are referred to as partly problem dependent:

- Initialization
- Recombination
- Mutation

The reason is that solution coding, like binary or permutation, is used to describe the genotype structure of an individual, i.e., how the representation of the solutions is chosen. Hence those operators are referred to as partly problem dependent here. The main advantage is the applicability of generic crossover and mutation operators, i.e., no problem specific operators must be developed.

The remaining two components, selection and replacement, are problem independent, as they typically only consider the solution quality.

3.2.3. Parameter control in EAs

One of the main issues related the description of a specific EA is the specification of its components, such as the choice of representation, selection, recombination, and mutation operators, thereby setting a framework while still leaving quite a few items undefined. For instance, a given EA might be stated by using binary representation, uniform crossover, bit-flip mutation, tournament selection, and generational replacement. For a full specification, however, further details have to be given, for instance, the population size, the probability of mutation and crossover, and the tournament size. These data called the *algorithm parameters* or *strategy parameters*. They important to complete the definition of the EA and they are necessary to produce an executable version. The values of these parameters greatly determine whether the algorithm will find an optimal or near-optimal solution and whether it will find such a solution efficiently [144, 148].

Globally, two major forms of setting parameter values can be distinguished: parameter tuning and parameter control. By parameter tuning we mean the commonly practiced approach that amounts to finding good values for the parameters before the run of the algorithm and then running the algorithm using these values, which remain fixed during the run. Parameter control forms an alternative, as it amounts to starting a run with initial parameter values that are changed during the run [142, 144].

3.3. Description of Some Important Issues in EAs

The next two subsections discuss how to deal with infeasible solutions and how to maintain the necessary exploration and exploitations in the EA problem solving procedure.

3.3.1. Solution infeasibility

Another important point in EAs is how to deal with *infeasible solutions*. Basically infeasible solution is no solution at all, because it violates at least one constraint, so the result is unusable. Typically, the occurrence of infeasible solutions depends on the design of the recombination and mutation operators [141].

Basically, there are three options to deal with infeasible solutions [141, 150]:

1. discard infeasible solutions,
2. penalize infeasible solutions, and
3. repair infeasible solutions.

Each of the three options has advantages and disadvantages. Obviously the easiest method for handling infeasible solutions is to discard them. But if finding a solution is computationally expensive this method will not be appropriate. The second approach is penalizing infeasible solutions. That is, the amount of infeasibility must somehow be measured and then worsens the quality of the solution, i.e., the fitness of a solution degrades. This raises another problem of finding a penalty function which allows infeasible solutions but drives the search into the feasible area in the search space. The last option is repairing the infeasible solution. Repairing an infeasible solution is problem dependent and requires certain design decisions. In some cases repairing a solution is as difficult as constructing a feasible solution to the given problem.

3.3.2. Exploration and exploitation

Exploration and *exploitation* (a.k.a., *diversification* and *intensification*, respectively) are the two cornerstones of problem solving by search. They are often used to categorize distinct phases of the search process. Roughly speaking, exploration is the generation of

new individuals from the untested regions of the search space, while exploitation means the concentration of the search in the vicinity of known good solutions. Evolutionary search processes are often referred to in terms of a trade-off between exploration and exploitation. Too much of exploitation can lead to a propensity to focus the search too quickly, while too much exploration leads to slow convergence rates [151].

In EAs, search operators (mutation and recombination) and selection have been characterized by their contribution to the explorative and exploitative aspects of the search. Selection is commonly seen as the source of exploitation, while exploration is attributed to the operators of mutation and recombination. Selection operator selects individuals for mating or reproduction from already existed solutions according to their fitness concentrating the search in the vicinity of known good solutions. The search operators generate new solutions forming the random part of the algorithm. They explore new regions of the search space to ensure the necessary diversity to fuel the evolutionary process [152].

3.4. Evolutionary Multi-objective Optimization

EAs are developed to solve real-world problems, such as clustering and scheduling. Many of these problems involve the simultaneous optimization of several competing objectives and constraints that are difficult, if not impossible, to be solved without the aid of powerful optimization algorithms. One way to deal with them is to combine all objectives into one single fitness function. However, the drawback is that one must explicitly state the influence of each part in the overall fitness function. This gets even harder if the unit of measurement is different for every objective. Multi-Objective Evolutionary Algorithms (MOEA) are efficient optimization methods used in solving problems with multiple conflicting objectives in various branches of engineering, science, and commerce. The goal of MOEAs is to provide the decision maker with a complete set of solutions such that no other solutions in the search space are better than them with respect to all the considered objectives.

This section describes the application of evolutionary techniques to a particular class of problems, namely multi-objective optimization. First, an introduction to this class of problems and definitions of multi-objective optimization and single objective optimization

problems is presented. An important explanations and notations of Pareto optimality is given next. Finally, this section ends by presenting algorithmic framework of elitist Non-dominated Sorting Genetic Algorithm II (NSGA-II) as one of the attractive and prominent MOEAs.

3.4.1. Multi-objective optimization problems (MOPs)

In order to develop an understanding of MOPs and the ability to design MOEAs to solve them, a series of formal non-ambiguous definitions are required. To give a simple illustration, we begin by defining the Single-objective Optimization Problems (SOPs) as follows [149]:

Definition (Single-objective Optimization Problem): *General SOP is defined as minimizing (or maximizing) $f(x)$ subject to $g_i(x) \leq 0, i = \{1, \dots, m\}$, and $h_j(x) = 0, j = \{1, \dots, p\}$ $x \in \Omega$. Then, a solution minimizes (or maximizes) the scalar $f(x)$ where x is a n -dimensional decision variable vector $x = (x_1, \dots, x_n)$ from some search space Ω .*

In this definition, $g_i(x) \leq 0$ and $h_j(x) = 0$ represent the inequality and equality constraints respectively, that must be fulfilled while optimizing (minimizing or maximizing) $f(x)$. Ω contains all possible x that can be used to satisfy an evaluation of $f(x)$ and its constraints. Also in this definition, the “decision variable vector x ” is a vector of the numerical quantities for which values are to be chosen in an optimization problem.

In practice it turns out that a great many applications that have traditionally been tackled by defining a single objective function (quality function) have at their heart a MOP that has been transformed into SOP in order to make optimization tractable [151]. MOPs (as a rule) present a possibly uncountable set of solutions, which when evaluated, produce vectors whose components represent trade-offs in objective space. A decision maker then implicitly chooses an acceptable solution (or solutions) by selecting one or more of these vectors. Then, the MOP (also called multicriteria optimization, multiperformance or vector optimization problem) can be informally defined as the problem of finding [151, 153]:

“A vector of decision variables which satisfies constraints and optimizes a vector function whose elements represent the objective functions. These functions form a mathematical description of performance criteria which are usually in conflict with each other. Hence, the term “optimize” means finding such a solution which would give the values of all the objective functions acceptable to the decision maker.”

More precisely, MOPs are those problems where the goal is to optimize k objective functions simultaneously. This may involve the maximization of all k functions, the minimization of all k functions or a combination of maximization and minimization of these k functions. Formally, the general MOP (global minimum or maximum problem) can be then defined as follows [151,152, 154-159]:

Definition (Multi-objective Optimization Problem): *General MOP is defined as minimizing (or maximizing) $F(x) = [f_1(x), \dots, f_k(x)]$ subject to $g_i(x) \leq 0, i = \{1, \dots, m\}$, and $h_j(x) = 0, j = \{1, \dots, p\}$ $x \in \Omega$. An MOP solution minimizes (or maximizes) the components of a vector $F(x)$ where x is a n -dimensional decision variable vector $x = [x_1, \dots, x_n]$ from some search space Ω .*

In the definition above, it is noted that $g_i(x) \leq 0$ and $h_j(x) = 0$ represent constraints that must be fulfilled while minimizing (or maximizing) $F(x)$ and Ω contains all possible x that can be used to satisfy an evaluation of $F(x)$.

3.4.2. Dominance concept and Pareto optimality

In an MOP of k functions, $f_1(x), f_2(x), \dots, f_k(x)$ one candidate solution x_1 may be better than another solution x_2 , with respect to f_i (i.e., without loss of generality, for maximization problem $f_i(x_1) < f_i(x_2)$), but worse with respect to f_j (i.e., for maximization problem $f_j(x_1) > f_j(x_2)$). In other words, solution x_1 has a mutually better than and mutually worse than relation with solution x_2 . Thus, x_1 and x_2 have a *non-domination* relationship (this also known as *Pareto dominance*). In more general formalization, MOP can be redefined according to the domination concept as: finding a vector $x^* = [x_1^*, \dots, x_n^*]$ optimizing the vector function $F(x) = [f_1(x), \dots, f_k(x)]$ where $x = [x_1, \dots, x_n]$ is the decision variables vector. Here, the optimization of $F(x)$ is based on

the domination concept. To understand the domination concept, let us consider two arbitrary solutions \mathbb{U} and \mathbb{V} from the solution space Ω , both of which have scores according to some set of objective values. Then, solution \mathbb{U} is said to *dominate* \mathbb{V} if and only if the following two conditions hold [5,149,160-162]:

- Solution \mathbb{U} is no worse than solution \mathbb{V} in all objectives. For example and without loss of generality in maximization, the word “no worse” means $f_i(\mathbb{U}) \nless f_i(\mathbb{V})$ for all $i = 1, 2, \dots, k$.
- Solution \mathbb{U} is strictly better than solution \mathbb{V} in at least one objective. For example in maximization, the word “strictly better” means $f_i(\mathbb{U}) > f_i(\mathbb{V})$ for at least one $i \in 1, 2, \dots, k$.

For conflicting objectives, there exists no single solution that dominates all others, and we will call a solution *global non-dominated* (or *Pareto Optimal*) if it is not dominated by any other solution in the search space. The goal of multi-objective optimization algorithms is to preserve the global non-dominated points (or at least the near-global non-dominated points) in objective space and associated solution points in decision space to provide the decision maker with a set of non-dominated solutions x^* . Hence, a *global non-dominated set* (also called *Pareto Optimal Set*) can be defined as: among a set of solutions in the solution space Ω , the non-dominated solutions set Ω' are those that are not dominated by any member of the set Ω . All non-dominated solutions possess the attribute that their quality cannot be increased with respect to any of the objective functions without detrimentally affecting one of the others (see Figure 3.2). The set of all non-dominated solutions in one front is called *non-dominated front*, *Pareto set* or *Pareto front*.

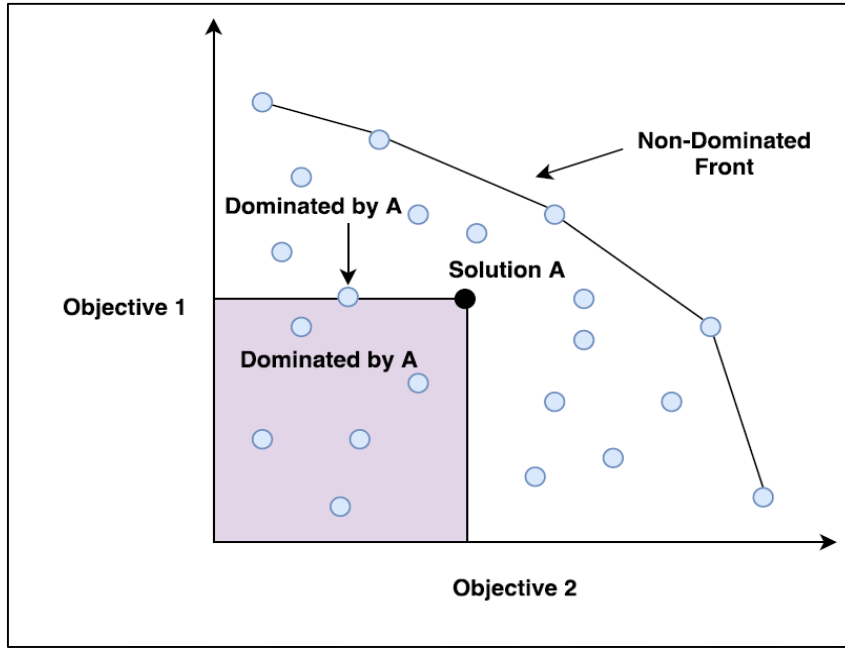


Figure 3.2. Multi-objective optimization problem based on domination concept while maximizing two objectives

EAs have a proven ability to identify high-quality solutions in high-dimensional search spaces containing difficult features such as discontinuities and multiple constraints. When coupled with their population-based nature and their ability for finding and preserving diverse sets of good solutions, it is not surprising that EA-based methods are currently the state of the art in many multi-objective optimization problems [151].

3.4.3. Elitist non-dominated sorting genetic algorithm II (NSGA-II)

EAs designed for MOPs are called *Multi-Objective EAs* (MOEAs), and this kind of optimization is called *Evolutionary Multi-objective Optimization* (EMO). EAs have attracted to solve MOPs due to their ability to search for multiple solutions in parallel (so that a family of feasible solutions to the problem is found) as well as handle complex features such as multimodality, and disjoint objective spaces [145, 161, 163, 164]. In literature, there have been many approaches to EMOs. All have the following common primary goals [158]:

- Preserve global non-dominated points (or at least their nearest points) in objective space and associated solution points in decision space.

- Continue to make algorithmic progress towards the global non-dominated set in objective function space.
- Maintain a good diversity and distribution of generated solutions along the non-dominated front.
- Provide the decision maker with enough but yet limited number of non-dominated points for selection resulting in decision variable values.

An example of the classical MOEAs is elitist Non-dominated Sorting Genetic Algorithm II (NSGA-II) [161, 162, 165]. NSGA-II is one of the most attractive and prominent MOEAs published to date [160]. NSGA-II is a population-based optimization algorithm that assigns fitness based on dividing the population into a number of fronts of equal domination. To achieve this, the algorithm iteratively seeks all the non-dominated points in the population that have not been labeled as belonging to a previous front. It then labels the new set as belonging to the current front, and increments the front count, repeating until all solutions have been labeled. Each point in a given front gets as its raw fitness the count of all solutions in inferior fronts. In this algorithm the diversity is implemented by considering the members from that individual's front [152].

The general framework of NSGA-II (as presented in Algorithm 3.2) begins with an initial population of solutions and employs Genetic Algorithm (GA) with the traditional crossover and mutation operators. The selection process is derived by sorting the individuals according to a ranking and crowding distance schemes. All non-dominated individuals are classified into one front based on non-domination. Then, this group of classified individuals is ignored and another front of non-dominated individuals is considered. The process continues until all individuals in the population are classified. Individuals in the first front form completely non-dominant set in the current population whereas Individuals in the second front become dominated by the individuals in the first front and the fronts go so on. Each individual in each front assigned a rank (a dummy fitness) value based on the front in which it belongs. Individuals in the first front are given a rank value 1 and individuals in the second front are given rank value 2 and so on. Once the non-dominated sort is completed, a crowding distance is then calculated and added to each individual in the population in front-wise manner. The crowding distance computation requires sorting each front according to each objective function value in ascending order of magnitude. Then, for each objective function, the boundary individuals

(solutions with smallest and largest function values) are assigned an infinite distance value. All other intermediate individuals are assigned a Distance Value (DV) equal to the absolute normalized difference in the function values of two adjacent individuals (previous and next individuals) as in the equation below:

$$DV(\text{current individual}) = f_i(\text{Current individual}) + \frac{f_i(\text{next}) - f_i(\text{previous})}{f_i(\text{max}) - f_i(\text{min})} \quad (3.1)$$

where f_i is the fitness function $\forall i \in (1, \dots, k)$, max is the maximum value in this front for f_i , and min is the minimum value in this front for f_i . Then, this calculation is continued with other objective functions. The overall crowding distance value is calculated as the sum of individual distance values corresponding to each objective. The crowding distance is a measure of how close an individual is to its neighbors. A large average crowding distance value results in better diversity in the population. Within the selection and population reduction phases, a binary tournament selection method is used and individuals are compared based on their contribution to the diversity of the population. Binary tournament selection operator criterion is based on the non-domination and crowded comparison. Individuals with lesser ranks of non-dominant fronts or with equal ranks (i.e., belong to same front) but with greater crowding distances are selected to form mating pool. Since individuals in the first front have minimum rank value, they always get more copies than the rest of the population. This allows search for non-dominated regions, and results in convergence of the population toward such regions. Finally, after selection and variation operations (crossover and mutation operators), population with the current individuals (parents) and off-springs is sorted again based on non-domination and only the best N individuals are selected, where N is the population size. The selection is based on the rank value and on the crowding distance value in the last front. This evolutionary process continues until a previously determined stopping criterion is met.

Algorithm 3.2. Algorithmic Framework of NSGA-II

Input:

- Multi-objective problem $F(x) = (f_1(x), \dots, f_k(x))$, k is number of objective functions
- Population size, N
- Stopping criterion, τ

Output:

- First front of the non-domination fronts

Step 1 – Setup

- $gen = 0$.
- Initialize random population of solutions, $= \{P_1, \dots, P_N\}$.
- Evaluate objectives vector $f_i(P_j), \forall i \in \{1, \dots, k\}, \forall j \in \{1, \dots, N\}$.
- Assign rank $r(P_i), \forall i \in \{1, \dots, N\}$ based on non-domination fronts.
- Assign crowding distance rank $c(P_i), \forall i \in \{1, \dots, N\}$

Step 2 – Update:

- Generate a new population $P'_i = \{P'_1, \dots, P'_N\}$ where Child P'_i is obtained by using binary tournament selection, recombination, and mutation.
- For each Parent in P and Child in P'
 - i. Assign rank based on non-domination.
 - ii. Generate sets of non-dominated vectors.
 - iii. Determine crowding distance between points on each front.
 - iv. Elitist selection: The new generation is filled by each front subsequently until the population size exceeds the current population size. If by adding all the individuals in front j the population exceeds N then individuals in front j are selected based on their crowding distance in the descending order until the population size is N .

Step 3 – Stopping criteria

- If $\tau = \text{True}$, then stop and output is the first non-dominated front, otherwise $gen = gen + 1$, go to Step 2.

4. THE PROPOSED TASK ALLOCATION PROTOCOLS WITH METAHEURISTIC METHODOLOGY

IoT is an environment that contains different embedded devices interacting with each other to perform tasks related to information collection, communications and processing. Successful applications of IoT aim to interconnect objects with various capabilities within the same heterogeneous network. The goal is to allow network entities to cooperate and make their resources available to reach the optimal allocation of tasks among themselves in order to perform the demanded task. However, assigning tasks to group of heterogeneous objects that are equipped with limited resources poses a challenging task. The most limited and valuable resource for variety of IoT objects is battery power. Therefore, minimizing the energy consumption in task allocation process is one of the primary objectives.

The problem of task allocation is extensively studied in the field of WSNs. However, the task allocation problem in IoT is an open research issue. Currently, almost all existing studies employ heuristic optimizations to cope with different aspects of task allocation problems without considering the heterogeneity in terms of energy levels and computational power of network entities. CBATA [8, 12] is an interesting and prominent work that attempts to solve the problem of task allocation in IoT by applying heuristic steps while employing task groups and virtual objects concept. Although, CBATA provides robustness against links and nodes failures, it requires a heavy message exchange procedure which is very intensive with regard to communication, computation, storage, and energy overheads. To the best of our knowledge, no attempt has been made to employ task groups and virtual objects-based framework to solve the problem of task allocation in IoT using meta-heuristic methods. The work in this thesis is among the first works to propose task groups and virtual objects based framework to solve task allocation problem in IoT and the first work that adopts meta-heuristic methods for this purpose. Moreover, it represents the first work attempting to address computational power utilization and energy efficiency problems in task allocation of IoT as a MOP.

The first part of this chapter is devoted to present four task allocation protocols in IoT using EA technique. In these protocols the scenario where each object is capable of performing one task and can be a member in one task group is assumed. These for protocols are: Evolutionary Task Allocation Protocol-1 (ETAP1), Evolutionary Task

Allocation Protocol-2 (ETAP2), Stable Evolutionary Task Allocation Protocol-1 (SETAP1) and Stable Evolutionary Task Allocation Protocol-2 (SETAP2). ETAP1 and ETAP2 are designed for application that is supposed to work in typical environments with the goal of minimizing the energy expenditures for tasks allocation to ensure maximal network longevity. Their proposed objective functions take into consideration the energies and the distance between the objects to form efficient clusters in terms of network lifetime, and energy consumption. On the other hand, SETAP1 and SETAP2 are designed for crucial applications motivating by the fact that most IoT based networks are heterogeneous especially in terms of energy levels. The key idea of SETAP1 and SETAP2 is to hypothesize a possible energy based heuristic for individual solution initialization and mutation to properly maintain longer stability periods.

The second part of this chapter, assumes a more realistic and complex scenario where objects are capable of performing a variety of tasks and can be members in more than one group. In this context two novel protocols are proposed. These protocols are: Stady Task Allocation Protocol (STAP) and Multi-Objective Task Allocation Protocol (MOTAP). STAP formats the problem of task allocation as a meta-heuristic optimization problem with the goal of increasing stability and operational periods of the network by developing a novel single objective optimization algorithm. MOTAP jointly formulates the computational power utilization and energy efficiency problems in task allocation of IoT as a novel MOP. Again, up to the best of our knowledge, this is the first work attempting to address computational power utilization together with energy efficiency problems in task allocation of IoT. Also considering this scenario, we modified the most relative method to the stated problem, namely CBATA [8, 12], and redirect its goal toward energy efficiency by developing Modified-CBATA (M-CBATA) algorithm.

This chapter organized as follows: the chapter begins by defining the problem and the solution concept used in this thesis for task allocation and then describes the system model. Afterwards, ETAP1 and ETAP2 that are developed within the framework of the first scenario are presented by giving an overview of their algorithmic framework. The operations and operators that are used in the protocols, such as the way in which the virtual objects will be elected and the evolutionary operators used for that are then explained. This is followed by explaining the association and a formal layout of the proposed protocols. Later, the details of the heuristics that required to redirect the goal of ETAP1 and ETAP2

toward stability awareness protocols (i.e., SETAP1 and SETAP2) are presented. The protocols of the second scenario are then presented, beginning by explaining the environment of the second scenario. Thereafter, giving an illustrative example the modified version of CBATA (i.e., M-CBATA) is presented. Later, STAP protocol with its conceptual framework and its ejected energy aware heuristics is explained. Finally, the chapter is closed with presenting the details of MOTAP and defining its operations and operators both formally and informally.

4.1. Problem Definition and System Model

In our protocols, to model IoT network, we assume a static two-dimensional area \mathbb{A} with size $(\mathbb{X}_{max}, \mathbb{Y}_{max})$. The internet coverage is provided in this area via an access point \mathbb{AP} located at the center of the area (e.g., \mathbb{AP} located at $(\mathbb{X}_{max}/2, \mathbb{Y}_{max}/2)$). We also assume that \mathbb{A} has a set O of n objects with known coordinates $O = \{(o_{1x,y}), (o_{2x,y}), \dots, (o_{nx,y})\}$. These objects are randomly deployed in \mathbb{A} ($1 \leq \forall i \leq n \mid o_{i,x,y} = ([0, \mathbb{X}_{max}], [0, \mathbb{Y}_{max}])$) and they are equipped with different energy levels. Furthermore, we assume that the IoT network is capable of performing T different tasks simultaneously.

In our protocols, each task in the network is represented by a group of virtual objects. These virtual objects are responsible of assigning the task to their connected objects. In order to define our task allocation system that is based on task groups and virtual objects concept, we assume that a task $t, \forall t \in \{1, \dots, T\}$ is needed to be performed. Each virtual object that belongs to the task group of task t sends Indication Messages (IMs) to their connected objects to inform them that the task associated with the task group (i.e., task t) is needed to be performed. Then, task group objects that received IMs perform the task and send the results back to the virtual objects. Virtual objects aggregate these result messages and send the aggregated results to \mathbb{AP} to be available in the cyberspace.

To calculate power consumption for transmitting and receiving data packets among network objects, virtual objects and access point, we employed the first-order radio model [166], which is illustrated in Figure 4.1.

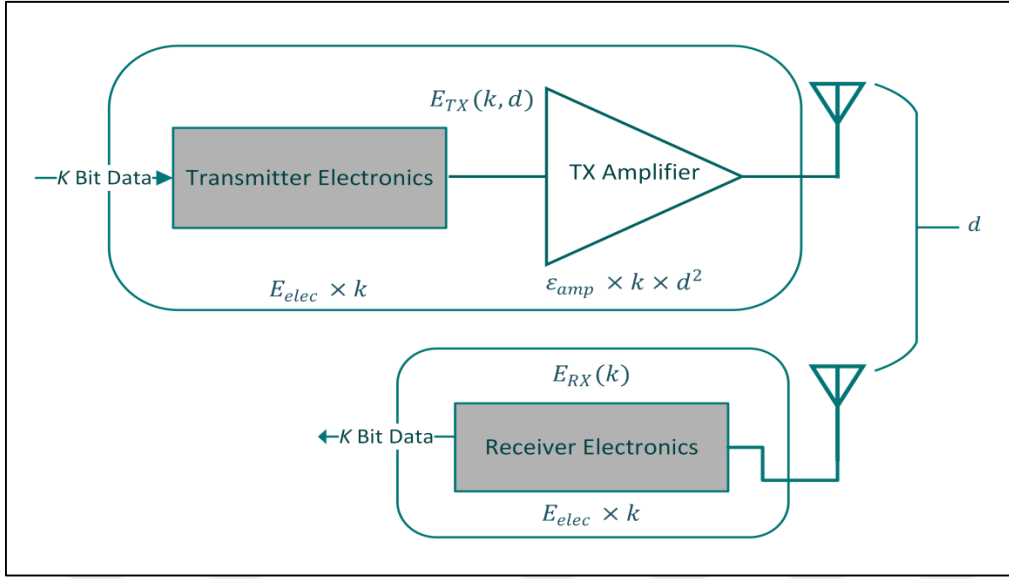


Figure 4.1. First-order radio model

In first-order radio model the energy expenditures are measured as a function of distance. That is, the energy consumption to transmit a message of k -bit over a distance d is:

$$E_{Tx}(k, d) = E_{elec} \times k + \epsilon_{amp} \times k \times d^2 \quad (4.1)$$

while the energy consumption to receive this message is:

$$E_{Rx}(k) = E_{elec} \times k \quad (4.2)$$

in the above equations, E_{elec} represents the energy dissipated per bit to run the transmitter or the receiver circuit and ϵ_{amp} represents an electronic amplifier used in transmitter circuit to convert a low-power radio-frequency signal into a higher power signal in order to achieve an acceptable Signal-to-Noise Ratio (SNR). We also consider data aggregation in our protocol. We suppose that the data gathered at vertical objects are processed before being transmitted. In this case, the number of bits to be sent would be smaller, and therefore the transmission energy consumption would be lower.

4.2. Overview of the Proposed Protocols in Context of Scenario #1

This scenario considers the IoT platforms where each object in the network is capable of performing only one task (e.g., measuring the humidity, record the temperature, monitoring

traffic congestions, tracking the movement of objects or persons, detecting the risky conditions in public facilities, and so on). Accordingly, a group of objects that are capable of performing similar and replaceable tasks can be classified under one a task group. It is obvious that in most IoT applications there may be several task groups in the network. From each task group some objects (or an object) are selected to be virtual objects that represent the task group. The rest of the objects within the task group are then clustered and connected to one virtual object. Figure 4.2 gives visualize illustration of scenario #1.

In the context of this scenario four protocols are developed. These protocols are ETAP1 and ETAP2 and their stability redirected protocols SETAP1 and SETAP2. In their essential, the proposed evolutionary task allocation protocols are centralized protocols where the Central Control Station (CCS) has to determine the task allocation configuration in terms of virtual objects in each task group. The CCS uses evolutionary algorithm as a tool (with a problem specific specializations) for creating an energy efficient clusters (i.e., by selecting the appropriate objects as virtual objects) at each task group for a given number of objects.

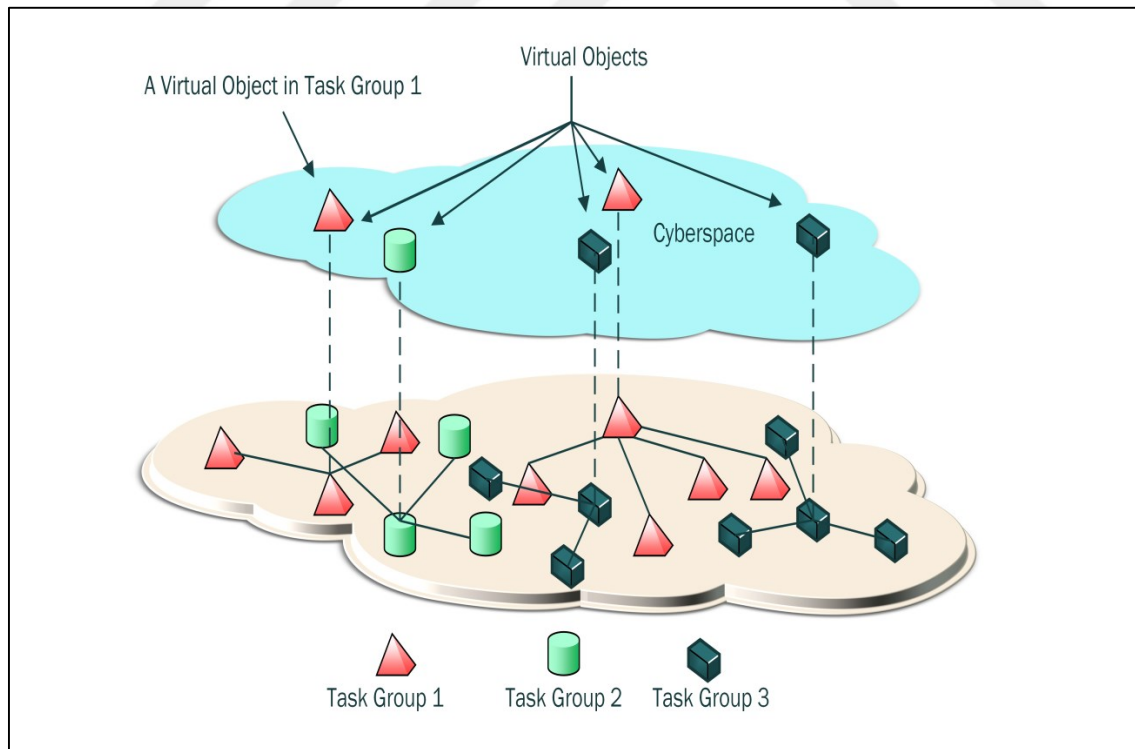


Figure 4.2. The concept of task groups and virtual objects in scenario #1

The configurations for task allocation in the proposed protocols consist of two phases: election phase and association phase. In the election phase, virtual objects are chosen using an adopted evolutionary algorithm. The evolutionary algorithm in the election phase begins with generating an initial population of solutions. The complete solutions are encoded as individuals that determine where the virtual objects and member objects are located in each task group in the network. At this step, each individual is evaluated and assigned a fitness value based on the estimated energy consumption, which is determined by the proposed objective function parameters. The population approach allows simultaneous search of multiple basins of attraction and eliminates noise in evaluating solution quality. After the population initialization phase, the evolutionary loop begins, where individuals go through evolutionary operators –selection and variation operators– with pre-determined probabilities to improve the quality of the individuals. In selection operator, the above average individuals with a proportion to their fitness values are selected (reproduced) to form a mating pool and become parents of the next generation's individuals more often than below average individuals. On the other hand, in variation operator, which consists of evolutionary recombination and evolutionary mutation, new unexplored solutions are derived. Recombination involves exchanging information between two (or more) parents from the mating pool to produce new offspring while mutation involves posing small changes in the offspring in order to promote the search for better individuals. At the end of each variation operator the fitness value is calculated and assigned to offspring individuals. It is obvious that, the selection operator pokes the search toward superior solutions, whereas the variation operators promote wider exploration. The variation operators are followed by an elitism selection. In elitism selection the best N individuals are selected and allowed to be transformed to next generation, where N is the population size. These operations iterated for a number of generations until the termination criteria is satisfied. Then, the best individual solution will be used to seed the next phase- the association phase. In association phase, the non-virtual object members of each task group are associated to their virtual objects to form clusters. The general framework of evolutionary algorithms that the evolutionary task allocation protocols follow is depicted in Algorithm 4.1.

Algorithm 4.1. Outline of the evolutionary algorithm trajectory

Input:

- An optimization problem $F(\mathbb{X})$, where \mathbb{X} is n-dimensional decision variable $\mathbb{X} = \{\chi_1, \dots, \chi_n\}$
- Population size, N
- Termination criterion $\tau \rightarrow \{\text{true}, \text{false}\}$

Output:

- Best solution in the last generation according to fitness values

Begin:

Step 1 – Initialize Population:

- $\text{gen} = 0$
- Initialize random population of solutions, $I^0 = \{I_1^0, \dots, I_N^0\}$
- Calculate fitness value $\Phi : I_i^0 \rightarrow \mathbb{R}, \forall i \in \{1, \dots, N\}$.

Step 2 – Perform Evolutionary Loop:

- Perform Selection: $P(\text{gen}) = \{S(I^{\text{gen}})_1, \dots, S(I^{\text{gen}})_k\}$, where S is selection operator and k is the size of mating pool
- Perform Recombination: $P'(\text{gen}) = \{R(P(\text{gen}))\}$, where R is recombination operator
- Calculate fitness value $\Phi : P'(\text{gen}) \rightarrow \mathbb{R}$.
- Perform Mutation: $P''(\text{gen}) = \{M(P'(\text{gen}))\}$, where M is mutation operator
- Calculate fitness value $\Phi : P''(\text{gen}) \rightarrow \mathbb{R}$.
- Perform Elitist selection: $I^{\text{gen}+1} = \text{the best } N \text{ individuals from the set } \{P \cup P' \cup P''\}$

Step 3 – Test Stopping criteria:

- if $(\tau \{\text{gen}\}) = \text{true}$, then stop and the output is the best solution in the last generation, otherwise $\text{gen} = \text{gen} + 1$, go to Step 2

End.

The conceptual visualization of the proposed evolutionary task allocation protocols is shown in Figure 4.3.

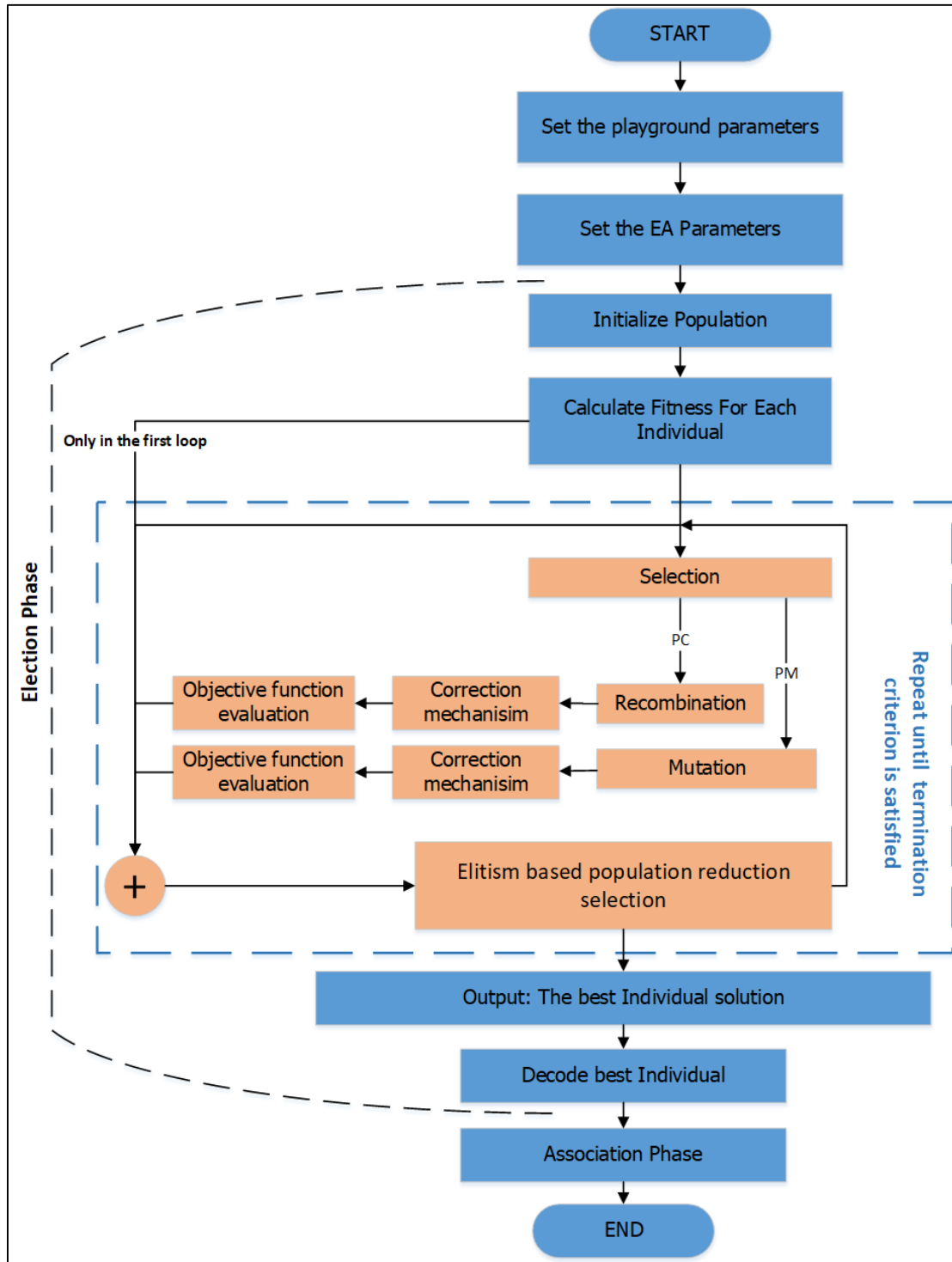


Figure 4.3. The conceptual visualization of the proposed evolutionary task allocation protocols in scenario #1

Thus, the conceptual evolutionary based task allocation protocols can be described informally as follows:

1. Set the playground parameters: Include network dimensions (X_{max}, Y_{max}), Access point location ($AP_{x,y}$) (i.e., $AP_{x,y} = (X_{max}/2, Y_{max}/2)$), Number of objects (n), Coordinates of objects ($O_{1,x,y}, \dots, (O_{n,x,y})$, Objects' energy ($E_{O_1}, \dots, (E_{O_n})$, Number of tasks that the network is capable of performing (T), Objects tasks ($t_{O_1}, \dots, (t_{O_n})$, Amplifier energy ϵ_{amp} , Transceiver circuit energy (E_{elec}), and Data aggregation energy (E_{DA}).
2. Set the evolutionary algorithm parameters: Include population size (N), Probability of recombination (p_c), Probability of mutation (p_m), Mutation rate (m_r) and Number of generations.
3. Virtual objects election phase:
 - a. Population Initialization: Generate an initial EA population.
 - b. Objective function evaluation: Calculate the fitness function of each individual in the population.
 - c. Evolutionary operators: Create a new population of individuals through:
 - Selections: Select individual mates from old population to form the mating pool using tournament selection.
 - Recombination: Make offspring between two randomly selected mates from the mating pool via two-point crossover operator.
 - Mutation: Mutate a ratio of $p_m \times N$ of mates from the mating pool.
 - Correction mechanism: Ensure the feasibility of the new offspring.
 - Evaluate: Calculate the fitness function of each individual in the new population.
 - d. Elitism: Selects the fittest N individuals from the set of parents and offspring.
 - e. Replace: Use the new population instead of the old one for further run of the virtual object election algorithm.
 - f. Stop: Test if the stopping condition of the EA is satisfied. If such, get the best individual found in the current population based on its fitness. If not, go to step c.
 - g. Decoding: Decode the genotype of the best individual into its phenotype.
4. Association phase: CCS associates each the non-virtual object members of each task group to their virtual objects to form clusters.

4.2.1. Virtual objects election phase

Selecting the appropriate collection of virtual objects at each task group has critical importance for energy efficient task allocation. During the virtual objects election phase, an EA is adopted to form the selection of virtual objects and partitions each task group into clusters accordingly. The process of EA begins by generating an initial population of candidate solutions. Each of these individuals is evaluated using a fitness function. Then, these individuals go through evolutionary genetic operators which include selection and variation operators (i.e., recombination and mutation). These evolutionary operators are applied with pre-determined probabilities for exploring new solutions and to improve the quality of the existing individuals. Each of the variation operators is then followed by a feasibility insurance mechanism. Finally, an elitism mechanism is applied to restrict the number of the individuals to the pre-determined population size. This evolutionary process is repeated until the termination condition is satisfied. Then, the solution with the best objective value is chosen as the final solution. The following subsections present each step of the virtual objects election phase in both informal and formal details.

Individual's genotype encoding/ phenotype decoding

EAs encode the decision variables (or input parameters) of the underlying problem into solution strings of individuals or chromosomes. Containers of the string are called *genes*. The position and the value in the string of a gene are called *locus* and *allele*, respectively. Each individual represents a complete and candidate solution to the given problem, encoded in a form that facilitates the evaluation of objectives and the execution of variation operators. This form represents the necessary information required by the algorithm and encoded as the individual's *genotype*. Genotype denotes the coding of the variables from the problem space into the solution space. In contrast, *phenotype* represents the decoding of genotype (i.e., the variables themselves).

In the context of EAs, the proposed evolutionary task allocation protocols consider a population of individuals. Let P be a set of population individuals. That is $|P| = N$ (N is population size). In our algorithms each individual $I_i \in P, \forall i \in (1, \dots, N)$ is regarded as a fixed length vector of size equal to the total number of objects in the IoT network. Thus, for an IoT network with n objects, the individual consists of n genes. In these individuals

each object is mapped into a gene. A gene value (allele) can be either 1 or 0 and it indicates whether the object represented by this gene is a virtual object or not, respectively. In this manner, the genotype representation of population $P = I^N = \{I^1, \dots, I^N\}$ of N individual solutions can be formally specified as follows:

$$\forall i \in \{1, \dots, N\}, \forall j \in \{1, \dots, n\} \text{ and } \forall t \in \{1, \dots, T\}$$

$$I_{j,t}^i = \begin{cases} 1 & \text{if object } j \text{ is a virtual object} \\ 0 & \text{otherwise} \end{cases} \quad (4.3)$$

where T is the number of task groups in the network (i.e., T is the number of tasks that can be performed in the network). For example, $I_{1,3}^2$ means the first gene (i.e., object) of the second individual of the population belongs to task group 3.

This representation implicitly facilitates the formation of the selection of virtual objects at each task group. Gene j represents the corresponding objects' identification numbers (IDs). Each ID can be used to retrieve the corresponding object's information such as object's x, y coordinates, energy level and the task that is capable of performing.

While the genotype has a particular set of genes in the genome representation, the phenotype, on the other hand, has the physical characteristic of the corresponding genotype. So we need a decoding operator to map the individual from the genotype representation in the problem space to a real solution in the phenotype space. Formally, a gene j of individual i in the proposed evolutionary task allocation protocols is decoded using the following equation:

$$\forall i \in \{1, \dots, N\}, \forall j \in \{1, \dots, n\} \text{ and } \forall t \in \{1, \dots, T\}$$

$$Object_j = \begin{cases} \text{virtual_object} & \text{if Gene}_j^i = 1 \\ \text{non_virtual_object} & \text{if Gene}_j^i = 0 \end{cases} \quad (4.4)$$

Figure 4.4 depicts genotype and phenotype examples for an IoT network capable of performing 4 tasks and consists of 250 objects from the initial population. Figure 4.4 (a), represents the genotype solution corresponded to the phenotype in Figure 4.4 (b). The

phenotype task allocation solution clarifies the form of: Access point, virtual objects of each Task Group (TG), and member objects (non-virtual objects) in each task group.

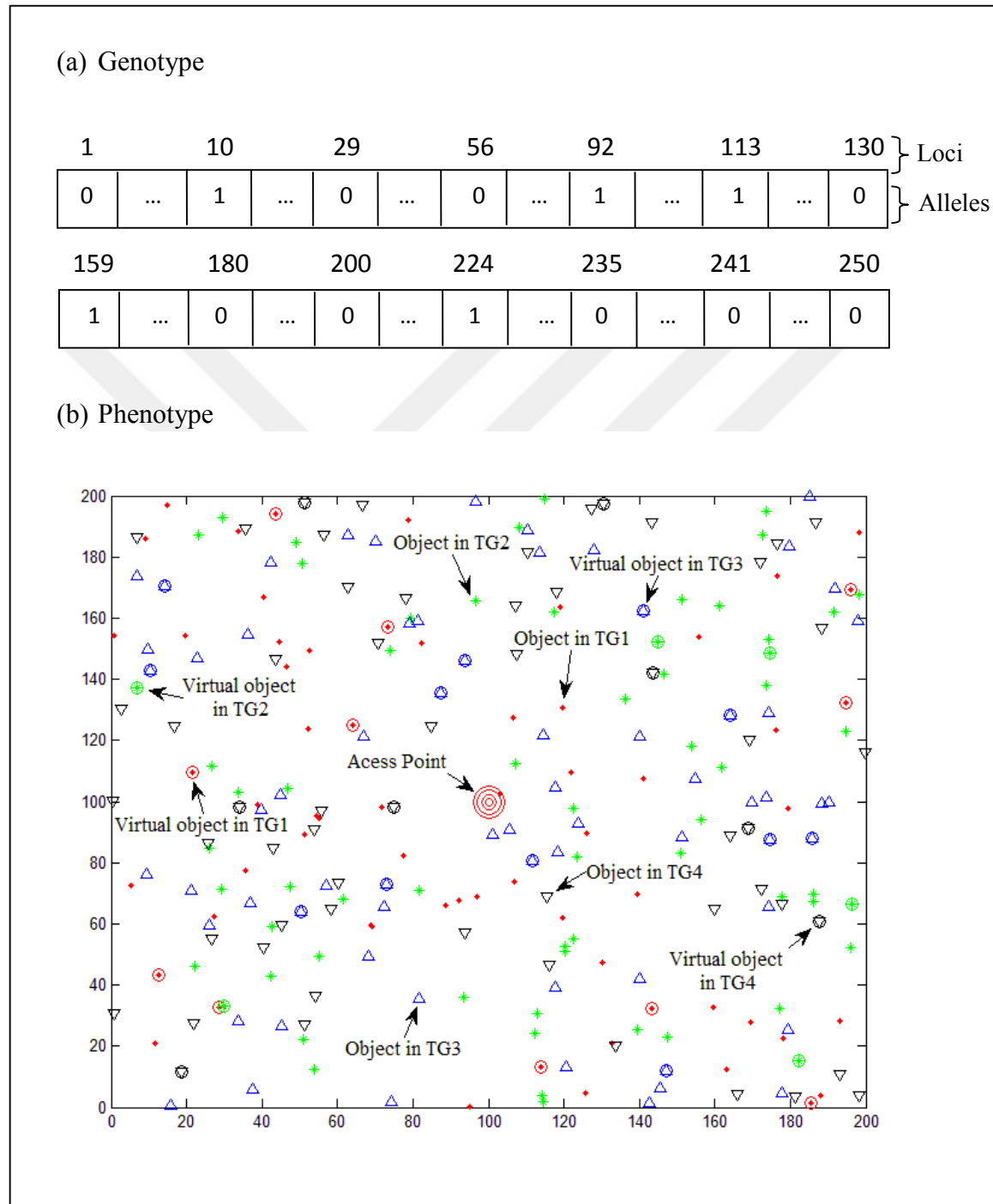


Figure 4.4. Examples for genotype (a) and phenotype (b) individuals of evolutionary task allocation protocols in scenario #1

Population initialization

In initialization phase, techniques based on randomness are usually used to generate the initial population. In our evolutionary task allocation protocol, in order to maintain more genetic diversity in initial population, we applied two methods for initialization as shown in Algorithm 4.2.

Algorithm 4.2. Initial population generation algorithm for protocols in scenario #1

Input:

- IoT network = $\{O_{i,t}, \dots, O_{n,t}\}$, where $t \in \{1, \dots, T\}$
- Population size, N

Output:

- Initial population, $I^N(0)$

Begin:

Step 1 –Setup:

- $I(0) = \theta$
- Set $PC = 1$

Step 2 – Operation:

- if $\text{random}_{PC} \leq 0.5$

Then

- 2.1: Set $TC = 1$
- 2.2: Let TG be a set of all objects belong to the task group TC , $TG = \{\forall O_{i,t} \in \text{IoT network}, t = TC\}$
- 2.3: Let $O_{u,t}$ be a random object belongs to TG
- 2.4: Set $I_{u,TC}^{PC} = 1$
- 2.5: $\forall O_{v,t} \in TG, v \neq u$ set $I_{v,TC}^{PC} = 0$
- 2.6: if $TC \neq T$, then $TC = TC + 1$ go to step 2.2, otherwise go to step 3

Else

- 2.7: Set $TC = 1$
- 2.8: Let TG be a set of all objects belong to the task group TC , $TG = \{\forall O_{i,t} \in \text{IoT network}, t = TC\}$
- 2.9: Obtain, $K_{\text{opt}} = \sqrt{\frac{|TG|}{2\pi}} \frac{2}{0.765}$ and $P_{\text{opt}} = \frac{K_{\text{opt}}}{|TG|}$

Algorithm 4.2 (Continue). Initial population generation algorithm for protocols in scenario #1

- 2.10: $\forall j \in TG$
 - if $\text{random}_{j,TC} \leq P_{opt}$ then $I_{j,TC}^{PC} = 1$
 - if $\text{random}_{j,TC} > P_{opt}$ then $I_{j,TC}^{PC} = 0$
- 2.11: if $TC \neq T$, then $TC = TC + 1$ go to step 2.8

Step 3 – Perform Concatenation:

- $I(0) = I(0) \cup I^{PC}$

Step 4 – Test Stopping condition:

- If $PC = N$, then stop and output is $I(0)$, otherwise $PC = PC + 1$, go to Step 2

End.

Each of these methods is applied with the probability of 0,5. Thus, each method creates the half of the individuals in the initial population. In the first method, an object in each task group is chosen randomly to become a virtual object within the corresponding task group. In the second method, in each task group each object becomes a virtual object with the probability of P_{opt} , where P_{opt} is the optimal election probability used in [167]. P_{opt} is calculated as:

$$P_{opt} = \frac{K_{opt}}{l} \quad (4.5)$$

where l is the number of objects in a task group and K_{opt} is the desired percentage of virtual objects in this task group. The calculation of K_{opt} is derived from [167] and [168]. To calculate K_{opt} , we assume that objects of each task group are uniformly distributed in an area $\mathbb{A} = \mathbb{M} \times \mathbb{M}$. For simplicity, we ignore the energy dissipation for transmitting and reserving IMs. Thus, for assigning a task and reserving the results, the energy dissipated in a virtual object is given by the following formula:

$$E_{VO} = \left(\frac{l}{C} - 1\right) \times k \times E_{elec} + \frac{l}{C} \times k \times E_{DA} + k \times \varepsilon_{amp} \times d_{to \mathbb{AP}}^2 \quad (4.6)$$

where C is the number of clusters (i.e., the number of virtual objects) in a task group, k is the number of bits to be sent or received, E_{DA} is the energy dissipated for data aggregation

and $d_{to \mathbb{AP}}$ is the average distance between a virtual object and the access point. The energy spent in a non-virtual object entity is equal to:

$$E_O = k \times E_{elec} + k \times \varepsilon_{amp} \times d_{to VO}^2 \quad (4.7)$$

In Equation (4.7), $d_{to VO}$ is average distance between a cluster member and its virtual object. The area occupied by each cluster is approximately $\frac{\mathbb{M}^2}{2}$, by assuming that this area is a circle with the radius $R = \frac{\mathbb{M}}{\sqrt{\pi C}}$. Considering the uniform distribution of objects, it can be shown that:

$$d_{to VO}^2 = \int_{x=0}^{x=\mathbb{X}_{\max}} \int_{y=0}^{y=\mathbb{Y}_{\max}} (x^2 + y^2) p(x, y) dx dy = \frac{\mathbb{M}^2}{2\pi C} \quad (4.8)$$

where $p(x, y)$ is object distribution. The energy spent in a virtual object cluster during a task execution for communication and aggregation (ignoring the nature of the task itself) is given by:

$$E_{VO-cluster} \approx E_{VO} + \frac{l}{C} \times E_O \quad (4.9)$$

The total energy dissipated in a task group is equal to:

$$E_{TG} = k \left(2lE_{elec} + lE_{DA} + \varepsilon_{amp}(Cd_{to \mathbb{AP}}^2 + ld_{to VO}^2) \right) \quad (4.10)$$

By differentiating E_{TG} and setting its derivative with respect to C to zero, we can obtain the optimal number of virtual objects (i.e., optimal number of clusters) as:

$$K_{opt} = \sqrt{\frac{l}{2\pi}} \times \frac{\mathbb{M}}{d_{to \mathbb{AP}}} \quad (4.11)$$

The average distance from a virtual object to the access point is given by [169]:

$$d_{to \mathbb{AP}} = \int_{\mathbb{A}} \sqrt{x^2 + y^2} \frac{1}{\mathbb{A}} d\mathbb{A} = 0,765 \frac{\mathbb{M}}{2} \quad (4.12)$$

In this way, the desired percentage of virtual objects in a task group (K_{opt}) becomes:

$$K_{opt} = \sqrt{\frac{l}{2\pi}} \frac{2}{0,765} \quad (4.13)$$

Each gene $j \in \{1, \dots, n\}$ in each task group $t \in \{1, \dots, T\}$ belongs to each individual, $I^i, \forall i \in \{1, \dots, N\}$, is randomly initialized with 1s and 0s according to P_{opt} of the desired percentage of virtual objects as:

$$\forall i \in \{1, \dots, N\}, \forall j \in \{1, \dots, n\} \text{ and } \forall t \in \{1, \dots, T\}$$

$$I_{j,t}^i = \begin{cases} 1 & \text{if } random_{j,t} \leq P_{opt} \\ 0 & \text{if } random_{j,t} > P_{opt} \end{cases} \quad (4.14)$$

Fitness evaluations

To quantify the quality of each individual, a fitness value measured by an objective function Φ is assigned to each solution. An objective function measures how good an individual solution is for task allocation problem by differentiating between good and bad solutions. The objective function accomplishes this by interpreting the individual in terms of physical representation and evaluates its fitness based on desired traits (in the solution). The fitness function forms the bridge between the task allocation problem itself and the evolutionary algorithm.

All objective functions used in evolutionary task allocation protocols of scenario #1 represent minimization problems, where the most fitted individuals will have the lowest numerical value of the associated function. The following describes the fitness parameters used to form the fitness function in Evolutionary Task Allocation Protocol-1 (ETAP1) and Evolutionary Task Allocation Protocol-2 (ETAP2):

- Evolutionary Task Allocation Protocol-1 (ETAP1): The goal of ETAP1 objective function Φ_{ETAP1} is to minimize the total energy dissipated in the network for communication in the process of task allocation. It achieves this goal by forming the selection of virtual objects in a manner that minimizes the energy consumption at each

task group. In our protocols, virtual objects are responsible for assigning tasks to their connected objects. The total energy spent for this operation represents the energy consumed for task allocation. Φ_{ETAP1} , quantifies this operation by summing: the total energy spent by virtual objects to send IMs to their connected objects, the total energy consumed by these objects to receive the IMs, and the total dissipated energy by virtual objects to receive, aggregate and send the aggregated messages to AP. It is clear that the overall energy dissipated in all task groups represents the total energy dissipated in the network. Formally speaking, the objective function used to evaluate the individual solutions in ETAP1 protocol becomes:

$$\Phi_{ETAP1} = \sum_{t=1}^T \left(\left(\sum_{v=1}^{|VO_t|} \sum_{j=1}^{|O_{v_t}|} IM_E_{TX_{v,j}} + IM_E_{RX} + E_{TX_{j,v}} + E_{RX} + E_{DA} \right) + \sum_{v=1}^{|VO_t|} E_{TX_{v,AP}} \right) \quad (4.15)$$

where T is the number of the tasks that the IoT network can perform (i.e., the number of task groups), $|VO_t|$ is the number of virtual objects at the t th task group, $|O_{v_t}|$ is the number of objects associated to the v th virtual object of the t th task group and $IM_E_{TX_{object_1, object_2}}$ and $E_{TX_{object_1, object_2}}$ are energy consumptions for transmitting indication messages and result messages from $object_1$ to $object_2$, respectively. IM_E_{RX} and E_{RX} determine the energy spent for receiving IM by IoT objects and the energy spent for receiving data (or result) messages by virtual objects. E_{DA} is energy dissipated for data aggregating.

- Evolutionary Task Allocation Protocol-2 (ETAP2): From Equation 1, it is deduced that the consumed energy in a transmission depends on the distance between the source and the destination. That is the longer the transmission distance is, the larger the dissipated energy will be. Based on this fact, we formulate the objective function of ETAP2 as Φ_{ETAP2} . Φ_{ETAP2} is composed of two parts: compactness (i.e., intra-distance) and separation (i.e., inter-distance). The compactness part aims to minimize the distances between the virtual objects and their connected objects to reduce the energy consumption when virtual objects send indication messages to their connected objects or when these objects transmit data to their associated virtual objects. On the other hand, separation part aims to maximize the distances among virtual objects of each task

group to form a good distribution of virtual objects within each task group. The compactness part of Φ_{ETAP2} is the sum of all distances between the network objects and their associated virtual objects whereas the separation part is the total distances among virtual objects of each task group. For normalization purposes both compactness and separation part are divided by the number of the tasks that the IoT network can perform. In this way Φ_{ETAP2} can be quantified by:

$$\Phi_{ETAP2} = \frac{(compactness \setminus T)}{(separation \setminus T)} = \frac{\left(\sum_{t=1}^T \left(\sum_{v=1}^{|VO_t|} \sum_{j=1}^{|O_{v_t}|} d(Vobject_v, Object_j) \right) \right) \setminus T}{\left(\sum_{t=1}^T \left(\sum_{v=1}^{|VO_t|} \sum_{u=1}^{|VO_t|} \forall v \neq u, d(Vobject_v, Vobject_u) \right) \right) \setminus T} \quad (4.16)$$

where T is the number of the tasks that the IoT network can perform (i.e., the number of task groups), $|VO_t|$ is the number of virtual objects at the t th task group, and $|O_{v_t}|$ is the number of objects associated to the v th virtual object of the t th task group. $Vobject_v$ and $Object_j$ are the v th virtual object of the t th task group and the j th object associated with that virtual object, respectively. $d(Vobject_v, Object_j)$ and $d(Vobject_v, Vobject_u)$ are the Euclidean distances. The Euclidean distances for $d(Vobject_v, Object_j)$ can be calculated as $\sqrt{(x_{Vobject_v} - x_{Object_j})^2 + (y_{Vobject_v} - y_{Object_j})^2}$. Finally, it is worth to note that the denominator in Equation 15 needs to be maximized while the numerator needs to be minimized. As a result, the objective function is formulated as minimization problem. It is also worth to mention that parameter T could be excluded from the formula and it is included only for normalization and analytical purposes.

The evolutionary process

In the evolutionary process, selection operator provides exploitation (i.e., intensification), compared with the exploration for new and better solutions (i.e., diversification) which provided by recombination and mutation operators. In the following each of the evolutionary operators that is used in evolutionary task allocation protocols of scenario #1 is illustrated:

- Selection operator: In the context of EA, the next component is selection operator. Selection acts as a force to increase the mean quality of solutions in the population. It

preserves the good solutions by electing parents with superior fitness values and transferring them to the mating pool for reproduction in the next generation. In our proposed protocols we adopted Binary Tournament Selection (BTS) as selection operator since it is simple and effective [23]. BTS selects the best individual based on its fitness value from two randomly selected individuals of the population and repeats this operation N times to create the mating pool of the size equal to the size of population. The formal definition of BTS, $S_{BTS}: I^{\wedge 2} \rightarrow I'$, is as follows: Let $r_1, r_2 \sim U\{1, \dots, N\}$ be two uniformly distributed random numbers from the set $\{1, \dots, N\}$ and let I^{r_1}, I^{r_2} be two individual solutions. Then, our selection operator becomes as follows:

$$I^{i'} \forall i \in \{1, \dots, N\} = \begin{cases} I^{r_1} & \text{if } \Phi(I^{r_1}) \leq \Phi(I^{r_2}) \\ I^{r_2} & \text{otherwise.} \end{cases} \quad (4.17)$$

- **Recombination:** In recombination a cut and cross fill based method is applied. In this method a ratio of $(p_c \times \frac{N}{2})$, where p_c is probability of recombination) of pair of parents are chosen randomly from mating pool for recombination. Then, two cut points are randomly chosen in each task group and the segment of genes between these cut points of the two parent individuals are exchanged with each other. Formally, let r_1 and r_2 be two random numbers selected from the range $\{1, \dots, N\}$ and let $I^{r_1'}, I^{r_2'}$ be two individuals randomly selected based on r_1 and r_2 from the mating pool. For each task group $TG_t, \forall t \in \{1, \dots, T\}$ we generate two random cut points c_1 and c_2 in the range $\{1, \dots, |TG_t| - 1\}$, where $|TG_t|$ is t th task group size. Then, the recombination operation for each task group can be represented as follows:

$$\begin{aligned} I_{i_t, t}^{r''} &= (I_{1_t, t}^{r'_1}, \dots, I_{c_1, t}^{r'_1}, I_{c_1+1, t}^{r'_2}, \dots, I_{c_2-1, t}^{r'_2}, I_{c_2, t}^{r'_1}, \dots, I_{|TG_t|, t}^{r'_1}) \\ I_{i_t, t}^{r''} &= (I_{1_t, t}^{r'_2}, \dots, I_{c_1, t}^{r'_2}, I_{c_1+1, t}^{r'_1}, \dots, I_{c_2-1, t}^{r'_1}, I_{c_2, t}^{r'_2}, \dots, I_{|TG_t|, t}^{r'_2}) \end{aligned} \quad (4.18)$$

where $i_t \forall i \in \{1, \dots, |TG_t|\}$ is a gene that belongs to the t th task group. This operation is iterated $\forall t \in \{1, \dots, T\}$. Figure 4.5 depicts an example of two-point cut and cross fill based recombination. The figure captured for 250 object and network with 4 task groups.

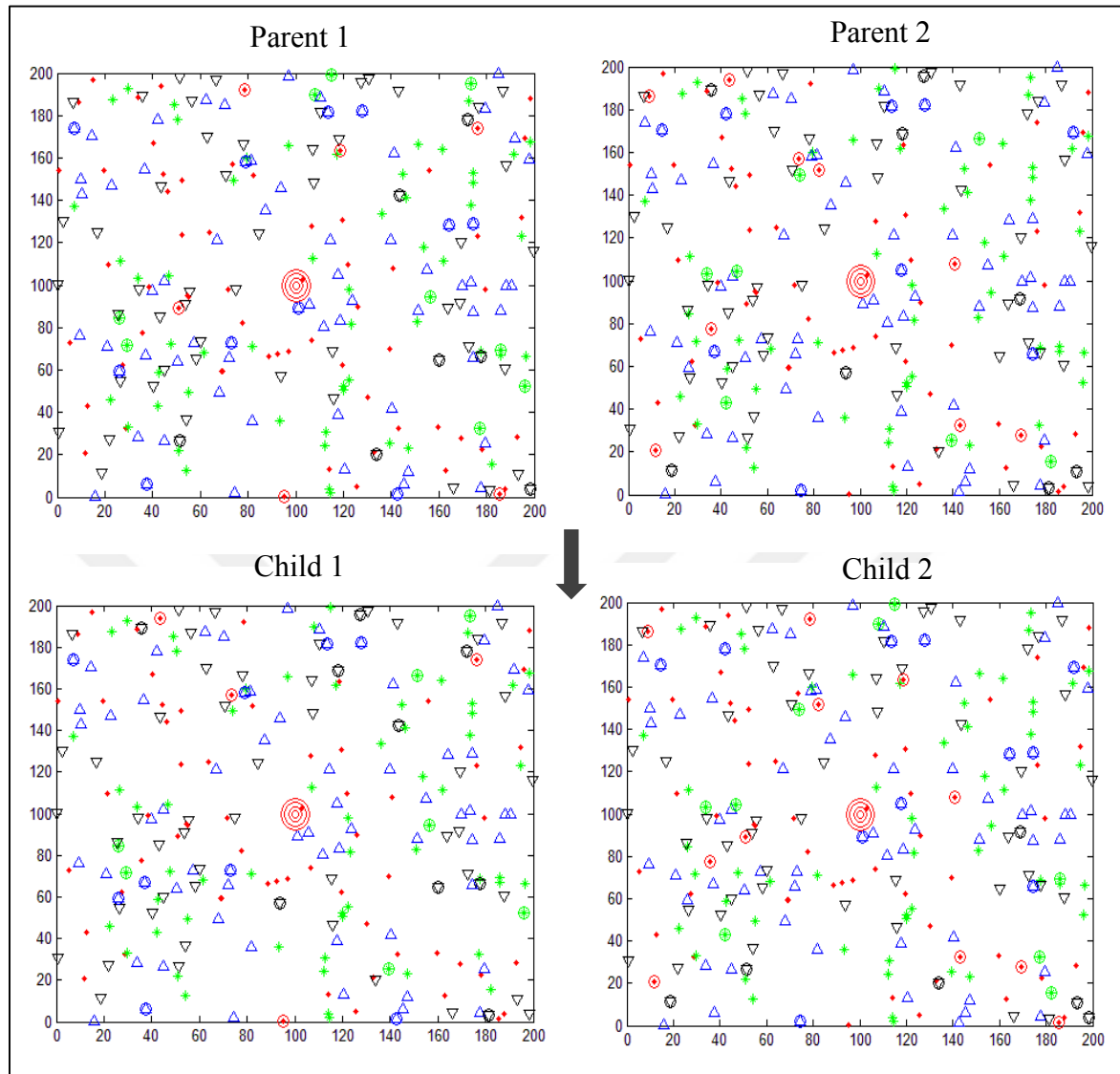


Figure 4.5. Two-point cut and cross fill

- **Mutation:** The next variation operator is mutation. A ratio of $p_m \times N$ (where p_m is probability of mutation) of population individuals are undergone the mutation operation. For mutation operator a Bit Flop Mutation (BFM) method is adopted. BFM is a unary variation operator applied to each gene in each task group with mutation rate of m_r . Once a gene is chosen for mutation, its value is flipped from 1 to 0 and vice versa as follows:

$$\forall t \in \{1, \dots, T\}$$

$$I_{i_t,t}^{r'''} = \begin{cases} I_{i_t,t}^{r'} & \text{if } \text{random} > m_r \\ 1 - I_{i_t,t}^{r'} & \text{otherwise} \end{cases} \quad (4.19)$$

In Equation (4.19), r is a random number in range $\{1, \dots, N\}$, $I^{r'}$ is an individual chosen randomly based on r from the mating pool and finally, $i_t \forall i \in \{1, \dots, |TG_t|\}$ is a gene belongs to the t th task group. Again, the operation is repeated for each task group. Figure 4.6 depicts an example of mutation. In this figure the network consists of 250 objects and capable of performing 4 tasks (i.e., there are 4 task groups in the network).

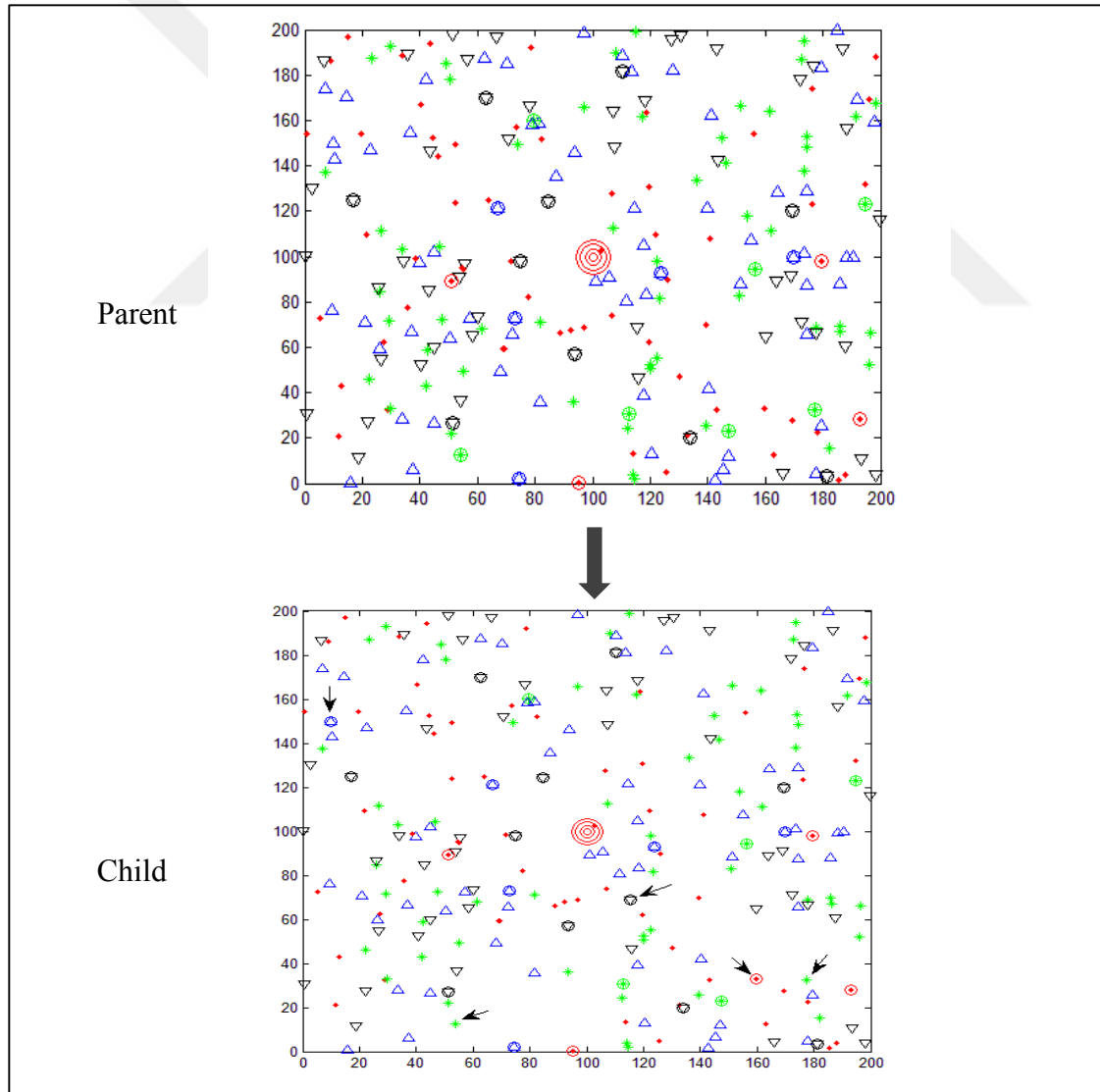


Figure 4.6. An example of mutation

- Correction mechanism: In order to guarantee the validity of the new solutions both recombination and mutation operators are followed by a feasibility insurance mechanism that tests and repairs the infeasible individual solutions. This mechanism tests each task group and randomly selects a virtual object for task groups that become without virtual objects due to execution of variation operators.
- Elitism selection: The execution of variation operators creates new sets of candidate solutions (offspring individuals) from the already existing solutions in mating pool (parent individuals). Therefore, an elitism scheme is needed to restrict the number of individuals in the population to N individuals (hence, N is the population size). In our protocols, an elitism mechanism based on fitness values is applied. That is, parents and offspring individual sets are concatenated and sorted in ascending order and then only the first N individuals are truncated and transferred to the next generation. In this way we retained the best individuals (or a few best individuals) in the next generation and prevented the loss of the fittest members of the population.

Termination criterion

The most common terminating condition used in evolution algorithms is to allow the algorithm to run to a maximum number of generations. In our protocols, the algorithm terminates when the specified number of generations is reached. In the last generation the solution with the best fitness value (minimum value) is selected to ensure that the transmission power required by each object in the process of task allocation is minimized and hence the total energy dissipation in the network decreases.

Finally, the selected best individual solution that obtained by the evolutionary algorithm undergo through the decoding process. In the decoding process the genotype of the best individual is decoded to obtain the phenotype result. The best individual solution $Best_I$ can be formally specified as:

$$\nexists I_i \in P, \forall i \in (1, \dots, N): \Phi(I_i) < \Phi(Best_I) \quad (4.20)$$

where, P is the population set, N is population size, and Φ is fitness function.

4.2.2. Association phase

In this phase, connections are established among the nodes based on the roles that were assigned during the first phase (i.e., virtual objects election phase). The CCS determines for each non virtual object entity to which virtual object it belongs by choosing the virtual object that requires the minimum communication energy at each task group. Based on Euclidean distances, the virtual object with minimum distance will be chosen in order to achieve minimum energy consumption.

Formally speaking, let $|VO_{t-best}|$ be the number of virtual objects elected by the best individual, Best_I in the task group t , and let O_t be a non-virtual object member in task group t , then:

$$\forall i, j \in \{1, \dots, |VO_{t-best}|\}$$

$$O_t \in C_i, d(O_t, Virtual_Object_i) < d(O_t, Virtual_Object_j) \quad (4.21)$$

where, C_i is cluster consists of $Virtual_Object_i$ and all its connected objects, d is the Euclidean distance.

4.2.3. Toward stability aware protocols

In an attempt to harness the strength of the proposed evolutionary protocols toward stabilization purpose, we exploit the variation in energy levels of objects and redirected the proposed ETAP1 and ETAP2 toward stability awareness goal. The so-called Stable Evolutionary Task Allocation Protocol-1 (SETAP1) and Stable Evolutionary Task Allocation Protocol-2 (SETAP2) are derived from ETAP1 and ETAP2 with the goal of maximizing the time interval before the death of the first virtual object in each task group. The basic idea of SETAP1 and SETAP2 is to inject an energy aware heuristic in both population initialization and mutation operator of ETAP1 and ETAP2 to be collaborated with the proposed fitness functions in order to maintain a robust performance.

In population initialization we modified the initialization phase previously described in Algorithm 4.2. In the modified algorithm shown in Algorithm 4.3, only objects with energy level greater than (or at least equal to) the average energy level in each task group are allowed to become a virtual object in that task group. Accordingly, in the first method of initialization phase, in each task group a random object with energy level greater or equal to the average energy level in the corresponding task group is chosen to become a virtual object. In the second method, each gene $j \in \{1, \dots, n\}$ in each task group $t \in \{1, \dots, T\}$ belongs to each individual, $I^i, \forall i \in \{1, \dots, N\}$, can be initialized with 1s and 0s as follows:

$\forall i \in \{1, \dots, N\}, \forall j \in \{1, \dots, n\}$ and $\forall t \in \{1, \dots, T\}$

$$I_{j,t}^i = \begin{cases} 1 & \text{if } \text{random}_{j,t} \leq P_{opt} \wedge \text{Energy}(\text{object}_{j,t}) \geq E_{avg_t} \\ 0 & \text{otherwise} \end{cases} \quad (4.22)$$

where, P_{opt} is the desired percentage of the virtual objects in each task group defined in Equation (4.5) and E_{avg_t} is average energy level in the t th task group.

Algorithm 4.3. Population initialization algorithm in SETAP1 and SETAP2

Input:

- IoT network = $\{O_{i,t}, \dots, O_{n,t}\}, \forall t \in \{1, \dots, T\}$
- Population size, N
- A set of average energy levels for each task group, $E_{avg_t}, \forall t \in \{1, \dots, T\}$

Output:

- Initial population, $I^N(0)$

Begin:

Step 1 – Setup:

- $I(0) = \theta$
- Set $PC = 1$

Step 2 – Operation:

- if $\text{random}_{PC} \leq 0.5$

Then

- 2.1: Set $TC = 1$

Algorithm 4.3 (Continue). Population initialization algorithm in SETAP1 and SETAP2

- 2.2: Let TG be a set of all objects belong to the task group TC ,TG = $\{\forall O_{i,t} \in \text{IoT network}, t = \text{TC}\}$
- 2.3: Let $O_{u,t}$ be a random object belongs to TG
- 2.4: if $\text{Energy}(\text{object}_{u,t}) < E_{\text{avgTC}}$ then go to Step 2.3
- 2.5: Set $I_{u,\text{TC}}^{\text{PC}} = 1$
- 2.6: $\forall O_{v,t} \in \text{TG}, v \neq u$ set $I_{v,\text{TC}}^{\text{PC}} = 0$
- 2.7: if $\text{TC} \neq T$, then $\text{TC} = \text{TC} + 1$ go to step 2.2, otherwise go to step 3
- Else
- 2.8: Set $\text{TC} = 1$
- 2.9: Let TG be a set of all objects belong to the task group TC ,TG = $\{\forall O_{i,t} \in \text{IoT network}, t = \text{TC}\}$
- 2.10: Obtain, $K_{\text{opt}} = \sqrt{\frac{|\text{TG}|}{2\pi}} \frac{2}{0.765}$ and $P_{\text{opt}} = \frac{K_{\text{opt}}}{|\text{TG}|}$
- 2.11: $\forall j \in \text{TG}$
- if $\text{random}_{j,\text{TC}} \leq P_{\text{opt}} \wedge \text{Energy}(\text{object}_{j,\text{TC}}) \geq E_{\text{avgTC}}$ then $I_{j,\text{TC}}^{\text{PC}} = 1$
- if $\text{random}_{j,\text{TC}} > P_{\text{opt}}$ then $I_{j,\text{TC}}^{\text{PC}} = 0$
- 2.12: if $\text{TC} \neq T$, then $\text{TC} = \text{TC} + 1$ go to step 2.9

Step 3 – Concatenation:

- $I(0) = I(0) \cup I^{\text{PC}}$

Step 4 – Stopping condition:

- if $\text{PC} = N$, then stop and output is $I(0)$, otherwise $\text{PC} = \text{PC} + 1$, go to Step 2

End.

Associated with each individual is a fitness value measured by an objective function, Φ , which numerically quantifies how good that individual is for the task allocation problem. For SETAP1 and SETAP2, the objective functions are as defined previously in Equations (4.15) and (4.16), respectively.

Both selection and recombination operators are also as defined previously in Equation (4.17) and Equation (4.18), respectively. The mutation operator, on the other hand, has an additional heuristic. Each gene in each task group of the individual selected for mutation is

mutated by applying BFM with mutation rate of m_r . Here, in this heuristic only objects with above average energy level are considered under mutation. Once a gene is chosen for mutation, its value is inverted from 1 to 0 and vice versa:

$\forall r \in \{1, \dots, N\}, i_t \forall i \in \{1, \dots, |TG_t|\}$ and $\forall t \in \{1, \dots, T\}$

$$I_{i,t}''' = \begin{cases} 1 - I_{i,t}^{r'} & \text{if } Energy(object_{i,t}) \geq E_{avg_t} \wedge random \leq m_r \\ I_{i,t}^{r'} & \text{otherwise} \end{cases} \quad (4.23)$$

In the Equation (4.23), TG_t is a set of all objects that belong to task group t , $\forall t \in \{1, \dots, T\}$ and $i_t \forall i \in \{1, \dots, |TG_t|\}$ is a gene belongs to the t th task group.

4.3. Overview of the Proposed Protocols in Context of Scenario #2

This scenario considers the applications of IoT where objects are capable of performing several tasks. In a these applications, some of the task groups may intersect as the network may contain several objects with different skills belonging to several task groups. Each task group is assigned to a relevant task by a virtual object(s) which virtualize the physical objects connected to them in the cyberspace and they are in charge of processing the requests to these physical objects. To understand this scenario, assume that the network is performing a humidity and temperature sensing in a specific area. In this case, the network contains two task groups: humidity measurement task group, which contains only the objects that are equipped with humidity sensors, and temperature measurement task group, which includes the objects that are equipped with temperature sensors. It is obvious that these two task groups could intersect when some objects are equipped with both humidity and temperature sensors. Then, in each task group, some objects are selected as virtual objects. The rest of the task group objects are clustered and connected to one virtual object. At this point, allocating proper resources to the required task is the duty of the virtual objects. It is worth to mention that, objects that are participated in several task groups could be virtual objects in more than one task group. Figure 4.7 depicts the concept of task groups and virtual objects in scenario #2.

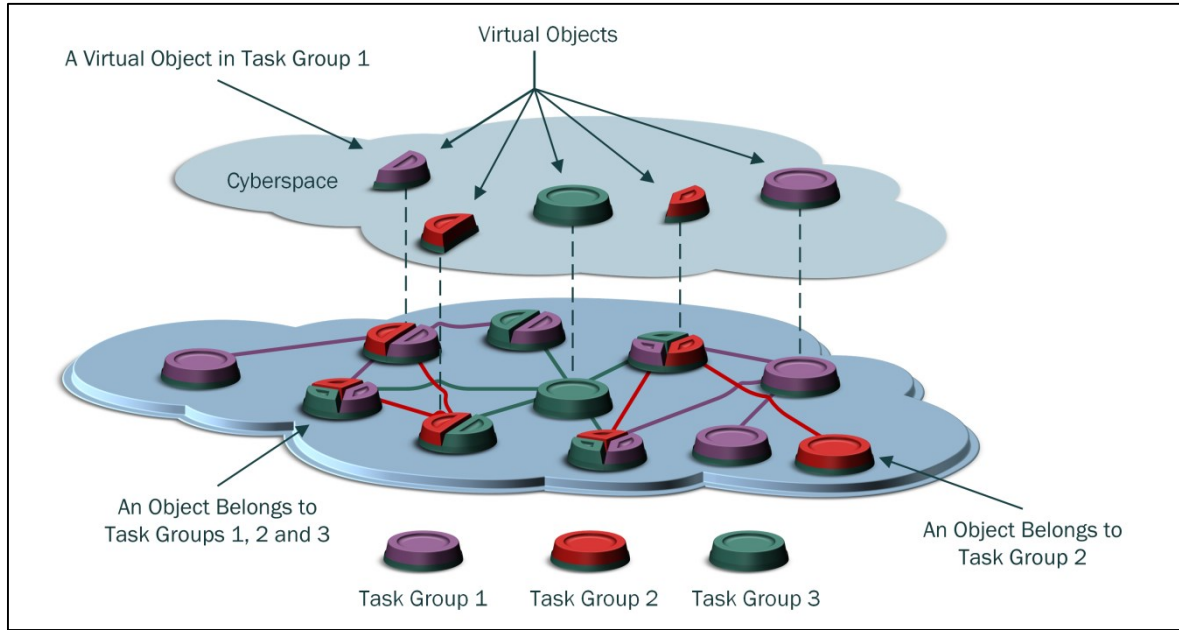


Figure 4.7. The concept of task groups and virtual objects in scenario #2

In the context of scenario #2, two protocols are developed and an existing work in the literature is modified to fit the energy efficiency requirement of task allocation problem. All of these protocols are located at a CCS responsible of determining the task allocation configuration in terms of the virtual objects and the connections in each task group. These protocols are Modified-CBATA (M-CBATA) which is the modified version of the existing work in [8] and [12], Steady Task Allocation Protocol (STAP), and Multi-Objective Task Allocation Protocol (MOTAP). Starting with M-CBATA, the next subsections present illustrations for each of these protocols.

4.3.1. Modified-CBATA (M-CBATA)

The authors in [8] and [12] proposed Consensus Based Approach for Task Allocation (CBATA) to solve the problem of resource allocation in IoT networks. CBATA is a heuristic approach that employs the concept of task groups and virtual objects to formulate the problem of task allocation. In this algorithm, the first object is selected as virtual object in each task group and the next object in the list is selected as vice-virtual object. Considering that the objects have different capabilities to perform variety of tasks, one of the main drawbacks of this method is the big probability of some objects to be a virtual object in many task groups. It is clear that being a virtual object is a very energy consuming operation and being a virtual object in many task groups increases this effect.

Therefore, objects that are virtual objects in many task groups quickly deplete their energies leading to very short stability and operational periods. As an example, let us consider the network in Figure 4.8 (a). The network is composed of two task groups (TG1 and TG2), where each task group has its virtual object assigned to it (i.e., VO1 assigned to TG1 and VO2 assigned to TG2). Now, assume an object i that is capable of performing tasks 1, 2, 3, and 4 is joined to the network. In this example, the virtual objects VO1 and VO2 are assigned to task groups TG1 and TG2, and no virtual object has been yet assigned to task groups 3 and 4. Thus, VO1 and VO2 add object i to their list of objects and acknowledge it. Obviously, object i does not receive acknowledgements for task groups 3 and 4. Hence, object i assumes the role of VO3 and VO4 and then it informs the CCS (see Figure 4.8 (b)).

In the previous example, object i undertook the duty of being virtual object in two task groups. The situation could be more severe if an object that is associated with large number of task groups that do not have virtual objects is joined to the network. Considering this drawback, we modified CBATA in such a way that, no object will be a virtual object in more than one task group if it is possible. We named the modified CBATA algorithm as Modified-CBATA (M-CBATA). The goal of M-CBATA is to minimize the number of objects that are virtual objects in more than one task group or at least reduce the number of task groups that the object is virtual object in them. In M-CBATA, when an object joins to the network it sequentially tests the following steps and performs one or more of them accordingly:

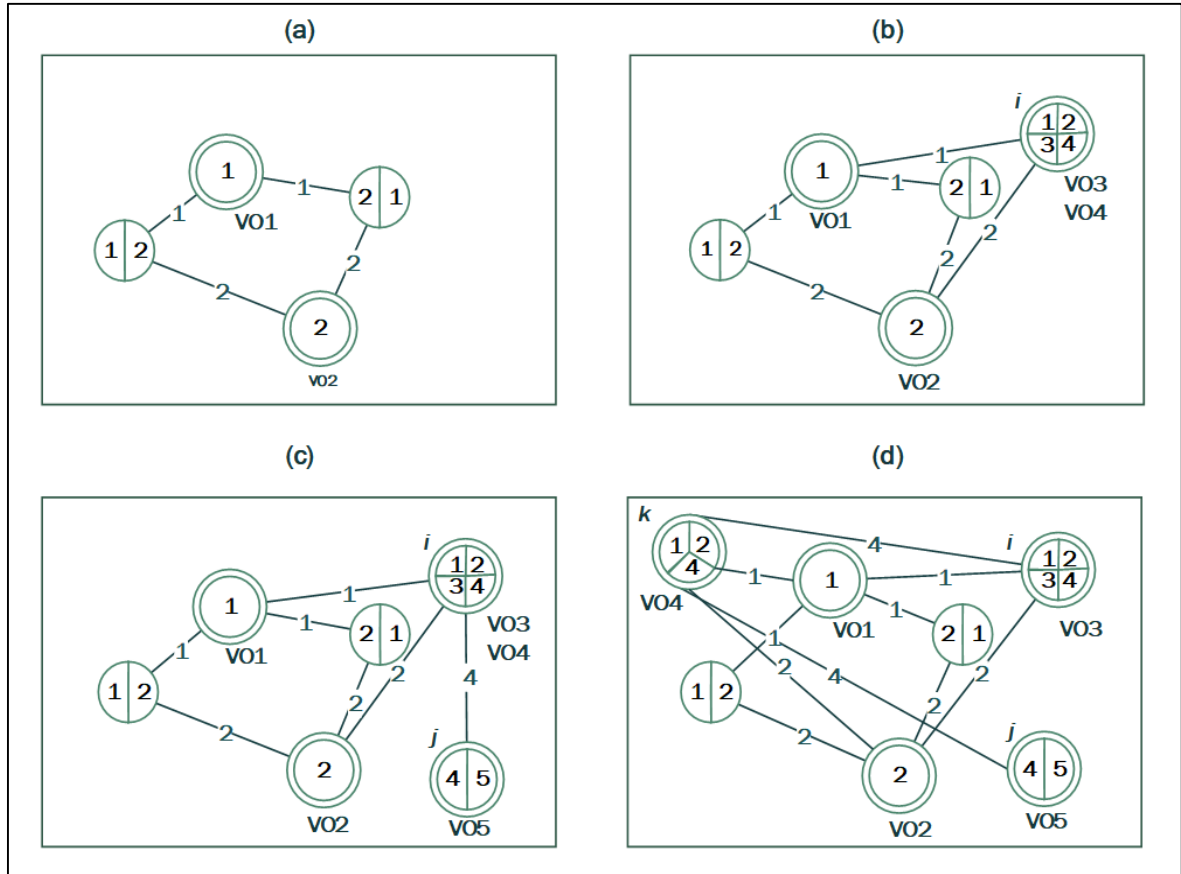


Figure 4.8. Illustrative example for CBATA and M-CBATA (a): A network with two task groups (b): Object i joined to the network (c): Object j joined to the network (d): Object k joined to the network

1. If the object is associated with one or more tasks that their task groups have no virtual objects yet, the object becomes a virtual object in these task groups.
2. If the object is associated with some tasks and their task groups have virtual objects that are associated with more than one task group then:
 - a. If the object already selected as virtual object in another task group, then it associates itself to these virtual objects.
 - b. If the object is not selected as a virtual object, then it informs one of the virtual objects (i.e., the virtual objects that are associated with more than one task group) to exclude itself from being a virtual object. The newly joined object announces itself as the new virtual object in the corresponding task group.
 - c. If the object is associated with task groups that have virtual objects and those virtual objects are associated with one task group, then the object joins to those virtual objects.

Illustrative Example

Let us reconsider the example in Figure 4.8 (a). When the object i wants to join the network (see Figure 4.8 (b)) it tests the network. Since task groups 3 and 4 have no virtual objects assigned to them, object i performs step 1 and becomes a virtual object in these task groups. Since task groups 1 and 2 have their virtual objects assigned to them and each of these virtual objects are assigned to one task group, object i performs step 3 and associate itself with VO1 and VO2. Now, let us assume the following scenarios:

- Suppose that an object j that can execute tasks 4 and 5 wants to join the network (see Figure 4.8 (c)). Since there is no virtual object assigned to task group 5 the object j performs step 1 and announces itself as VO5. On the other hand, although object i (i.e., VO4) is virtual object in more than one task group; object j follows step 2-a and joins to VO4 to perform task 4 and takes no action since object j has already been selected as VO5.
- Suppose that an object k that can perform tasks 1, 2 and 4 wants to join the network (see Figure 4.8 (d)). Since VO1 and VO2 both are objects which are virtual objects in only one task group, object k follows step 3 and associates itself with VO1 and VO2. Then, object k follows step 2-b and informs object i to exclude itself from being VO4 and object k announce itself as VO4 informing the list of objects belonging to task group 4. At this point, all virtual objects in the network are virtual objects in only one task group.

4.3.2. Steady task allocation protocol (STAP)

STAP aims to boost the energy efficiency of the task allocation in the direction of maximizing the stability period (the time interval until the first virtual object depletes its energy in each task group) as well as increasing the operational period (the interval until all virtual objects deplete their energies in each task group). In order to accomplish this goal we developed a single objective optimization algorithm with heterogeneity aware heuristics to ensure more extension of stability and operational periods. The proposed algorithm attempts to select objects with energy levels above the average energy level of a task group to be virtual objects at that task group and also attempts to reduce energy

consumptions of these virtual objects. It forms the collaboration between an efficient object function and heterogeneity aware heuristics injected to population initialization and mutation operators. The next subsections shade the light on the special design futures of system model and the algorithmic framework of STAP.

System model for STAP

STAP formulate the problem using the system model defined in Section 4.1. In addition, STAP assumes a heterogeneous IoT objects. These objects are heterogeneous in terms of their energy levels and the number and the type of tasks they perform (i.e., each object is capable of performing $t_o | t_o \leq T$ tasks).

The network in STAP is partitioned into T task groups with each contains the objects that are capable of performing similar tasks. At each task group some objects that satisfies specific design features are selected to be virtual objects. These virtual objects are responsible of representing the task group and allocating tasks to the objects connecting to them. Task allocation is done by the virtual objects by sending an Indication Messages (IMs) to their connected objects. As the objects can participate in multiple task groups it is important to attach Task Identification TID (i.e., $TID = t$) with each IM, here t represents the demanded task. Then, each object resaves the IM performs the task that is assigned in the TID and sends the results back to the virtual object. Virtual objects aggregate the data and send the aggregated results to \mathbb{AP} .

The algorithmic framework in STAP

STAP is centralized algorithm located at CCS that is equipped with unlimited resources. It needs to operate one time in order to forms the selection of virtual objects and partitions the network into clusters accordingly. As a single objective evolutionary optimization algorithm, the algorithmic framework of STAP uses the general underlying of the EAs which is presented in Figure 4.3.

STAP begins by considering a population of individuals P of size N (i.e., $|P| = N$). Each individual $I_i \in P, \forall i \in (1, \dots, N)$ is divided into a set of sub-individuals. Each sub-individual represents a task group and is expressed as a fixed length vector of size equal to

the number of objects (n). In this manner, each gene can be represented as $I_{i(j,k)}, \forall i \in (1, \dots, N), \forall j \in (1, \dots, T), \forall k \in (1, \dots, n)$ where T is the number of task groups in the network (i.e., the number of tasks that the network is capable of performing). Each gene encodes one object in the network in each task group and it is represented by an allele that can take a value of 1, 0 or -1 . The allele takes the value of 1 if the object is a virtual object and the value of 0 if the object is not a virtual object but a member of the task group. Finally, it takes the value of -1 if the corresponding object is not a member of the task group. Figure 4.9 illustrates an abstract view of a population. Accordingly, a population $P = \{I_1, I_2, \dots, I_N\}$, can be formalized as:

$$\forall i \in (1, \dots, N), \forall j \in (1, \dots, T), \forall k \in (1, \dots, n)$$

$$I_{i(j,k)} = \begin{cases} 1 & \text{if } Object_k \in j^{th} \text{ task group} \wedge Object_k \in VO_j \\ 0 & \text{if } Object_k \in j^{th} \text{ task group} \wedge Object_k \notin VO_j \\ -1 & \text{if } Object_k \notin j^{th} \text{ task group} \end{cases} \quad (4.24)$$

where, VO_j is the set of virtual objects in the j th task group. For example, $I_{2(4,1)} = 1$ means that the 1st gene (i.e., the first object in the network) of the 4th sub-individual (i.e., the fourth task group) of the 2nd individual of the population is a virtual object in the 4th task group.

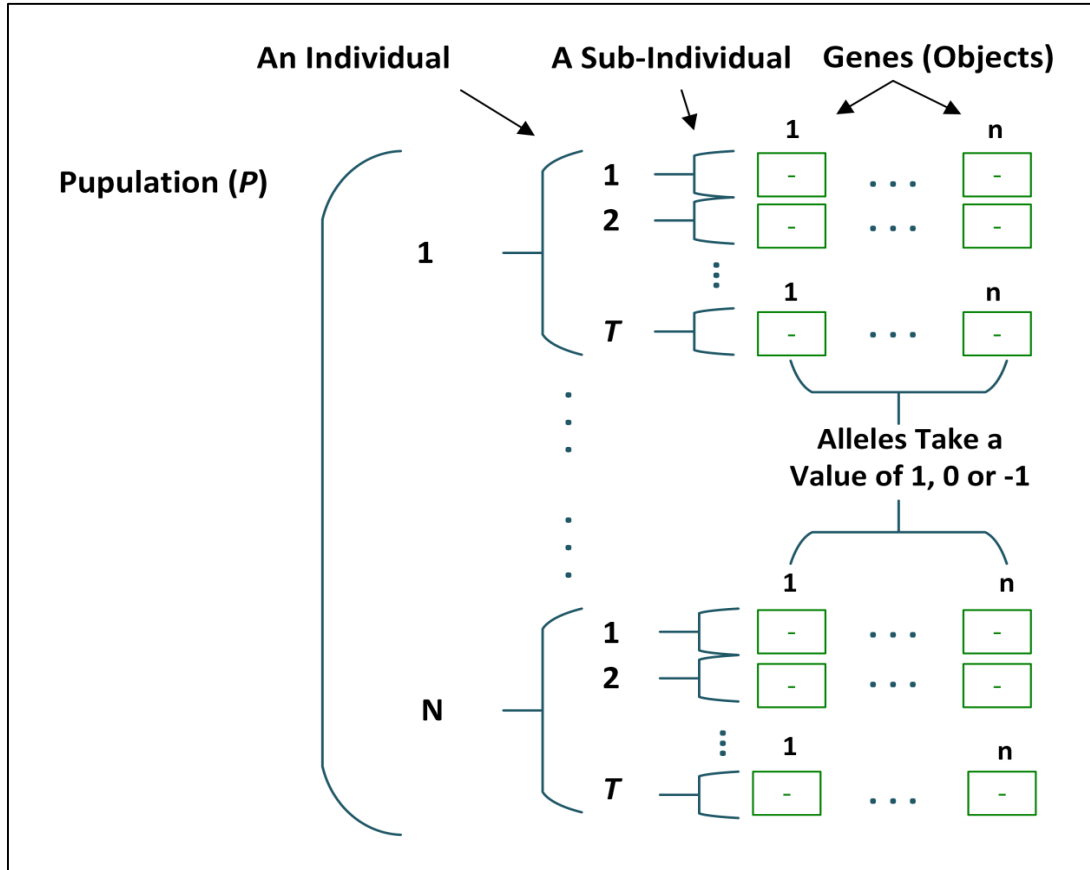


Figure 4.9. Population visualization in STAP

In the initialization phase, in order to maintain more genetic diversity, STAP employs two methods to set the initial population. Each method is applied with probability of 0,5 to each individual. In both methods a heterogeneity aware heuristic are applied to support the energy efficiency of the selected virtual objects towards more extension of stability and operational periods. The heterogeneity aware heuristics ensure that only objects with energy levels above the average energy level of the task group are permitted to be virtual objects. In the first method, an object has the property of satisfying the heterogeneity aware heuristic condition is selected randomly from each task group to carry out the duty of being a virtual object within the task group. The second method uses the principle of optimal election probability presented in [167]. In this method, in each task group the virtual objects are determined according to heterogeneity aware heuristic condition and the optimal election probability $P_{opt} = K_{opt}/l$, where l , is the number of objects in each task group, and $K_{opt} = \sqrt{(l/2\pi)} \cdot 2/0,765$ is the optimal number of constructed clusters in each task group [167]. In this manner, each gene is initialized as follows:

$$\forall i \in (1, \dots, N), \forall j \in (1, \dots, T), \forall k \in (1, \dots, n)$$

$$I_{i(j,k)} = \begin{cases} 1 & \text{if } Object_k \in j^{th} \text{ task group} \wedge E(Object_k) > E(Avg_j) \wedge random_{k,j} \leq P_{opt} \\ 0 & \text{if } Object_k \in j^{th} \text{ task group} \wedge random_{k,j} > P_{opt} \\ -1 & \text{if } Object_k \notin j^{th} \text{ task group} \end{cases} \quad (4.24)$$

where, $E(Object_k)$ is the energy of the j th object and $E(Avg_j)$ is the average energy of j th task group.

To measure the quality of each solution quantitatively an objective function Φ that is subject to minimization is attached to each individual. The objective function of STAP, Φ_{STAP} aims to minimize energy consumptions of virtual objects as well as maximizing the mean value of their energies. Φ_{STAP} , is composed of three parts: Individual virtual objects dissipated energy, VO_{DisE} , the total number of task groups for objects that are simultaneously assigned as virtual objects in multiple task groups, TG_{multi_VO} , and the average energy of virtual objects, VO_E . The first part (i.e., VO_{DisE}) attempts to minimize the average energy consumption of virtual objects. In STAP virtual objects are responsible of assigning tasks to their connected objects. VO_{DisE} , attempts to minimize the average energy consumed of virtual objects to perform this operation. This can be quantified by summing: the total energy spent by virtual objects to send IMs to their connected objects and the total energy dissipated by virtual objects to receive, aggregate and send the aggregated messages to \mathbb{AP} . Accordingly, VO_{DisE} can be formalized as follows:

$$VO_{DisE} = \sum_{j=1}^T \left(\left(\sum_{l=1}^{|VO_j|} \sum_{o=1}^{|O_{l_j}|} IM_{ETX_{l,o}} + Data_{ERX} + E_{DA} \right) + \sum_{l=1}^{|VO_j|} E_{TX_{L,AP}} \right) / |VO| \quad (4.25)$$

where, $|VO_j|$ is the number of virtual objects in the j th task group, $|O_{l_j}|$ is the number of objects connected to the l th virtual object of the j th task group. $IM_{ETX_{object_1, object_2}}$, is energy expenditure for sending indication message from $object_1$ to $object_2$. $Data_{ERX}$, determine the energy spent for receiving result messages by virtual objects. E_{DA} , and $E_{TX_{L,AP}}$ are energy consumed for data aggregation and energy for sending aggregated

message from virtual object l to \mathbb{AP} , respectively. Finally, $|VO|$ is the total number of virtual objects in the network.

The second part of, Φ_{STAP} , TG_{multi_VO} considers the fact that being a virtual object is extremely energy consuming operation. So, it attempts to minimize the number of task groups that an object is virtual object at them to at most one task group. Formally, TG_{multi_VO} , can be expressed as:

$$\forall k \in (1, \dots, n)$$

$$TG_{multi_VO} = \sum_{k=1}^n (|VO(O_k)| \mid |VO(O_k)| > 1) \quad (4.26)$$

where, $|VO(O_k)|$, is the number of task groups that object O_k is virtual object at them. Note, if the virtual objects is represent only one task group, then TG_{multi_VO} is set to 0.

Finally, the goal of the third part of the objective function VO_E , is maximizing the mean value of energy of virtual objects by selecting objects with high energy level as virtual objects as follows:

$$VO_E = \frac{\sum_{j=1}^T \sum_{l=1}^{|VO_j|} E_{VO_l}}{|VO|} \quad (4.27)$$

where, E_{VO_l} , is the energy amount of the l th virtual object of the j th task group. Then, by combining the three parts we can calculate Φ_{STAP} as follows:

$$\Phi_{STAP} = \frac{VO_{DisE} + TG_{multi_VO}}{VO_E} \quad (4.28)$$

The next component is the evolutionary genetic operators which includes selection and variation operators. Selection is an intensification operator acts as a force to increase the mean quality of solutions in the population. In STAP we applied the canonical binary tournament selection which elects the individual with the superior fitness value from two randomly selected individuals of the population set transferring it to the mating pool. This

operation is repeated N times to build a mating pool of the size equal to the size of population.

On the other hand, variation operators are diversification operators used to explore new and better solutions. Variation operators include recombination and mutation. In recombination, a proportion, $PC \times N / 2$ (where PC is probability of recombination) of pairs of parents are chosen from the mating pool and undergo 2-point multilevel cut and cross fill. For each sub-individual from the pair of parents, two cut points, $1 \leq cp_1, cp_2 \leq n - 1$, are randomly selected and the genes at $cp_1 \dots cp_2$ are swapped between the participated sub-individual. In mutation, a multilevel bit flop mutation is applied on a portion $PM \times N$ (where PM is probability of mutation) of individuals in the mating pool. Each gene with a value of allele not equal to -1 at each sub-individual is mutated with a mutation rate of m_r . If the value of the allele is 1, then is directly flipped to 0. However, the value of allele is flipped to 1 from 0 only if the energy of the corresponding object is greater than the average energy of its task group. Formally, mutation operator can be expressed as follows:

$$\forall k \in (1, \dots, n), \forall j \in (1, \dots, T)$$

$$I'_{r(j,k)} = \begin{cases} I_{r(j,k)} & \text{if } random > m_r \vee I_{r(j,k)} = -1 \\ 0 & \text{if } random \leq m_r \wedge I_{r(j,k)} = 1 \\ 1 & \text{if } random \leq m_r \wedge I_{r(j,k)} = 0 \wedge E(Object_k) > E(Avg_j) \end{cases} \quad (4.29)$$

where, r is a random number in range $\{1, \dots, N\}$, I'_r is an individual chosen randomly based on r from the mating pool. This operation repeated sequentially for each sub-individual $\forall j \in (1, \dots, T)$.

Executing variation operators creates new sets of candidate solutions, thus a correction mechanism is applied to guarantee the validity of the new solutions. Thus, if a task group becomes without a virtual object an object with energy level greater than the average energy of the task group is randomly selected and assigned as a virtual object within the task group.

In population reduction phase, an elitism selection is applied where the parent individuals of the population combined with the new solutions generated by the variation operators. The combined set is then sorted in ascending order based on the value of objective function and the first N individuals are preserved and transferred to the next generation.

This evolutionary process is repeated until the termination condition is satisfied. In STAP, the algorithm terminates when the specified number of generations is reached. Then, the solution with the best objective value is chosen as the final solution. The best individual is then decoded and undergo through association phase. In association phase, the CCS constructs the clusters and builds the connections between the virtual objects and non-virtual objects entities based on the Euclidean distances.

4.3.3. Multi-objective task allocation protocol (MOTAP)

MOTAP aims to increase the energy efficiency of the network as well as maximizing the computation power utilization of virtual objects while assigning tasks to objects. However, selecting the virtual objects that maximize the total computational power in use could conflict with the goal of energy efficiency. For instance, selecting a large number of objects as virtual objects to maximize the total computational power is inefficient since a large number of virtual objects will consume more energy and in turn, degrade the energy efficiency of the network. Therefore, it is desirable to maximize the total computational power in use while minimizing the number of virtual objects (i.e., maximizing the mean value of computational power) in each task group. However, minimizing the number of virtual objects increases the energy load on individual virtual objects and therefore decreases the stability period (i.e., the time interval in which the first virtual object completely depletes its energy in each task group). The same situation occurs when an object with several skills and high computational power is selected as a virtual object in many task groups.

Multi-Objective Evolutionary Algorithms (MOEAs) involve optimizing a number of objectives that conflict with each other and providing an optimal decision with a set of trade-off solutions between the conflicting objectives. Thus, MOTAP uses the underlying of MOEAs in order to jointly formulate the computational power utilization and energy

efficiency problems in task allocation of IoT as a novel Multi-objective Optimization Problem (MOP).

The next subsections illustrate the special design features of MOTAP and sight the light on its algorithmic framework.

System model for MOTAP

MOTAP uses the system model defined in Section 4.1 with some application specific design features. MOTAP assumes that the IoT objects are equipped with different energy levels and have different computation power capabilities supplied with processors with different frequencies. Furthermore, MOTAP assumes that the IoT network is eligible of performing T tasks and each object is capable of performing t_o ($t_o \leq T$) tasks.

In our model, the network is partitioned into different task groups, each of which contains the objects that are capable of performing the corresponding task of the task group. In each task group, a set of objects are selected as virtual objects to represent the task group and they act as the task allocator within their groups. When, a task t , $\forall t \in \{1, \dots, T\}$, is needed to be performed, the virtual objects of the task group corresponding t sends Indication Messages (IMs) to their connected objects. Each IM is attached with a Task Identification TID (i.e., $TID = t$) since the objects can participate in multiple task groups. Then, the objects perform the task that is assigned in the TID after receiving the IMs and send the results back to the virtual objects. Virtual objects perform data aggregation and send the aggregated results to \mathbb{AP} .

The algorithmic framework of multi-objective task allocation protocol (MOTAP)

MOTAP is a centralized protocol resides in the CCS that is equipped with unlimited resources. Although there are several variants of MOEAs, it is out of the scope of this thesis to investigate the applicability of different MOEAs. MOTAP adopts a custom designed framework based on elitist Non-Dominated Sorting Genetic Algorithm II (NSGA-II) as its underlying algorithmic paradigm.

The evolutionary process of the general framework of MOTAP (see Figure 4.10) can be illustrated as follows:

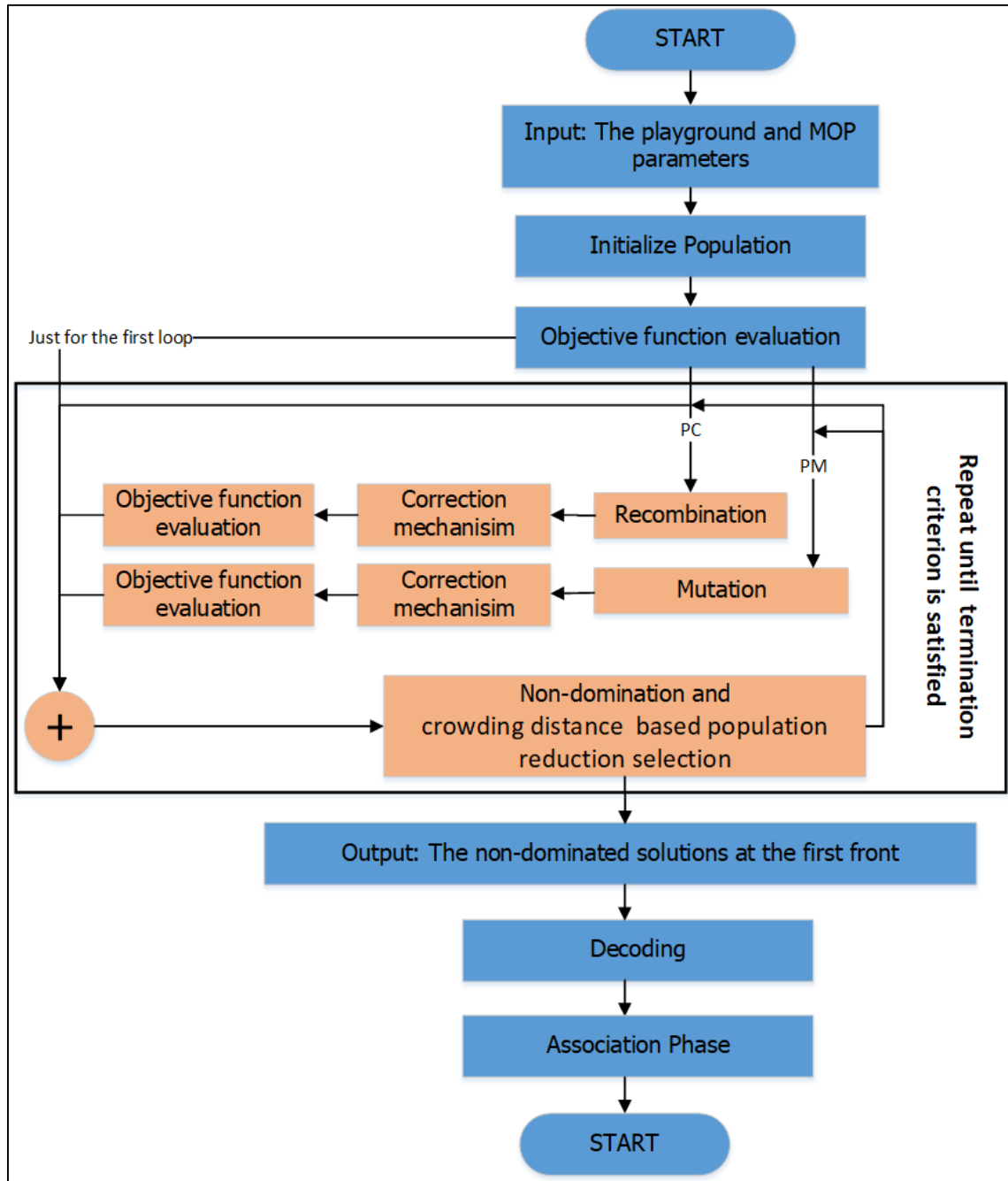


Figure 4.10. The General layout of MOTAP

Individuals encoding in the solution space: MOTAP It begins by considering a population P which consists of a group of N individuals. Each individual $I_i \in P, \forall i \in (1, \dots, N)$ is divided into a set of sub-individuals. Each of these sub-individuals represents a task group in the network and its length equal to the number of objects (n). MOTAP uses the structural view that is presented in Figure 4.9 to store the population. In this structure, each gene encodes one object in the network in each task group and takes a value of $-1, 0$ or 1 as follows:

- -1: if the object is not a member of the task group.
- 0: if the object is a member of the task group but it is not a virtual object.
- 1: if the object is a member of the task group and it is a virtual object in that task group.

Figure 4.11 (a) and (b) show genotype and phenotype examples in MOTAP, respectively. This figure abstracts a 200×200 Unit IoT network capable of performing 4 tasks and consists of 50 objects from the initial population. Figure 4.11 (a), represents the genotype solution corresponded to the phenotype in Figure 4.11 (b). The phenotype solution clarifies the form of: Access point, virtual objects of each Task Group (TG), and member objects (non-virtual objects) in each task group. The tasks of each object are specified by arrow to the right of the object. Red colored tasks mean the object is a virtual object in the corresponding task group.

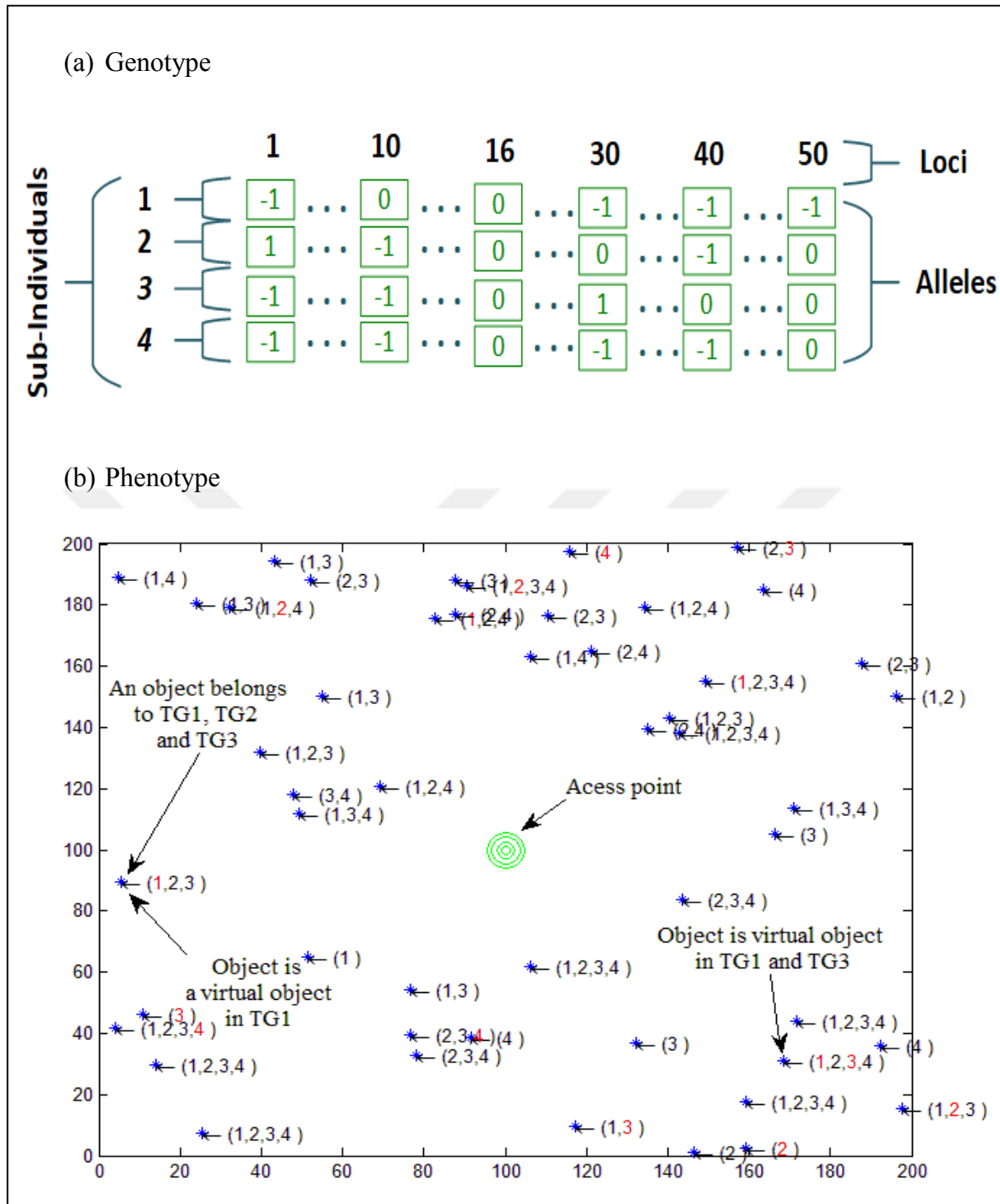


Figure 4.11. Examples for genotype (a) and phenotype (b) individuals of MOTAP

Population initialization: In the initialization phase, MOTAP employs two strategies to ensure better exploration of different points in the search space. Each strategy creates half of the population with a probability of 0,5 to create each individual. In the first strategy, an object in each task group is chosen randomly to become a virtual object within the task group. In other words, a gene from the set of gens corresponding the set of objects belong

to the task group in each sub-individual is chosen randomly and its allele value is set to 1. The second strategy adopts the principle of optimal election probability [167]. According to Heinzelman et al. [167], the percentage of virtual objects in each task group can be determined by an optimal election probability $P_{opt} = K_{opt}/l$ where, l is the number of objects in each task group, and K_{opt} is the optimal number of constructed clusters in each task group which is previously calculated in Equation (4.13). Then, each gene is randomly initialized to -1 , 0 or 1 depending on the membership of the corresponding object to the task group and P_{opt} of the desired percentage of virtual objects as follows:

$$\forall i \in (1, \dots, N), \forall j \in (1, \dots, T), \forall k \in (1, \dots, n)$$

$$I_{i(j,k)} = \begin{cases} 1 & \text{if } Object_k \in j^{th} \text{ task group} \wedge random_{k,j} \leq P_{opt} \\ 0 & \text{if } Object_k \in j^{th} \text{ task group} \wedge random_{k,j} > P_{opt} \\ -1 & \text{if } Object_k \notin j^{th} \text{ task group} \end{cases} \quad (4.30)$$

Objective functions evaluation: $\Phi: P_i \rightarrow R^2$ denotes the objective function vector that is assigned to each individual. This vector attaches a quality measure to each individual. In MOTAP, there are two objective functions: \mathcal{F}_1 and \mathcal{F}_2 , where both are subject to maximization. \mathcal{F}_1 involves maximizing the mean value of computation power of virtual objects. It accomplishes this by selecting objects with powerful processing units as virtual objects while minimizing the total number of virtual objects in each task group. Formally, $\Phi_{\mathcal{F}_1}$ can be calculated as follows:

$$\Phi_{\mathcal{F}_1} = \frac{\sum_{j=1}^T \sum_{l=1}^{|VO_t|} pp(VO_{t_l})}{\sum_{j=1}^T |VO_t|} \quad (4.31)$$

In the equation above, $|VO_t|$ represents the number of virtual objects at the t th task group, and $pp(VO_{t_l})$ represents the processing power (i.e., processing unit's capacity) of the l th virtual object of the t th task group. It is obvious that the genotype coding in of MOTAP facilitates the computation of $\Phi_{\mathcal{F}_1}$, which can be expressed as:

$$\Phi_{\mathcal{F}_1} = \frac{\sum_{j=1}^T \sum_{l=1}^{|1's_t|} pp(1_{t_l})}{\sum_{j=1}^T |1's_t|} \quad (4.32)$$

In this equation, $|1's_t|$ is the number of 1's (i.e., the number of virtual objects) at the t th sub-individual and $pp(1_{t_l})$ is the processing power of the l th virtual object of the t th task group.

\mathcal{F}_2 maximizes the stability of each task group and minimizes energy dissipation for task allocation, which is expressed as communication overhead. It consists two parts: stability, S , and energy efficiency, EF . S maximizes the stability of each task group by ensuring that only the objects with high energy levels could be selected as virtual object. Hence, it maximizes the mean value of virtual objects' energy. S also considers the fact that being a virtual object is an energy consuming operation. So, it attempts to minimize the number of objects that are virtual objects in more than one task group. However, while S extends the stability periods of task groups, it also extends the operational periods of these groups. S can be formally expressed as follows:

$$S = \frac{\sum_{j=1}^T [\sum_{l=1}^{|VO_t|} E(VO_{t_l})] / |VO_t|}{|multi_TG(VO)|} \quad (4.33)$$

where, $E(VO_{t_l})$ is the energy equipped with l th virtual object of the t th task group and $|multi_TG(VO)|$ is the number of objects that are virtual objects in more than one task groups. S is expressed as maximization and the value at the denominator is minimized.

The second part of \mathcal{F}_2 is energy efficiency (EF) with the goal of minimizing the total energy consumption for task allocation. This goal is conducted by selecting the virtual objects such that the energy consumption from the communication overhead for assigning tasks and receiving results at each task group is minimized. We formally define EF as:

$$EF = \sum_{t=1}^T \left(\left(\sum_{l=1}^{|VO_t|} \sum_{j=1}^{|O_{t_l}|} IM_E_{TX_{l,j}} + IM_E_{RX} + E_{TX_{j,l}} + E_{RX} + E_{DA} \right) + \sum_{l=1}^{|VO_t|} E_{TX_{l,AP}} \right) \quad (4.35)$$

where, $|O_{t_l}|$ is the number of objects connected to the l th virtual object of the t th task group and $IM_E_{TX_{object_1, object_2}}$ and $E_{TX_{object_1, object_2}}$ are the energy expenditures for sending the indication and result messages from $object_1$ to $object_2$, respectively. IM_E_{RX} and E_{RX}

determine the energy consumption for receiving IM by IoT objects and the energy consumption for receiving data (or result) messages by virtual objects. E_{DA} is the energy dissipated for data aggregation.

Then, by combining Equations (4.34) and (4.35), we can calculate $\Phi_{\mathcal{F}_2}$ as follows:

$$\Phi_{\mathcal{F}_2} = S/EF \quad (4.36)$$

Variation operators and reduction selection: Variation operators are used to alter the task allocation solutions found in the population and explore new points of the search space. They include recombination and mutation operators. In recombination, a ratio of pairs of randomly selected parents ($PC \times \frac{N}{2}$, where PC is probability of recombination) undergo 2-point multilevel cut and cross fill. For each pair of parents, the recombination is divided into T levels, where T is the number of sub-individuals as well as the number of task groups. At each level two cut points, $1 \leq cp_1, cp_2 \leq n - 1$, are randomly selected and then, each sub-individual of the participating parents is swapped at genes cp_1 and cp_2 . This operation is repeated at each level until all sub-individuals are subject to 2-point cut and cross fill. Formally, let I_{r_1}, I_{r_2} be a pair of parent individuals randomly selected from the range $\{1, \dots, N\}$ of the population. Then, the following operation is repeated at each level $\forall j \in (1, \dots, T)$:

$$\forall k \in (1, \dots, n)$$

$$\begin{aligned} I'_{r_1(j,k)} &= \left(I_{r_1(j,1)}, \dots, I_{r_1(j,cp_1)}, I_{r_2(j,cp_1+1)}, \dots, \right. \\ &\quad \left. \dots, I_{r_2(j,cp_1-1)}, I_{r_1(j,cp_2)}, \dots, I_{r_1(j,n)} \right) \\ I'_{r_2(j,k)} &= \left(I_{r_2(j,1)}, \dots, I_{r_2(j,cp_1)}, I_{r_1(j,cp_1+1)}, \dots, \right. \\ &\quad \left. \dots, I_{r_1(j,cp_1-1)}, I_{r_2(j,cp_2)}, \dots, I_{r_2(j,n)} \right) \end{aligned} \quad (4.37)$$

In mutation, a proportion $PM \times N$ (where PM is probability of mutation) individuals in the population are mutated. When an individual I_r is chosen for mutation, each gene at each sub-individual with an allele not equal to -1 is mutated with a mutation rate of m_r by applying Bit Flop Mutation (BFM). In BFM, once a gene is chosen for mutation, its value is flipped from 1 to 0 and vice versa:

$\forall k \in (1, \dots, n)$

$$I'_{r(j,k)} = \begin{cases} I_{r(j,k)} & \text{if } I_{r(j,k)} = -1 \text{ or } \text{random} > m_r \\ 1 - I_{r(j,k)} & \text{otherwise} \end{cases} \quad (4.38)$$

This operation is repeated for each sub-individual $\forall j \in (1, \dots, T)$.

Reduction selection is used to restrict the population size to N . In reduction selection phase of MOTAP, an elitism scheme based on non-domination and crowding distance concept is applied. Thus, the individuals with lower ranks of non-dominant fronts or with equal ranks (i.e., individuals belonging to the same front) but with greater crowding distances are selected to be transferred to the next generation.

The termination criterion: The termination criterion in MOTAP is a pre-specified number of generations. In the last generation, the non-dominated solutions of the first front are introduced to the decision maker. These solutions represent a set of optimal trade-off solutions between the conflicting objectives of computation power and energy efficiency.

The decision maker at this point is responsible of selecting the most appropriate individual to the network from the set of non-dominated solutions. Then, selected solution is decoded and transferred to the association phase where the clusters are constructed and the connections between virtual objects and non-virtual objects entities are established.



5. SIMULATION RESULTS AND DISCUSSION

This chapter presents the experimental results of the proposed evolutionary task allocation protocols. In literature, there is no existing work for the problem of task allocation that considers the heterogeneous IoT network in terms of energy levels using the concept of task groups and virtual objects in an evolutionary algorithm framework. Hence, in the simulations, for the purpose of benchmarking our protocols, we compared our methods with the most relevant algorithm in the literature (CBATA) [8, 12]. For simulation purposes, CBATA is implemented from the scratch to test its energy consumption performance. The evaluations consider different performance measures related to the energy efficiency, duration of the stability and operational periods, computational power utilization and evolutionary algorithm quality.

This chapter is mainly divided into two parts. The first part demonstrates the performance of the protocols that are developed within the framework of scenario #1. This part begins by investigating the performance of ETAP1 and ETAP2 in terms of energy efficiency, number of operational rounds, and the quality of the proposed evolutionary algorithm. Next, the simulations examine the effects of energy aware heuristics and evaluate the performance of SETAP1 and SETAP2 in terms of length of stability periods and average available energy in virtual objects. On the other hand, the second part investigates the performance of the protocols in the context of scenario #2. In order to evaluate MOTAP, STAP and M-CBATA, several evaluation metrics related to computational power optimization, energy efficiency, duration of the stability, and operational periods are used. Moreover, the quality of the EA that is used to develop STAP is measured by adopting convergence to optimal solution metric. Furthermore, in order to evaluate the quality of our multi-objective optimization algorithm that is used in MOTAP, additional measurements such as the diversity maintained in the non-dominate optimal set and the consistent convergence to the optimal solutions are considered.

5.1. Performance Analysis of Protocols in Context of Scenario #1

This section expounds the performance of ETAP1, ETAP2, SETAP1, and SETAP2 against the most relevant work in exist literature (CBATA). The evaluation taking place considering different performance evaluation metrics related to the energy efficiency, duration of the stability and operational periods and evolutionary algorithm quality. Energy efficiency is measured by calculating the total dissipated energy in the network and the total dissipated energy in each task group. Additionally, in order to give a deeper insight, the execution of tasks is broken up into rounds and assumed the most extreme case where at each round all tasks are needed to be performed. Then, the number of rounds until the network or task groups become non-operational is depicted. The stability period in a task group is defined as the time interval before the death of the first virtual object in that task group. We evaluated the length of the stability periods of the proposed protocols by transferring task allocation process into rounds and obtaining the number of rounds till a virtual object explodes its energy in each task group. Finally, the quality of the proposed evolutionary algorithms is evaluated in terms of consistent convergence to the optimal solution.

The next subsections begin by illustrating the simulation environment and the control parameters' settings that used to obtain the experimental results. Then, the analysis results of ETAP1 and ETAP2 against CBATA are presented. Finally, the results of the stability aware protocols (i.e., SETA1 and SETAP2) are investigated.

5.1.1. Simulation environment

The simulation environment is designed adopting a custom simulator using MATLAB. MATLAB is a high-level language and interactive environment to analyze and design systems for solving different engineering and scientific problems [170, 171]. The custom designed simulator enables us to generate random network topologies, to execute the protocols on the generated topologies, to obtain the desired evaluation metrics, and to benchmark the outputs of the simulations. The communication links in this simulator are assumed to be bi-directional. Moreover, to maintain the poor radio conditions in most IoT applications, a retransmission mechanism that retransmits erroneous or lost packets up to 5 times is applied. The simulations are executed under several bit error rates to evaluate the

effect of retransmission mechanism. It is observed that increasing the packet error rate by 1% increases the total energy consumption 3,2% on average (See Table 5.1). However, only the worst-case scenario, which is the case when we have 10% packet error rate is given. We experimentally observed and selected this error rate as a severe network condition.

Table 5.1. Average dissipated energy (in joules) in 5 test instances for 200×200 network scale, 250 object and 4 task groups and different packet error rates

Packet Error Rate	ETAP1	ETAP2	Packet Error Rate	ETAP1	ETAP2
0%	0,27934	0,59381	6%	0,33320	0,71314
1%	0,28772	0,61221	7%	0,34352	0,73809
2%	0,29750	0,63118	8%	0,35279	0,75580
3%	0,30583	0,65011	9%	0,36125	0,77545
4%	0,31500	0,66961	10%	0,36874	0,78384
5%	0,32508	0,69237	-	-	-

In the experiment, n objects are randomly deployed in a square shaped area with the dimensions of $M \times M$ unit. This means that the horizontal and vertical coordinates of each object are randomly selected between zero and the maximum value of the dimension. The internet coverage in this area is provided by an access point AP located at the center of the area. The network is capable of performing T tasks. In the simulation it is assumed that each object can perform one task and a task number between $[1, T]$ is randomly associated to each object. The simulation is mainly divided into three groups based on number of objects (n), number of tasks in the network (T) and the dimension of the area (M). Unless otherwise stated, we suppose a default setting for these parameters (i.e., $n = 250, M = 200, T = 4$) and then to get different simulation scenarios we vary value of one parameter while fixing values of other two parameters. For number of objects we vary the value of n from 200 to 600 objects with an incremental value of 50. In this way we generated 9 different test instances. For number of tasks in the network, T takes 5 different values (2; 4; 6; 8 and 10). Finally, we changed the area dimensions to 7 different values from 100 to 250 with an incremental value of 25. To insure fairness of simulations regarding the aforementioned randomness in objects' coordinates, objects' energy levels and the tasks are associated with each object. We generated 5 different network topologies for each simulation settings. Then, to maintain the probabilistic feature of the evolutionary task

allocation protocols each of these settings, are executed 5 times. Then, the average result of each network is obtained and the final results are calculated by averaging the results of the networks. Hence, the overall simulation tests a total of 525 random networks for each protocol.

5.1.2. Network energy model

Each object is randomly initialized with an energy value between [0,5; 2] Joules. For network energy model we adopted first-order radio model [166] with its default parameter settings. In this model the energy needed to run transceiver circuit is set to $E_{elec} = 50$ nJ/bit. The energy expenditures in amplifier is $\epsilon_{amp} = 100$ pJ/bit/m². The message size in our protocols is set to 4000 bits for data messages and 8 bits for indication messages. Another parameter also taken into account is the energy spent for data aggregation, which is set to $E_{DA} = 5$ nJ/bit.

5.1.3. Parameters and rules settings in the evolutionary task allocation algorithms

In the simulation, all routing protocols assume the following:

- The central control station is aware of its location and these of the IoT objects.
- The central control station and the IoT objects are stationary during the simulation.
- The central control station is equipped with unlimited amount of energy.

Other parameter settings that complete the characteristics of the evolutionary task allocation protocols are: binary tournament selection, two-point crossover with 0,6 recombination probability (p_c), bit flop mutation with Mutation probability (p_m) = 0,03 and mutation rate (m_r) = 0,02, population reduction based on elitism selection and population size (N) of 50 individuals allowed to evolve for 50 generations. Table 5,2 summarizes the particular rules and control parameters that are used in the evolutionary algorithm.

Table 5.2. Parameters and rules used in the evolutionary algorithms of scenario #1

Population size (N)	50
Stopping criteria	Number of generations = 50
Selection mechanism	Binary Tournament selection (BTS)
Recombination operator	Two-point crossover
Mutation operator	Bit Flop Mutation (BFM)
Recombination probability(p_c)	0,6
Mutation probability(p_m)	0,03
Mutation rate (m_r)	0,02
Replacement policy	Tournament selection based elitism

5.1.4. Results of ETAP1 and ETAP2

The next subsections evaluate ETAP1 and ETAP2 against CBATA considering different performance evaluation metrics related to the energy efficiency, duration of the operational period and evolutionary algorithm quality.

Protocols evaluation for energy efficiency

Tables 5.3 and 5.4 present the performance of ETAP1, ETAP2 against CBATA regarding to total dissipated energy in the network for task allocation. The results in Table 5.3 are obtained by varying network scale and setting number of objects and number of tasks that the network can perform to 250 objects and 4 tasks, respectively. In Table 5.4, the results are for network with 4 tasks, 200×200 unit side length and different objects densities. Moreover, the results in these tables are averaged over 5 test instances and the best results are shown in italic font. From these results it can be observed that both ETAP1 and ETAP2 outperform CBATA. ETAP1 with its objective function that directly deals with energy consumption performs better than ETAP2. ETAP1 directly tackles the problem of energy consumption in task allocation by minimizing the total energy needs for this operation. On the other hand, ETAP2 deals with the problem indirectly by forming the problem as function to distance and attempts to compose a good distribution of virtual objects within each task group. As can be expected, energy consumption increases when network scale increases or the number of object increases. However, the increase rate in ETAP1 and ETAP2 are much less than the increase rate in CBATA. For example in Table 5.3, one can see that, the energy dissipation in ETAP1 in the most extreme case in this scenario (when

network side length is 250) is less than the energy dissipation in CBATA in the most slight case (i.e., when network side length is 100) by 45,15 milijoule. The savings obtained in this simulation seem to be negligible; however, it should be noted that each object is equipped with an energy value between $[0,5; 2]$ Joules. Also in this table, the average increment in energy consumption at each increment of 25×25 unit of network size is 38,712 milijoule, 93,385 milijoule and 274,452 milijoule in ETAP1, ETAP2 and CBATA, respectively. The last two columns in the table summarize the gain percentage of ETAP1 and ETAP2 over CBATA.

Table 5.3. Average dissipated energy (in joules) in 5 test instances for different network scales

Network Side Length	ETAP1	ETAP2	CBATA	ETAP1 gain over CBATA	ETAP2 gain over CBATA
100	0,13647	0,22353	0,41389	67,02%	45,99%
125	0,15889	0,27794	0,63432	74,95%	56,18%
150	0,19032	0,34504	0,84772	77,54%	59,29%
175	0,22679	0,43837	1,10410	79,45%	60,29%
200	0,26496	0,65772	1,52680	82,64%	56,92%
225	0,31443	0,66562	1,88100	83,28%	64,61%
250	0,36874	0,78384	2,06060	82,10%	61,96%

Table 5.4. Average dissipated energy (in joules) in 5 test instances for different object density settings

Number of Objects	ETAP1	ETAP2	CBATA	ETAP1 gain over CBATA	ETAP2 gain over CBATA
200	0,22026	0,49525	1,0885	79,76%	54,50%
250	0,26999	0,51884	1,5260	82,30%	66,00%
300	0,30875	0,64170	1,8607	83,40%	65,51%
350	0,34148	0,80132	1,8931	81,96%	57,67%
400	0,38107	0,86787	2,3330	83,66%	62,80%
450	0,41409	0,93076	2,5528	83,77%	63,53%
500	0,44988	0,95843	2,6295	82,89%	63,55%
550	0,48696	1,00990	2,9359	83,41%	65,60%
600	0,52240	1,15840	3,8259	86,34%	69,72%

Another outcome that supports the previous results is presented in Table 5.5. This table illustrates protocols performance in terms of energy consumption in the network for task allocation for network dimension of 200×200 unit, 250 objects, and up to 10 tasks. In these results ETAP1 outperforms both CBATA and ETAP2. In turn ETAP2 performs better than CBATA. In ETAP1 and ETAP2 energy consumptions grow in parallel with increased number of tasks and fixed number of objects and network dimension. CBATA depicts an opposite behavior where energy consumption degrades when the number of tasks increases. Although this behavior has positive affect on energy consumption, it could be neglected considering the limited computational power of IoT objects which may be capable of performing several though limited number of tasks [3].

Table 5.5. Average dissipated energy (in joules) for 5 test instances for different number of tasks

Number of Tasks	ETAP1	ETAP2	CBATA	ETAP1 gain over CBATA	ETAP2 gain over CBATA
2	0,21515	0,40125	1,4969	85,62%	73,19%
4	0,26837	0,58609	1,3806	80,56%	57,54%
6	0,29768	0,59308	1,3784	78,40%	56,97%
8	0,32584	0,56590	1,4738	77,89%	61,60%
10	0,34815	0,68737	1,3641	74,47%	49,61%

Figures 5.1 and 5.2 qualitatively depict the performance of the protocols in terms of total dissipated energy in the network to allocate tasks to objects for different number of objects and different number of tasks, respectively. The results indicate that ETAP1 performs better than ETAP2 and CBATA; on the other hand, ETAP2 performs better than CBATA. Considering different evaluation metrics, as seen in Figure 5.1, increasing the number of objects from 200 to 600 increases energy dissipation by 0,30214; 0,66315 and 2,7374 in ETAP1, ETAP2 and CBATA, respectively. In these results the averages of increase ratios in energy consumption are 0,037768; 0,082894 and 0,342175 Joule for each of ETAP1, ETAP2 and CBATA. The increase ratio of CBATA (i.e., 0,342175 Joule) indicates a great disparity in the behavior of the protocol compared to both ETAP1 and ETAP2. One can attribute that to the behavior of CBATA which selects only one virtual object in each task group and replace it only when it depletes its energy. In contrast with that, both ETAP1 and ETAP2 maintain a load balance and distribute the burden of task allocation among

good selected virtual objects in each task group. Another evaluation metric can be obtained from Figure 5.2. In this figure, the big gap in energy dissipation between both ETAP1 and ETAP2, and CBATA becomes more obvious. When the number of tasks is equal to 2, CBATA consumes 1.4969 Joule for task allocation. This value in ETAP2 is 0,40125 and 0,21515 in ETAP2. Increasing the number of tasks to 6 increases the energy consumption to 1,3784; 0,59308 and 0,29768 Joule in CBATA, ETAP2 and ETAP1, respectively. The results show that ETAP1 makes the best score followed by ETAP2.

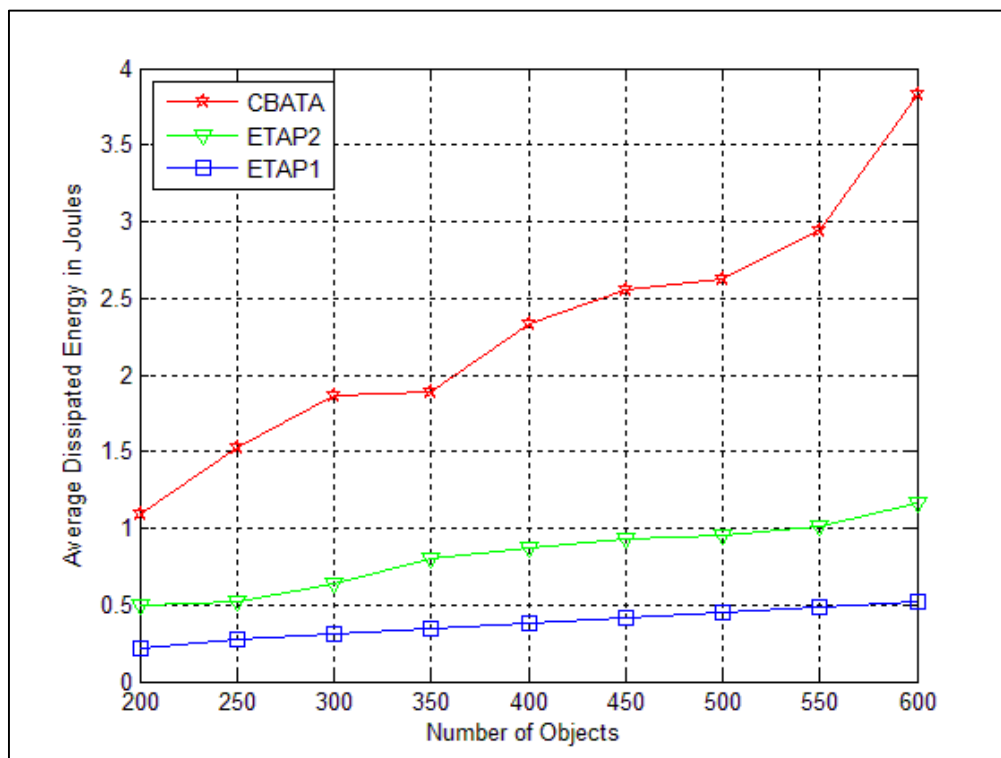


Figure 5.1. Average dissipated energy in 5 test instances for different object density settings

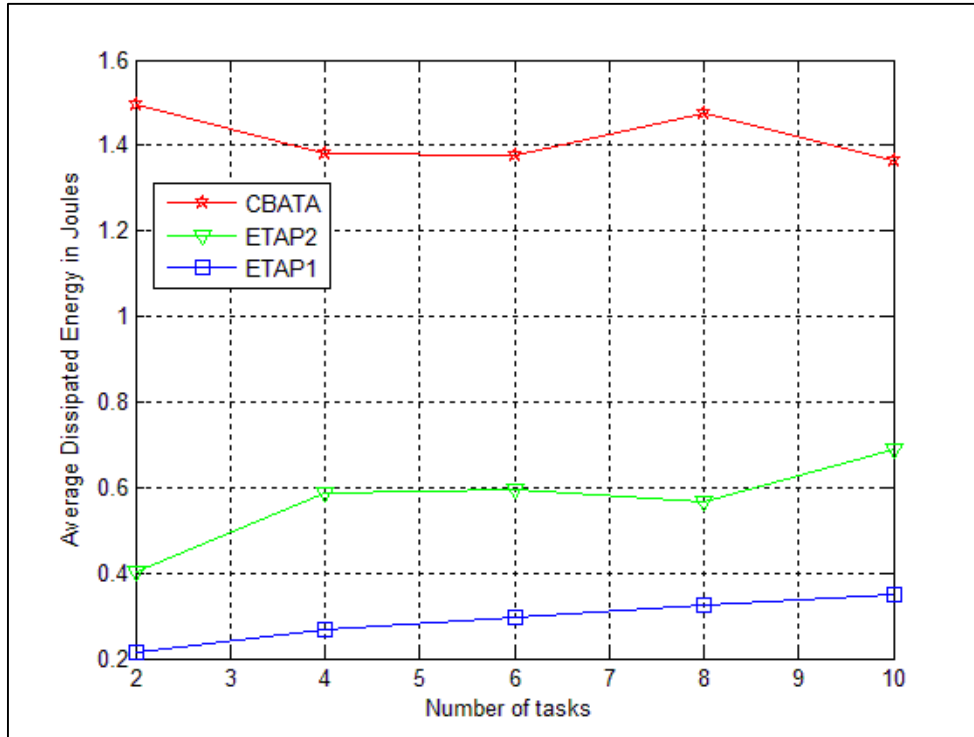


Figure 5.2. Average dissipated energy in 5 test instances for different number of tasks

For detailed view, Figure 5.3 and Tables 5.6, 5.7 and 5.8 investigate energy dissipation in each task group that resulted from task allocation. The results of Figure 5.3 are obtained in a network with 200 unit side length, 250 objects, and 6 task groups. On the other hand, the results in Tables 5.6, 5.7 and 5.8 are obtained with default setting of the network while varying number of network tasks, network dimensions, and number of objects respectively. In these tables the best results are indicated by italic font. The results are averaged over 5 test instances with each executed 5 times. It is worth to note that, the total energy in the network is distributed evenly among all task groups in the network in this scenario. It is clear that ETAP1 evenly selects the virtual objects in the network as a result energy expenditures are evenly distributed among objects of the network in different task groups. This situation is different in CBATA. Thus, there are big differences among energy consumptions of different task groups. In ETAP2 this situation is less severe. For example in Figure 5.3 and Table 5.6 only task group 3 and task group 5 consume more energy compared to other task groups. Quantitatively (also from Figure 5.3 and Table 5.6), the energy differences between one task group and another are 0,00234; 0,02008 and 0,0571 for ETAP1, ETAP2, and CBATA, respectively.

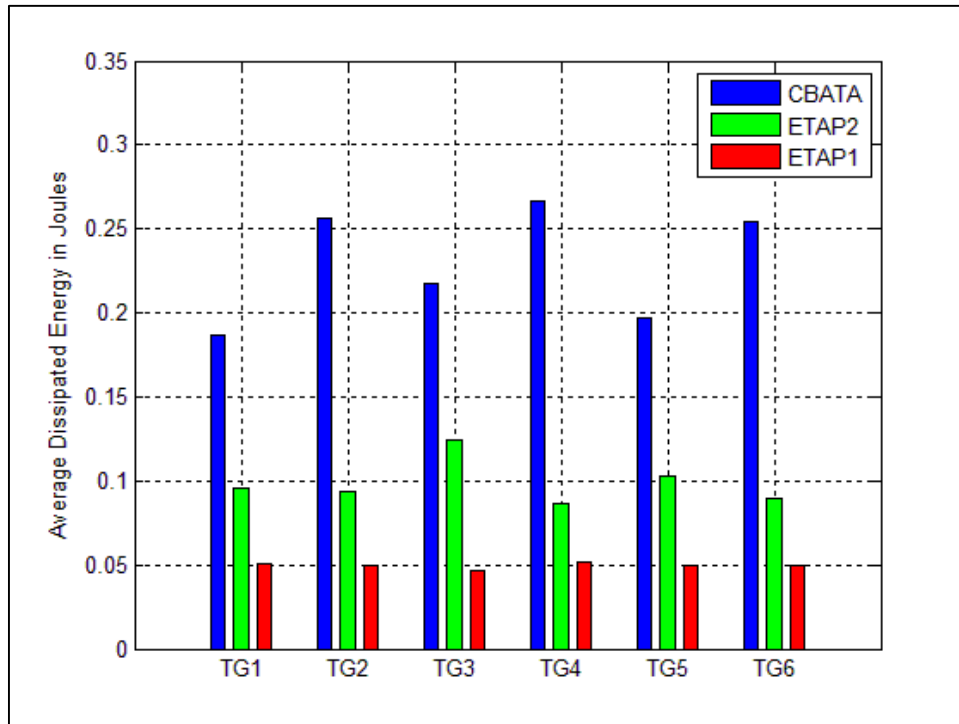


Figure 5.3. Average dissipated energy in each task group for 5 test instances with network dimension = 200×200 , object density = 250 and number of tasks = 6

Table 5.6. Average dissipated energy in each task group for 5 test instances with network dimension = 200×200 , object density = 250 and number of tasks = 6

Task Group	ETAP1	ETAP2	CBATA
1	0,0511	0,0953	0,1867
2	0,0500	0,0940	0,2561
3	0,0464	0,1245	0,2171
4	0,0514	0,0865	0,2670
5	0,0495	0,1033	0,1971
6	0,0494	0,0895	0,2544

Table 5.7. Average dissipated energy in each task group for 5 test instances with object density = 250, number of tasks = 4 and different network dimensions

Network	ETAP1				ETAP2			
Dimension	TG1	TG2	TG3	TG4	TG1	TG2	TG3	TG4
100 × 100	0,0337	0,0338	0,0346	0,0344	0,0529	0,0498	0,0649	0,0559
125 × 125	0,0423	0,0377	0,0406	0,0382	0,0524	0,0683	0,0815	0,0757
150 × 150	0,0462	0,0477	0,0494	0,0470	0,0764	0,1067	0,0661	0,0958
175 × 175	0,0596	0,0554	0,0531	0,0587	0,1062	0,1052	0,0957	0,1312
200 × 200	0,0700	0,0653	0,0622	0,0675	0,1272	0,1878	0,2008	0,1420
225 × 225	0,0763	0,0809	0,0765	0,0807	0,1735	0,2009	0,1273	0,1640
250 × 250	0,0869	0,0962	0,0921	0,0935	0,1827	0,1998	0,1806	0,2207
Network	CBATA							
Dimension	TG1	TG2	TG3	TG4				
100 × 100	0,1110	0,0848	0,1125	0,1056				
125 × 125	0,1724	0,1788	0,1409	0,1422				
150 × 150	0,2061	0,2236	0,1908	0,2273				
175 × 175	0,2900	0,3013	0,1896	0,3232				
200 × 200	0,4175	0,4004	0,3613	0,3476				
225 × 225	0,4204	0,4188	0,5419	0,4998				
250 × 250	0,4428	0,4440	0,5364	0,6374				

Table 5.8. Average dissipated energy in each task group for 5 test instances with network dimension = 200×200 , number of tasks = 4 and different object density

Number of Objects	ETAP1				ETAP2			
	TG1	TG2	TG3	TG4	TG1	TG2	TG3	TG4
200	0,0499	0,0553	0,0603	0,0547	0,1616	0,1091	0,1026	0,1220
250	0,0680	0,0666	0,0688	0,0665	0,1373	0,1224	0,1346	0,1245
300	0,0755	0,0758	0,0806	0,0769	0,1560	0,1695	0,1447	0,1715
350	0,0878	0,0854	0,0844	0,0839	0,1498	0,2501	0,2392	0,1621
400	0,0980	0,0933	0,0951	0,0946	0,1507	0,2406	0,2456	0,2309
450	0,1016	0,1016	0,1037	0,1072	0,1864	0,2529	0,1999	0,2915
500	0,1124	0,1148	0,1109	0,1117	0,2310	0,2146	0,2735	0,2393
550	0,1236	0,1166	0,1230	0,1238	0,2310	0,2806	0,1847	0,3137
600	0,1369	0,1288	0,1259	0,1308	0,3928	0,2004	0,2780	0,2871
Number of Objects	CBATA							
	TG1	TG2	TG3	TG4				
200	0,2591	0,2666	0,2663	0,2966				
250	0,4556	0,3673	0,3597	0,3434				
300	0,3826	0,5124	0,4555	0,5102				
350	0,5020	0,4574	0,5100	0,4237				
400	0,6090	0,5165	0,5186	0,6890				
450	0,6642	0,4494	0,7457	0,6935				
500	0,5128	0,6865	0,6528	0,7774				
550	0,8354	0,6547	0,7551	0,6907				
600	0,8178	0,9885	0,8573	1,1623				

Protocols evaluation for duration of the operational period

In order to give a deeper insight of the performance of the protocols, Figures 5.4 and 5.5 breaks the execution of tasks into rounds and depicts the most extreme case that all tasks are needed to be performed at each round assuming a continuous and a uniform arrival of tasks. Figure 5.4 and Figure 5.5 show the number of rounds until each task group become non-operational (i.e., all its virtual objects deplete their energies) for scenarios that object's densities are 250 and 500, respectively. In both scenarios and in all task groups the longer operational time (i.e., the largest number of rounds) is achieved by ETAP1 followed by ETAP2 whereas CBATA becomes at the last rank. The reason is that both ETAP1 and ETAP2 with their main goal of minimizing energy consumption maintain a load balance and distribute the burden of task allocation among good selected virtual objects in each task group. Thus, when one virtual object in a task group depletes its energy the set of rest

virtual objects take the duty of keeping the operation continuity of that task group. On the other hand, CBATA selects only one virtual object in each task group. This lead to faster depletion of a virtual object's energy and thus shorter operational period. In Figures 5.4, the average numbers of rounds of all task groups are 1848,05; 339,45 and 66 for ETAP1, ETAP2 and CBATA. In Figures 5.5, the values are 1086,25; 216,05 and 40,8. In both scenarios the huge superiority of ETAP1 over both ETAP2 and CBATA is very obvious.

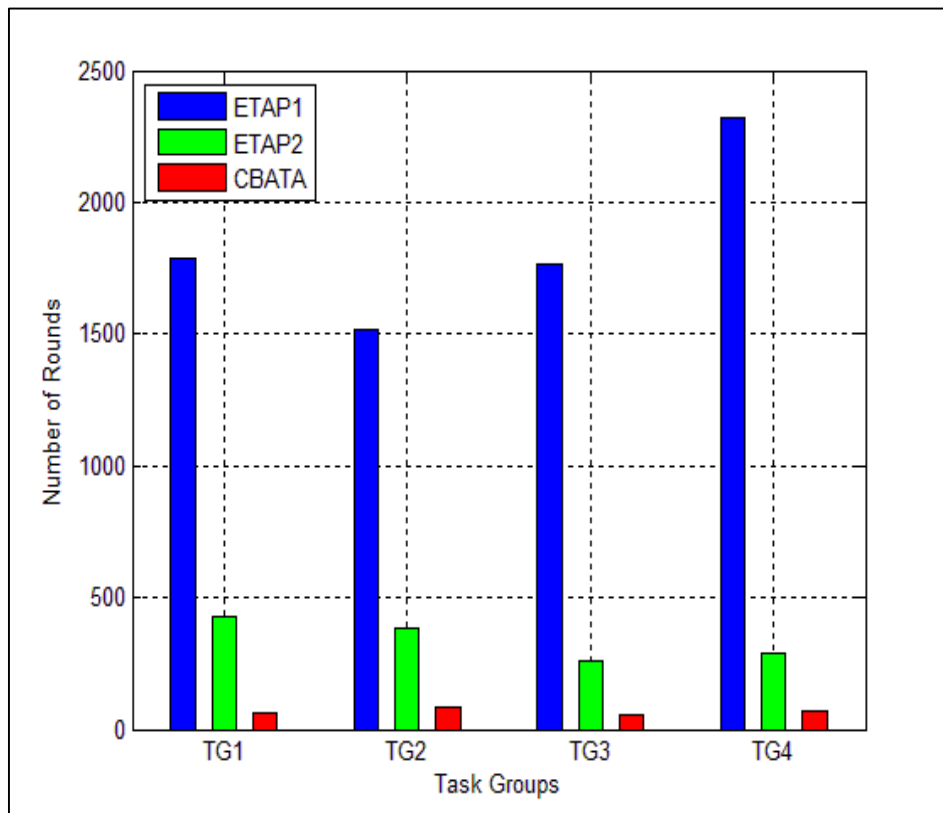


Figure 5.4. Average number of rounds until each task group becomes non-operational for objects density = 250

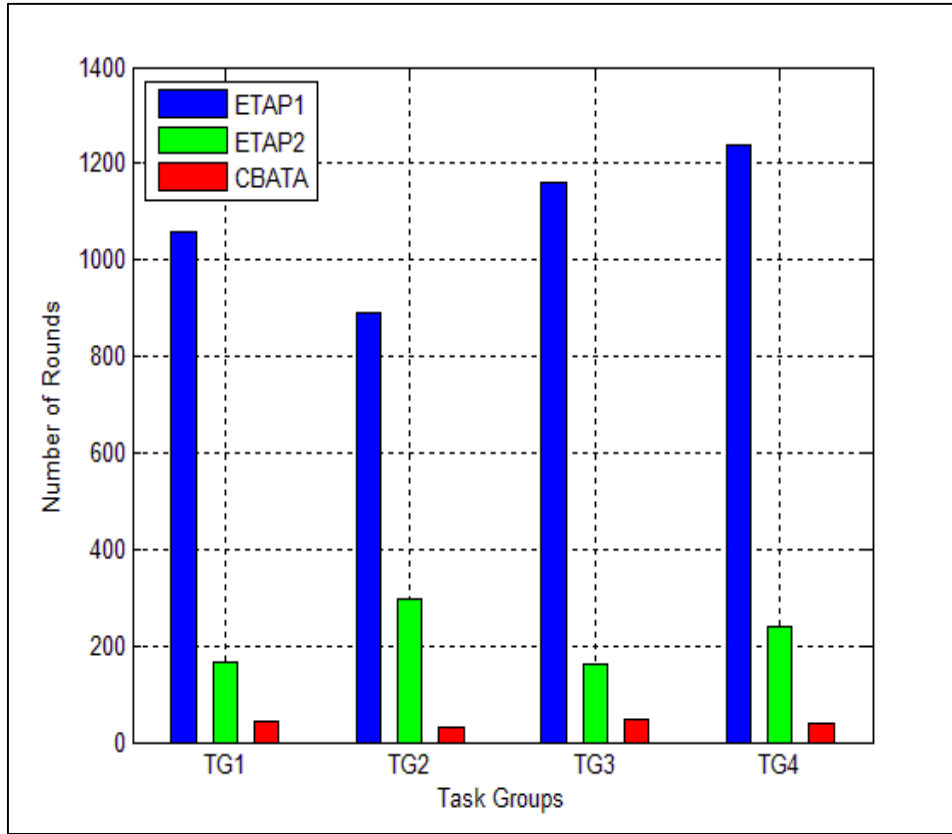


Figure 5.5. Average number of rounds until each task group becomes non-operational for objects density = 500

Protocols evaluation for evolutionary algorithm quality

For the sake of demonstrating the quality of the proposed evolutionary algorithms, the convergence to optimal solution metric is employed. This evaluation metric states that, solutions must evolve and converge to an optimal solution in an organized manner throughout the process of evolutionary algorithm loop [151]. In this context, Figures 5.6 and 5.7 visualizes quality of the proposed evolutionary algorithms regarding to convergence metric Figure 5.6 shows the best solution in each generation as well as in initial population in terms of fitness values for ETAP1. Figure 5.7 depicts the same results for ETAP2. The results are averaged over 5 test instances with default parameter settings. From the results, one can see that both ETAP1 and ETAP2 provide high-quality solutions that evolve orderly at each generation towards optimal solution.

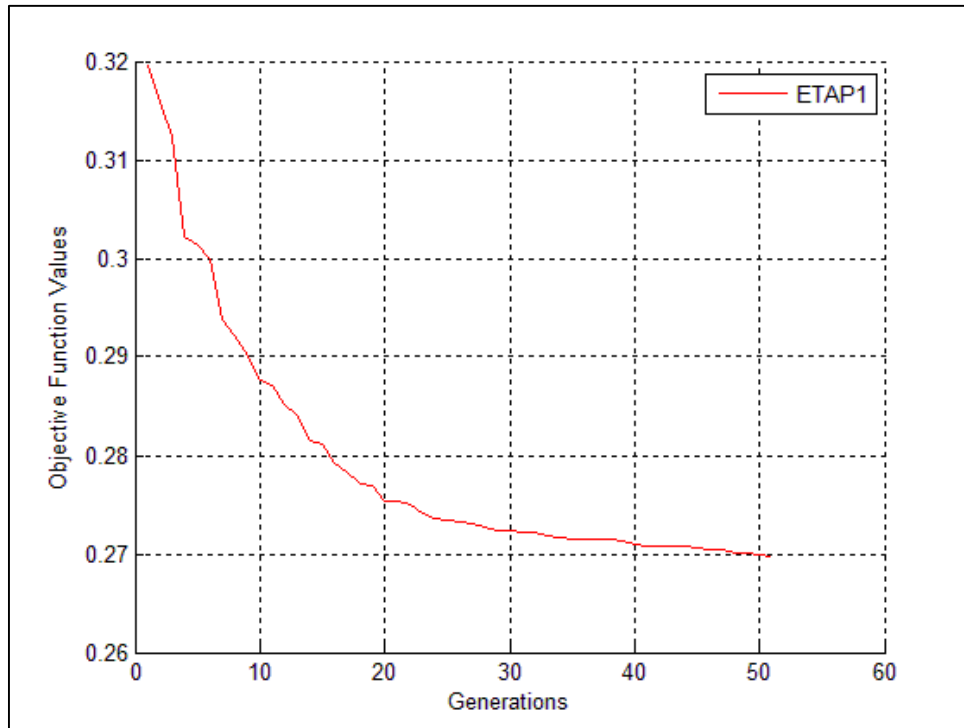


Figure 5.6. Convergence of ETAP1 toward optimal solution for 50 generations + initialization phase

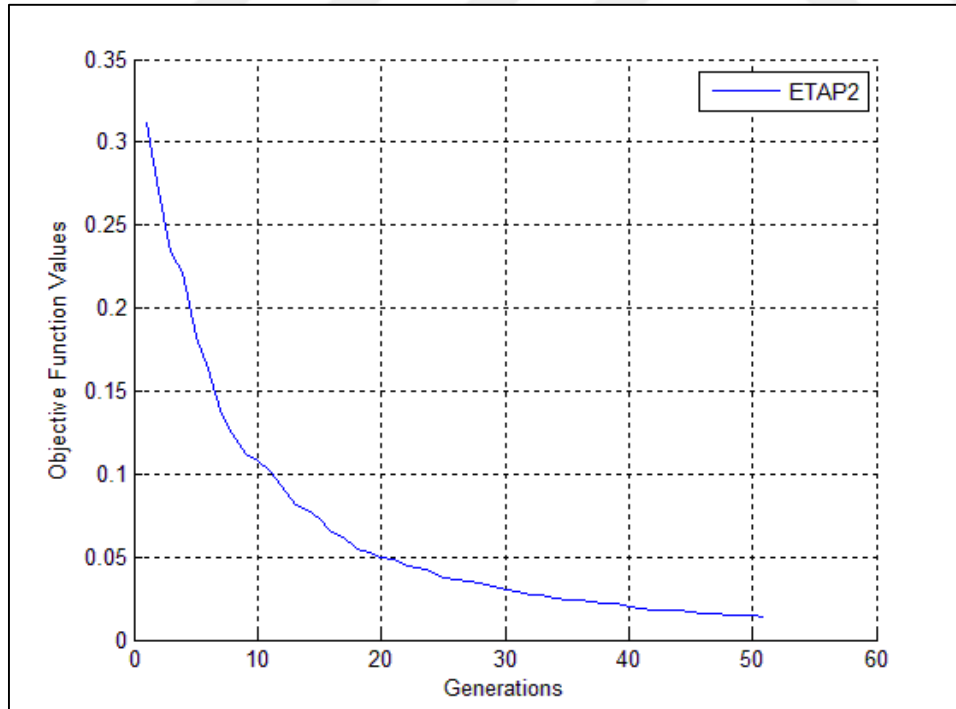


Figure 5.7. Convergence of ETAP2 toward optimal solution for 50 generations + initialization phase

5.1.5. Results of SETAP1 and SETAP2 (The effect of energy aware heuristic on ETAP1 and ETAP2)

The goal of the heuristics injected to ETAP1 and ETAP2 insure that only the objects with energy levels greater than average energy level in each task group are allowed to be virtual objects. These heuristics increase the mean value of energy of the virtual objects in each task group. Table 5.9 supports this claim. This table presents the mean value of virtual object energies for each task group. In this simulation the results are calculated in each case for 5 simulation settings with network dimension = 200×200 , the number of objects = 250 and the number of tasks = 8. Then, each simulation setting is executed 5 times and the results are averaged over execution results and the best results are shown in italic font. Table 5.9 reports that the mean values of virtual object energies in SETAP1 and SETAP2 are greater than other protocols. The table also reports that the heuristics injected to ETAP1 and ETAP2 are succeed in raising energy levels of virtual objects. It is worth to note that in some task groups (e.g., task group 3 and 5) the mean values of virtual objects energies in CBATA are greater than the values in ETAP1 and ETAP2. However, ETAP1 and ETAP2 with their less energy consumptions (as shown in Table 5.6) success in achieving better energy efficiency. The last two columns in the table summarize the gain percentage of ETAP1 and ETAP2 over CBATA.

Table 5.9. The mean value of virtual objects energies (in joules) for 8 task groups

Task Group	SETAP1	SETAP2	ETAP1	ETAP2	CBATA	SETAP1 gain over CBATA	SETAP2 gain over CBATA
1	<i>1,6442</i>	1,4930	1,2384	1,2963	1,0595	35,56%	29,03%
2	1,6618	<i>1,7226</i>	1,3788	1,2644	1,1089	33,27%	35,62%
3	1,5469	<i>1,5654</i>	1,2546	1,2831	1,3547	12,42%	13,45%
4	1,5789	<i>1,6022</i>	1,2442	1,1101	1,0612	32,78%	33,76%
5	<i>1,5193</i>	1,4717	1,1890	1,1456	1,3836	08,93%	05,98%
6	1,6364	<i>1,6655</i>	1,3353	1,2912	0,7998	51,12%	51,97%
7	1,6433	<i>1,6462</i>	1,3069	1,0990	1,1134	32,24%	32,36%
8	1,6002	<i>1,6689</i>	1,1974	1,0742	0,8924	44,23%	46,52%

Figure 5.8 and Figure 5.9 depicts the performance of the protocols in terms of stability periods. In these figures the execution of the protocols are break up into rounds supposing that all tasks are needed to be performed at each round in a continuous and a uniform

manner. The results are obtained for the scenarios of object's density = 250 (Figure 5.8) and object's density = 500 (Figure 5.9) to show the number of rounds until each task group becomes unstable (i.e., until a virtual object within a task group depletes its energy). In both Figures, it is clear that SETAP1 achieves longer stability periods in all task groups compared to other protocols. On the other hand, SETAP2 with its injected heuristics competes with ETAP1 and outperforms its unmodified version (ETAP2) as well as CBATA.

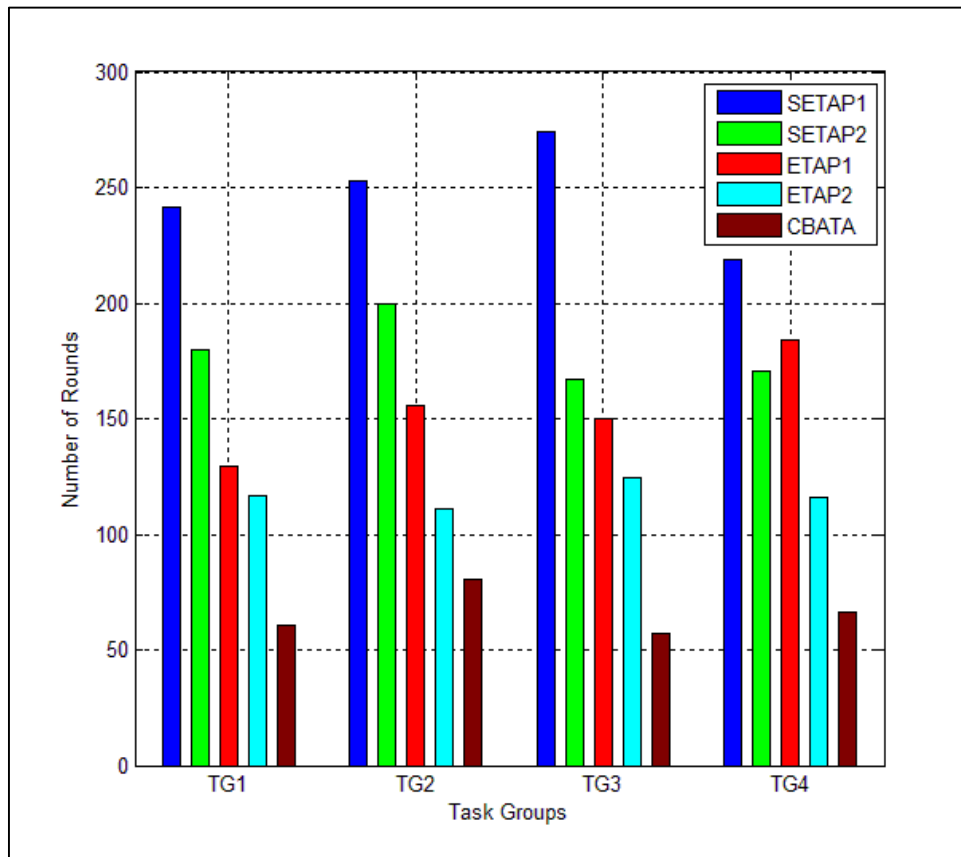


Figure 5.8. Number of rounds until each task group becomes unstable for network with objects density = 250

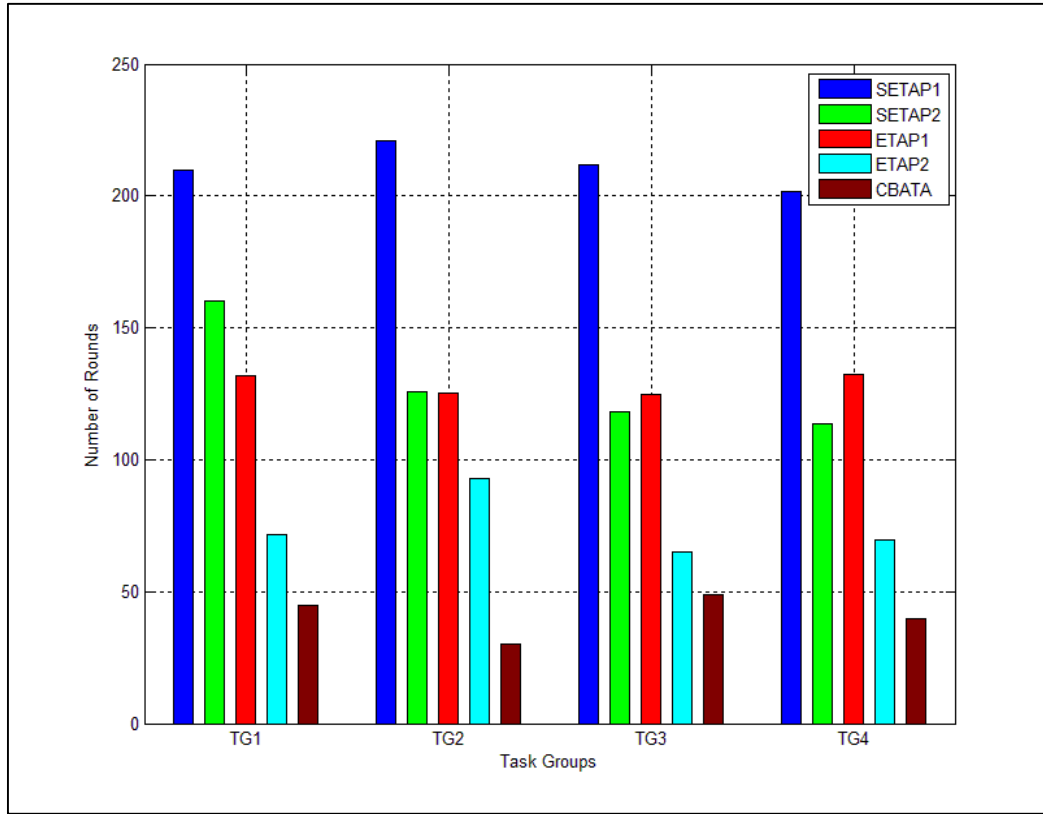


Figure 5.9. Number of rounds until each task group becomes unstable for network with objects density = 500

As it has been shown in Figures 5.8 and 5.9, SETAP1 and SETAP2 perform better than their unmodified versions (i.e., ETAP1 and ETAP2) in terms of extending the length of stability periods. However, to give a detailed view of the effect of injected heuristics in term of energy efficiency, Tables 5.10, 5.11 and 5.12 report the result of SETAP1 and SETAP2 in terms of dissipated energy in the network for different network scales, different number of objects, and different number of tasks, respectively. For completeness we attached the results of ETAP1 and ETAP2 from Tables 5.3, 5.4 and 5.5 with the corresponding Tables of 5.10, 5.11 and 5.12. It is clear that although the heuristics succeeded in providing longer stability periods they negatively affected the performance of the protocols in terms of total dissipated energy in the network. Realizing this fact, the motivations of ETAP1 and ETAP2 in one hand and the motivations of SETAP1 and SETAP2 in another hand for different application categories become more convenient.

Table 5.10. Average dissipated energy (in joules) in 5 test instances for different network scales in a single round

Network Side Length	SETAP1	ETAP1	SETAP2	ETAP2
100	0,14023	0,13647	0,22515	0,22353
125	0,16656	0,15889	0,28501	0,27794
150	0,19374	0,19032	0,34584	0,34504
175	0,23135	0,22679	0,43974	0,43837
200	0,28048	0,26496	0,65902	0,65772
225	0,32169	0,31443	0,66559	0,66562
250	0,37376	0,36874	0,80028	0,78384

Table 5.11. Average dissipated energy (in joules) in 5 test instances for different number of objects in a single round

Number of Objects	SETAP1	ETAP1	SETAP2	ETAP2
200	0,23055	0,22026	0,49939	0,49525
250	0,27613	0,26999	0,57787	0,51884
300	0,30963	0,30875	0,64980	0,64170
350	0,35412	0,34148	0,79989	0,80132
400	0,39349	0,38107	0,89376	0,86787
450	0,42281	0,41409	0,93579	0,93076
500	0,44989	0,44988	0,96525	0,95843
550	0,48923	0,48696	1,01431	1,00990
600	0,52764	0,52240	1,16562	1,15840

Table 5.12. Average dissipated energy (in joules) in 5 test instances for different number of tasks in a single round

Number of Tasks	SETAP1	ETAP1	SETAP2	ETAP2
2	0,21984	0,21515	0,42857	0,40125
4	0,27690	0,26837	0,61775	0,58609
6	0,32231	0,29768	0,66596	0,59308
8	0,34534	0,32584	0,61710	0,56590
10	0,38426	0,34815	0,68931	0,68737

5.2. Performance Analysis of Protocols in Context of Scenario #2

This subsection presents the simulation results of MOTAP and STAP comparing their performance against the most relevant algorithm in the literature, CBATA [8, 12] and the improved version of CBATA, M-CBATA. In comparisons several evaluation metrics related to computational power optimization, energy efficiency, duration of the stability and operational periods, and quality of the evolutionary algorithm are used. Moreover, in order to evaluate the quality of the proposed multi-objective optimization algorithm, additional measurements such as the diversity maintained in the non-dominate optimal set and the consistent convergence to the optimal solutions are considered.

In the following subsections the simulations tools and environment that are used to obtain the results are illustrated. Then, the parameter settings for network energy model and the evolutionary algorithm are defined. Finally, the evaluation results and discussions of MOTAP, STAP, M-CBATA and CBATA are investigated by presenting the results of simulations obtained by MATLAB simulator and then those obtained by OMNeT++ simulator.

5.2.1. Simulation environment

In the simulations, we developed a custom designed simulation environment in MATLAB [170, 171] to compute theoretical simulation results. For more complex and realistic application layer simulations, we used the network simulation framework OMNeT++ [172]. Both MATLAB and OMNeT++ results verify each other. In the simulations, a bi-directional communication links and 10% packet error rate are assumed. In case of a link failure, up to 5 times retransmission mechanism is implemented.

As the inputs to our simulators, 5 random network topologies are generated. In each network topology, n objects are randomly deployed in an $M \times M$ unit area and the internet coverage for these objects is provided by an access point \mathbb{AP} located at the center of the area. Each object is equipped with an energy resource and its level is randomly selected in the range $[0,5; 2]$ joule. Moreover, each object is attached with a processing unit of a clock speed value that is randomly chosen between 16 and 322 MHz. Here, the assumed 16 MHz and 322 MHz are the processing unit frequency of Arduino YÚN microcontroller

[173] and MIPS M5100 Core Processor [174], respectively. Both Arduino YÚN and MIPS M51xx Core Processor family are ideal for wide range of IoT, M2M, wearable, and other embedded and real-time applications [173, 174]. Finally, it is assumed that each object is assumed to be capable of performing up to T tasks with each task ID randomly selected from range of $[1, T]$.

Our simulation scenarios are mainly divided into three groups according to dimension of the area (M), objects density (n), and number of tasks that the network is capable of performing (T). Unless otherwise stated, the default settings for the simulations are $n = 200$, $M = 200$ unit, and $T = 4$. In order to generate different simulation scenarios the value of each parameter is varied while fixing values of other two parameters. In order to maintain the probabilistic feature of the evolutionary algorithms, each of the 5 network topologies is execute 5 times and the average result for each topology is obtained over these executions. Then, the final result for each setting is calculated by averaging the results of the networks. The network side length, M takes 7 different values starting with 100 up to 250 with an incremental value of 25. The number of objects in the network (n) takes 8 different values (50; 100; 150; 200; 250; 300; 350; and 400). Finally, the value of the number of tasks that the network could perform is changed to 5 different values from 2 to 10 with incremental value of 2. Overall, the simulation performs 500 tests for each protocol.

5.2.2. Parameters and rules settings for network energy model and the evolutionary task allocation algorithms

The network energy model parameters take values identical to the values in Section 5.1.2. Table 5.13 lists the energy model parameters.

Table 5.13. Network energy model parameters settings

Energy for transceiver circuit (E_{elec})	50 nJ/bit
Energy for amplifier (ϵ_{amp})	100 pJ/bit/m ²
Energy for data aggregation (E_{DA})	5 nJ/bit
Data message size	4000 bit
Indication message size	8 bit

In the simulation, the CCS is assumed to be aware of its location and those of the IoT objects. Moreover, the CCS is supposed to be equipped with unlimited amount of energy. Finally, both CCS and the IoT objects are supposed to be stationary during the simulation.

The parameters' settings of evolutionary algorithm used in STAP are similar to those listed in Table 5.2 with only difference in number of generations. In STAP the population is allowed to evolve for 100 generations. On the other hand, the particular rules and control parameters settings that used to complete the characteristics of MOTAP are: non-domination and crowding distance concept based selection, two-point multilevel crossover with 0,6 recombination probability (p_c), bit flop mutation with mutation probability (p_m) = 0,03 and Mutation rate (m_r) = 0,02, population reduction that adopts elitism scheme based on non-domination and crowding distance concept and population size (N) of 50 individuals allowed to evolve for 100 generations. Table 5.14 summarizes the particular rules and parameters' settings of MOTAP.

Table 5.14. MOTAP parameters and rules

Population size (N)	50
Termination criteria	Number of generations = 100
Recombination operator	2-point multilevel cut and cross fill
Mutation operator	Bit Flop Mutation (BFM)
Recombination probability (p_c)	0.6
Mutation probability (p_m)	0.03
Mutation rate (m_r)	0.02
Reduction selection	Elitism scheme based on non-domination and crowding distance concept

5.2.3. Results of simulation (by MATLAB simulator)

The next subsections evaluate the performance of STAP, MOTAP and M-CBATA against CBATA within the platform of MATLAB simulator. The evaluation considers several metrics in context of energy efficiency, duration of the stability and operational periods, computational power optimization, quality of the evolutionary algorithm, and computation time.

Protocols evaluation for energy efficiency

Tables 5.15, 5.16 and 5.17 illustrate the performance of the protocols from the perspective of total energy consumed in the network to allocate tasks to the objects and retrieve the results. The results in Tables 5.15, 5.16 and 5.17 are obtained under the default simulation setting values with varying network dimensions, number of objects and number of networks tasks for the tables respectively. In these tables the best results are indicated by italic font. From the results it is clear that both STAP and MOTAP exhibit comparable performance and the best results are exchange between them. Moreover, both (i.e., STAP and MOTAP) outperform CBATA and its modified version M-CBATA. Thus, STAP and MOTAP consume less energy for task allocation compared to CBATA and M-CBATA. STAP with its objective function in Equation 4.28 and MOTAP with its second objective in Equation 4.36 are very successful in minimizing the total energy that is dissipated in the network for the operation of task allocation. Another observation from the results is that both CBATA and M-CBATA are generally equal at their performance. The reason is that both algorithms form the selection of virtual objects in a blind manner regarding to energy consumptions. The main advantage of M-CBATA over CBATA is that it selects different virtual objects for each task group by distributing the load of being virtual object and extends the operational and stability periods.

Table 5.15. Average dissipated energy (in joules) in 5 test instances for different network scales

Network Side Length	STAP	MOTAP	M-CBATA	CBATA
100	<i>0,28115</i>	0,301468	0,91704	0,94636
125	<i>0,32667</i>	0,339224	1,18560	1,26910
150	<i>0,37996</i>	0,392548	1,70760	1,66090
175	<i>0,44361</i>	0,445192	2,17110	1,89030
200	0,54295	<i>0,523368</i>	2,89860	3,14310
225	<i>0,62468</i>	0,629360	3,92000	3,92200
250	0,80566	<i>0,701096</i>	4,36240	4,28320

Table 5.16. Average dissipated energy (in joules) in 5 test instances for different object density settings

Number of Objects	STAP	MOTAP	M-CBATA	CBATA
50	0,25015	0,19262	0,80095	0,66806
100	0,32248	0,32029	1,29830	1,58540
150	0,43639	0,42126	2,34400	1,76020
200	0,53060	0,52460	2,91530	2,76400
250	0,64588	0,62822	3,51570	3,55150
300	0,70987	0,72843	4,46310	4,70110
350	0,79829	0,80395	5,03240	4,69960
400	0,87933	0,90029	5,45800	5,88720

Table 5.17. Average dissipated energy (in joules) for 5 test instances for different number of tasks

Number of Tasks	STAP	MOTAP	M-CBATA	CBATA
2	0,28596	0,28219	1,48160	1,99890
4	0,52582	0,52658	2,63270	2,61580
6	0,89300	0,86055	3,70190	3,77790
8	1,42970	1,35360	5,35640	5,99170
10	2,16930	1,89750	6,33310	5,54200

Another considerable evaluation metric is scalability. It is obvious from results in Tables 5.15, 5.16 and 5.17 that both STAP and MOTAP handle the scalability in a more efficient manner than M-CBATA and CBATA. For example, in Table 5.15 increasing the dimension of the network from 100×100 unit to 250×250 unit increases the energy consumption in STAP by 0,52451 joule and in MOTAP by 0,399628 joule. This value dramatically increases in M-CBATA and CBATA to 3,44536 and 3,33684 joules, respectively. Another example is captured from Table 5.16. In this table increasing the number of objects by 350 objects magnifies energy consumption by 0,62918 joule in STAP and by 0,399628 joule in MOTAP. Whereas, the energy spent magnification in M-CBATA is 3,44536 joule and in CBATA is 3,33684 joule. This can be attributed to behavior of STAP and MOTAP which degrade the energy load on each task group by selecting several virtual objects for each task group. On the other hand, regardless of the number of objects, M-CBATA and CBATA select one virtual object for each task group. This behavior of M-CBATA and CBATA magnifies the energy load in each task group.

The results in Figures 5.10, 5.11 and 5.12 qualitatively supports the previous observations. The figures visualize the results in Tables 5.15, 5.16 and 5.17 respectively. It is clear from Figure 5.10 and Figure 5. 11, increasing the density of the objects, the dimension of network or the tasks that the network is capable of performing increase the energy consumption of the network in an uniform manner.

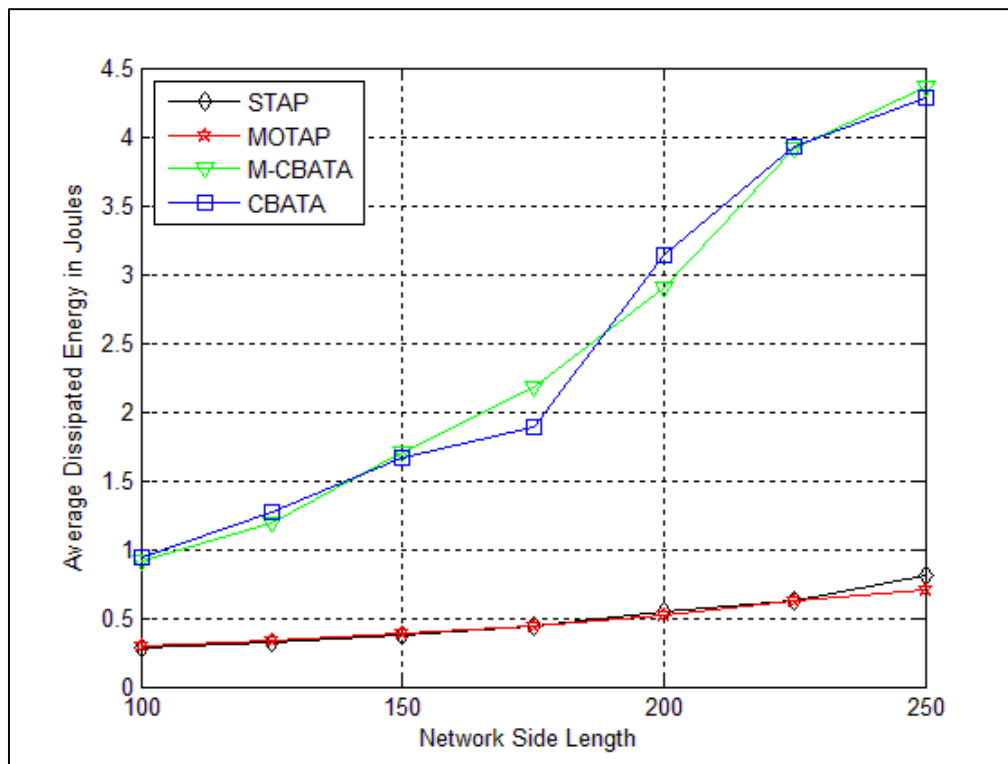


Figure 5.10. Average dissipated energy in 5 test instances for different network dimensions

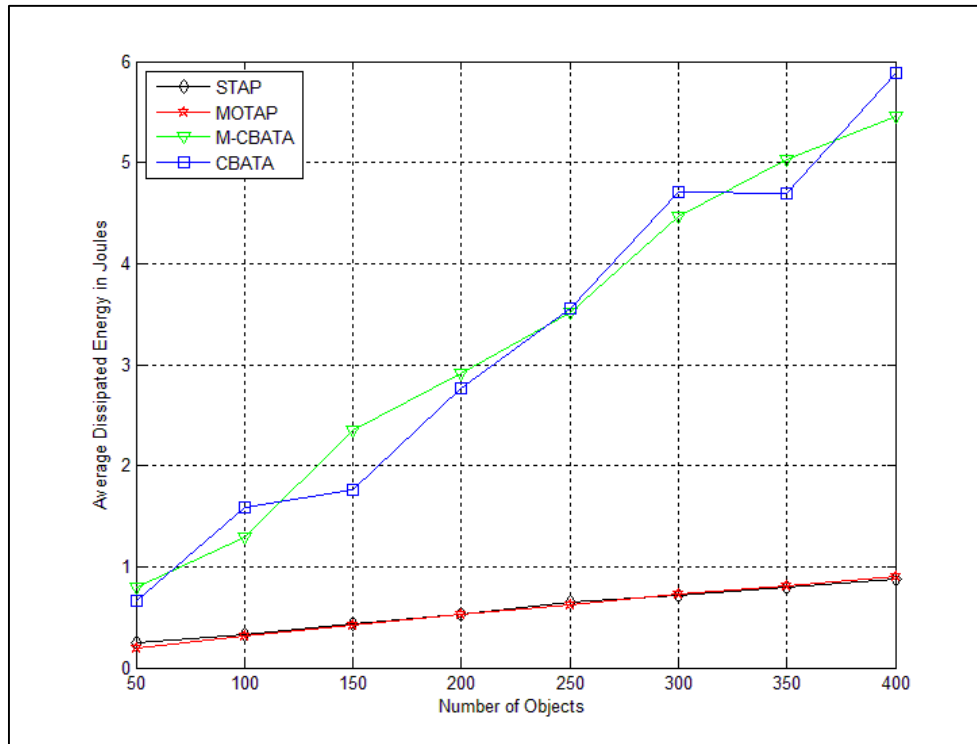


Figure 5.11. Average dissipated energy in 5 test instances for different number of objects

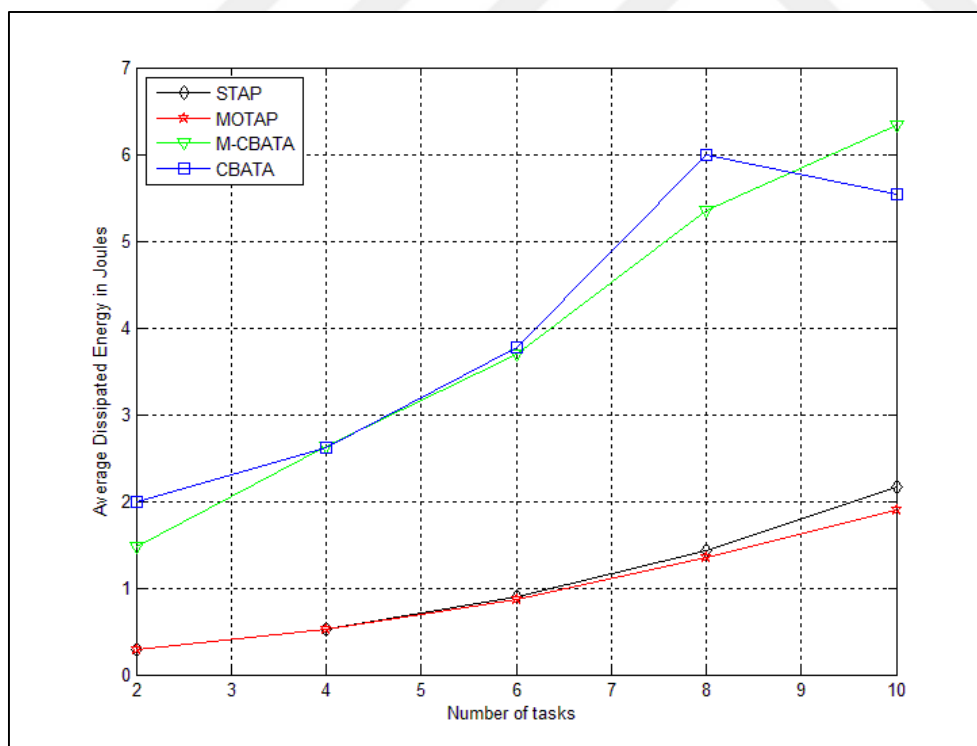


Figure 5.12. Average dissipated energy in 5 test instances for different number of tasks

A detailed view on energy dissipation in each task group is presented in Figure 5.13 and Table 5.18. In these results, the network side length is 200 units, the number of objects is

200, and the number of tasks is six. Moreover, the best results in Table 5.18 are indicated by italic font. The results clearly align with the previous results.

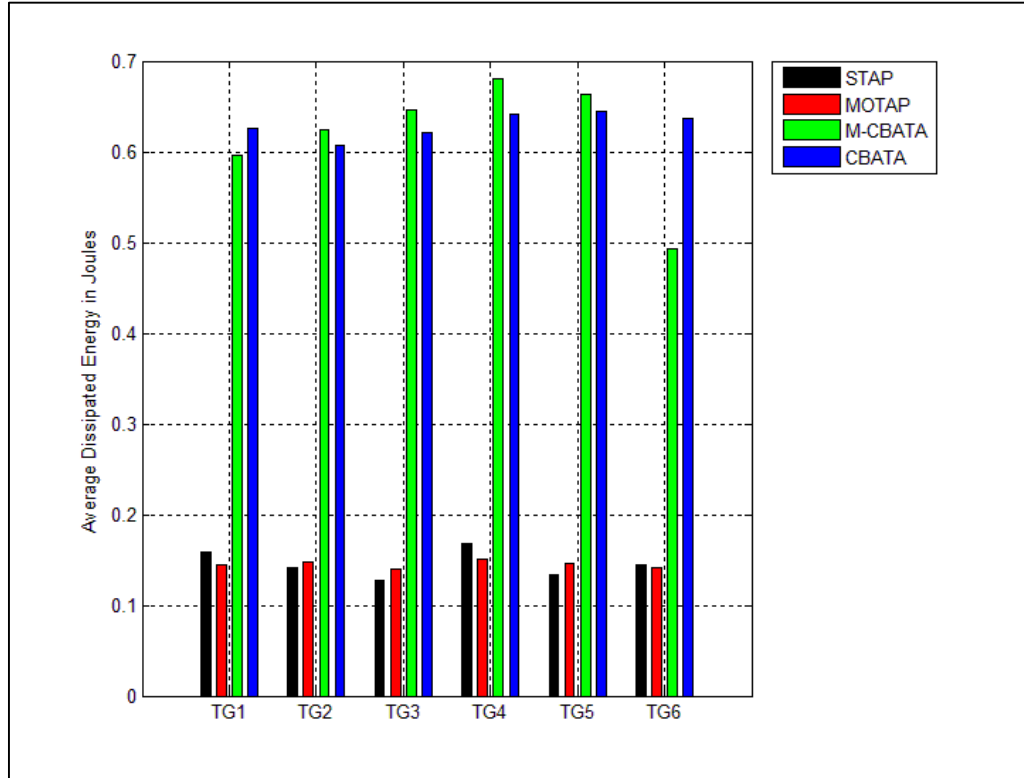


Figure 5.13. Average dissipated energy in each task group

Table 5.18. Average dissipated energy (in joules) for 5 test instances in each task group

Task Group#	STAP	MOTAP	M-CBATA	CBATA
1	0,15851	<i>0,14495</i>	0,59535	0,62550
2	<i>0,14111</i>	0,14814	0,62389	0,60718
3	<i>0,12807</i>	0,14005	0,64606	0,62173
4	0,16863	<i>0,15092</i>	0,68090	0,64204
5	<i>0,13412</i>	0,14649	0,66332	0,64438
6	0,14468	<i>0,14184</i>	0,49240	0,63707

Protocols evaluation for operational and stability periods

The length of the operational and stability periods is directly determined by the amount of energy and by the energy efficiency of virtual objects. Greater amount of energy coupled with optimal use of the virtual objects' available energy lead to extension in both operational and stability periods. Tables 5.19, 5.20, and 5.21 and their qualitative

representations in Figures 5.14, 5.15, and 5.16 depicts the average energy consumption of virtual objects with networks default settings and different network dimensions, different numbers of objects, and different number of networks tasks, respectively. The best results in Tables 5.19, 5.20, and 5.21 are specified by italic font.

Table 5.19. Average dissipated energy of virtual objects (in joules) for different network scales

Network Side Length	STAP	MOTAP	M-CBATA	CBATA
100	<i>0,0023014</i>	0,0044194	0,028795	0,042191
125	<i>0,0026217</i>	0,0043884	0,029080	0,070841
150	<i>0,0030248</i>	0,0047299	0,029503	0,047350
175	<i>0,0034822</i>	0,0050028	0,030014	0,105300
200	<i>0,0041447</i>	0,0056077	0,031352	0,084352
225	<i>0,0043909</i>	0,0063564	0,033352	0,051141
250	<i>0,0049590</i>	0,0069147	0,033868	0,066381

Table 5.20. Average dissipated energy of virtual objects (in joules) for different object density settings

Number of Objects	STAP	MOTAP	M-CBATA	CBATA
50	<i>0,0034104</i>	0,0039320	0,010541	0,022122
100	<i>0,0035954</i>	0,0046025	0,016913	0,037430
150	<i>0,0037572</i>	0,0052908	0,024930	0,058994
200	<i>0,0039846</i>	0,0055409	0,031903	0,068875
250	<i>0,0042673</i>	0,0063651	0,039015	0,140560
300	<i>0,0044266</i>	0,0062358	0,046221	0,163070
350	<i>0,0045405</i>	0,0067405	0,053408	0,167610
400	<i>0,0047949</i>	0,0066480	0,060386	0,170330

Table 5.21. Average dissipated energy of virtual objects (in joules) for different number of tasks

Number of Tasks	STAP	MOTAP	M-CBATA	CBATA
2	<i>0,0029236</i>	0,0054821	0,036027	0,054051
4	<i>0,0038314</i>	0,0055912	0,030888	0,058851
6	<i>0,0052032</i>	0,0065012	0,028655	0,074666
8	<i>0,0060637</i>	0,0071712	0,028824	0,102160
10	<i>0,0074432</i>	0,0083327	0,027983	0,112120

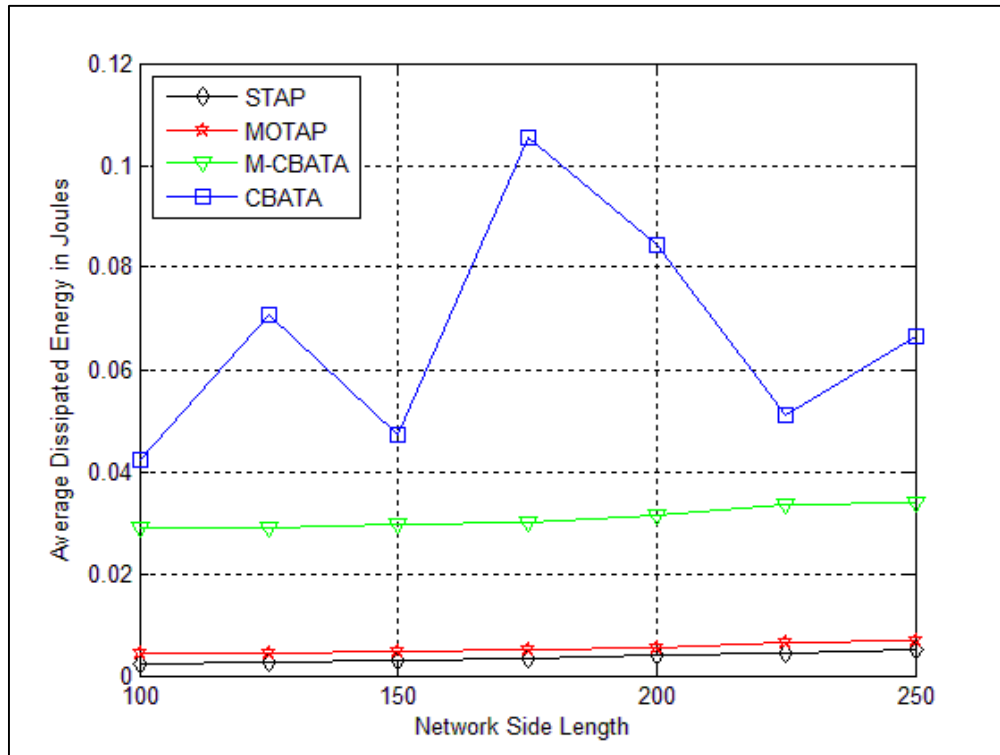


Figure 5.14. Average dissipated energy of virtual objects for different network dimensions

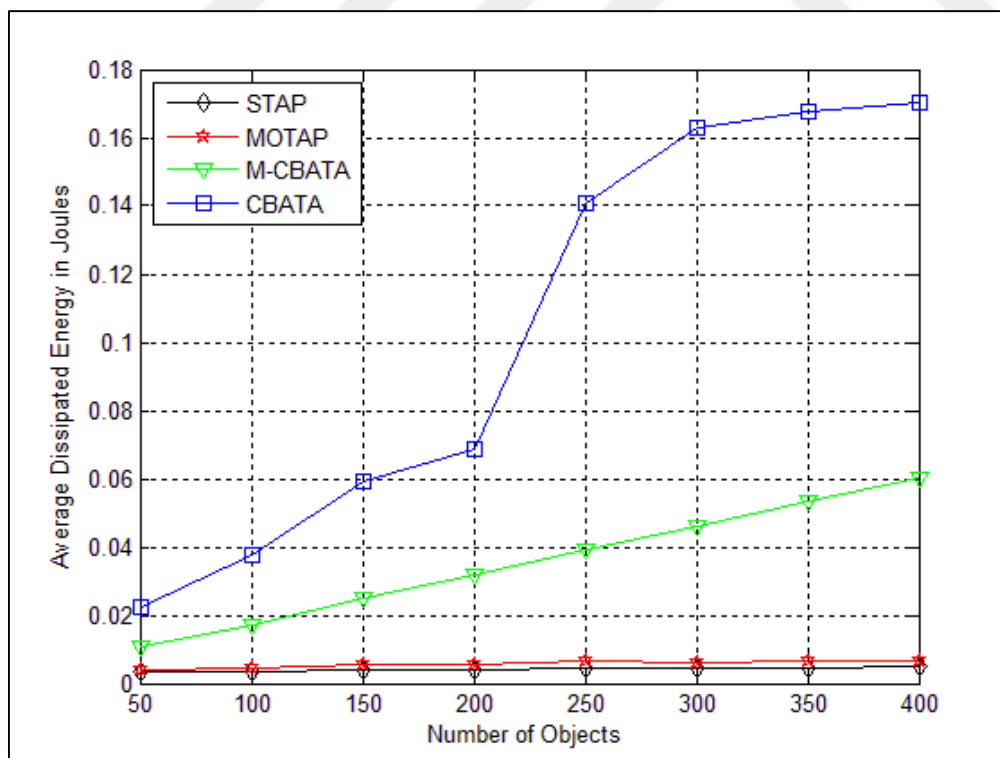


Figure 5.15. Average dissipated energy of virtual objects for different number of objects

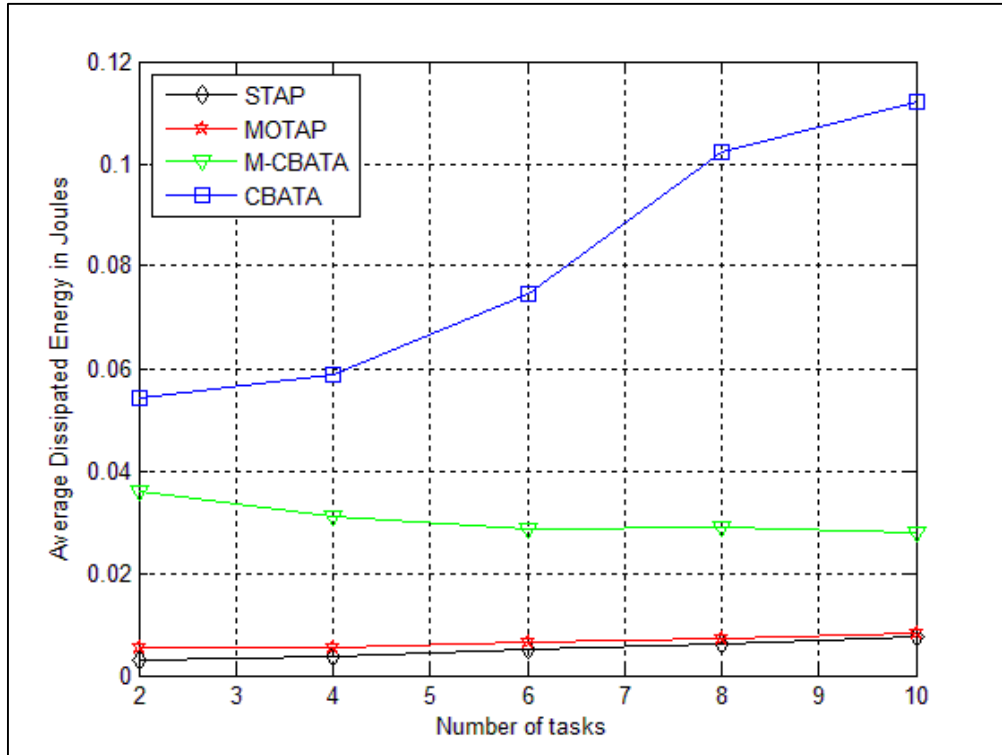


Figure 5.16. Average dissipated energy of virtual objects for different number of tasks

It is obvious from the results in Tables 5.19, 5.20, and 5.21 and Figures 5.14, 5.15, and 5.16 that STAP and MOTAP significantly perform better than both M-CBATA and CBATA. Also it is clear that STAP slightly outperforms MOTAP. The reason is that STAP is a single objective EA that has one goal of minimizing energy consumption of task allocation and optimizing the energy efficiency of virtual objects. On the other hand, MOTAP is MOEA that has two conflicted objectives of optimizing the energy efficiency of task allocation and utilizing the operational power of virtual objects. In MOTAP, none of these objectives can further optimize without deteriorating the other objectives.

Another observation from these results is the advantage of the M-CBATA over CBATA. In CBATA there is big opportunity for an object to serve as virtual object at several task groups. This behavior magnifies energy spent of virtual objects. In contrast, in M-CBATA it is more likely for objects to be virtual objects in reduced number of task groups compared to CBATA regardless of the number of tasks that the object is capable of performing. This behavior minimizes the work load on individual virtual objects and results in reduced energy consumption. This observation becomes very clear in Table 5.21 and Figure 5.17. Thus, increasing number of tasks at the same number of objects increases the number of tasks groups. This for CBATA means that an object could be serving within

larger number of task groups. Whereas in M-CBATA, since there is only one virtual object in each task group increasing number of virtual objects increases the number of clusters in the network which in turn means fewer number of objects at each task group that the virtual objects have to serve.

One of the primary goals of STAP is selecting virtual objects with leveraged energy levels. This goal yields by the energy aware heuristic that are injected in mutation and population initialization phase. As can be seen from Figure 5.17 and Table 5.22 this goal is efficiently satisfied. The results show the energy of virtual objects in each task group for number of tasks = 6. In Table 5.22 the best results are indicated by italic font. STAP by selecting virtual objects with higher residual energy outperforms other rival protocols.

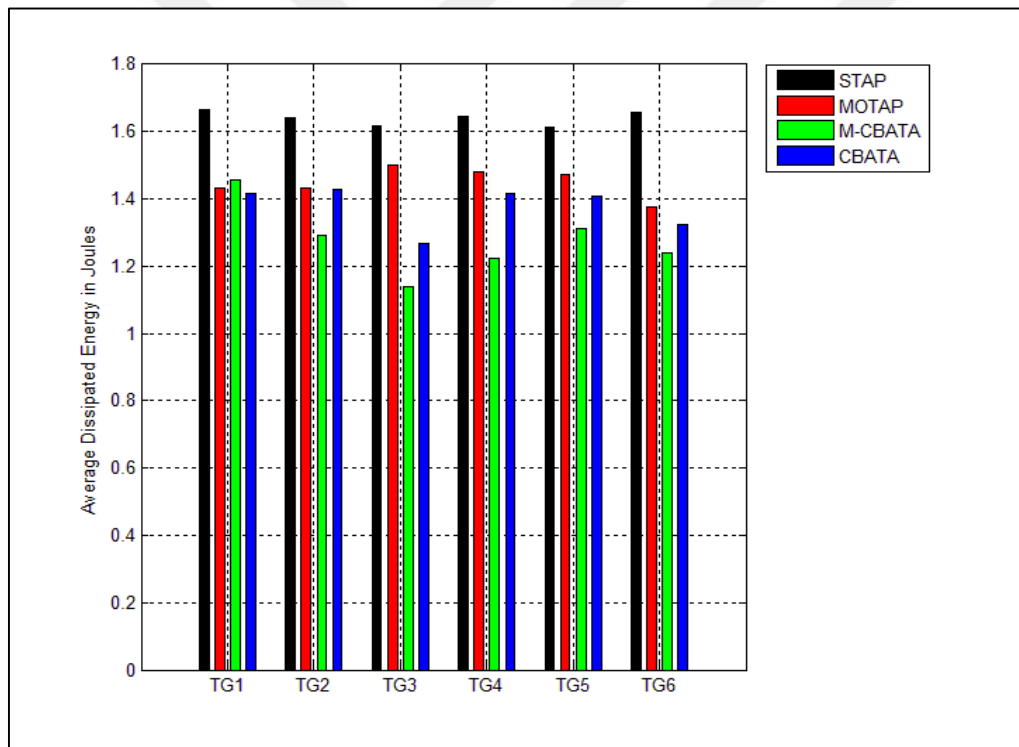


Figure 5.17. Average energy of virtual objects in each task group for number of objects = 200, network dimensions = 200 × 200 unit and number of tasks = 6

Table 5.22. Average energy of virtual objects (in joules) in each task group for number of objects = 200, network dimensions = 200×200 unit and number of tasks = 6

Task Group#	STAP	MOTAP	M-CBATA	CBATA
1	1,6635	1,4331	1,4581	1,4166
2	1,6399	1,4317	1,2898	1,4290
3	1,6173	1,4996	1,1409	1,2679
4	1,6436	1,4816	1,2244	1,4178
5	1,6141	1,4725	1,3113	1,4079
6	1,6564	1,3775	1,2413	1,3245

The results in Tables 5.23 and 5.24 and Figures 5.18 and 4.19 are obtained by breaking the execution of tasks into rounds and assuming the most extreme scenario where queries for execution of all tasks arrive in a continuous and a uniform manner at each round. Table 5.23 and Figure 5.18 depict the average length of the operational periods of each task group. Whereas, Table 5.24 and Figure 5.19 show the average length of stability period of each task group. The results in Tables 5.23 and 5.24 are captured for networks of 200×200 unit dimension, 4 tasks and different number of objectives. On the other hand, Figures 5.18 and 4.19 are obtained for the default setting network. The results come in align with the previous results and as can be seen STAP yields better performance in terms of stability and operational periods compared to other protocols. MOTAP with its energy efficient objective function comes at the second rank. On the other hand, M-CBATA with its modifications that minimize the work load on individual virtual objects and results in reduced energy consumption outperforms CBATA. It is worth to note that, in M-CBATA and CBATA operational and stability periods are identical since they select one virtual object at each task group.

Table 5.23. Average number of rounds until each task group becomes non-operational for different object density

Number of Objects	STAP				MOTAP			
	TG1	TG2	TG3	TG4	TG1	TG2	TG3	TG4
50	601,80	738,40	562,00	1558,2	553,6	890,6	972,0	632,6
100	1161,6	928,00	1194,2	1037,8	928,8	654,0	791,2	661,8
150	924,00	1527,6	1488,4	955,8	600,2	598,6	652,4	515,2
200	1189,8	1998,8	1584,8	1884,6	601,4	840,2	597,6	689,8
250	1955,6	1312,4	996,00	1617,4	486,6	605,0	519,4	388,4
300	1361,4	1308,0	1315,6	1656,8	643,6	477,8	354,4	706,4
350	1290,4	1321,6	1344,0	766,60	650,2	443,0	564,6	429,8
400	902,40	1404,0	1707,4	1333,4	546,4	554,0	922,6	466,8
Number of Objects	M-CBATA				CBATA			
	TG1	TG2	TG3	TG4	TG1	TG2	TG3	TG4
50	93,4	174,2	127,2	121,8	70,0	51,8	79,2	34,6
100	70,8	51,60	79,20	71,40	55,0	34,2	35,0	48,0
150	43,4	47,60	53,80	53,40	31,4	24,0	32,0	25,0
200	43,6	47,80	29,80	45,40	25,2	13,6	23,0	11,4
250	33,8	29,00	32,80	38,80	8,40	8,20	16,0	7,80
300	26,8	28,60	26,80	25,00	8,20	7,40	7,40	7,60
350	16,0	23,20	22,60	33,20	13,8	10,6	8,80	16,4
400	23,0	21,80	21,60	20,20	8,60	10,0	12,0	7,40

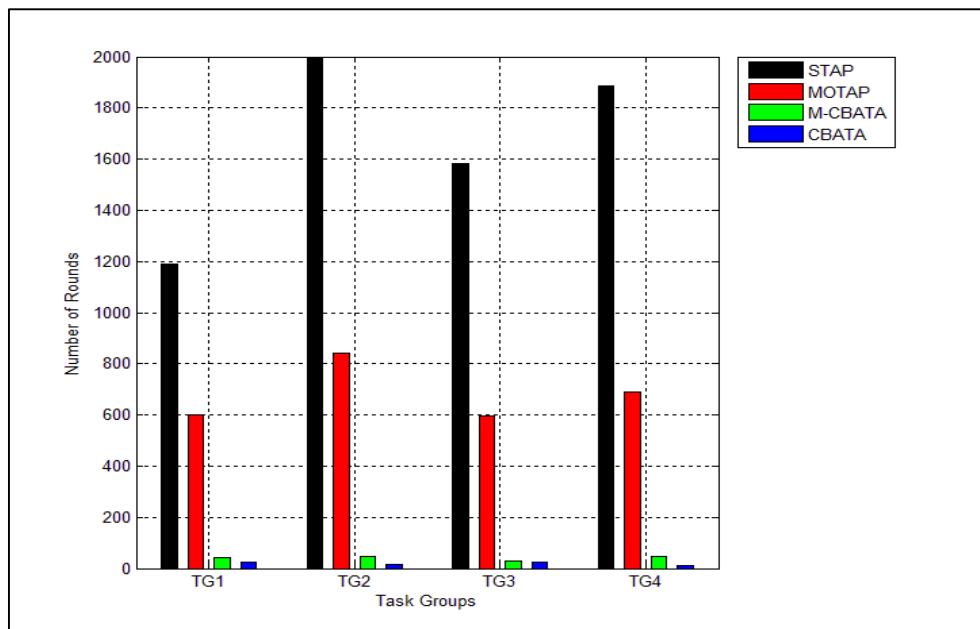


Figure 5.18. Average number of rounds until each task group becomes non-operational

Table 5.24. Average number of rounds until each task group becomes unstable for different object density

Number of Objects	STAP				MOTAP			
	TG1	TG2	TG3	TG4	TG1	TG2	TG3	TG4
50	293,0	343,2	312,4	438,6	198,2	195,8	205,4	247,4
100	308,4	226,4	291,0	250,6	183,4	181,0	144,6	192,0
150	237,4	221,6	272,6	224,4	187,0	158,6	154,0	162,8
200	229,2	221,2	227,8	208,4	118,6	109,0	169,8	107,6
250	191,2	197,2	207,2	207,8	123,2	129,0	108,0	136,4
300	194,4	192,8	186,4	161,8	106,0	151,4	100,4	97,00
350	189,0	185,8	177,8	193,6	105,4	91,60	122,8	77,40
400	190,4	171,8	166,4	167,2	74,60	91,40	108,8	95,60

Number of Objects	M-CBATA				CBATA			
	TG1	TG2	TG3	TG4	TG1	TG2	TG3	TG4
50	93,4	174,2	127,2	121,8	70,0	51,8	79,2	34,6
100	70,8	51,60	79,20	71,40	55,0	34,2	35,0	48,0
150	43,4	47,60	53,80	53,40	31,4	24,0	32,0	25,0
200	43,6	47,80	29,80	45,40	25,2	13,6	23,0	11,4
250	33,8	29,00	32,80	38,80	8,40	8,20	16,0	7,80
300	26,8	28,60	26,80	25,00	8,20	7,40	7,40	7,60
350	16,0	23,20	22,60	33,20	13,8	10,6	8,80	16,4
400	23,0	21,80	21,60	20,20	8,60	10,0	12,0	7,40

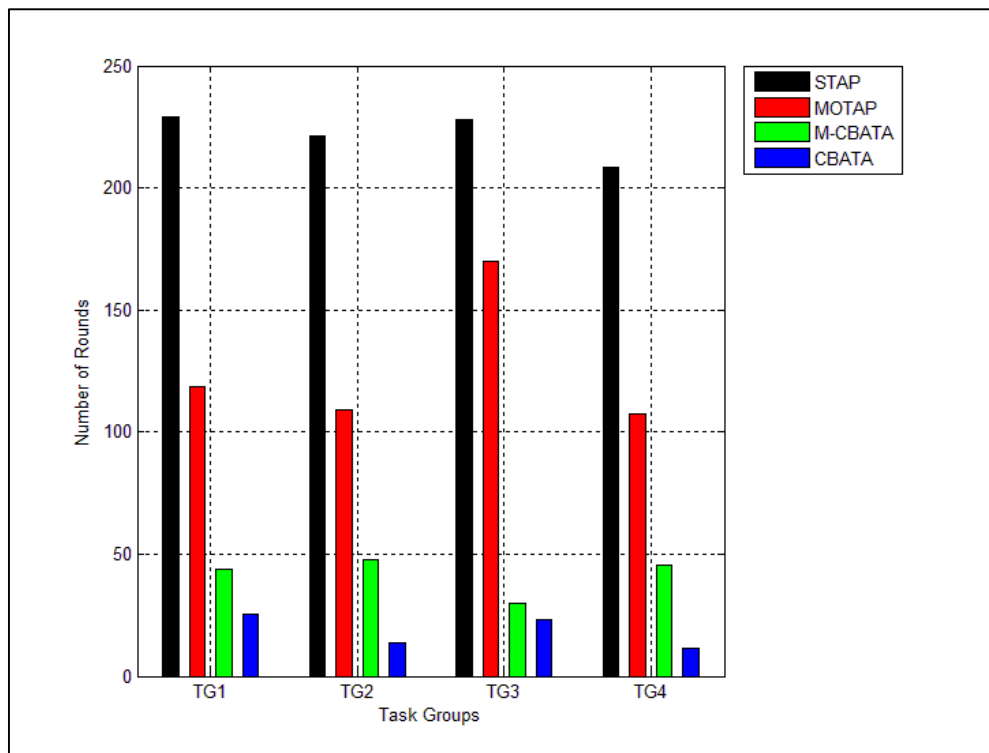


Figure 5.19. Average number of rounds until each task group becomes unstable

Protocols evaluation for computational power

MOTAP considers the fact that virtual objects are the basic units of operations and attempts to select objects with high processing unit capacities as virtual objects. Table 5.25 and Figure 5.20 illustrate the mean value of the processing units' frequencies for the virtual objects at each task group when the number of tasks is six. The best results in Table 5.25 are in italic font. MOTAP with its first objective function in Equation 4.31 succeeds at increasing the mean value of proceeding power of virtual objects compared to STAP, M-CBATA and CBATA.

On the other hand, except task group six, M-CBATA selects the virtual objects with higher computation power relative to CBATA. However, both M-CBATA and CBATA do not consider computation power when they select virtual objects. It is more likely for M-CBATA to select objects with different levels of processing power at different task groups. In CBATA, an object with a low capacity processing power could be selected as virtual object at several task groups.

Table 5.25. Average processing power of virtual objects (in MHz) in each task group for number of tasks = 6

Task Group#	STAP	MOTAP	M-CBATA	CBATA
1	161,47	<i>298,47</i>	163,20	149,00
2	174,58	<i>302,07</i>	197,00	103,80
3	175,39	<i>296,00</i>	209,00	145,60
4	164,84	<i>305,13</i>	140,20	137,80
5	165,57	<i>305,28</i>	209,00	127,80
6	165,78	<i>303,02</i>	106,20	137,80

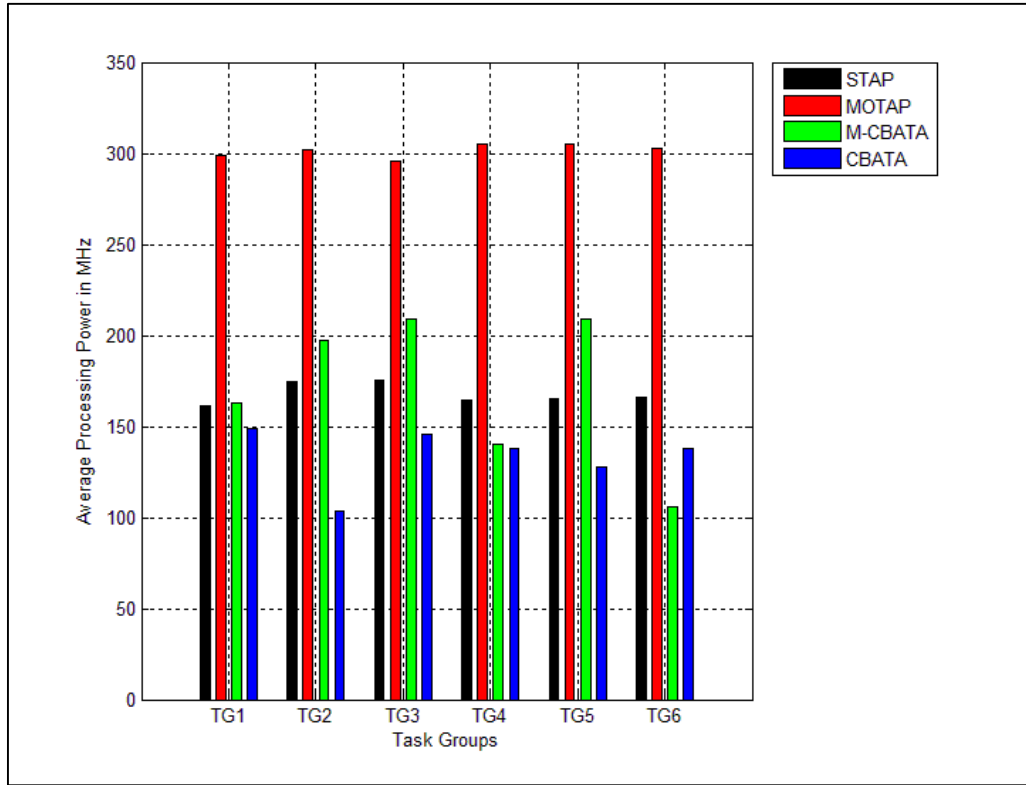


Figure 5.20. Average processing power of virtual objects in each task group for number of tasks = 6

Protocols evaluation for evolutionary algorithm quality

In order to demonstrating the quality of the proposed evolutionary algorithm in STAP, we employed convergence to optimal solution metric. Figure 5.21 visualizes quality of the proposed evolutionary algorithm in STAP regarding to convergence metric. The figure shows the best solution in each generation as well as in initial population in terms of fitness values. The results are obtained for a network under the default parameter settings (i.e., network dimensions= 200×200 unit, number of objects = 200, and number of networks' tasks = 4). From the results, one can see that STAP provides high-quality solutions that evolve in an organized manner at each generation towards optimal solution. It is worth to mention that in Figure 5.21 STAP evolved 56 times throughout the process of evolutionary algorithm.

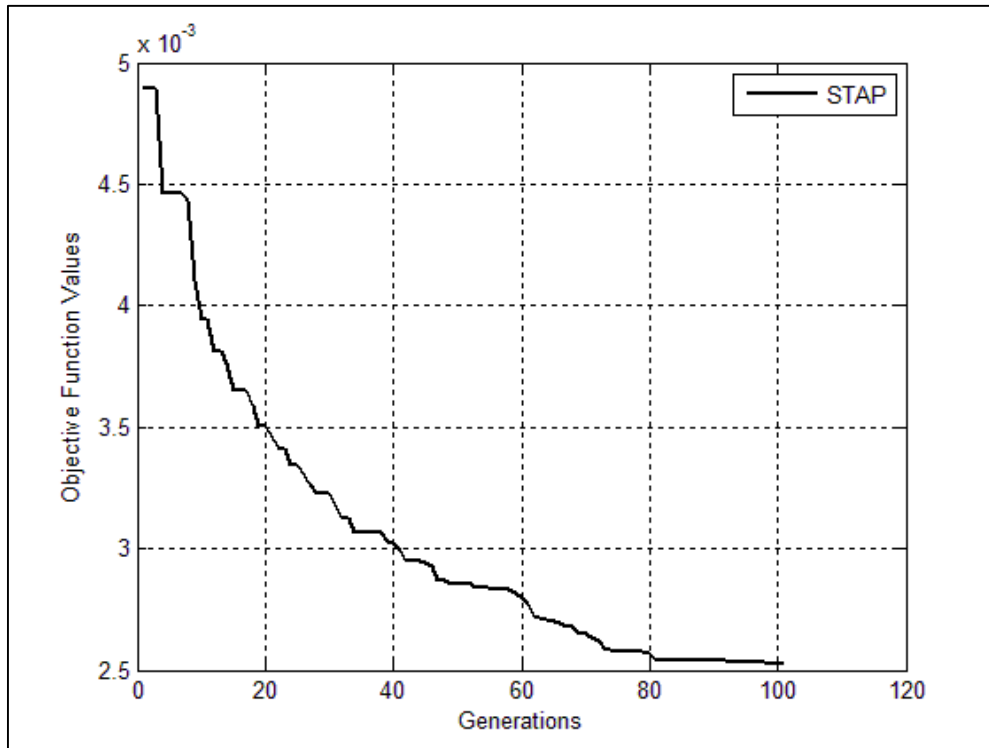


Figure 5.21. Convergence of evolutionary algorithm in STAP toward optimal solution for 100 generations + initialization phase

Figures 5.22 and 5.23 demonstrate the efficiency of MOTAP as a multi objective optimization algorithm. Both figures are obtained for a network with the default parameter settings. Figures 5.22(a) and (b) depict the convergence to optimal solution metric for first and second objectives of MOTAP respectively. On the other hand, Figure 5.23 shows the non-dominated solutions that are provided by MOTAP for one network. The x, y —coordinates correspond to computation power and energy efficiency objectives, respectively. We can observe from the results in Figure 5.22 and 5.23 that MOTAP provides high-quality solutions that maintain the diversity and covers the whole range between the two conflicted objectives.

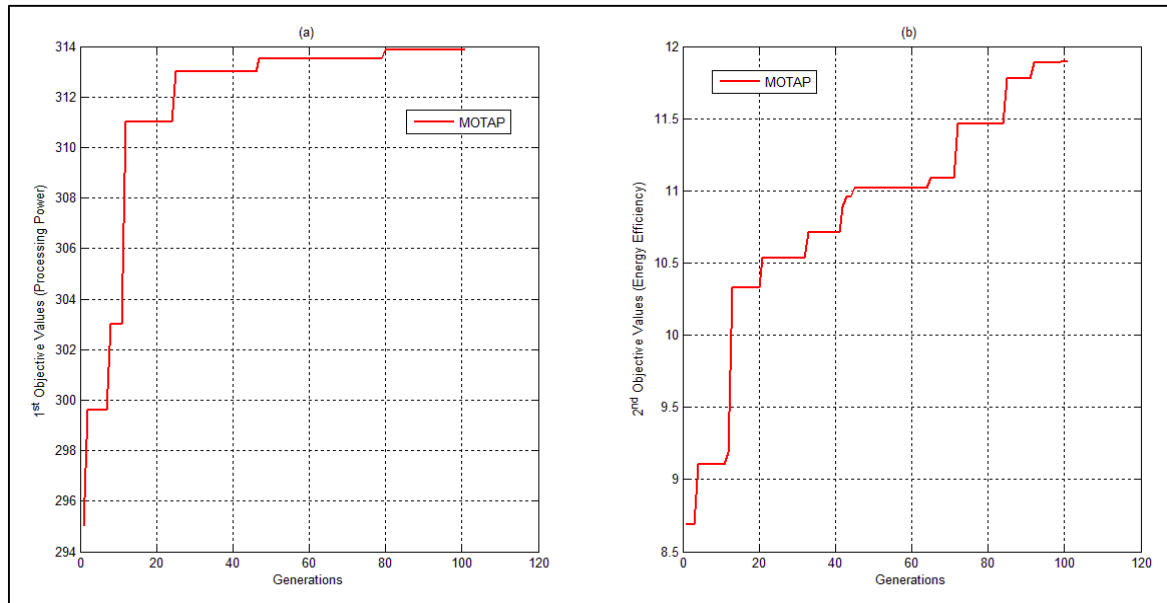


Figure 5.22. Convergence of evolutionary algorithm in MOTAP toward optimal solution for 100 generations + initialization phase (a): Convergence of first objective (processing power) (b): Convergence of second objective (energy efficiency)

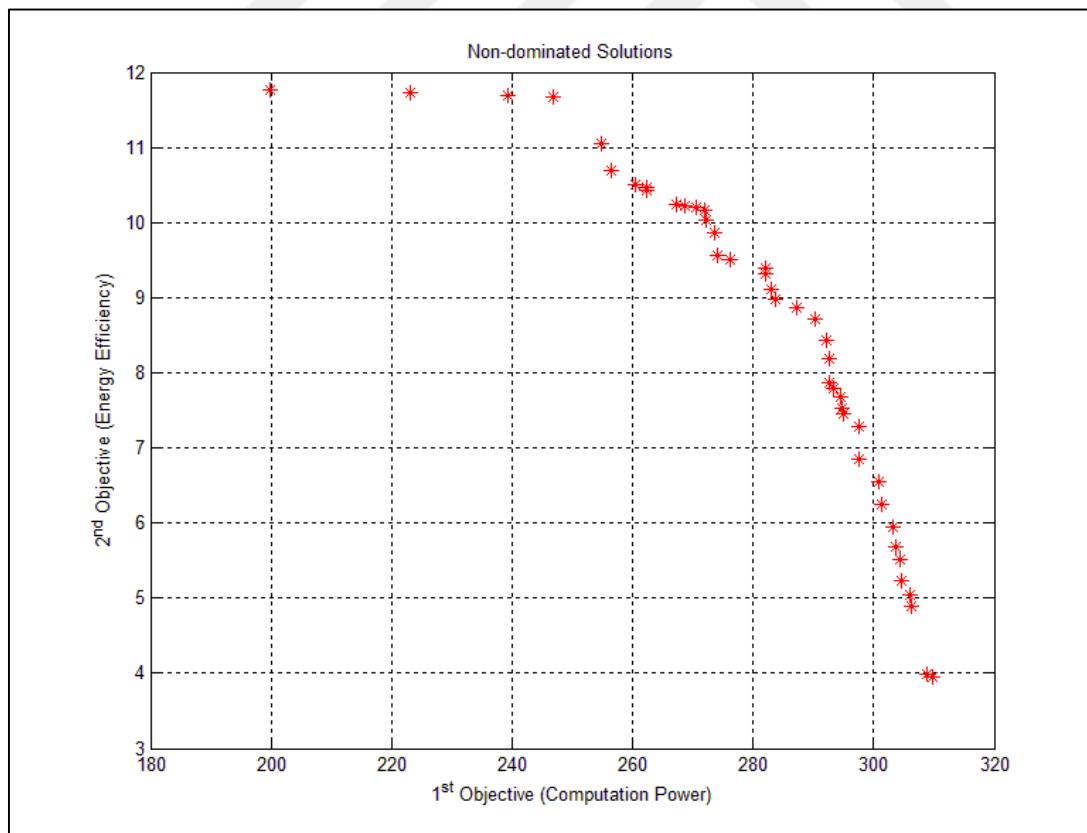


Figure 5.23 Non-dominated solutions of MOTAP for one network

Protocols evaluation for computation time

When using Intel Core i5 CPU 2.27GHz, EA-based protocols (STAP and MOTAP) take additional time to construct the task allocation configuration. This result comes naturally because these protocols handle more than one solution (as the case in M-CBATA and CBATA). For example, for 50 different individuals to be evolved in 100 generations STAP needs to process 5000 alternative solutions. The situation even worse in MOTAP which needs to evaluate two different objectives and perform domination based operations. In contrast, M-CBATA and CBATA constructs single solution for task allocation problem. Tables 5.26, 2.27, and 5.28 present the computation times for the protocols for different network scales, different number of objects, and different number of tasks. As it is expected, M-CBATA and CBATA execute in a shorter time compared to STAP and MOTAP. However, considering the technological advances of today's microprocessors and their super speeds the huge difference in execution times could be ignored using high end systems.

Table 5.26. Time duration (in seconds) for each protocol for different network scales

Network Side Length	STAP	MOTAP	M-CBATA	CBATA
100	103,821699	112,514876	0,051928	0,051238
125	107,032598	117,837922	0,056326	0,055142
150	103,427216	116,732613	0,053658	0,053233
175	96,1988790	119,688508	0,051328	0,051163
200	102,714103	121,264129	0,053716	0,055185
225	104,558569	121,676684	0,052952	0,055348
250	99,3646170	116,606447	0,054154	0,052953

Table 5.27. Time duration (in seconds) for each protocol for different number of objects

Number of Objects	STAP	MOTAP	M-CBATA	CBATA
50	20,462271	49,439396	0,015476	0,014475
100	46,970431	72,025544	0,028854	0,027756
150	71,300757	94,569715	0,038907	0,040615
200	102,714103	121,264129	0,053716	0,055185
250	135,925822	144,246529	0,066062	0,066095
300	166,067310	164,381075	0,077712	0,077904
350	206,766771	196,526132	0,093373	0,094018
400	249,616927	235,196962	0,109013	0,108227

Table 5.28. Time duration (in seconds) for each protocol for different number of tasks

Number of Tasks	STAP	MOTAP	M-CBATA	CBATA
2	71,289202	86,850474	0,031913	0,034245
4	102,714103	121,264129	0,053716	0,055185
6	126,075914	147,545239	0,074587	0,074004
8	148,242618	171,841962	0,092857	0,090801
10	182,077525	210,954588	0,114471	0,113979

It is obvious that increasing the number of generation will negatively affect the computation time. Although, the increased number of generations could lead to better solutions by exploring wider regions of search space. Therefore, a trade-off between the quality of the solutions and the computation time should be considered when selecting the number of generations. Table 5.29 demonstrates the vales of the fitness function in STAP and MOTAP for different number of generations. The results are obtained for the default parameter setting of the simulation. It is clear that the generation number of 100 could better maintain this trade-off.

Table 5.29. STAP and MOTAP objective functions' values for different number of generations

Number of Generations	Minimization	Maximization	
	Φ_{STAP}	$\Phi_{MOTAP_{\mathcal{F}_1}}$	$\Phi_{MOTAP_{\mathcal{F}_2}}$
25	0,0030212	294,8000	10,6614
50	0,0027608	286,6667	11,5872
75	0,0024912	308,4545	11,2626
100	0,0024666	316,2000	12,1810
150	0,0025090	317,7143	11,6423
200	0,0023897	301,5833	12,7094
500	0,0022055	318,4167	12,8769

5.2.4. Results of simulation (by OMNeT++ simulator)

To validate the results obtained by Matlab simulator and to give a deeper view of the performance of the protocols, application layer based simulations are performed by executing STAP, MOTAP, M-CBATA, and CBATA using a powerful network simulator, OMNeT++. The simulations consider a realistic scenario where the executions of tasks are broken into several rounds and at each round a continuous and a uniform arrival of tasks is supposed. To verify the protocols in a most extreme and dramatic case, we assumed that all tasks must be performed at each round. Finally, considering the complexity of the simulation the protocols are applied on a network with 25 objects. The dimensions of the networks is set to 100×100 units and the number of tasks is set 4. Then, five network topologies are generated and the results are averaged over these topologies. Figure 5.24 shows the number of rounds until each task group becomes non-operational (i.e., until all virtual objects completely consume their energies). Table 5.30 demonstrates the operational time of each task group for each of the generated 5 network topologies and attached with the table the average results of all networks. As it is expected, STAP outperforms other protocols. On the average of all task groups, STAP slightly outperforms MOTAP by 184,45 rounds. With a huge advantage STAP outperforms M-CBATA and CBATA by 922,25 and 1032,3 rounds respectively. On the other hand, MOTAP outperforms M-CBATA by 737,80 rounds and CBATA by 847,85 rounds. Finally, M-CBATA with its property of reducing the energy load on each virtual object outperforms CBATA by 110,05 rounds.

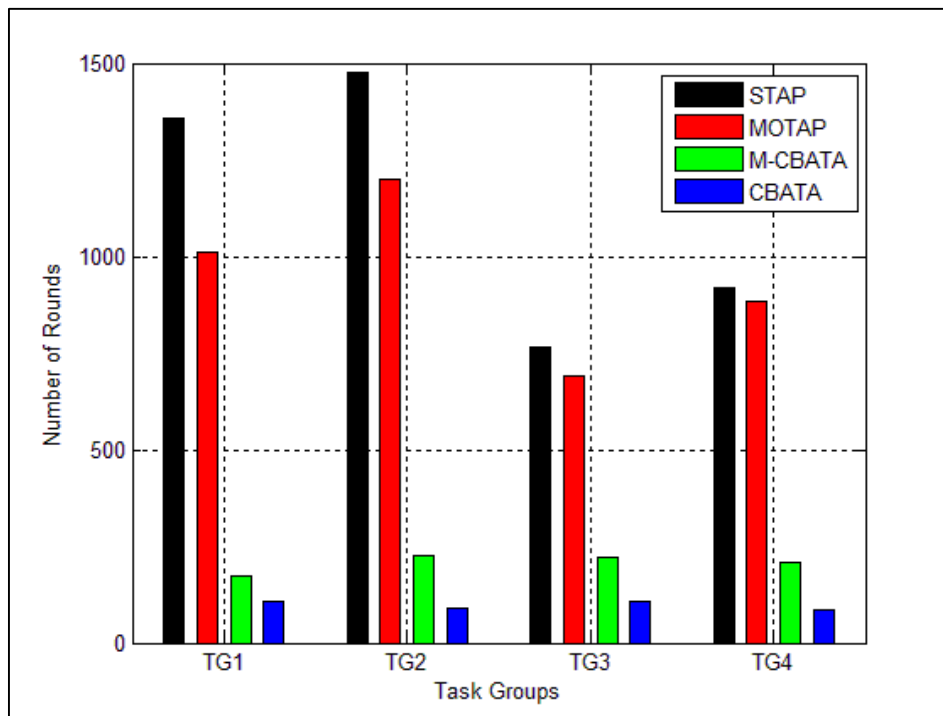


Figure 5.24. Number of rounds until each task group becomes non-operational

Table 5.30. Number of rounds until each task group becomes non-operational for 5 random generated networks

Network#	STAP				MOTAP			
	TG1	TG2	TG3	TG4	TG1	TG2	TG3	TG4
1	1231	1121	719	923	911	1564	452	792
2	1230	1622	934	875	1202	1508	672	796
3	1857	1898	850	1024	1527	1245	561	885
4	1434	880	551	1051	734	979	1147	845
5	1025	1867	774	730	678	697	611	1101
Average Results	1355,4	1477,6	765,60	920,60	1010,4	1198,6	688,60	883,80
Network#	M-CBATA				CBATA			
	TG1	TG2	TG3	TG4	TG1	TG2	TG3	TG4
1	320	135	164	124	66	66	66	66
2	63	328	195	392	146	145	136	136
3	172	270	229	297	79	78	78	78
4	57	139	366	133	93	93	92	92
5	253	267	159	88	156	64	156	64
Average Results	173,0	227,80	222,60	206,80	108,0	89,20	105,60	87,20

The performance of the protocols in terms of stability periods are shown in Figure 5.25 and Table 5.31. The results figure depicts the number of rounds until each task group becomes unstable (i.e., until a virtual object depletes its energy). Again, the best performance is captured by STAP. On the average of all task groups, STAP Compared to MOTAP produce longer stability periods by 86.40 rounds. Also, STAP outperforms M-CBATA and CBATA by 371,20 and 481,25 rounds respectively. On the other hand, MOTAP outperforms M-CBATA by 284,80 rounds and CBATA by 394,85 rounds. It is worth to note that since M-CBATA and CBATA select one virtual object for each task group their stability periods are similar to their operational periods.

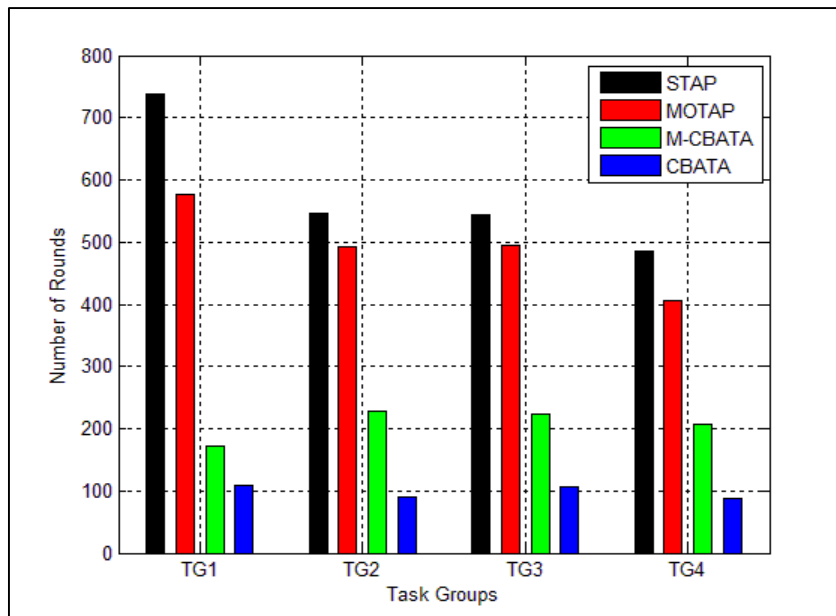


Figure 5.25. Number of rounds until each task group becomes unstable

Table 5.31. Number of rounds until each task group becomes unstable for 5 random generated networks

Network#	STAP				MOTAP			
	TG1	TG2	TG3	TG4	TG1	TG2	TG3	TG4
1	823	364	503	522	718	285	425	483
2	878	465	540	641	721	378	480	512
3	830	886	538	436	717	729	467	347
4	528	764	879	414	476	613	924	317
5	633	257	261	413	250	454	175	376
Average Results	738,40	547,20	544,20	485,20	576,40	491,80	494,20	407,0
Network#	M-CBATA				CBATA			
	TG1	TG2	TG3	TG4	TG1	TG2	TG3	TG4
1	320	135	164	124	66	66	66	66
2	63	328	195	392	146	145	136	136
3	172	270	229	297	79	78	78	78
4	57	139	366	133	93	93	92	92
5	253	267	159	88	156	64	156	64
Average Results	173,0	227,80	222,60	206,80	108,0	89,20	105,60	87,20

6. CONCLUSIONS AND FUTURE WORK

6.1. Conclusions

In this thesis the problem of task allocation is researched. The problem has been studied in the framework of IoT with the typical scenario where applications require the collaboration of different objects. The main challenges and constraints that make the task allocation in IoT an NP-hard are investigated. Then, to tackle with the problem task allocation and virtual object concept is adopted.

An important element in this thesis was the development evolutionary based task allocation protocols that take the different requirements of variety of IoT applications into consideration. In the first stage the thesis considered the scenario where objects are capable of performing only one task. Then, to increase the applicability of the proposed protocol a more complex and realistic scenario is assumed. In this scenario it is assumed that IoT objects have different skills and are capable of performing variety of tasks. Within the framework of the first scenario, for typical applications (e.g., smart environments) two protocols namely, Evolutionary Task Allocation Protocol-1 (ETAP1) and Evolutionary Task Allocation Protocol-2 (ETAP2) are developed. The main goal of ETAP1 and ETAP2 is to construct energy efficient task allocation topology by appropriate selection of virtual objects and suitable association of objects of each task group to the selected virtual objects. By accomplishing this goal, the operational period of each task group can be maximized to ensure extended network lifetime. Also within the framework of the first scenario, for security and reliability required applications (e.g., factory automation, environmental monitoring ...) the goals of ETAP1 and ETAP2 are redirected towarded stability awareness. The new derivative protocols are called Stabile Evolutionary Task Allocation Protocol-1 (SETAP1) and Stabile Evolutionary Task Allocation Protocol-2 (SETAP2). The basic idea behind these protocols is to inject energy aware heuristic into population initialization phase and into mutation operator to insure only objects with high residual energy are allowed to be virtual objects. These heuristic ensure a longer period of time before the first virtual object depletes its energy at each task group. On the other hand, in the context of the second scenario where objects are capable of performing different tasks and the task groups can intersect with each other two protocols are developed. The first protocol is

Steady Task Allocation Protocol (STAP). This protocol designed with stability is in mind. STAP attempts to maximize the mean value of virtual objects' energy whereas minimize their energy spends to insure longer stable performance. Finally, within the context of second scenario Multi-Objective Task Allocation Protocol (MOTAP) is proposed. The protocol considers the fact that each virtual object is a basic unit of computation. MOTAP attempts to build MOEA that maximize two contradicted objectives: computation power and energy efficiency. Also within the framework of this scenario, the most relative study in the literature is researched and its modified version is developed. Modified CBATA (M-CBATA) is the modified version of CBATA (Consensus Based Approach for Task Allocation). The main goal of M-CBATA is to ensure that each task group has its own virtual object that has not been a virtual object in other task groups. This goal guarantees an extended operational and stability period of M-CBATA compared to CBATA.

The propose protocols are extensively evaluated in a custom designed simulation environments using MATLAB and OMNeT++ simulators. In the simulation considering the randomness of the networks 5 network topologies are generated for each network setting. Then, the probabilistic feature of EA based protocols is considered and each network is executed 5 times. The final results are obtained by taking the average results of the execution and then averaging the results of the whole networks. The simulation results have demonstrated the superiority of the proposed protocols.

It is clear from what was mentioned, that the thesis attempts to present a full set of protocols to solve the task allocation problem in IoT. These protocols can be tailored to fit wide range of IoT applications. In what follows the contributions of this thesis are summarized. Then, the summarized evaluation results of the protocols are provided. The chapter closes by presenting some open issues and future research directions.

6.2. Summary of Thesis Contributions

In summary, the main achievements of this thesis are as following:

1. Presenting a comprehensive survey of the concept of IoT.

- Introducing a historical review of the concept of IoT technology. Highlighting the basic elements and technologies that helped for emergence of IoT.
 - Investigating characteristics, architecture, platforms and building blocks of this technology.
 - Defining the main services that enabled the wide range of IoT applications.
 - Reviewing applications domains of IoT and their relevant major scenarios.
 - Illustrating the challenges of this technology that can represent open research issues in the field of IoT.
2. The thesis studies the problem of task allocation in IoT considering special features and design characteristics of this technique. Then, by adopting bio-inspired meta-heuristic methods and considering two different realistic scenarios several novel protocols with application specific characteristics that can tackle with the problem of task allocation in IoT are developed. To the best of our knowledge, this work is the first work that adopts meta-heuristic methods for this purpose.
 3. In the scenario where each object is capable of performing only a single task, two sets of protocols are proposed: “protocols with the goal of maximizing the network lifetime” and “protocols with the main goal of extending the stability period of the network”.
 - For typical applications, two novel protocols with different objective models with the goals of minimizing energy consumption regarding to task allocation and maximizing the lifetime of the network are proposed.
 - For high reliability demanding crucial applications that require their entire task groups to be active for a long time, the concept of stability period is defined and two protocols with the goal of minimizing energy consumption and maximizing stability periods of each task group to ensure stable and balanced operation of whole network are proposed.
 4. In the second scenario a more realistic and complex scenario is considered where objects have different skills and could intersect with different groups. Within this context we developed two novel protocols as follows:

- A bio-inspired single objective protocol with heterogeneity aware heuristics to insure reliably task allocation in terms of boosting the energy efficiency in the direction of extending the operational and the stability periods of the network.
 - A protocol that formulates the problem of task allocation as a MOOP that simultaneously maximizes two contradictory objectives: computational power utilization and energy efficiency.
5. To evaluate the proposed protocols extensive MATLAB based analysis as well as application layer simulations based on OMNeT++ using several benchmarking metrics are applied. The results have demonstrated the superiority of the proposed protocol.

6.3. Summary of the Simulation Results

The formation of task allocation in IoT has turned out to be an NP-hard problem, and it attempts to satisfy various objectives such as increasing the operational time and the stability period and maximizing the computational power. The underlying works employee the task groups and virtual objects concept and present evolutionary-based task allocation protocols in IoT. The overall results of the proposed protocols, after performing extensive simulations based on MATLAB and OMNeT++ simulators can draw the following conclusions (in the tables below, the relation “>” reads “*better than*”, “<” reads “*worse than*”, and the relation “=” reads “*equal performance*”):

- Energy efficiency: It means energy consumption for communication in the process of task allocation. Table 6.1 shows the overall evaluation for protocols of scenario #1 (i.e., ETAP1, ETAP2, SETAP1, SETAP2, and CBATA). In these results ETAP1 outperforms other protocols. Whereas, Table 6.2 presents the overall evaluation of STAP, MOTAP, M-CBATA and CBATA of scenario #2. Here, “=” is used between STAP and MOTAP since the two protocols show similar performance considering different settings of network dimension, number of objects, and number of tasks. Thus, in some cases STAP outperforms MOTAP, while on other cases MOTAP perform better than STAP.

Table 6.1. Evaluation summary for the energy efficiency of scenario #1 protocols

Protocols	CBATA	ETAP1	ETAP2	SETAP1	SETAP2
ETAP1	>	-	>	>	>
ETAP2	>	<	-	<	>
SETAP1	>	<	>	-	>
SETAP2	>	<	<	<	-

Table 6.2. Evaluation summary for the energy efficiency of scenario #2 protocols

Protocols	CBATA	STAP	MOTAP	M-CBATA
STAP	>	-	=	>
MOTAP	>	=	-	>
M-CBATA	=	<	<	-

- Operational period: Regarding protocols of scenario #1, ETAP1 carry out longer operational period compared to other protocols. Table 6.3 summarizes the overall results of ETAP1, ETAP2 and CBATA in term of the operational period length. On the other hand, Table 6.4 shows the overall evaluation of protocols of scenario #2. This table reveals that STAP makes the longer operational periods in all task groups.

Table 6.3. Evaluation summary for the operational period of ETAP1, ETAP2 and CBATA within the context of scenario #1

Protocols	CBATA	ETAP1	ETAP2
ETAP1	>	-	>
ETAP2	>	<	-

Table 6.4. Evaluation summary for the operational period of scenario #2 protocols

Protocols	CBATA	STAP	MOTAP	M-CBATA
STAP	>	-	>	>
MOTAP	>	<	-	>
M-CBATA	>	<	<	-

- Stability period: With their energy aware heuristic SETAP1 and SETAP2 perform better than other protocols of scenario #1 regarding to stability period metric. Table 6.5 summarizes the overall performance of scenario #1 protocols. Regarding to protocols of

scenario #2, STAP outperforms other protocols. The overall results of protocol of scenario #2 are presented in Table 6.6.

Table 6.5. Evaluation summary for the stability period of scenario #1 protocols

Protocols	CBATA	ETAP1	ETAP2	SETAP1	SETAP2
ETAP1	>	-	>	<	<
ETAP2	>	<	-	<	<
SETAP1	>	>	>	-	>
SETAP2	>	>	>	<	-

Table 6.6. Evaluation summary for the stability period of scenario #2 protocols

Protocols	CBATA	STAP	MOTAP	M-CBATA
STAP	>	-	>	>
MOTAP	>	<	-	>
M-CBATA	>	<	<	-

- Computation power: MOTAP achieves better performance compared to STAP, M-CBATA, and CBATA. However, STAP, M-CBATA and CBATA do not consider computation power when they select virtual objects. It is more likely for M-CBATA to select objects with different levels of processing power at different task groups. Table 6.7 presents the overall performance of scenario #2 protocols in term of computation power.

Table 6.7. Evaluation summary for the computation power of scenario #2 protocols

Protocols	CBATA	STAP	MOTAP	M-CBATA
STAP	>	-	<	=
MOTAP	>	>	-	>
M-CBATA	>	=	<	-

- Computation time: Naturally EA based task allocation protocols take more time in computation compared to CBATA. However, considering the high quality results of EA based protocols and the advances of microprocessors speeds, the extra time required for computation become tolerable.

6.4. Future Research

In light of the results and discussions reported in this thesis, some of the open research issues that need to be explored further for future works in the area of task allocation in IoT are briefly listed as follows:

- Future research work needs to focus on exploring more complex task allocation models. For instance, instead of using two level hierarchy (or in other words two hops from objects to their virtual objects and from virtual objects to central control station) it may be more efficient to formulate the problem as a multi-hop task allocation by adopting local virtual objects. In this way at each task group global virtual objects communicate with local virtual objects to allocate tasks to objects of the task group. Using multi-hop task allocation reduces the latency of data traveling to the central control station.
- A complex communication models can be used. It is known that the strength of the transmitted signals is inversely proportional to the distance between sender and receiver. This fact can be used at the stage of virtual objects selection with the goal of maximizing the strength of the received signals from their objects in order to increase the reliability.
- Furthermore, additional heuristics may be studied and applied in the construction of the objective function and/or other EA components to provide more network stability or longevity periods.
- It may be an open research area to measure the effect of the type of evolutionary algorithm. Several types of evolutionary algorithms are suit well to different types of problems. An analytical work may re-implement the proposed protocols by applying different types of evolutionary algorithms.
- In some applications, the optimization problem involves more than two contradictory objectives where the improvement on one objective leads to the deterioration of others. Considering the problem of task allocation maximizing of reliability of the received signals can be on expense of the energy and the computation power of the virtual objects.
- Another interesting research inspired by realizing the mobility feature of most IoT objects. The communications between the virtual objects and their objects can be taken place on the opportunistic bases. In opportunistic networks, connections among nodes

are created dynamically in an infrastructure-less way: when forwarding a message, next hops are chosen opportunistically, on the basis of their likelihood to get the message closer to its destination [175].



REFERENCES

1. Miorandi, D. Sicari, S. De Pellegrini, F. and Chlamtac, I. (2012). Internet of things: Vision, applications and research challenges. *Ad Hoc Networks*, 10 (7), 1497– 1516.
2. Atzori, L. Iera, A. and Morabito, G. (2010). The internet of things: A survey. *Computer Networks*, 54 (15), 2787–2805.
3. Al-Fuqaha, A. Guizani, M. Mohammadi, M. Aledhari, M. and Ayyash, M. (2015). Internet of things: A survey on enabling technologies, protocols, and applications. *IEEE Communications Surveys & Tutorials*, 17 (4), 2347–2376.
4. Gubbi, J. Buyya, R. Marusic, S. and Palaniswami, M. (2013). Internet of Things (IoT): A vision, architectural elements, and future directions. *Future generation computer systems*, 29 (7), 1645–1660.
5. Khalil, E.A. and Ozdemir, S. (2017). Reliable and energy efficient topology control in probabilistic Wireless Sensor Networks via multi-objective optimization. *The Journal of Supercomputing*, 6 (73), 2632–2656.
6. Akyildiz, I. F. and Vuran, M. C. (2010). *Wireless Sensor Networks* (First Edition). John Wiley & Sons Ltd.
7. Buratti, C. Conti, A. Dardari, D. and Verdone, R. (2009). An overview on wireless sensor networks technology and evolution. *Sensors*, 9(9), 6869-6896.
8. Colistra, G. Pilloni, V. and Atzori, L. (2014). Task allocation in group of nodes in the IoT: A consensus approach. *In Communications (ICC), 2014 IEEE International Conference*, 3848-3853.
9. Pilloni, V. and Atzori, L. (2017). Consensus-based resource allocation among objects in the internet of things. *Annals of Telecommunications*, 72(7-8), 415-429, 2017.
10. Jin, S. Zhou, M. and Wu, A.S. (2003). Sensor network optimization using a genetic algorithm. *In: Proceedings of the 7th world multiconference on systemics, cybernetics and informatics*, 109-116.
11. Yang, X.S. (2010). *Nature-inspired metaheuristic algorithms* (Second Edition). Luniver Press.
12. Colistra, G. Pilloni, V. and Atzori, L. (2014). The problem of task allocation in the internet of things and the consensus-based approach. *Computer Networks*, 73, 98–111.
13. Coetzee, L. and Eksteen, J. (2011). The Internet of Things-promise for the future? An introduction. *In IST-Africa Conference Proceedings, IEEE*, 1-9.
14. Internet: Internet World Stats. International Website for up to date world Internet Usage and Statistics. URL: <http://www.internetworldstats.com/> (16.01.2017).

15. Macaulay, J. Buckalew, L. and Chung, G. (2015). Internet of Things in Logistics, A collaborative report by DHL and Cisco on implications and use cases for the logistics industry. **DHL Trend Research|Cisco Consulting Services, Troisdorf, Germany, 2015.**
16. Markets and Markets (M&M) Research Group. Internet of Things (IoT) & Machine-To-Machine (M2M) communication market by Technologies & Platforms, M&M Connections & IoT Components worldwide Market forecasts (2014–2019). **Researchandmarkets.com Publications**, Dublin, Ireland, Technical Report, 2785177, 2014.
17. Rose, K. Eldridge, S. and Chapin, L. (2015). The Internet of Things (IoT): An Overview-Understanding the Issues and Challenges of a More Connected World. **Internet Society**, Geneva, Switzerland, Technical Report, October 2015.
18. Saha, H.N. Mandal, A. and Sinha, A. (2017). Recent trends in the Internet of Things. **2017 IEEE 7th Annual Computing and Communication Workshop and Conference (CCWC)**, Las Vegas, NV, 2017,1-4.
19. Tsai, C.W. Lai, C.F. and Athanasios, V.V. (2014). Future internet of things: open issues and challenges. **Wireless Networks**, 20(8), 2201-2217.
20. The International Telecommunication Union. (2005) The Internet of Things. **ITU Internet Reports. 7th ed.** Hammamet, Tunis, November 2005.
21. Satyanarayanan, M. Simoens, P. Xiaoö, Y. Pillai, P. Chen, Z. Ha, K. Hu, W. and Amos, B. (2015). Edge analytics in the internet of things. **IEEE Pervasive Computing**, 14(2), 24-31.
22. Want, R. Schilit, B.N. and Jenson, S. (2015). Enabling the internet of things. **IEEE Computer**, 48(1), 28-35.
23. Wang, F. Hu, L. Hu, J. Zhou, J. and Zhao, K. (2017). Recent advances in the internet of things: multiple perspectives. **IETE Technical Review**, 34(2), 122-132.
24. Aggarwal, C.C. Ashish, N. and Sheth, A. (2013). The Internet of Things: A Survey from the Data-Centric Perspective. **Managing and mining sensor data**, C.C. Aggarwal, Ed. Springer US, 383-428.
25. Alam, F. Mehmood, R. Katib, I. and Albeshri, A. (2016). Analysis of eight data mining algorithms for smarter internet of things (IoT). **Procedia Computer Science**, 98, 437-442.
26. Madakam, S. Ramaswamy, R. and Tripathi, S. (2015). Internet of things (IoT): A literature review. **Journal of Computer and Communications**, 3(5), 164-173.
27. Magrassi, P. Panarella, A. Deighton, N. and Johnson, G. (2001). Computers to Acquire Control of the Physical World. **Gartner Research Report**, T-14-0301, Stamford, USA.

28. Li, J. Wang, Y. and Sun, T. (2013). A Hybrid Genetic Algorithm For Task Scheduling In Internet Of Things. *ICIT 2013 The 6th International Conference on Information Technology*, Amman, Jordan.
29. Shen, G. and Liu, B. (2011). The visions, technologies, applications and security issues of Internet of Things. *E -Business and E-Government (ICEE), 2011 International Conference*, Shanghai, China, 6-8 May 2011.
30. Tan, L. and Wang, N. (2010). Future internet: The Internet of Things. *2010 3rd International Conference on Advanced Computer Theory and Engineering(ICAETE)*, Chengdu, China, 20-22 August 2010.
31. Akyildiz, F. Su, W. Sankarasubramaniam, Y. and Cayirci, E. (2002). Wireless sensor networks: a survey. *Computer Networks*, 38(4), 393-422.
32. Jain, D. Krishna, P.V. and Saritha, V. (2012). A Study on Internet of Things based Applications. *arXiv preprint*, arXiv:1206.3891, 1-10.
33. Ferro, E. and Potorti, F. (2005). Bluetooth and Wi-Fi wireless protocols: A survey and a comparison. *IEEE Wireless Communications*, 12(1), 12-26.
34. IEEE Standards Association. ()2011. IEEE Standard for Local and Metropolitan Area Networks- Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs). New York, USA, *IEEE Standard 802.15*. 4-2011.
35. Collotta, M. Pau, G. Talty, T. and Tonguz, O.K. (2018). Bluetooth 5: A Concrete Step Forward toward the IoT. *IEEE Communications Magazine*, 56(7), 125-131.
36. Crosby, G.V. and Vafa, F. (2013). Wireless sensor networks and LTE-A network convergence. *38th Annual IEEE Conference on Local Computer Networks*, Sydney, Australia, 21-24 Oct. 2013.
37. Ghosh, A. Ratasuk, R. Mondal, B. Mangalvedhe, N. and Thomas, T. (2010). LTE-Advanced: Next-generation wireless broadband technology. *IEEE Wireless Communications*, 17(3), 10-22.
38. Xiong, X. Zheng, K. Xu, R. Xiang, W. and Chatzimisios, P. (2015). "Low power wide area machine-to-machine networks: Key techniques and prototype". *IEEE Communications Magazine*, 53(9), 64-71.
39. Sigfox. French research and development organization. URL: <http://www.sigfox.com/> (22.03.2017).
40. Sornin, N. Luis, M. Eirich, T. Kramp, T. and Hersent, O. (2015). Lorawan specification. Beaverton, USA, LoRa Alliance, *Technical Report*, V1.0.
41. Ingenu. American research and development organization. URL: <https://www.ingenu.com/> (22.03.2017).

42. Telensa. An international organization makes intelligent city control systems. URL: <http://www.telensa.com/> (22.03.2017).
43. Sanchez-Iborra, R. and Cano, M.D. (2016). State of the art in LP-Wan solutions for industrial IoT services. *Sensors*, 16(5), 708-722.
44. Raza, U. Kulkarni, P. and Sooriyabandara, M. (2017). Low power wide area networks: An overview. *IEEE Communications Surveys & Tutorials*, 19(2), 855-873.
45. Want, R. (2006). An introduction to RFID technology. *IEEE Pervasive Computing*, 5(1), 25-33.
46. Mulligan, G. (2007). The 6LoWPAN architecture. *4th Workshop on Embedded Networked Sensors*, Cork, Ireland, 25-26 June, 2007.
47. Shelby, Z. and Bormann, C. (2009). *6LoWPAN: The Wireless Embedded Internet* (First Edition). Wiltshire, UK, John Wiley & Sons.
48. Want, R. (2011). Near field communication. *IEEE Pervasive Computing*, 10(3), 4-7.
49. Kshetrimayum, R.S. (2009). An introduction to UWB communication systems. *IEEE Potentials*, 28(2), 9-13.
50. Thakare, S. Patil, A. and Siddiqui, A. (2016). The internet of things – emerging technologies, challenges and application. *International Journal of Computer Applications*, 149(10), 21-25.
51. Pilkington, K. (2014). Revolv teams up with Home Depot to keep your house connected. *Centre National d'Etudes des Telecommunications (CNET)*, 2014. URL: <https://www.cnet.com/news/revolv-teams-up-with-home-depot-to-keep-your-house-connected/> (16.01.2017).
52. Doukas, C. (2012). *Building Internet of Things with the ARDUINO* (First Edition). Dougherty, GA, USA, CreateSpace Independent Publishing Platform.
53. De-Sousa, M. (2015). *Internet of Things with Intel Galileo* (First Edition). Birmingham, UK, Packt Publishing Ltd.
54. Intel Galileo Board. Intel: American multinational technology organization. URL: <http://ark.intel.com/products/78919/Intel-Galileo-Board> (16.01.2017).
55. Raspberry PI. A UK-based charity that works to put the power of digital making into the hands of people all over the world. URL: <https://www.raspberrypi.org/> (16.01.2017).
56. Maksimović, M. Vujović, V. Davidović, N. Milošević, V. and Perišić, B. (2014). Raspberry Pi as Internet of things hardware: performances and constraints. *1st International Conference on Electrical, Electronic and Computing Engineering*, Vrnjačka Banja, Serbia, 2-5 June 2014.

57. Hodges, S. Taylor, S. Villar, N. Scott, J. Bial, D. and Fischer, P.T. (2013). Prototyping connected devices for the internet of things. *Computer*, 46(2), 26-34.
58. Beagleboard. Community supported open hardware computers for making. URL: <https://beagleboard.org/> (16.01.2017).
59. Principi, E. Colagiacomio, V. Squartini, S. and Piazza, F. (2012). Low power high-performance computing on the beagleboard platform. *5th European DSP Education and Research Conference (EDERC)*, Amsterdam, Netherlands, 13-14 September 2012.
60. Kruger, C.P. and Hancke, G.P. (2014). Benchmarking Internet of things devices. *12th IEEE International Conference on Industrial Informatics (INDIN)*, Porto Alegre, Brazil, 27-30 July 2014.
61. Cubieboard. Chinese company develops a series of open source hardware. URL: <http://cubieboard.org/> (16.01.2017).
62. UDOO. unique open-source project bringing Mini PC with Android, Linux and Arduino together in functional all-in-one embedded system. URL: <http://www.udoo.org/> (15.01.2017).
63. Zolertia. Spanish company offers hardware solutions for creating Internet of things applications. URL: <http://zolertia.io> (16.01.2017).
64. Eistec. Mülle: Mülle wireless sensor platform. URL: <http://www.eistec.se/mulle/> (16.01.2017).
65. Dunkels, A. Gronvall, B. and Voigt, T. (2004). Contiki-A lightweight and flexible operating system for tiny networked sensors. *29th Annual IEEE International Conference on Local Computer Networks (LCN'04)*, Tampa, FL, USA, 16-18 Nov. 2004.
66. Cao, Q. Abdelzaher, T. Stankovic, J. and He, T. (2008). The LiteOS operating system: Towards Unix-like abstractions for wireless sensor networks. *In Information Processing in Sensor Networks, 2008. IPSN'08. International Conference, IEEE*, 233–244, St. Louis, MO, USA, USA, 22-24 April 2008.
67. Levis, P. Madden, S. Polastre, J. Szewczyk, R. Whitehouse, K. Woo, A. Gay, D. Hill, J. Welsh, M. Brewer, E. and Culler, D. (2005). *Tinyos: An operating system for sensor networks* (First Edition). Heidelberg, Berlin, Springer, 2005.
68. Baccelli, E. Hahm, O. Gunes, M. Wahlisch, M. and Schmidt, T.C. (2013). RIOT OS: Towards an OS for the Internet of Things. *In Computer Communications Workshops (INFOCOM WKSHPS), 2013 IEEE Conference*, Turin, Italy, 14-19 April 2013.
69. Open Auto Alliance. A global alliance of technology and auto industry leaders committed to bringing the Android platform to cars since 2015. URL: <http://www.openautoalliance.net/> (16.01.2017).

70. Android. Mobile operating system developed by Google. URL: <https://www.android.com/> (16.01.2017).
71. Koshizuka, N. and Sakamura, K. (2010). Ubiquitous ID: Standards for ubiquitous computing and the Internet of Things. *IEEE Pervasive Computing*, 9(4), 98–101.
72. Kushalnagar, N. Montenegro, G. and Schumacher, C. (2007). *IPv6 over low-power wireless personal area networks (6LoWPANs): overview, assumptions, problem statement, and goals* (First Edition). Fremont, USA, Internet Engineering Task Force (IETF), No. RFC 4919, 2007.
73. Barnaghi, P. Wang, W. Henson, C. and Taylor, K. (2012). Semantics for the Internet of Things: Early progress and back to the future. *International Journal on Semantic Web and Information Systems (IJSWIS)*, 8(1), 1-21.
74. Schneider, J. Kamiya, T. Peintner, D. and Kyusakov, R. (2014). Efficient XML Interchange (EXI) Format 1.0. (Second Edition). Cambridge, Massachusetts, USA, World Wide Web Consortium, Recommend. REC-Exi-20110310.
75. Gigli, M. and Koo, S. (2011). Internet of Things: Services and applications categorization. *Advances in Internet of Things*, 1(2), 27-31.
76. Xiaojiang, X. Jianli, W. and Mingdong, L. (2010). Services and key technologies of the Internet of Things. *ZTE Communications*, 8(2), 26–29.
77. Desai, P. Sheth, A. and Anantharam, P. (2015). Semantic gateway as a service architecture for IoT interoperability. *2015 IEEE International Conference on Mobile Services*, New York, NY, 27 June-2 July 2015.
78. Ning, H. and Hu, S. (2012). Technology classification, industry, and education for future internet of things. *International Journal of Communication Systems*, 25(9), 1230-1241.
79. Yun, M. and Yuxin, B. (2010). Research on the architecture and key technology of internet of things (IoT) applied on smart grid. *In Proceedings of the International Conference on Advances in Energy Engineering*, Beijing, China, 19-20 June 2010.
80. Romero, C.D.G. Barriga, J.K.D. and Molano, J.I.R. (2016). Big data meaning in the architecture of IoT for smart cities. *Data Mining and Big Data: First International Conference, DMBD 2016*, Bali, Indonesia, 25-30 June, 2016.
81. Bandyopadhyay, D. and Sen, J. (2011). Internet of things: Applications and challenges in technology and standardization. *Wireless Personal Communications*, 58(1), 49-69.
82. Khan, R. Khan, S.U. Zaheer, R. and Khan, S. (2012). Future Internet: The Internet of Things architecture, possible applications and key challenges. *10th International Conference on Frontiers of Information Technology, Islamabad*, India, 17-19 December 2012.

83. Yang, Z. Yue, Y. Yang, Y. Peng, Y. Wang, X. and Liu, W. (2011). Study and application on the architecture and key technologies for IoT. **2011 International Conference on Multimedia Technology**, Hangzhou, China, 26-28 July 2011.
84. Wu, M. Lu, T.J. Ling, F.Y. Sun, J. and Du, H.Y. (2010). Research on the architecture of Internet of things. **3rd International Conference on Advanced Computer Theory and Engineering (ICACTE)**, Chengdu, China, 20-22 August 2010.
85. Balamuralidhara, P. Misra, P. and Pal, A. (2013). Software platforms for internet of things and M2M. **Journal of the Indian Institute of Science**, 93(3), 487-498.
86. Dayarathna, M. Comparing 11 IoT Development Platforms. URL: <https://dzone.com/articles/iot-software-platform-comparison> (25.03.2017).
87. Perera, S. IoT Analytics: Using Big Data to Architect IoT Solutions. URL: <http://wso2.com/whitepapers/iot-analytics-using-big-data-to-architect-iot-solutions/> (25.03.2017).
88. Gluhak, A. Krco, S. Nati, M. Pfisterer, D. Mitton, N. and Razafindralambo, T. (2011). A Survey on Facilities for Experimental Internet of Things Research, **IEEE Communications Magazine**, 49(11), 58–67.
89. Abdmeziem, R. and Tandjaoui, D. (2014). Internet of Things: Concept, Building blocks, Applications and Challenges. **arXiv preprint**, arXiv:1401.6877.
90. Giusto, D. Iera, A. Morabito, G. and Atzori, L. (2010). *The Internet of Things: 20th Tyrrhenian Workshop on Digital Communication* (First Edition). New York, USA, Springer-Verlag New York, 2010.
91. Jara, A.J. Zamora, M.A. and Skarmeta, A.F.G. (2009). An Ambient Assisted Living System for Telemedicine with Detection of Symptoms. Editors: Mira, J. Ferrández, J.M. Álvarez, J.R. de la Paz F. Toledo, F.J. **Bioinspired Applications in Artificial and Natural Computation Lecture Notes in Computer Science**, 75-84, Heidelberg, Berlin, Germany, Springer, Berlin, Heidelberg.
92. Resch, A. and Blecker, T. (2012). Smart Logistics—a Literature Review. Editors: Blecker, T. Kersten, W. Ringle, C.M. **Pioneering Supply Chain Design: A Comprehensive Insight Into Emerging Trends, Technologies and Applications**, 91-102, Germany, Josef Eul Verlag GmbH.
93. Hipp, C. Sellner, T. Bierkandt, J. and Holtewert, P. (2012). Smart factory: System logic of the project epic. **1st International Conference on Smart Systems, Devices and Technologies**, Stuttgart, Germany, 27 May-1 June 2012.
94. Tong-Ke, F. (2013). Smart Agriculture based on cloud computing and IoT. **Journal of Convergence Information Technology**, 8(2), 210-216.
95. Jayaraman, P.P. Yavari, A. Georgakopoulos, D. Morshed, A. and Zaslavsky, A. (2016). Internet of things platform for smart farming: experiences and lessons learnt. **Sensors**, 16(11), 1884.

96. Mukhopadhyay, S.C. and Suryadevara, N.K. (2014) Internet of Things: Challenges and Opportunities. In: Mukhopadhyay S. (eds) Internet of Things. **Smart Sensors, Measurement and Instrumentation**, vol 9. Springer, Cham.
97. Mattern, F. and Floerkemeier, C. (2010). From the Internet of Computers to the Internet of Things. Editors: Sachs, K. Petrov, I. From Active Data Management to Event-Based Systems and More, 242–259, Berlin, Germany, **Springer**, Berlin, Heidelberg, 2010.
98. Li, S. Xu, L.D. and Wang, X. (2013). Compressed sensing signal and data acquisition in wireless sensor networks and internet of things. **IEEE Transactions on Industrial Informatics**, 9(4), 2177-2186.
99. Vljajic, N. and Xia, D. (2006). Wireless sensor networks: to cluster or not to cluster?. **International Symposium on World of Wireless, Mobile and Multimedia Networks**, Buffalo, New York, USA, 26-29 June 2006.
100. Postel, J. (1998). Internet official protocol standards. **The Internet Society**, Technical report, United States, RFC 1800.
101. Goth, G. (2012). The End of IPv4 is Nearly Here — Really. **IEEE Internet Computing**, 16(2), 7-11.
102. Weber, S. and Cheng, L. (2004). A survey of anycast in ipv6 networks. **IEEE Communications Magazine**, 42(1), 127–132.
103. Edwards, W.K. (2006). Discovery systems in ubiquitous computing. **IEEE Pervasive Computing**, 5(2), 70–77.
104. Zhang, Z.K. Cho, M.C.Y. Wang, C.W. Hsu, C.W. Chen, C.K. and Shieh, S. (2014). IoT security: ongoing challenges and research opportunities. **7th International Conference on Service-Oriented Computing and Applications**, Matsue, Japan, 17-19 November 2014.
105. Borgia, E. (2014). The internet of things vision: key features, applications and open issues. **Computer Communications**, Vol. 54, 1-31.
106. Chen, Y.K. (2012). Challenges and opportunities of internet of things. **17th Asia and South Pacific Design Automation Conference**, Sydney, NSW, Australia, 30 January-2 February 2012.
107. Blackford, J. Digdon, P.M. and Aptean. (2013). CPE WAN Management Protocol. Broadband Forum, Fremont, USA, Technical Report, TR-069.
108. Harrington, D. Presuhn, R. and Wijnen, B. An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks. URL: <https://tools.ietf.org/html/rfc3411.html> (28.03.2017).
109. Enns, R. Bjorklund, M. Schoenwaelder, J. and Bierman, A. Network Configuration Protocol (NETCONF). URL: <https://tools.ietf.org/html/rfc6241> (29.03.2017).

110. Gordon, D.M. (2002). The organization of work in social insect colonies. *Complexity*, 8(1), 43-46.
111. Heinzelman, W.B. Chandrakasan, A. and Balakrishnan, H. (2002). An application-specific protocol architecture for wireless microsensor networks. *IEEE Transactions on Wireless Communications*, 1(4), 660-670.
112. Colistra, G. (2015). Task allocation in the Internet of Things. Doctoral dissertation, Università degli Studi di Cagliari, Italy.
113. Huang, C.M. Lan, K.C. and Tsai, C.Z. (2008). A survey of opportunistic networks. *In Advanced Information Networking and Applications-Workshops (AINAW), 22nd International Conference on. IEEE*, 1672–1677, 2008.
114. Yu, Y. and Prasanna, V.K. (2005). Energy-balanced task allocation for collaborative processing in wireless sensor networks. *Mobile Networks and Applications*, 10(1-2), 115-131.
115. Edalat, N. Xiao, W. Tham, C.K. Keikha, E. and Ong, L.L. (2009). A price-based adaptive task allocation for wireless sensor network. *In: Mobile Adhoc and Sensor Systems, 2009. MASS'09. IEEE 6th International Conference*, 2009, pp. 888-893.
116. Pilloni, V. and Atzori, L. (2011). Deployment of distributed applications in wireless sensor networks. *Sensors*, 11(8), 7395-7419.
117. Jin, Y. Jin, J. Gluhak, A. Moessner, K. and Palaniswami, M. (2012). An intelligent task allocation scheme for multihop wireless networks. *IEEE Transactions on Parallel and Distributed Systems*. 23(3), 444-451.
118. Zhu, J. Li, J. and Gao, H. (2007). Tasks allocation for real-time applications in heterogeneous sensor networks for energy minimization. *In: Proceedings of the Eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/ Distributed Computing*. Vol. 2, 20-25.
119. Abdelhak, S. Gurram, C.S. Ghosh, S. and Bayoumi, M. (2010). Energy-balancing task allocation on wireless sensor networks for extending the lifetime, *In: Circuits and Systems (MWSCAS), 2010 53rd IEEE International Midwest Symposium*, 2010, pp. 781-784.
120. Ferjani, A.A. Liouane, N. and Kacem, I. (2016). Task allocation for wireless sensor network using logic gate-based evolutionary algorithm. *In: Control, Decision and Information Technologies (CoDIT), 2016 International Conference*, 2016, pp. 654-658.
121. Pilloni, V. Franceschelli, M. Atzori, L. and Giua, A. (2012). A decentralized lifetime maximization algorithm for distributed applications in wireless sensor networks. *In: Communications (ICC), 2012 IEEE International Conference*, 2012, pp. 1372-1377.

122. Shen, Y. and Ju, H. (2011). Energy-efficient task assignment based on entropy theory and particle swarm optimization algorithm for wireless sensor networks. *In: Proceedings of the 2011 IEEE/ACM International Conference on Green Computing and Communications, IEEE Computer Society*, 2011, pp. 120–123.
123. Pilloni, V. Navaratnam, P. Vural, S. Atzori, L. and Tafazolli, R. (2013). Cooperative task assignment for distributed deployment of applications in WSNs. *IEEE International Conference on Communications (ICC), Budapest*, 2013, pp. 2229–2234.
124. Li, W. Delicato, F.C. Pires, P.F. Lee, Y.C. Zomaya, A.Y. Miceli, C. and Pirmez, L. (2014). Efficient allocation of resources in multiple heterogeneous wireless sensor networks. *Journal of Parallel and Distributed Computing*, 74(1), 1775–1788.
125. Kim, M. and Ko, I.Y. (2015). An Efficient Resource Allocation Approach Based on a Genetic Algorithm for Composite Services in IoT Environments. *In: Web Services (ICWS), 2015 IEEE International Conference*, 2015, pp. 543–550.
126. Guinard, D. Trifa, V. Mattern, F. and Wilde, E. (2011). From the internet of things to the web of things: Resource-oriented architecture and best practices. *In: Architecting the Internet of things, Springer*, Berlin, Heidelberg, pp. 97–129.
127. Silverajan, B. and Harju, I. (2009). Developing network software and communications protocols towards the internet of things. *In: Proceedings of the Fourth International ICST Conference on COMMunication System softWARE and middlewaRE, ACM*, 2009, p 9.
128. Haller, S. (2010). The things in the internet of things. *Poster at the (IoT 2010)*, Tokyo, Japan, November, 2010, vol. 5, p. 26.
129. Chen, S.E. (1995). Quicktime vr: An image-based approach to virtual environment navigation. *In Proceedings of the 22nd annual conference on Computer graphics and interactive technique, ACM*, 1995, pp. 29–38.
130. Zyda, M. (2005). From visual simulation to virtual reality to games. *Computer*, 38(9), 25–32.
131. CONVERGENCE, (2010). Convergence. URL: <http://www.ict-convergence.eu/>. (29.07.2018).
132. Römer, K. Schoch, T. Mattern, F. and Dübendorfer, T. (2004). Smart identification frameworks for ubiquitous computing applications. *Wireless Networks*, 10(6), 689–700.
133. Tsiatsis, V. Gluhak, A. Bauge, T. Montagut, F. Bernat, J. Bauer, M. Villalonga, C. Barnaghi, P. and Krco, S. (2010). The SENSEI real world internet architecture. *In: Towards the Future Internet, IOS Press*, pp. 247–256.
134. IoT-A project. Internet of things architecture, Ref. D1.2. URL: <http://www.iot-a.eu>. (29.07.2018).

- 135.COMPOSE, (2012). Collaborative open market to place objects at your service. URL: <http://www.compose-project.eu/>. (07.08.2018).
- 136.iCore, (2011). Empowering IoT through cognitive technologies. URL: <http://www.iot-icore.eu/>. (08.08.2018).
- 137.Espada, J.P. Martínez, O.S. Bustelo, B.C.P.G. and Lovelle, J.M.C. (2011). Virtual objects on the internet of things. *International Journal of Artificial Intelligence and Interactive Multimedia*, 1(4), 23–29.
- 138.Jin, V. Zhou, M. and Wu, A.S. (2003). Sensor Network Optimization Using a Genetic Algorithm. *In: the Proceedings of the 7th World Multiconference on Systemics, Cybernetics, and Informatics*, Orlando, FL, July 2003.
- 139.Hoang, D.C. Yadav, P. Kumar, R. and Panda, S.K. (2010). A robust harmony search algorithm based clustering protocol for wireless sensor networks. *In: 2010 IEEE International Conference on Communications Workshops (ICC)*, 2010.
- 140.Yang, X.S. (2009). Harmony search as a metaheuristic algorithm. *In: Music-inspired harmony search algorithm, Springer*, Berlin, Heidelberg, 2009, pp. 1-14.
- 141.Zäpfel, G. Braune, R. and Bögl, M. (2010). *Metaheuristic search concepts: A tutorial with applications to production and logistics* (First Edition). Springer Science & Business Media.
- 142.Yu, X. and Gen, M. (2010). *Introduction to evolutionary algorithms* (First Edition). Springer Science & Business Media.
- 143.Cotta, C. Sevaux, M. and Sörensen, K. (2008). Adaptive and multilevel metaheuristics. Springer, Vol. 136.
- 144.Khalil, E.A. (2008). An Evolutionary Routing Protocol for Dynamic Clustering of Wireless Sensor Networks. M.S. dissertation, University of Baghdad, Iraq.
- 145.Ahn, C.W. (2006). *Advances in Evolutionary Algorithms Theory, Design and Practice* (First Edition). Part of the Studies in Computational Intelligence book series, Vol. 18, Springer.
- 146.Engelbrecht, A.P. (2007). *Computational Intelligence an Introduction* (Second Edition). John Wiley & Sons Ltd.
- 147.Sumathi, S. and Surekha, P. (2010). Computational Intelligence Paradigms: Theory and Applications Using MATLAB, CRC Press.
- 148.Eiben, A.E. Michalewicz, Z. Schoenauer, M. and Smith, J.E. (2007). Parameter control in evolutionary algorithms. In Parameter setting in evolutionary algorithms, pp. 19-46, Springer, Berlin, Heidelberg.

149. Coello, C.A.C. Lamont, G.B. and Van-Veldhuizen, D.A. (2007). *Evolutionary algorithms for Solving Multi-Objective Problems* (Second Edition). Springer, New York, USA.
150. Blum, C. and Roli, A. (2003). Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Computing Surveys*, 35(3), 268-308.
151. Eiben, A.E. and Smith, J.E. (2015). *Introduction to evolutionary computing* (Second Edition). In natural computing series, Springer-Verlag, Berlin Heidelberg, 2015.
152. Eiben, A.E. and Schippers, C.A. (1998). On evolutionary exploration and exploitation. *Fundamenta Informaticae*, 35(1-4), 35-50.
153. Osyczka, A. (1985). Multicriteria optimization for engineering design. *In: Design Optimization*, pp. 193–227.
154. Chen, J. H. (2004). Theory and Applications of Efficient Multiobjective Evolutionary Algorithms, Doctoral dissertation, Feng Chia University, Taichung, Taiwan, R.O.C.
155. Fonseca, C. M. and Fleming, P. J. (1995). An overview of evolutionary algorithms in multiobjective optimization. *Evolutionary Computation*, 3(1), 1-16.
156. Zitzler, E. (1999). Evolutionary algorithms for multiobjective optimization: Methods and applications. Doctoral dissertation, Swiss Federal Institute of Technology (ETH), Zurich, Switzerland.
157. Coello, C.A.C. (2002). Theoretical and Numerical Constraint-Handling Techniques used with Evolutionary Algorithms: A Survey of the State of the Art. *Computer Methods in Applied Mechanics and Engineering*, 191(11–12), 1245– 1287.
158. Coello, C.A.C. and Lamont, G. B. (2004). *Applications of Multi-Objective Evolutionary Algorithms* (First Edition). World Scientific, Singapore, 2004. ISBN 981-256-106-4.
159. Van-Veldhuizen, D.A. (1999). Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations. Doctoral dissertation, Department of Electrical and Computer Engineering. Graduate School of Engineering. Air Force Institute of Technology, Wright-Patterson AFB, Ohio.
160. Attea, B.A. Khalil, E.A. and Zdemir, S. (2014). Biologically inspired probabilistic coverage for mobile sensor networks. *Soft Computing*, 18(11), 2313-2322.
161. Deb, K. Pratap, A. Agarwal, S. and Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2), 182-197.
162. Srinivas, N. and Deb, K. (1994). Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary computation*, 2(3), 221-248.

163. Lu, H. and Yen, G.G. (2003). Rank-density-based multiobjective genetic algorithm and benchmark test function study. *IEEE Transactions on Evolutionary Computation*, 7(4), 325-343.
164. Zitzler, E. and Thiele, L. (1999). Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach. *IEEE Transactions on Evolutionary Computation*, 3(4), 257-271.
165. Deb, K. Pratap, A. Agarwal, S. and Meyarivan, T. (2000). A fast an elitist multi-objective genetic algorithm: NSGA-II. *In: Proceedings Parallel Problem Solving from Nature VI*, pp. 849-858.
166. Heinzelman, W.R. Chandrakasan, A. and Balakrishnan, H. (2000). Energy-efficient communication protocol for wireless microsensor networks. *In: System sciences, Proceedings of the 33rd annual Hawaii international conference*, 2000, p. 10.
167. Heinzelman, W.B. (2000). Application-Specific Protocol Architectures for Wireless Networks. Doctoral dissertation, Massachusetts Institute of Technology, USA.
168. Smaragdakis, G. Matta, I. and Bestavros, A. (2004). SEP: A stable election protocol for clustered heterogeneous wireless sensor networks. *In: 2nd International Workshop on Sensor and Actor Network Protocols and Applications (SANPA 2004)*, Boston MA, Aug. 2004.
169. Bandyopadhyay, S. and Coyle, E.J. (2004). Minimizing communication costs in hierarchically-clustered networks of wireless sensors. *Computer Networks*, 44(1), 1-16.
170. The MathWorks, Inc. MATLAB Primer. (2017) URL: <https://www.mathworks.com/help/matlab/index.html>. (07.09.17).
171. Messac, A. (2015). Optimization in Practice with MATLAB: For Engineering Students and Professionals, *Cambridge University Press*, Cambridge.
172. András Varga and OpenSim Ltd., (2016). OMNeT++ discrete event simulation system version 5.3 user manual. URL: <https://www.omnetpp.org/doc/omnetpp/SimulationManual.pdf>. (11.04.18).
173. “ARDUINO YÚN”, (2018). URL; <https://store.arduino.cc/usa/arduino-yun>, (04.05.18).
174. “M-class m51xx core family”, (2018). URA: <https://www.mips.com/products/warrior/m-class-m51xx-core-family/>. (04.05.2018).
175. Huang, C.M. Lan, K.C. and Tsai, C.Z. (2008). A survey of opportunistic networks. *In: Advanced Information Networking and Applications-Workshops (AINAW 2008)*, IEEE, pp. 1672–1677.



CURRICULUM VITAE

Personal Information

Surname, Name : SAFFAR, Enan Ameen Khalil
 Nationality : Iraq
 Date and place of birth : 6 February 1986, Baghdad
 Marital status : Single
 Phone number : +90 543 853 91 22
 E-mail : enanameen@yahoo.com



Education

Degree	School/ Program	Graduation Date
PhD	University of Gazi / Computer engineering	Ongoing
MSc	University of Baghdad / Computer Science	2011
Undergraduate	University of Baghdad / Computer Science	2008
High School	Al-Iraq Al Gedid/Baghdad	2004

Professional Experience

Year	Place of Work	Position
2011-Ongoing	Gazi Üniversitesi- KAVEM	Research collaborator

Foreign Language

English, Turkish, Arabic

Publications

1. Khalil, E.A. Ozdemir, S. and Tosun, S. (2018). Evolutionary task allocation in Internet of Things-based application domains. *Future Generation Computer Systems*, vol. 86, pp. 121-133.
2. Khalil, E.A. and Ozdemir, S. (2018). Overview of Internet of Things: Concept, Characteristics, Challenges and Opportunities. *Pamukkale University Journal of Engineering Sciences*, 24(2), 311-326.

3. Khalil, E.A. and Ozdemir, S. (2017). Reliable and Energy Efficient Topology Control in Probabilistic Wireless Sensor Networks via Multi-Objective Optimization. *The Journal of Supercomputing*, Springer, 73(6), 2632-2656.
4. Attea, B.A. Khalil, E.A. and Cosar, A. (2015). Multi-objective evolutionary routing protocol for efficient coverage in mobile sensor networks. *Soft Computing*, 19(10), 2983-2995.
5. Attea, B.A. Khalil, E.A. Ozdemir, S. Yildiz, O. (2015). A Multi-Objective Disjoint Set Covers for Reliable Lifetime Maximization of Wireless Sensor Networks. *Wireless Personal Communications*, Springer, 81(2), 819-838.
6. Khalil, E.A. and Ozdemir, S. (2015). Prolonging Stability Period of CDS Based Wireless Sensor Networks. *In: Wireless Communications and Mobile Computing Conference (IWCMC), IEEE*, pp. 776-781.
7. Khalil, E.A. and Ozdemir, S. (2015). CDS Based Reliable Topology Control in Wireless Sensor Networks. *In: Networks, Computers and Communications (ISNCC), IEEE*, pp. 1-5.
8. Attea, B.A. Khalil, E.A. Ozdemir, S. (2014). Biologically Inspired Probabilistic Coverage for Mobile Sensor Networks. *Soft Computing, Springer*, 18(11), 2313-2322.
9. Khalil, E.A. and Ozdemir, S. (2013). Energy Aware Evolutionary Routing Protocol with Probabilistic Sensing Model and Wake-Up Scheduling. *In: Globecom Workshops (GC Wkshps), 2013 IEEE*, pp. 873-878.
10. Attea, B.A. and Khalil, E.A. (2013). Stable-aware evolutionary routing protocol for wireless sensor networks. *Wireless Personal Communications, Springer*, 69(4), 1799-1817.
11. Attea, B.A. and Khalil, E.A. (2012). A new evolutionary based routing protocol for clustered heterogeneous wireless sensor networks. *Applied Soft Computing*, 12(7), 1950-1957.
12. Khalil, E.A. and Attea, B.A. (2011). Energy-aware evolutionary routing protocol for dynamic clustering of wireless sensor networks. *Swarm and Evolutionary Computation*, 1(4), 195-203.

Hobbies

Reading, playing sports



GAZİ GELECEKTİR...