Author:
Singh, Raghubir

Title:
Computation Offloading inHeterogeneous Multi-access EdgeComputing

Computation Offloading in Heterogeneous Multi-access Edge Computing

By

RAGHUBIR SINGH

Department of Electrical and Electronic Engineering
UNIVERSITY OF BRISTOL

A dissertation submitted to the University of Bristol in accordance with the requirements of the degree of DOCTOR OF PHILOSOPHY in the Faculty of Engineering.

SEPTEMBER 2021

Word count: 38025

Abstract

In recent years, there has been considerable interest in computational offloading from mobile devices (such as smartphones) to reduce computational task completion time and the associated energy consumption by battery-dependent mobile devices.

The most researched strategy involves offloading to Cloud Computing in Mobile Cloud Computing, where the Cloud offers not only storage but Software as a Service. Resorting to physically distant consolidated data centres, however, has excessive latency and low bandwidth issues. Multi-access Edge Computing (MEC) networks offer improved solutions to real-time and delay-sensitive mobile applications with closer proximity of cellular networks and potentially much larger numbers of hardware units accessible by mobile devices.

This thesis explores different scenarios for offloading computational tasks to MEC servers from multiple users with a range of mobile devices. A unified and coherent approach presents detailed simulation data for how offloading can be beneficial to reduce total task completion time and local (mobile device) energy use in MEC networks with varying quantitative mobile user demand, heterogeneity in mobile device on-board and MEC processor speeds, communication speeds, link access delays and mobile device numbers. The analysis is then extended to show that the relationship between CPU workloads on the mobile device and a MEC server and the link speed between them are crucial parameters that determine the success of offloading to the MEC network to reduce total task completion time and mobile device energy use.

Furthermore, novel distributed heuristic algorithms have been developed that allow a mobile device to decide how to select the least-time schedules of multiple jobs to be offloaded and to identify least-time solutions for multiple mobile devices simultaneously offloading jobs to multiple MEC servers. The proposed heuristic algorithms are tested in a range of numerical simulations and the results demonstrate that the heuristic approach can produce reasonable quality solutions in comparison with linear programming.

Heuristic algorithms have been extended to incorporate time and energy in a network with multiple MEC servers and mobile devices MDs to focus on an objective function with variable weighting factors for time and local energy use; this approach is designed to give the use of a mobile device the maximum flexibility in choosing savings for time and energy use. Numerical simulations in test cases, evaluate the impact of changing weighting factors. The objective function shows a continuous variation as the emphasis is placed on either time or energy saving by the weighting factors. The numerical tests also demonstrate that the proposed heuristic algorithms produce near-optimal computational offloading solutions using a combined weighted score for schedule task completion time and energy.

A preliminary multi-factorial analysis includes economic cost factors to explore how a subscription service could reflect mobile device users' varying requirements in minimising task completion time or extending the battery lifetime of mobile devices.

In the memory of my late Grandmother and Grandfather
and dedicated to my late cousin Kulwinder Singh Bassi
With love and respect.

Never confuse education with intelligence, you can have a PhD
and still be an idiot.

RICHARD P. FEYNMAN

DEDICATION AND ACKNOWLEDGEMENTS

Firstly I would like to express my heartfelt gratitude to my supervisors Dr George Oiknomou and Dr Simon Armour, for their dedication, patience, and encouragement through my PhD study. Their insightful and enlightening supervision has been of important effects on my research. Also, I would like to thank the Engineering and Physical Sciences Research Council and Toshiba Bristol Research and Innovation Laboratory for sponsoring my PhD study. I owe great gratitude to Dr Mahesh Sooriyabandara and Dr Aftab Khan for their unlimited support and encouragement during my study. A special thank goes to Professor. Robert Piechocki for his valuable comments during the review processing.

Also, I am very grateful to my friends and colleagues at the University of Bristol, especially Mr Uful Erol, Dr Nakrop Jinaporn and Dr Ghaith Radhi Hassan Al-Juboori, for their kind support during my research.

I would like to present my honest thankfulness to my parents and sister, for their magnificent effect in my life and their numerous sacrifices for me. I also wish to thank my uncles and brother-in-law and their families for unconditional support and love during my study. Without them and their support, I could not complete my PhD.

AUTHOR'S DECLARATION

I declare that the work in this dissertation was carried out in accordance with the requirements of the University's Regulations and Code of Practice for Research Degree Programmes and that it has not been submitted for any other academic award. Except where indicated by specific reference in the text, the work is the candidate's own work. Work done in collaboration with, or with the assistance of, others, is indicated as such. Any views expressed in the dissertation are those of the author.

SIGNED: . [redacted] ..... DATE: 30/09/2021

TABLE OF CONTENTS

LIST OF TABLES

TABLE **Page**

List of Figures

Figure **Page**

GLOSSARY

3G Third Generation. 14, 90

4G Fourth Generation. 14, 55, 90

5G Fifth Generation. xii, 1, 3, 4, 12, 14, 18, 20, 25, 27, 28, 55, 59, 90, 119, 123, 124

CCTV Closed-circuit television. 15, 17

CPI Cycle per instruction. 37

CPU Central Processing Unit. xvi, 8, 36–38, 41, 49, 53–59, 61, 88, 89, 119–121

ETSI European Telecommunications Standards. 14, 21, 22

FC Fog Computing. 14, 15, 18–20, 22, 26

IIoT Industrial Internet of Things. 3, 4, 20, 25

IoT Internet of Things. 3–5, 14, 15, 18–22, 25–27

IP Internet Protocol. 22

IT Information Technology. 1, 3, 4, 13, 14, 21, 29

LTE Long-Term Evolution. 23

MCC Mobile Cloud Computing. ix, xv, 9, 15, 16, 28–31, 35–38, 49, 52, 93

MD Mobile Device. x–xvii, xxi, xxii, 7–9, 11, 28, 29, 36–38, 40, 41, 48–56, 58, 59, 61–67, 69, 71–75, 77–80, 82, 84–86, 88, 89, 93–105, 109, 110, 112–117, 119–123, 125

mDCs Micro Data Centers. 14, 15, 18

MEC Multi-access Edge Computing. ix–xvii, xxi–xxiii, 1, 3–6, 8, 9, 11, 12, 17, 26–31, 35–42, 44, 46, 48–54, 56–59, 61–84, 86, 88–90, 92–117, 119–125

NFV Network Function Virtualisation. 15

QoE Quality of Experience. 5, 13, 14, 16, 21, 24, 26, 58

RAN Radio Access Network. 3, 5, 21

SDN Software-defined networking. 15

SLA Service Level Agreements. 111

NOMENCLATURE

A_i	Job Allocation Matrix of a MD i
C^{MD}	Total number of instructions to compute a job on the MD i
D_i^{MD}	Amount of data processed on the MD i
$E_{i,c}^{\mathrm{DL}}$	Total down-link energy consumption on a link connecting the MD i and MEC c
E_i^{MD}	Total energy consumption to process all jobs on a MD i
E^{Total}	Total energy consumption of all jobs
$E_{i,c}^{\mathrm{UL}}$	Total up-link energy consumption on a link connecting the MMD i and MEC c
E_i^{idle}	Total idling energy of a MD i
$E_{\max}$	Maximum energy consumption to process all jobs
$F_{i,c}$	Amount of data flowing over the links between the MD i and MEC c
L_i^{MD}	Given CPU workload on a MD i
L_c^{MEC}	Given CPU workload on a MEC c
L_{c}	Loading on MEC c
M^{J}	MECs allocation based on job size
M^{R}	Random allocation of MECs
M^{T}	MECs allocation based on computational time
M^{W}	MECs allocation based on minimum score
O^{GS}	Offloading decision based on guided search
O^{DD}	Offloading decision based on data size

O^{DFP} Offloading decision based on fixed probability

O^{DPD} Offloading decision based on probability distribution

P_i^{MD} Power required of an embedded processor on a MD i

P_i^{idle} Idling power required of a MD i

P_c^{rec} Power required of a MEC c

P_i^{send} Power required of a MD i

P_j^i Probability of job j on MD i

$S_{i,c}^{\mathrm{Max}}$ Transmission capacity of the link i, c

T^{Total} Total computational time of all jobs

T_c^{Total} Total offloading time of job j

T_i^{MD} Total computational time to process the jobs on a MD i

T_{max} Maximum computational offloading time to process all jobs

$T_{c,i,j}$ Receiving time of offloading job j from MEC c to MD i

$T_{i,c,j}$ Transmission time of offloading job j from MD i to MEC c

X_j^{MD} Data size of a job j on a MD

X_j^{MEC} Data size of job j on a MEC

$\Gamma_{i,c}$ Link speed between MD i and MEC c

Π_c Proportion of data size reduction after processing on MEC c

α_i The computing capability of MD i

β_c The computing capability of MEC c

η A bias correction coefficient

$\gamma_{i,c}^{\mathrm{DL}}$ Downlink speed between a MEC c and a MD i

$\gamma_{i,c}^{\mathrm{UL}}$ Uplink speed between a MD i and a MEC c

λ_i Complexity of an application on a mobile device i

$c \in C$ $\{c = 1, \ldots, n\}$ set of MECs

$i \in M$ $\{i = 1, \ldots, m\}$ set of mobile devices

$j \in J^i$ $\{s = 1, \ldots, m_i\}$ set of jobs on mobile device i

$u_{j,c}$ Binary variable denoting offloading decision of job j on MEC c

w_e Weighting on computational energy in the multi-objective function $(0 \leq w_e \leq 1)$

w_t Weighting on computational time in the multi-objective function. $(0 \leq w_t \leq 1)$

CHAPTER 1

INTRODUCTION

An important aspect of Computing Science is efficient and economic problem-solving. Over the years, significant advances have been made in enhancing computing capabilities for problem solving. Figure 1.1 presents the timeline of historic advancements in the area of remote computing. These advances include creating the world wide web in 1980 and the concept of cloud computing in the late 1990s. More recently, advances in Artificial Intelligence have aided applications in, for example, autonomous vehicles [45].

The reach of Computing Science applications in real-world problem solving is extensive. This thesis is concerned with a particular area of Computing Science focused on enhancing mobile device users' experience using Multi-access Edge Computing (MEC) networks. MEC is a paradigm of Edge Computing that offers improved solutions to real-time and delay-sensitive mobile applications within the proximity of cellular networks [129]. It analyses, processes and stores data closer to the customer, reduces latency and provides an improved experience for latency-sensitive and high-bandwidth applications, whereas in Cloud Computing architecture all data are sent to distant cloud data centres for processing.

The year 2020 saw the first commercial publication dedicated to Multi-access Edge Computing: *Multi-Access Edge Computing in Action [129]*. The implication of the title is that the deployment of MEC has reached a stage where an informed assessment could be made as to its further development as an IT sector complementary to established Cloud Computing services.

A close reading of the volume reveals, however, that MEC remains at the level of working groups, public-private initiatives, proof of concept studies and academic-private sector research consortia. The central dilemma is that, while the emphasis is placed on the roles of innovators and vendors to develop applications that take advantage of MEC services, the architecture and provision of MEC platforms remain subjects for debate and rival proposals. In addition, MEC is increasingly discussed as essential for the practical implementation of 5G technology; as stated

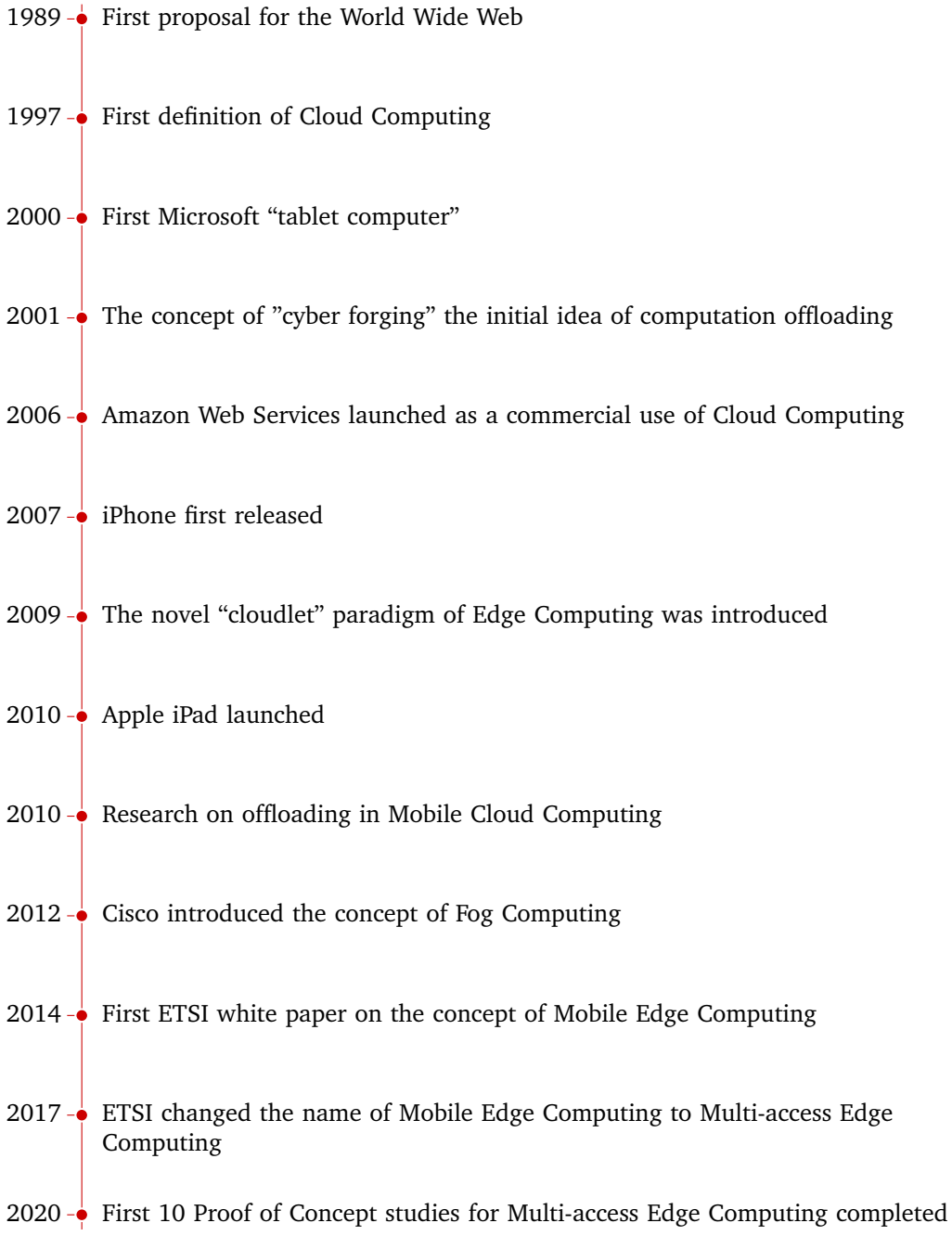


Figure 1.1: Evolution of Technologies Relevant to Multi-access Edge Computing

in [129]:

MEC thus represents a key technology and architectural concept to enable the evolution to 5G, since it helps advance the transformation of the mobile broadband network into a programmable world and contributes to satisfying the demanding requirements of 5G in terms of expected throughput, latency, scalability and automation.

A key word is "latency". Edge Computing as a general concept is invariably contrasted to Cloud Computing. In an IT world dominated by distant consolidated data centres, the finite speed of electromagnetic waves imposes minimum round-trip times of 60 milliseconds or more if a return trip of 20,000 kilometres is required. If applications cannot be successfully run with latency times exceeding a few milliseconds, resort to distant consolidated data centres is not viable. The full roll out of 5G communication links would, seen from this angle, be essential for MEC deployment. However, relatively few applications will demand millisecond resolution; of eight classes of 5G applications that are widely considered - Pervasive Video, 50+ Mbps Everywhere, High-Speed Train, Sensor Networks, Tactile Internet, Disaster Response, eHealth services and broadcast services applications require end-to-end latencies as low as 1 millisecond [129].

Another reason for the slower-than-expected deployment of MEC is also implicit in its all-inclusiveness [124, 125]. This continues to generate a confused literature in which the original term "Mobile Edge Computing" is still used but where "Mobile" has different interpretations [49, 127, 164]. This aspect is discussed in more detail in Section 2.1.

Edge Computing is also not a necessary consequence of increasing traffic to consolidated data centres (Figure 1.2). The year 2017 saw global internet traffic reach 1 zettabyte (1×10^{21} bytes) but investment in the construction and commissioning of data centres is accelerating [126]. Neither is the energy demand of the worlds' data centres a serious issue: 2018 data centre usage was only 1% of global electricity demand and energy usage per server is predicted to decline as larger data centres are built with more than 1×10^5 servers, especially in colder climates were ambient air can be used for cooling [126].

1.1 Introduction to the Edge Computing Paradigm

As discussed in detail in Chapter 2, Edge Computing evolved to serve two distinct sectors:

1. The Internet of Things (IoT) and the Industrial Internet of Things (IIoT), i.e., devices with connectivity which send steams of data to be analysed and stored remotely, for which Fog Computing was designed.

2. Mobile computing and telecommunications devices such as laptops, tablets and smartphones; Mobile Edge Computing originally aimed to provide "IT and cloud-computing capabilities within the Radio Access Network (RAN) in close proximity to mobile subscribers" [52].

[129] also considers the following broad service scenarios for MEC systems:

- Intelligent video acceleration
- Video stream analysis
- Augmented reality
- Assistance for intensive computation
- Enterprise
- Connected vehicle
- IoT gateway

Of these, one aspect of "assistance for intensive computation", i.e. computational offloading, forms the subject of this thesis. With the exception of augmented reality, the other items in the list belong to the IoT or the IIoT.

In September 2020, the following enterprises offered Multi-access (or Multi Access) Edge computing services on their websites:

- Verizon (telecommunications provider and supplier) – for business customers, faster processing, increased bandwidth, ultra-low latency, localized data and expanded IoT potential.
- Intel (semiconductor chip manufacture) – the Smart Edge offering is described as a MEC platform commercialized for market use cases and on-premise enterprise deployments.
- Vodafone (telecommunications provider and supplier) – with mobile and IoT devices, MEC services reduce network congestion and speed up application performance, with high definition graphics, virtual reality, etc.
- Ericsson (networking and telecommunications) – assesses the majority of MEC/5G revenue potential to come from enterprise and IoT services but, for private consumers, virtual reality and gaming applications will be important features of MEC services.
- AT&T (telecommunications) – services in retail, manufacturing, healthcare and sports stadium IT and data management.
- SkyLab Network (on-site data processing and analysis) – offers MEC devices deployed at construction and engineering sites with video analytics and facility management applications to monitor and process data from cameras and on-site sensors (motion, fire, water, etc.).

- Cisco (networking hardware, software, telecommunications equipment, etc.) - mobile content delivery network for providers of video entertainment products.

The clear emphasis on IoT applications in the above examples is reinforced by the analysis presented in [129] of MEC platforms offered to software developers for IoT devices: Cloud IoT Edge (Google), Greengrass (Amazon Web Services) and Azure IoT Edge (Microsoft).

1.2 MEC/Human User Interfaces

A second medium is that of a private car, where safety information and context awareness as well as video steaming, augmented reality and infotainment services can be efficiently operated via MEC networks than by Cloud Computing [123].

Mobile devices such as smartphones are also self-evidently mobile in public service vehicles and private cars. For this thesis, however, mobile devices will be considered to remain within the base station Radio Access Network (RAN) of a single MEC service. The physical extent of implemented MEC networks remains unclear; some authors consider whole cities or geographical regions to be a single network [27]. Most authors consider much smaller ranges for RAN-assisted MEC networks.

The interfaces between human users of mobile devices and MEC networks is still very much being defined for services and commercial offerings. The original motivation for Edge Computing was straightforward in outline: Edge Computing proposes moving data processing capability away from distant consolidated data centres “in the cloud” to servers usually (but not exclusively) physically closer to the end user in order to support high Quality of Experience (QoE) applications for heterogeneous mobile device users and fixed internet-connected streaming devices via wireless networks. Of the four major paradigms for Edge Computing that were advanced between 2008 and 2016, the human connections to Cloudlets, Mobile Edge Computing Micro Data Centres were easily identified. In contrast, Fog Computing was put forward as the optimal solution for IoT applications to minimize the time required for time-critical processing and Big Data analytics.

The redefining of Mobile Edge Computing as Multi-access Edge Computing, however, subsequently amalgamated all the paradigms under one overall concept in which the new key criterion became that of access for mobile, fixed and vehicular devices for communication with Edge Computing services. Both fixed and mobile devices can use RAN to access MEC servers. For example, if a portable server is installed in a remote site, this becomes the “micro data centre” or if a server is installed in a local enterprise (such as a coffee bar) with non-proprietary software, the result is a “Cloudlet” (as originally defined). In many ways, Multi-access Edge Computing combined Mobile Edge Computing with Fog Computing to become the totality of Edge Computing concepts under one heading. For the analyses presented in this thesis, the Mobile Edge Computing component of MEC networks is represented by mobile devices such as

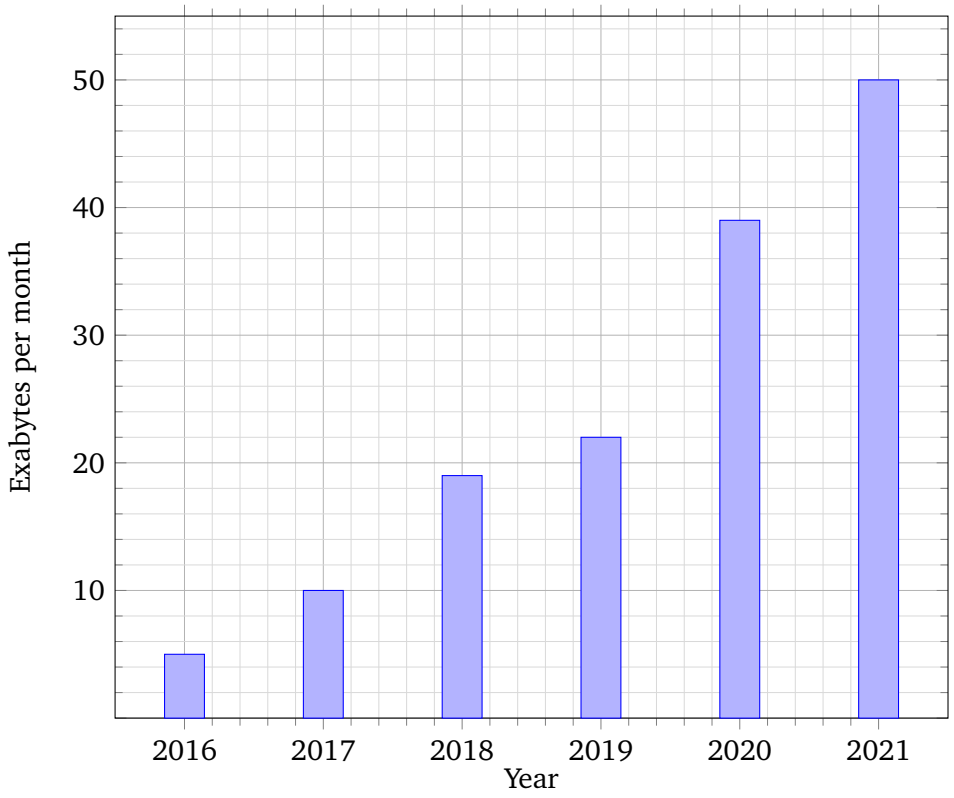


Figure 1.2: Historical and projected trend of the mobile data traffic (redrawn from data in [129])

tablet and smartphones communicating with local base stations to access superior computing resources and to extend battery lifetime.

1.3 Motivation of this Thesis and Research Outcomes

1.3.1 Motivation

At the start of this research project, an increasing number of reports had been and were being published about computation offloading to Cloud Computing centres; in contrast, offloading to MEC networks was considerably less investigated. The aim of the research was to comprehensively analyse offloading to heterogeneous MEC networks in which mobile device ("on-board") processor speeds, MEC server ("server-side") processors speeds and communication link speeds in the network to MEC servers were variables to reflect practical scenarios. The research outcomes of this Thesis would be used to lead to conclusions for implementation strategies for MEC networks offering offloading as a service as part of the assistance for intensive computation provided by Edge Computing.

Table 1.1: Summary of research questions and outcomes, their relevant chapters in the thesis and the publications derived from the work

Key Findings	**Research Question**	**Thesis Chapter**	**Publication**
The amount of data transmitted for a computational job from the MD is a neutral parameter as regards offloading/local processing decisions. Increasing data size results in both longer local processing and transmission/reception times and cannot produce a "crossover" effect.	Question 1	Chapter 3	Singh et al (2020a)
High computation complexity tasks are loaded at much lower communication link speeds than lower complexity tasks. Both types of task can yield very high (>98%) local energy saving but high-complexity tasks achieve this at much lower link speeds.	Question 1	Chapter 3	Singh et al (2019)
Link access delays can eliminate any time advantage of offloading; lower complexity tasks tolerate shorter link access delays.	Question 1	Chapter 3	Singh et al (2019)
High-complexity tasks are offloaded with a time advantage at lower server:on-board processor speed ratios especially at higher link speeds.	Question 1	Chapter 3	Singh et al (2019)
High-complexity tasks are offloaded with a time advantage with greater numbers of MD competing for access than is the case with lower complexity tasks.	Question 1	Chapter 3	Singh et al (2019)
With high-complexity applications, the time advantage of offloading was only lost at very high server CPU workloads; with high-complexity applications, the advantage of local processing was lost at high MD CPU workloads.	Question 1	Chapter 3	Singh et al (2020a)
Distributed heuristic algorithms were developed to closely rival optimum solutions from linear processing when one MD attempts to offload up to 20 jobs.	Question 2	Chapter 4	Singh et al (2020b)
Centralised heuristic algorithms were developed to closely rival optimum solutions from linear processing when multiple MDs attempt to offload up to 115 jobs.	Question 2	Chapter 4	Singh et al (2020b)
Distributed and centralised heuristic algorithms closely rival optimum solutions from linear processing for single or multiple MDs. Combining time and local energy shows that there are only trade-offs between them	Question 3	Chapter 5	Singh et al (2021)

1.3.2 Research Questions and Problem Formulations

The following research questions were formulated to translate the motivation of the project into specific research topics.

> **[Q1]:** What type of job on a mobile device (MD) benefits from offloading in terms of either total task completion time and/or energy usage by the MD given inevitable limitations in both hardware (communication link speeds, link access delays, fixed on-board processor speeds, the number of MD users attempting to offload to a MEC server simultaneously, CPU usage of the MEC server, etc.) and software considerations, in particular the computational complexity of any task to be offloaded?
>
> **[Q2]:** When a MD or multiple MDs offload multiple jobs to a MEC network with multiple servers can a schedule be locally or centrally optimised to minimise total task completion time?
>
> **[Q3]:** Can both time and energy savings from offloading be combined to explore optima either for a single MD or for multiple MD's and, if the relationship between a MD and a MEC network is on a subscription basis, can incurred costs of offloading be further used with time and energy factors?

1.4 Thesis Outcomes

1.4.1 Contributions of the Thesis

The specific research questions that this thesis answers are listed in Section 1.3.2. The high-level solutions to the research questions are provided in the Table 1.1. In terms of the subject-specific contributions, this thesis contributes to the state-of-the-art in the following three ways:

- An investigation of the key factors that affect computation offloading from a single mobile device in heterogeneous MEC networks to reduce task computation time and energy use by the mobile device;
- Proposed heuristic approaches for near-optimal offloading in a MEC network using a schedule approach of jobs to be offloaded from a single mobile device or from multiple mobile devices to reduce task computation time;
- Proposed heuristic approaches for near-optimal offloading in a MEC network using a schedule approach of jobs to be offloaded from a single mobile device or from multiple mobile devices to incorporate savings in computation time and energy use by mobile devices.

The contributions mentioned above are discussed in the following subsections.

1.4.1.1 Factors Determining the Advantage of Computation Offloading in Multi-Access Edge Computing

To identify answers to Q1, parameters in mathematical models require reliable quantitative estimates. An exhaustive literature search located very few reports with definite numerical values for computational task complexity, processor speeds or MD energy power ratings. One of these reports outlined a conceptual and mathematical model for offloading in Mobile Cloud Computing (MCC) and this model was initially adapted and extended for MEC networks to answer Q1 (Table 1.1).

Several critical parameters are considered to see how a heterogeneous population of mobile devices with different MEC server-side processors interact in decisions affecting offloading. In particular, different mobile device and server-side processing speeds are considered with respect to achieving successful offloading for shorter task completion time. Modelling how link access delays, excessive MEC server load caused by large numbers of users and different data transmission speeds to the MEC servers are considered. A detailed description of the mathematical model and the results of numerical simulations are presented in Chapter 3. This multi-variable analysis of the offloading process was an advance on previous studies and guided the choice of numerical parameters for further work.

A new model was then constructed to include effects of CPU workloads in both on-board and sever-side processors; the first case explored increasing activity on the MD while the second was another reflection of overloading in the MEC network.

The amount of data transmitted from a MD was also investigated as a factor influencing offload/local processing decisions (Q1).

1.4.1.2 A Heuristic Approach to Optimising Offloading Schedules in Heterogeneous Multi-Access Edge Computing Networks

Work to identify answers to Q2 involved exploring the concept of the "makespan", i.e. the last operation to finish in a schedule of operations, to define total task completion time. Preliminary direct-calculation studies with 81 or 1024 different offloading schedules from one MD to two MEC servers were used to validate a linear programming approach.

With larger number and MDs, jobs and MEC servers, the numbers of possible solutions increase dramatically and a heuristic approach to identifying near-optimal solutions was adopted (Table 1.1). Several different heuristic algorithms were developed to run in a distributed manner on individual MDs to investigate if this approach could closely rival optimum solutions from linear processing. Centralised heuristic algorithms were then developed for multiple MDs attempting to offload multiple jobs; such algorithms were envisaged as candidate central resource allocators in a MEC network.

1.4.1.3 Multi-Criteria Heuristic Optimization for Computational Offloading in Multi-Access Edge Computing

To identify answers to Q3 (Table 1.1), preliminary studies with relatively small numbers of possible offloading schedules were performed to align results with those from linear programming. Heuristic algorithms were then adapted to incorporate both time and energy factors for single and multiple MDs.

To extend the analysis to economic costs, illustrative relative costs were taken from established Internet services. The approach taken was to define if reduced task completion time, reduced local energy use or lower incurred cost depending could be combined in a flexible manner depending on individual circumstances; for example, low battery charge might favour a high emphasis placed on offloading more jobs to minimise local energy use but avoiding high costs of offloading might favour strategies to either minimise time or energy use.

1.4.2 Publications

The research presented in this thesis led to the publication of the following articles:

1.4.2.1 Published

1. R. Singh, S. Armour, A. Khan, M. Sooriyabandara and G. Oikonomou, "The Advantage of Computation Offloading in Multi-Access Edge Computing", Fourth IEEE International Conference on Fog and Mobile Edge Computing (FMEC), Rome, Italy, 2019.

2. R. Singh, S. Armour, A. Khan, M. Sooriyabandara and G. Oikonomou, Identification of the Key Parameters for Computational Offloading in Multi-Access Edge Computing, IEEE Cloud Summit 2020

3. R. Singh, S. Armour, A. Khan, M. Sooriyabandara and G. Oikonomou, Towards Multi-Criteria Heuristic Optimization for Computational Offloading in Multi-Access Edge Computing, IEEE 21st International Conference on High Performance Switching and Routing, 2021

4. R. Singh, S. Armour, A. Khan, M. Sooriyabandara and G. Oikonomou, "Heuristic Approaches for Computational Offloading in Multi-Access Edge Computing Networks", IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (IEEE PIMRC 2020)

1.5 Structure of the Thesis

Chapter 2 presents an in-depth analysis of how MEC evolved as a concept inside Edge Computing and the several proposed implementations of Edge Computing.

Chapter 3 analyses how different parameters in a heterogeneous MEC network determine whether or not offloading results in time and energy savings for the user (Q1).

Detailed numerical simulations were performed to explore how offloading can be beneficial in a MEC network with varying quantitative mobile user demand, heterogeneity in mobile device on-board and MEC processor speeds, computational task complexity, communication speeds, link access delays and mobile device user numbers. The range of link speeds was deliberately wide and two types of communication delay were also included.

A novel mathematical model of offloading was developed to include CPU workload in both the on-board and server-side processors (Q1). With this model, the impact of changing CPU workloads was investigated. Scenarios are developed using two applications of varying complexity offloaded from the MD.

Data size (in MB) of the task to be offloaded, the balance between the CPU workloads on the MD and MEC were investigated in relation to communication speeds required for shorter completion time by offloading. In addition, MD energy usage, modelled to increase linearly as MD CPU usage increases, was estimated for high-complexity and low-complexity application tasks run on the MD or offloaded in the MEC network.

Chapter 4 presents a model of a system that consists of multiple MEC servers and multiple MD users (Q2). Each MD was given multiple computational tasks to perform, and each task could either be computed locally on the MD or be offloaded to one of the MEC servers. The theoretical optimal allocation was computed with linear programming that minimises the time required to complete the computation of all tasks.

A distributed heuristic algorithm was then devised that allows each MD to independently, and using local knowledge only, decide how to handle each individual job. Three approaches were tested in algorithms to decide whether to offload each individual job; three distinct mechanisms were then used to determine which MEC server each task should be offloaded to. Simulations were used to evaluate those approaches in terms of how well they could approximate the theoretical optimum.

Chapter 5 describes a procedure to extend the model presented in Chapter 4 to incorporate both time completion time and local (MD) energy use. In the simulated heterogeneous MEC network, each MD had multiple computational jobs to process, and each task could again be processed locally or offloaded to one of the MEC servers. Several heuristic offloading options are developed and tested with an objective function for both time and energy with a range of weightings for optimizing time and energy.

The approaches were demonstrated in three test cases, which evaluated the impact of changing weighting factors for time and energy. The objective function was investigated as the

emphasis was placed on either time or energy saving by changing the linked the weighting factors. Numerical tests were used to explore if heuristic algorithms could produce near-optimal computational offloading solutions at different combinations of weighting factors for schedule task completion time and energy.

Chapter 6 summarises the research outcomes from Chapters 3-5 as conclusions and the implications for the Quality of Experience for the users of mobile devices accessing MEC networks. The discussion then analyses possible extensions of the work presented in this thesis, with particular emphasis on how 5G networks will affect offloading efficiencies.

CHAPTER 2

MULTI-ACCESS EDGE COMPUTING: A LITERATURE SURVEY

2.1 Introduction

The purpose of this Chapter is to place Multi-access Edge Computing in the broader landscape of Edge Computing and to trace its historical development via Mobile Edge Computing to identify its particular characteristics to contribute to contemporary telecommunications and IT and its suitability as a medium for computation offloading.

As presented by [98], IT developed after 2000 to offer new services and Cloud Computing was established as a novel computing infrastructure for the internet, based on highly resourced data centres. Interest in and adoption of Cloud Computing services has increased to the extent that global Cloud IP traffic will account for more than 90% of total data centre traffic by 2020 [36]. The main advantages of the Cloud computing paradigm remain "unlimited" storage capacity and computing resources, reduced capital expenditure and minimized carbon footprints [23, 25]. However, this technology faces key issues: speed of services and slow connections, which are often combined as low bandwidth/high latency and jitter as mobile devices offload computational and processing capacity to Cloud Computing services [30]. These challenges have been exacerbated by the continued proliferation of mobile and fixed internet-connected devices.

Problems of high latency and narrow bandwidths with reduced Quality of Experience (QoE) for users led to proposals to re-imagine the cloud: rather than being thought of as a homogeneous entity, the cloud would have a distinct "edge" separate from the core in which large-scale processing and storage would occur. Devices could, therefore, communicate with local servers unless a need arose for contact with the cloud's core competencies. This view was first articulated as the challenge to the rapidly increasing reliance on mega-data centres for hosting cloud computing [34]. These authors argued that geo-diverse multiple data centres

would provide a superior model for applications such as email distribution, using "local" servers to filter out spam and blocking undesirable forms of traffic closer to their points of origin. This formed, in effect, the first proposal for "edge" computing based on the deployment of "micro data centers" (mDCs) as advanced by Microsoft, Inc., which can be seen as a highly distributed cloud focused on mobile users and connected devices and requiring the installation of a global infrastructure of hardware sites, each with a limited number of servers (up to 10 per centre) and supplied with several terabytes of memory [34, 12].

Shortly afterwards, the paradigm of the "cloudlet" for small numbers of casual and transient mobile device users in locations such as coffee shops and restaurants etc. was articulated [137]. A third concept developed from the increasing availability and use of fixed internet-connected sensors (the "Internet of Things") requiring fast responses; this was structured as the "Fog Computing" (FC) paradigm [18].

Mobile vendors have brought powerful smart mobile devices to change the fundamentals of how people interact with IT and telecommunications. Due to greatly increased demands of mobile devices such as smartphones and tablets and because intensive mobile applications require high levels of processing and rely on remote data centres, accessing mobile services at "anytime, anywhere" increasingly clashes with users' QoE and their sense of personal privacy and control [57]. By its very nature, Edge Computing must be accessible by (and respond to) a heterogeneous collection of devices in wireless networks, which may be Wi-Fi, 3G, 4G and (in the future) 5G. To ensure the key essentials of low latency and high bandwidth in this highly flexible and highly changeable system, wireless interference must be minimized [52].

Multi Edge Computing initially emerged as an edge computing paradigm where a mobile user does not need to access cloud computing for data or computing capabilities in remote data centres but can use "edge" computing resources. The fundamentals were discussed in a white paper published by the European Telecommunications Standards (ETSI) in 2014 [52]. The concept of Mobile Edge Computing is simply to provide mobile and cloud computing services within close proximity of the mobile user, i.e. the provision of computing power in a delocalized manner close to mobile users (smartphones, tablets, etc.), aiming to decrease latency, achieve as high throughput as possible and provide direct access to real-time network information. The renaming of Mobile Edge Computing as Multi-Access Edge Computing reflects aims for applications development in 2017 for non-mobile devices; this is a crucial change of direction and its full implications will be discussed later in this Chapter [1].

The key objective of Edge Computing is to put resources within close proximity to the users and sources of data and information to help overcome cloud computing's recognised weaknesses of high latency, jitter and narrow bandwidth [100, 137]. These performance parameters are particularly important and relevant for wireless access networks in the context of a user's computing devices as well as in the Internet of Things (IoT):

[1] http://www.ETSI.org/news-events/news/1180-2017-03-news-etsi-

- Agility of Services: Mobile devices and fixed IoT sensors generate enormous amounts of data sent to the central Cloud. However, due to the centralized approach of the Cloud, this has lacked various important features such as contextual and location awareness. If Edge Computing processes data at the edge network, then context and location awareness are much more readily obtainable and achievable.
- Low latency: Reducing the time required when a packet travels from a node to the destination is critical in high processing applications such as augmented reality and gaming where mobile users expect uninterrupted services from the content provider.
- Coherence: The Edge Computing architecture can determine where to offload data, either on the local device or the edge network. Smart sensors make decisions and this improves the performance of the overall network and sends only useful data to the cloud. For example, a closed-circuit television (CCTV camera captures and transmits information only when movement occurs in close proximity to the camera.
- No Single Point of Failure: Edge Computing stores the limited amount of resources that allows applications to control computing offloading and networking resources to achieve the high level of efficiency and performance. Additionally, the architecture of Edge Computing provides a distributed approach if the primary edge network resources pool fails, "instantaneously" redirecting traffic to alternative edge network resources. With technologies, such as software-defined networking (SDN) and Network Function Virtualisation (NFV), this enables reliability and robustness of the network and improves integration with existing IoT environments.

Figure 2.1 illustrates the four Edge Computing paradigms in three-tier hierarchies and shows where actual functionality can be implemented either at the end device or at the edge network. FC end devices such as CCTVs can do some processing and send useful data fog nodes in the Fog's core. A simple cloudlet server at the business premise can perform processing itself rather than at the end devices or in combination with end devices. An mDC processes multiple users' requests locally. Mobile Edge Computing introduced a base station to act as the primary call site for mobile devices. Lastly, the concept of Mobile Cloud Computing (MCC) repeats many features of Cloud Computing but, because of the constraints of mobile devices (processing power and battery life), data processing is forwarded to Cloud data centres and is therefore not Edge Computing.

Each of these four Edge Computing concepts, Cloudlet, FC, Mobile Edge Computing and mDCs, shares a perceived vision of the future of the internet that addresses the mismatch between Cloud Computing (with its finite number of distant data centres) and the increasing number of mobile users competing for access with a continued increase in edge devices [154]. In addition, studies have already begun to explore how edge servers will be able to adapt to the anticipated heavy usage [135].

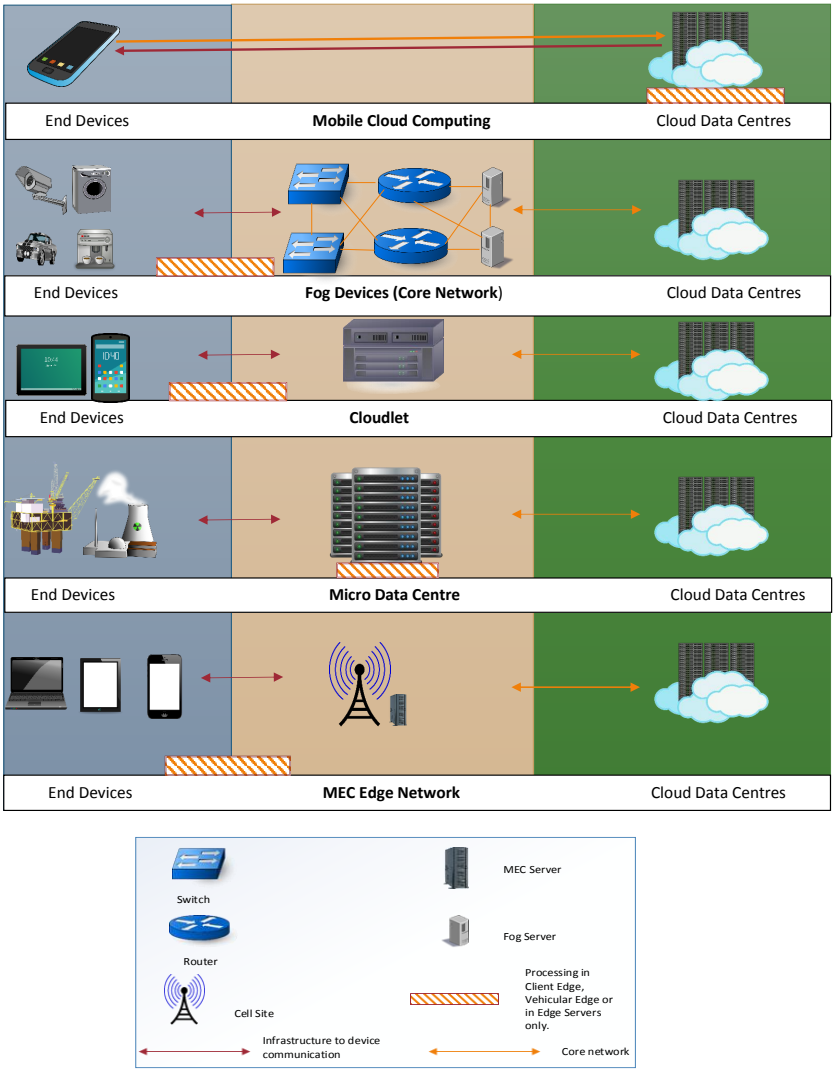


Figure 2.1: The Three-tier relationship between users/devices and cloud computing and four paradigms of intervening Edge Computing as well as MCC

Illustrative examples of the broad area of applications considered for Edge Computing are:

- Context aware applications: Previously, when the internet user frequently web surfed to see favourite content, in order to fulfil the user's request, the internet provider used historic information in their database, but increasingly internet service providers can provide users' favourite contents via geographical locations or analysed information from the application [111]. With Edge Computing, content providers can host services at the edge of the network with accurate user location within a radio access network and this can improve the QoE for mobile users.

- Smart Transport: Many cities are trying to implement various forms of this, for example

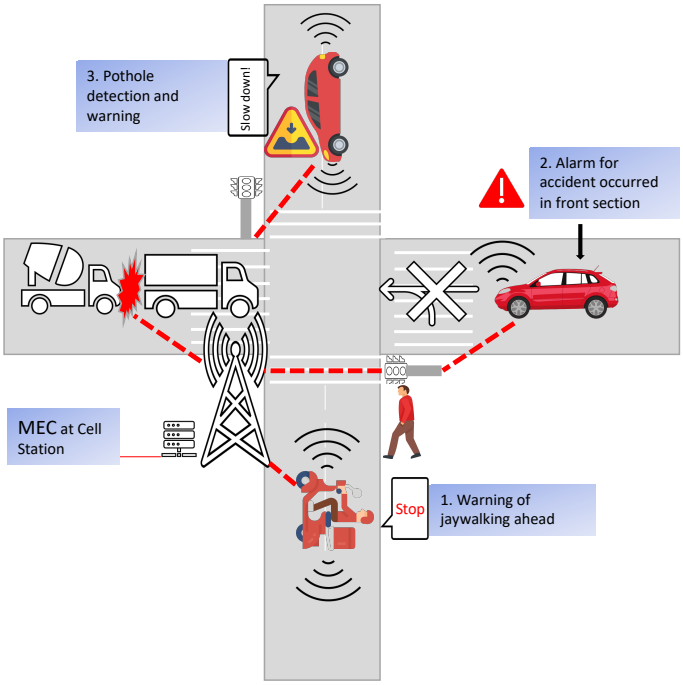


Figure 2.2: MEC enabled Traffic safety service

when poor traffic decisions are made by traffic controllers, adverse weather conditions and delays caused by road re-construction all add to traffic congestion and inefficient fuel usage [38]. With Edge Computing, city traffic can be automatically managed by edge servers via data collected from intelligent sensors at traffic lights and CCTV cameras on highways (Figure 2.2). Each sensor detects car movements and makes decisions accordingly and traffic lights can react to this processed information. Another important scenario is smart car parking, where users will access information about urban car parking spaces according to their precise geographical location. Currently, a smart transportation environment uses cloud computing where all the processing is done at remote data centres but cloud computing lacks key safety features in smart transportation. For example, if a driver-less car needs to stop in case of a dangerous situation, it has to upload the data to the cloud which then performs a computing process and sends the “stop” command to the car, when the car finally acts upon the instruction. A more rapid solution is to bring computation capability close and Edge Computing can provide limited levels of computation capability to make quick (lower latency) decisions.

To more clearly define how Edge Computing paradigms have been unified in a multi-access approach to heterogeneous networks that employ Wi-Fi technologies and which will

evolve to utilize 5G technologies, the approach used in the Chapter is as follows. Section 2.2 briefly explores the background to individual variants proposed for the implementation of Edge Computing. Section 2.3 analyses the limited number of published accounts of experimental demonstrations of direct comparisons of communication speeds in Edge and Cloud Computing. Section 2.4 presents a novel analysis that compares the different Edge Computing paradigms from the viewpoints of applications, functionalities and technologies. Section 2.5 discusses how FC and Mobile Edge Computing have been unified as Multi-access Edge Computing to offer practical solutions to a range of Edge Computing requirements in heterogeneous wireless networks. Finally, Section 2.6 links the concepts discussed in this Chapter to how computation offloading , which was initially established in Mobile Cloud Computing, could use the resources of Multi-access Edge Computing to develop improved offloading functionalities.

2.2 Edge Computing Approaches

2.2.1 Micro-Data Centres

Ruggedized mDCs, able to be sited out of doors, are already available for installation in remote sites, for example for oil/gas exploration, and any industrial application requiring sensor data, machine-to-machine communication and control and automation technologies will be amenable to this form of edge computing. Bahl [12] discusses mobile devices with wireless connections to mDCs achieving improved battery life between recharging events and high-end game stream but many applications are clearly IoT-related. Juniper presents industrial scenarios in which locatable mDCs can be operated in extreme environments or on a temporary basis [20, 23].

2.2.2 Cloudlets

The concept of the cloudlet was derived from two basic premises: firstly, mobile devices (excluding laptops and notebooks) were “resource poor”, i.e. compared to static PCs and laptops/notebooks, mobile devices have little computing power; secondly, providing a “data centre in a box” offered a small number of mobile device users in a private business (for example, a coffee store) the ability to leverage computing power [137]. “Computing power” was – in this original context – a suite of open-access software options incorporating Linux applications for word processing, spreadsheet data processing, etc. These resources would be combined in a maintenance-free virtual machine environment with post-use clean-up for users accessing the cloudlet transiently via short-range Wi-Fi connections.

Cloudlets as a business concept have failed to gain traction and the default offering from small enterprises has become free Wi-Fi for mobile devices (including laptops and notebooks). The driver for this has undoubtedly been the near-pervasive use of social media by mobile users, the vast majority of whom have shown little taste or need for a “data centre in a box”. This

in turn evolved the cloudlet concept into providing one link in a three-tier hierarchy: mobile device/cloudlet/Cloud [136].

Such an arrangement was made explicit in an application demonstrated for cloudlet computing in cognitive assistance [136]. This proof of concept study utilized Google Glass, streaming video from the Google Glass device to the cloudlet. The cloudlet can subsequently link with the "traditional" Cloud for services including centralized error reporting, usage logging and pre-collection of data. Logically, this hierarchical architecture has gradually linked cloudlets to other forms of Edge Computing, in particular as local solutions to high latency and low bandwidth problems for IoT applications [138]. In practical terms, this closer proximity might be implementable by augmenting Wi-Fi access points by adding processing, memory and storage; a desirable side-effect of such an arrangement would be to extend the battery life of the expected mobile device by requiring less energy usage for transactions with the cloudlet [65].

Application use cases proposed for cloudlets have included linking mobile devices to large public screens [37], applications such as face and speech recognition, object identification, physical simulation and rendering and Augmented Reality [64] and cognitive assistance [136]. Such experimental demonstrations have shown the power of a local cloudlet in accelerating the completion of computing tasks by resource-poor mobile devices. In other words, it is not the lack of effective functioning of cloudlets that have made the commercial take up of the cloudlet paradigm minimal, rather it is the lack of profitability in the conceptual business model. Therefore, dynamic cloudlets were proposed from any mobile devices in the network that possessed the necessary computing resources [97, 155]. In this analysis, an infrastructure co-located with the Wi-Fi access point was proposed which was also capable of discovery locally devices that could share computing resources. The same fundamental idea has been elaborated into the FemtoCloud proposal, in which mobile devices with significant idle computing power can link via a client service installed on the devices [65].

Attempts to standardise cloudlet technologies have been made by the Open Edge Computing project [2].

2.2.3 Fog Computing

The origins of Fog Computing (FC) can be traced to a conference presentation by Cisco Systems, Inc. in 2012 [18]. The authors considered the problems inherent in devices accessing cloud computing resources and, like the cloudlet architecture, proposed the insertion of an extra layer between the end user device and the cloud: embedded systems and sensors linked first to a "field area network" that comprised the FC ("distributed intelligence") element which itself communicated with the Cloud.

As FC ideas expanded, a concept advanced by Cisco became much-quoted: "Analysing data close to the device that collected the data can make the difference between averting disaster

[2] `http://openedgecomputing.org`

and a cascading system failure". Furthermore, the same document [35] states "Any device with computing, storage, and network connectivity can be a fog node. Examples include industrial controllers, switches, routers, embedded servers, and video surveillance cameras". This explicit linkage to fixed devices with connectivity and intelligent data analysing and data processing capabilities clearly distinguished FC from cloudlets. The scenarios of interest identified were: networks of wireless sensors, connected vehicles, smart grid distribution networks and in any context where data is collected at the "edge": vehicles, ships, factory floors, roadways, railways, etc. [18]. The great value of this approach was noted as the dense geographical distribution with a local focus which can analyse big data faster [132].

Smart traffic lights in vehicular networks, self-driving vehicles, smart meters monitoring domestic energy use and, pipeline monitoring, wind farms, closed loop control of industrial systems, and applications in the oil and gas sector were soon added to the list of things and fog functional relationships [17, 147, 153]. The relationship between FC and the Cloud was succinctly described as: "Thus, the solution to this problem is a multi-tiered architecture (with at least three tiers) whereby an IoT application is deployed as follows: a part on the "thing" (e.g. a car), a second part on the fog platform (e.g. a roadside cabinet or a router in a wireless access network or an LTE base station), and in the case of three tiers a final third part in a data-centre of the main cloud (e.g. Amazon EC2)" [86]. However, even by 2013, degradation of the link between the IoT and FC began to be evident. One of the features of FC was claimed to be the "great support for mobility", although "mobility" was not further defined [132]. Furthermore, [32] included 5G mobile devices along with the IoT, cyber-physical systems and data analytics in the applications open to FC. Similarly, mobile users were considered to use applications that could benefit from FC [42, 101, 107, 150].

Nevertheless, focusing FC concepts on real-time data processing and analytics as opposed to the human user-generated demand for computing and processing power was emphasised in a proposal for "Edge-centric Computing" [57]. These authors also asserted that trust in the security of personal and social sensitive data would be increased if the management of such sensitive data could be ensured at the edge, rather than being centred in distant data centres.

The authors of [151] discussed five likely areas for Fog Computing deployment: healthcare, smart grids, smart vehicles, urgent computing and Augmented Reality. Healthcare applications have been extensively investigated [5, 39, 56, 141, 147]. Other authors [112] studied smart grids based on meters in domestic and industrial settings to provide real-time data on power. Vehicular FC is a novel proposal to combine computational devices both on board vehicles and the power and resources of Edge Computing [71]. As mentioned above, urgent computing in disaster and emergency situations is an FC application that can greatly speed up response times and optimize responses [151].

To summarise, FC is focused on IoT and IIoT applications with services offering computation, data storage and networking among devices [19, 165]. The standardisation of FC technolo-

gies has been addressed by the OpenFog Computing project [3] and the Industrial Internet Consortium [4].

2.2.4 Mobile Edge Computing

The definition provided by the first paper was "Mobile-Edge Computing provides IT and cloud computing capabilities within the RAN in close proximity to mobile subscribers" [52]. This highlighted the role of telecommunications in Mobile Edge Computing, which could be viewed as uniting the telecommunications industry with IT at the mobile network edge. A variant term - "mobile edge cloud computing" - has also been used but this is conceptually indistinguishable from Mobile Edge Computing [30].

As its core concept, Mobile Edge Computing re-configures the devices already deployed at the mobile edge as mobile access points, i.e. base stations forward traffic but also add computing and storage capabilities to act as Mobile Edge Computing servers [14]. Four distinct stakeholders contributed to this early vision of Mobile Edge Computing: mobile users connecting to base stations, Mobile Edge Computing servers and other hardware owned and maintained by network operators, while Internet providers added connectivity to Cloud elements (data centres and content distribution networks) in which application service providers host applications. This architecture aims to reduce latency, improve bandwidth and enhance scalability for mobile users while catalysing the development of entirely new services.

The original ETSI white paper [52] envisaged six use cases: active device location tracking, Augmented Reality content delivery, video analytics, "Radio Access Network" (RAN) aware content optimization, distributed content and domain name system caching and application-aware performance optimization. Of these, video analytics was presented as an IoT application use case in which video streams from cameras were utilized for public safety and smart city data collection. The other five were, however, genuinely aimed at mobile device users.

Subsequently, ETSI sponsored proof of concept (PoC) studies intended to demonstrate the viability of the Mobile Edge Computing concept [5]; these studies included:

- PoC 1: Video User Experience Optimization via Mobile Edge Computing. This Mobile Edge Computing application is intended to recognize paid video streams from the content provider and to assign a higher priority to those streams, providing them with a higher bit rate. Paid subscribers would, therefore, have a higher QoE.

- PoC 2: Edge Video Orchestration and Video Clip Replay. This application running on the Mobile Edge Computing server enables the mobile user to receive live video streams from professional stadium cameras, choosing camera angles, etc. This model potentially

[3] https://www.w3.org/2017/05/wot-f2f/slides/OpenFog-Overview-W3C-Open-Day-in-May-2017.pdf
[4] https://opcfoundation.org/markets-collaboration/openfog/
[5] https://www.etsi.org/technologies/multi-access-edge-computing/mec-poc

enables unique stadium services, such as the ability to view in-game "player cams" to enrich fan experience or could support venue services such as food ordering.

- PoC 3: Healthcare-Dynamic Hospital User, IoT and Alert Status management. A typical 'Healthcare' application is considered in which a hospital can devise a cellular access hierarchy and open access to local systems based on managed access rights. This PoC has strong IoT Component.

Mobile Edge Computing evolved rapidly. In addition to, or in place of base stations, Mobile Edge Computing could utilize more cost-effective points in Internet Protocol (IP) networks to adopt Network Functions Virtualization technologies in distributed Mobile Edge Computing platforms [20]. However, another trend was that of incorporating IoT in Mobile Edge Computing schemes and implementation scenarios [130]. This was at least partly envisaged by the original document for Mobile Edge Computing including machine-to-machine scenarios connecting sensors to Mobile Edge Computing servers [12]; an IoT Gateway was subsequently discussed in an ETSI white paper [73]. This was unfortunate because confusion with Fog Computing was made possible [101, 107, 113, 152]. Nevertheless, different drivers have been emphasized when these two edge computing paradigms are compared and a key difference between FC and Mobile Edge Computing was that wireless IoT networks could be viewed as the principal driver for FC whilst low latency and resource efficiency in cellular networks were dominant considerations for Mobile Edge Computing [139]. ETSI's Mobile Edge Computing Industry Specification Group (Mobile Edge Computing ISG) was formed in 2014 to begin the process of standardisation [12].

2.3 Direct Experimental Comparison of Edge and Cloud Computing

Other than speculating on business models, the original presentation of the cloudlet concept left unanswered important questions concerning how much computing power would be required and the acceptable access time for any individual user [137]. More detailed analysis of the operating parameters of a cloudlet required specific application use cases and a seminal one was provided by a study of cloudlet functioning to facilitate the appropriation of a digital screen by a mobile device user [37]. The scenario envisioned was dramatic but realistic: a doctor (physician) was interrupted over dinner in a restaurant and asked to scrutinize a pathology slide in real time as a surgical operation continued; a smartphone was used to access a large lobby screen to fully visualize the medical state of affairs. To simulate this, a simple display screen game was played using cloudlets and a commercial cloud to assess the impact of physical distance on user experience (Table 2.1).

To some extent at least, physical proximity must reduce latency; considerably longer latency delays have, under experimental conditions, been measured than the "ideal" latencies imposed by geographical distance, due to queuing and other delays (Table 2.1). In the study reported in [37], a mobile device in the UK experienced latencies while connecting to cloudlet computers in Europe that were as little as 17.5% those with distant Cloud centres; connecting to a more physically distant cloudlet produced much longer latencies. Roundtrip times could be reduced by up to 90% and mobile device energy use by up to 88% [64, 72]. In a separate practical test of cloudlet versus Cloud, reductions in power usage by 95, and a nearly 10-fold reduction in task completion time were achieved [78].

These positive results for cloudlets were extended in a study of cloudlet versus Cloud performance using both Wi-Fi and radio access Long-Term Evolution (LTE) means of communication from mobile devices [37]. With a variety of applications, the use of a local cloudlet greatly improved response times (in some instances by a factor of approximately five-fold); the greater the geographical distance to the Cloud, the more the response time was elongated (Table 2.1). In addition, energy usage by the cloudlet was always reduced when compared with any Cloud option, in some cases by factors of 7-10 [64, 72]. In a separate practical test of cloudlet versus Cloud, reductions in power usage by 50%, a nearly 10-fold reduction in delay time and an increased throughput of 10-fold were achieved using a single-device cloudlet with one mobile user [78].

Data from [37, 64, 78] were captured using free and research-grade applications using study groups in the UK, Eastern US and Central Europe sending and receiving messages to and from laboratory cloudlets in the UK, Central Europe and the US or to and from four commercial clouds. However, the cloudlet model has been shown to suffer performance degradation if too many intermediary steps ("hops"), where bandwidth was reduced were required between the mobile user and the cloudlet target in a simulation study [53]. With one or two cloudlet wireless "hops" used to transfer data, the cloudlet outperformed the cloud-based approach for application scenarios that included file editing and video streaming; with more intermediate steps, the cloud option performed better because of smaller request transfer delays.

On a much larger scale – considering an area of 931 km^2 with approximately one base station per 2 km^2 and servicing 180,000 users generating daily traffic in excess of 10 TB, a "cloudlet network" has been simulated and mathematically analysed in [28]. The results suggest, however, that the problem addressed in that study was more related to Mobile Edge Computing than to cloudlets because so many devices were included in the network. While these authors did not explicitly compare Mobile Edge Computing and Cloud performance parameters, the results demonstrated advantages in an optimized "cloudlet network" in terms of traffic volume and reduction of latency [28].

Extrapolating from results with various definitions of cloudlets, it can be concluded from the published work that interposing edge servers between users and the Cloud is a viable means of

Table 2.1: Selected Performance Metrics for Edge Computing/Cloud Comparisons

Source	Connection	Mean Latency (ms)	Mean Round Trip Time (ms)	Energy (J)	Time to completion (s)
[37]	Cloudlet EU	93			
	Cloudlet UK	59			
	Cloudlet US	186			
	Cloudlet IRL	91			
	Cloud US-East	161			
	Cloud US-West	228			
	Cloud Asia	337			
[78]	Cloudlet			103	226
	Cloud			2052	2223
[64]	Cloudlet		80	1.1	
	Cloud EU		420	5.2	
	Cloud US-East		260	3.1	
	Cloud US-West		420	5.2	
	Cloud Asia		800	9.4	
[72]	Cloudlet		80	0.3	
	Cloud EU		500	1	
	Cloud US-East		320	0.3	
	Cloud US-West		420	0.9	
	Cloud Asia		770	1.2	

improving QoE and overall system performance in terms of latency, bandwidth and consistency of service. Edge Computing is particularly applicable in the case of wireless networks where bandwidth is limited and latency could be high.

2.4 Comparison of Edge Computing Paradigms

The Edge Computing paradigms are clearly related but can be distinguished on the bases of applications, functionalities, technology and implementation (commercialization and business models). A "taxonomy" for Edge Computing is presented in (Tables 2.2-2.5). The majority of the criteria used were derived from tables, data and text in published sources [137, 153, 132, 12, 19, 101, 140]

The principal differentiators of the paradigms are the intended users of Edge Computing (Table 2.2). From the viewpoint of applications in Edge Computing, the application domains and the compatible devices are clearly differentiated (Table 2.3). Applications which have been promoted for Edge Computing include Smart Parking [13, 24], Smart Energy cars [106], drones

Table 2.2: Proposed Characteristics for Variants of Edge Computing

	Cloudlets	**Fog Computing**	**Mobile Edge Computing**	**Micro-Data Centre**
Rapid Response	No	Yes	Yes	Yes
Latency	Low	Low	Low	Low
Mobility	Yes	Yes	Yes	Yes
User	Local and Smart City	Security industry and network providers	Telecommunication and Software Providers	Hardware
Security Provider	None	Service Provider	Service Provider and Hosted	Service Provider
Service Level Agreement	None	Essential	Essential	Essential
Academic research input	High	Moderate	Moderate	Low

Table 2.3: Proposed Taxonomy for Variants of Edge Computing, Application-Driven View

	Cloudlets	**Fog Computing**	**Mobile Edge Computing**	**Micro-Data Centre**
Application Domain	Mobile	IoT and Mobile	Mobile	IIoT
Real-time interaction	Yes	Yes	Yes	Yes
5G	No	Yes	Yes	Linked to/with
Tactile Internet	Not Possible	Possible	Possible	Possible
Smart City	No	Yes	Yes	Yes
Working Environment	"Data centre in a box" at the business premises	Indoor and Outdoor	Indoor and Outdoor	Indoor and Outdoor
Service type	Mobile Device	Fixed Device	Mobile Device	Movable Device

in IIoT [23], Mobile Health and eHealth [22, 39, 56] and Augmented Reality [9, 169].

Functionality-Driven View assesses different functionalities for application scenarios in Edge Computing, all of these which can be incorporated into the overall aim of performance enhancement (Table 2.4). Examples include resource management [69, 109, 160], energy efficiency [48, 79, 134], data analysis [43, 82], data traffic caching and storage [2, 82], distributed data mining [170], smart sensors [55], network resilience [50] and – the central research area of this thesis – computation offloading [16, 67, 113, 119, 6].

From the Technology-Driven View, a major strategy of Edge Computing is to focus around virtualization and industrialised applications at the edge network (Table 2.5). Such an evolution to Edge Computing would be enabled by technologies such as Software Defined Networking [19, 77], Network Functions Virtualization [19, 139], Information Centric Networking [4, 1], and emerging simulation tools [62, 132, 63, 133]. These new technologies provide novel tools that increase flexibility in designing networks. Complementary technologies will enable programmability of control and network functions and eventual migration of these key constituents of the network to Cloud Computing.

Table 2.4: Proposed Taxonomy for Variants of Edge Computing, Functionality-Driven View

	Cloudlets	**Fog Computing**	**Mobile Edge Computing**	**Micro-Data Centre**
Location Awareness	Limited	Unlimited	Unlimited	Limited
Number of users	Few users at a time 25–50	Many Users at a time 100 –2000	Many Users at a time 100–1000	Many Users at a time 100–4000
Content Consumption	Fixed locale	Anywhere	Anywhere	Fixed but mobile
Mobile Management	Yes	No	Yes	No

Table 2.5: Proposed Taxonomy for Variants of Edge Computing, Technology-Driven View

	Cloudlets	**Fog Computing**	**Mobile Edge Computing**	**Micro-Data Centre**
Connectivity	WLAN and Wired	WLAN, Wired, Cellular	WLAN, Wired, Cellular	WLAN and Wired
Numbers of servers	One	Numerous	Numerous	Numerous
Client Hardware requirements	Local mobile device	Distributed/Hierachical	Distributed/Hierachical	Local-mobile

2.5 Multi-access Edge Computing and its Deployment

An important term used by the authors of [129] when described Multi-access Edge Computing (MEC) is “chimera”, i.e. an animal with head, body and tail from different biological species; MEC was described as being formed “from the convergence of several disparate trends” and these trends were micro-data centres, cloudlets, FC and Mobile Edge Computing. The change from Mobile Edge to Multi-access was presented as being to “better reflect non-cellular operators’ requirements” and broaden its relevance to the IoT. This explicitly incorporates mobile and fixed devices and – following the approach adopted in this Chapter – amalgamates all forms of Edge Computing and their different network architectures [28, 95, 135] under one heading.

The assumption is that MEC systems will evolve to offer a broad range of services to users [101, 12, 27]. This broadening of service options will bring inevitable challenges and there is an urgent requirement to develop experimental test beds for a wide variety of purposes, including deployment scenarios and economic modelling [15, 110] as well as the security of personal data [128], network flexibility and recovery from failures [135] and users’ QoE [142, 14, 57]. In addition, this process of development and deployment must content with very fragment markets of widely different sizes [122]. Industrial users are increasingly seeing potential benefits to productivity from linking IoT devices to MEC systems and it is this sector that may show the largest long-term benefit from technological changes in this area [127, 41, 118]. Nevertheless, people-oriented applications are very likely to be developed for devices such as smartphones [54].

The implementation of MEC systems for applications targeting users of MDs faces an

unavoidable problem: that of highly fluctuating user demand, which can be on an hour-by-hour basis. MEC servers will not only receive and download potentially vast amounts of data but will need to adapt to very different computational demands [103]. This could easily lead to an over-structured and oversupplied hardware infrastructure which is often underused but with high economic capital costs and operational costs [104]. Using city-wide data arrays from MDs, adding information about required computational and processing requirements for MEC servers to the amounts and types of data from MDs has been shown to yield valuable information on how to more efficiently configure MEC networks [103].

As noted by the authors of [129], MEC technologies are not part of 5G but MEC systems and networks can benefit greatly from wireless communication speeds of 100-250 Mbps (i.e. up to 10 times faster than 4G services). If 5G can give ultra-low latency to MEC networks, an viable architecture could redirect Cloud Computing requests from the IoT to an accessible MEC system using Software-Defined Networking [66].

MEC networks are considered to be particularly valuable for vehicular communications because of low latency and high bandwidth provision; sensors and other hardware can be deployed in road infrastructure [59]. This approach can be extended to include other forms of traffic on streets and roads (pedestrians and cyclists) using smartphones to access MEC networks [108]. Future autonomous operation of robots and vehicles could benefit from blockchains, distributed peer-to-peer networks in which all network participants have access to a central ledger and its unchangeable record of transactions; this would provide security in providing services and managing resources in MEC systems [120].

Inherent in the concept of MEC is a multi-path transmission of information and service requests; both wired (fixed) and wireless (mobile) access routes have been incorporated into fibre-wireless access networks, which have been tested in experimental set-ups and shown to have improved performance [99]. If a heterogeneous collection of IoT devices connect to a MEC network, an application resource allocation mechanism (to decide the computing resources to be allocated to each application on MEC servers to efficiently process tasks within delay requirements) and a task scheduler (determining the order in which tasks are processed) are necessary [7]. Ultimately, however, MEC network congestion will erode any benefits of using the system and an architecture has been proposed by which users of the network could access real-time information about the state of the network [146].

While the issue of high energy usage in Cloud Computing consolidated data centres is known to be problematic and the subject of much research [44], energy use in MEC networks is more difficult to assess because of the complexity of traffic between multiple servers in MEC networks and Cloud Computing storage [81]. A ground-breaking analysis of possible energy savings by using MEC rather than Cloud Computing concluded that energy savings increased as the numbers of MEC users increased and that the total Cloud plus Edge Computing energy consumption could be reduced by 50% by careful scheduling of tasks [93].

Non-Orthogonal-Multiple-Access is a multiple access technique planned to be used for 5G cellular wireless networks with massive connectivity and potential overloading of the network; power allocation, time slot scheduling and offloading task assignment can be jointly optimised to minimize energy consumption in a MEC system [94].

2.6 Computation Offloading in MCC and MEC

2.6.1 Modalities of Cloud and Edge Computing interacting with MDs

In principle, there are three different modalities which could relate MDs to superior computing power in Edge Computing networks and Cloud Computing data centres:

1. MD to Edge – a one-to-one communication between an individual MD and a base station in direct linkage to one or more MEC servers. This basic modality assumes a client-service provider relationship at the level of the MEC network only, in which an MD offloads computational tasks to a MEC network whose servers possess all the computational functionalities to complete the requested task and transmitted processed data back to the MD.

2. MD to Edge to Cloud – initially a one-to-one communication between MD and MEC network but which then hands on complex computational tasks to Cloud Computing providers for processing before returning data back to the MD. This has been described as an architectural flow from edge devices via edge nodes to Cloud data centres [98].

3. MD to Edge to Edge – initially a one-to-one communication between MD and MEC network but which then hands on either complex computational tasks to another linked MEC network with the requisite software or routine tasks which cannot be rapidly processed (within the terms of a Service Level Agreement between the user and a service provider) because of overloading of the first-contact MEC network.

Superimposed on these three different modalities is the storage of data from any of the three to Cloud data centres for data storage and Big Data analytics [98]. This additional layer in the overall relationships does not affect the mechanism of offloading as perceived by the user of the MD because parameters such as task completion time and local (MD) energy use would not be affected; issues such as data security, confidentiality and privacy are, however, highly pertinent to any involvement of Cloud data centres not clearly specified by the terms of a Service Level Agreement between the user and a service provider or with this form of involvement which may be subject to malicious attack on either Edge or the Cloud components in the relationship.

2.6.2 Cloud-dependent offloading

The MD to Edge to Cloud modality has no self-evident advantage over MCC; the comparative immaturity of MEC technologies may, however, pose operating challenges which could persuade IT enterprises to incorporate the Cloud as an initial service offering [83].

Computational offloading has been a topic explored in depth for MCC scenarios [47, 171]. Computational offloading is a means of transferring computing-intensive tasks over to Cloud resources to overcome technical limitations in mobile devices, in particular to improve battery life of the mobile devices and increase computational performance but also to reduce the total energy consumed [33, 90]. Direct testing under defined experimental conditions has confirmed the benefits (shorter processing times and reduced energy consumption) of computational offloading from, for example, smartphones [40, 87, 89].

A specialised form of Cloud-dependent offloading could be a Software as a Service offering; this is a mature area in which major commercial enterprises operate and can provide tailored, specific software applications [76]. The implementation of MEC networks has, as its central logic, supplanting dependency on Cloud data centres to support novel IT services for MDs [83].

2.6.3 Cloud-independent offloading

While the fine details of computational offloading in Edge Computing may not always precisely mirror those in MCC, the benefits are anticipated to be broadly similar [89, 114, 167].

For optimal offloading in MEC, both radio/wireless and computation resources must be considered for multiple users; a study with a single MEC server considered users being able offload its various proportions of task and being allocated only some of the total computation power available but with the aim of minimizing the time required for each user task [91]. Edge-to-Edge communication has been considered essential to provide resilience to MEC networks [135].

Functionally, the Edge can be divided into “near” and “far” components [83]. For fixed devices in Fog Computing, data processing is performed in the “near Edge” with a maximum latencies of a few milliseconds in, for example, industrial gateways. MDs would, correspondingly communicate with 5G base stations. The “far Edge” processes data within approximately 10 milliseconds and could be features of smart city applications. These times contrast greatly with Cloud data centres, where full processing might require more than hundreds of milliseconds.

2.6.4 Research questions in offloading to MEC networks

Assuming that appropriate computational resources are available in the “edge”, quantitative decisions on computational offloading must be seamless and automatically made if MDs are to function effectively in such an architecture using computation offloading algorithms to find effective and (for mobile devices) energy-efficient solutions [7].

These considerations were central to framing the Research Questions for this thesis (Chapter 1, Section 1.3.2). Understanding what types of application and computational tasks from MDs could benefit or benefit most from offloading (Research Question 1) implies a knowledge of what types of software would be available to process offloaded jobs. The original proposal for cloudlets [137] envisaged non-proprietary software only and this would greatly limit the options for users of MD seeking to offload (for example, word processing). Mobile Edge Computing offered servers or server clusters and MEC implicitly continued this scenario; however, the full portfolio of software accessible by MD users remains undefined. The assumption made in this thesis is that a MEC server (stand-alone or in a cluster much smaller than in a remote consolidated data centre for Cloud Computing) would host software useful for applications such as Augmented Reality or facial recognition to meet ad hoc demands from MD users without the necessity to route offloaded jobs to Cloud Computing centres. The shorter round-trip times of "local" MEC servers would be better suited to low- and ultra-low-latency tasks.

Unlike fixed devices [42], MDs forward distinct data sets (for example, multiple digital photographic image files) in a sporadic basis and both Mobile and Multi-access Edge Computing were designed for this type of demand rather than Fog Computing and its continuous stream of data for analysis (Research Question 1).

The provision of multiple base stations for MEC networks addresses the problem of congestion and queuing (Research Question 1) by increasing the number of first-access points for MD users seeking to offload tasks beyond those already available in the distant consolidated data centres [30, 34].

Some early Proof of Concept studies for Mobile Edge Computing implied a subscription basis for individual users of MDs (Section 2.2.4). This would accelerate service innovation and deployment but economic cost factors must be included in in-depth analyses of subscription-based offloading where hardware limitations might occur due to limited or delayed upgrades (Research Question 3).

For these reasons, offloading to a selected MEC network as a service provider without referral of jobs to either Cloud data centres or linked MEC networks was selected as the basis for constructing and testing mathematical models (Research Question 1) and heuristic algorithms (Research Question 2-3) in a defined environment to explore the flexibility and capability of the offloading process.

2.6.5 Research trends in offloading to MEC systems

Selected published reports of computation offloading in MCC and MEC are given a detailed presentation in later Chapters of this thesis where key technical aspects are discussed:

1. Quantitative parameters used for numerical simulation tests (Chapter 3)

2. Features of mathematical modelling of the offloading process (Chapters 3 and 4)

3. Task scheduling (Chapter 4 and 5)

4. Multi-factorial optimisation (Chapter 5).

Recent trends in MEC R&D relevant to Edge-to-Device communication will now be addressed.

2.6.5.1 Partitioning of tasks for offloading

While an entire task can be offloaded to shorten computation time or conserve energy in MDs, a different approach is to partition resource-intensive of tasks so that only part of an individual task needs to be offloaded; this can be described as task/component offloading [98]. This approach was initially investigated in MCC but much of the work was focused on the development of novel applications for the MCC environment rather than optimising client-initiated offloading [46]. Partitioning of sub-tasks between the MD and the Cloud was also shown to reduce MD energy usage [74, 159].

In Mobile/Multi-access Edge Computing, task partitioning has been investigated for Virtual Reality and Augmented Reality applications [163]. The detailed partitioning mechanism adopted, however, is highly dependent on the application considered and conclusions may not be portable to other and even similar applications [98].

2.6.5.2 Caching mechanisms and strategies in MEC

In the broadest perspective, Edge-to-Cloud and Cloud-to-Edge transfer of software and data greatly increases the capacity of MEC networks to store data and access multiple (including specialised) applications [98]. Network architecture can, however, be a problematical in such communication links because of backhaul and latency delays; new system architectures have been explored to which bring both functions and contents closer to Edge components [158].

One example of this caching via intermediate servers is that of video content; distributed caching mechanism have been proposed for video files at base stations in Edge networks to increase video capacity and enhance user experience [3]. This issue has also been explored for Augmented Reality [51] and data-intensive applications [98]. Content popularity could impose major ”spikes” in demand and proactive caching has been proposed based on high prediction accuracy using neural network approaches [8], a collaborative effort between Edge servers and MD users using Federated Learning [168] and learning-based optimisation [70]. In contrast, reactive strategies have been proposed but these either seek to harmonise Cloud-to-Edge and Edge-to-Device transfer rates [85] or use Deep Learning approaches [148].

A similar problem is that of storing software programs in Edge servers when local storage capacity is limited; this was analysed in detail for the special case of the user of a MD seeking to offload customised code to perform offloaded computational tasks where Integer Linear Programming could identify optimum solutions given constraints in software and hardware [11].

In these and similar scenarios, policies for MEC networks must be developed for the management of caching insertion and expulsion [131].

2.6.5.3 Dynamic pricing for MEC access

In a model with a single MEC server accessed by multiple MDs, algorithms have been proposed for dynamic pricing strategies to maximise revenues for the service provider [29, 31].

For all Edge Computing paradigms (including MEC), auction-based access is always possible but a uniform-pricing mechanism may be subjectively more attractive to MD users [11]. Subscribers to a MEC service may prefer fixed-price schemes (per MB uploaded or downloaded) and ETSI Proof-of-Concept studies emphasised guaranteed Service Level Agreements [6] "Guest" users of MEC services may, of course, be open to dynamic or volatile costs for access.

2.7 Simulation approaches to MEC systems

A dedicated simulation package for Edge Computing has been presented [145]. EdgeCloudSim included a wireless local area network (WLAN) and a Wide Area Network (WAN) communication model, incorporated mobile nodes and mobility support and included a virtual machine (VM) utilization model and could, therefore, be considered as a suitable simulation software for MEC. In the published account of EdgeCloudSim, the WAN link functioned to send information to and from Cloud facilities and this generated two architectures: a 1-tier Edge with no link to the Cloud and a 2-tier Edge/Cloud. The study only presented results for Edge delays, Edge/Cloud delays and offloaded task failures. The Edge delays were much shorter than Edge/Cloud delays and both types of delay increased as the number of mobile devices increased from 50 to 250; the numbers of failures also increased as the total mobile devices increased even with a maximum CPU workload of 10% in the Edge; increasing the WAN speed by ten-fold did not improve the failure rates.

2.8 MEC Proof of Concept Studies Relevant to this Thesis

As discussed in Section 2.2.4, Proof of Concept studies in Mobile Edge Computing and (later) in Multi-access Edge Computing, had specific aims and ambitions. By 2020, 13 such studies were underway or had been completed [7]. None of these targeted computation offloading from MDs and only one explicitly targeted MDs (PoC 13, "MEC infotainment for smart roads and city hot spots"). PoC 13 focused on 4G/5G infotainment services for pedestrians and car drivers/passengers in smart roads and city hot spots.

[6] :https://www.etsi.org/newsroom/news/1037-2015-12-etsi-mobile-edge-computing-isg-announces-first-proofs-

[7] https://mecwiki.etsi.org/index.php?title=Ongoing_PoCs

The PoCs can be viewed as demonstrating an "Edge-to-Device" formalism designed for telecommunications and related industries with subscribers accessing premium services. These PoCs included:

1. PoC 1, Video User Experience Optimization via MEC
2. PoC 2, Edge Video Orchestration and Video Clip Replay via MEC
3. PoC 3, Radio aware video optimization in a fully virtualized network
4. PoC 7, Multi-Service MEC Platform for Advanced Service Delivery
5. PoC 13, MEC infotainment for smart roads and city hot spots. A second set of PoCs included areas of Fog Computing in which fixed devices communicated in a Device-to-Edge manner, for example:
6. PoC 6, Healthcare – Dynamic Hospital User, IoT and Alert Status management
7. PoC 8, Video Analytics
8. PoC 9, MEC platform to enable low-latency Industrial IoT
9. PoC 11, Communication Traffic Management for V2X.

PoCs 4, 5, 10 and 12 were aimed at developing or assisting enterprise-level operations.

None of the PoCs, therefore, involved studies which overlapped with the work to be presented in this Thesis. Nevertheless. the expanding literature on academic research into offloading in MEC provides evidence of the continued interest in this aspect of the relationship between MDs and Edge Computing [98, 80].

For the technical chapters of this Thesis, the implicit assumption is made that the software shared between MDs and MEC servers is at the level of and covers aspects of Video Analytics, for example facial recognition, which would be of broad interest to users of MDs. This level of Device-Edge/Edge-Device interaction has been discussed previously [166]. A potential associated service could be user-initiated parking space searching in urban environments [60].

2.9 Conclusions Relevant to Offloading to Edge Computing Networks

As discussed in Chapter 1, MEC is an evolving technology and its implementation is fragmentary. The Proof-of-Concept studies referenced in this Chapter acted as introductions to MEC concepts for telecommunications and other providers and to explore possible service offerings.

Based on this literature review, conclusions could be drawn to better focus the work presented in this Thesis:

1. To analyse the Edge-to-Device component in depth and to include as many hardware and software features potentially involved in the offloading process, only full offloading of a task would be considered because task partitioning is too dependent on the exact task and results would not be fully portable.

2. The software capabilities for task computation in both MD and MEC server would be matched because offloading program code or Cloud-to-Edge transfer of program code would potentially greatly extend total task completion time and reduce any advantage of offloading.

3. No data caching from the offloading process – other than in very transient events in data processing and return the MD – is required; this is equivalent to each offloaded task being unique and not repeatable for offloading. This also avoids any issues of data security of user privacy.

4. The approach for assessing offloading optimisation would take the novel approach of considering offloading multiple files as a scheduling exercise which could use linear programming or heuristic algorithms to provide solutions.

5. Optimal scheduling would be twofold: for an individual MD and for multiple MDs without any functional linkages between them.

6. Resource allocation and management in MEC networks would be approached from the standpoint of strategies to avoid overloading of a “first contact” MEC server/base station combination - as quantified via CPU workloads and access delays that result from excessive numbers of MD users simultaneously accessing a MEC network.

7. The application paradigm would be a facial recognition software for digital images of small to moderate size (for example, 4 MB). This would generate, for an individual MD, a task schedule which would later be repeated sequentially but for a different set of files and, for multiple MDs, a schedule which would be terminated for all MDs at closely similar times and subsequently lead to a ”n+1” set of tasks to be analysed to identify a scheduling optimum. In this second scenario, the resource allocator in the MEC network would be continuously seeking optimum schedules without resorting to the “first come, first served” option to be run ad hoc once the initial optimum solution had terminated [8].

8. Access to MEC services is assumed to be on a subscription basis with known costs, although service providers may offer premium-rate services in addition to the terms of Service Level Agreements with subscribers.

[8] https://www.etsi.org/newsroom/news/1037-2015-12-etsi-mobile-edge-computing-isg-announces-first-proofs-c

CHAPTER 3

KEY FACTORS DETERMINING THE ADVANTAGE OF COMPUTATION OFFLOADING IN MEC

3.1 Introduction

Although computation offloading in both Mobile Cloud Computing (MCC) and Multi-Access Edge Computing (MEC) has, as was discussed in Chapter 2, been researched and different mechanisms for offloading from mobile devices (MDs) to either the Cloud or to MEC servers have been proposed, the central question has been rarely approached: which computational tasks on MDs benefit from computation offloading?

In part, this research topic in computation offloading has been relatively slow to develop because, as will be argued in this Chapter, the question requires detailed numerical analysis for its investigation and this inhibits the testing of generic approaches. In addition, any of the key parameters are not available for the general hardware and software involved in computation offloading. Multiple parameters require quantifying for any mathematical model defining task completion times to be rigorously tested; likely relevant factors include processor speeds, file sizes, communication speeds, the computational complexity of tasks and the ease of access (or not, in the case of congested networks and overloaded servers) to servers for task offloading. Finally, in order to quantify the energy consumption and consequent savings that could be made by reducing completion time, power rating of each model device needs to be defined before solving the problem.

Key Research Question: What are the important factors in quantitative models that affect computation offloading for reducing task completion times and/or energy usage by MDs?

This Chapter makes two major contributions:

- An in-depth analysis of the effects of multiple factors on the advantages of offloading

tasks to a MEC server in terms of reduced task completion time and reduced energy use by a MD.

- The development of a new model to incorporate CPU workloads into the savings in time and MDenergy use possible by offloading in a MEC network.

3.1.1 Publications

The work presented in this chapter has led to the publication of the following two articles.

- R. Singh, S. Armour, A. Khan, M. Sooriyabandara and G. Oikonomou, "The Advantage of Computation Offloading in Multi-Access Edge Computing", Fourth IEEE International Conference on Fog and Mobile Edge Computing (FMEC), Rome, Italy, 2019.

- R. Singh, S. Armour, A. Khan, M. Sooriyabandara and G. Oikonomou, "Identification of the Key Parameters for Computational Offloading in Multi-Access Edge Computing", IEEE Cloud Summit 2020.

3.2 Relevant Studies in Computation Offloading: What to Offload?

In [87], the authors developed Mobile Augmentation Cloud Services (MACS) middle ware to offload Android applications from a MD in MCC and tested the offloading process empirically using designated hardware and two applications: a mathematical puzzle "N-Queens Problem" and facial detection and recognition from video files of increasing length and increasing file size. The conclusions from their study was that not every application is suitable for offloading, however, offloading provides advantages for applications which require high computational processing power.

In [105], a broader range of scientific software applications was considered (including computational programs for computational chemistry, astronomy and hydrology) because the authors were able to access critical data for these, specifically the ratios of bits to instructions for the applications. With this parameter, file sizes and processor speeds could be directly converted to task completion times. The conclusions from quantitative testing in MCC) scenarios were that the critical parameters to ensure offloading would result in smaller task completion times were:

- the on-board and server-side processor speeds;

- the minimum ("bottleneck") data transfer link speed (either 1 kbps or 1 Mbps);

- the computational complexity of the application.

Crucial to the analysis presented in [105] was the bits per instruction parameter. Even with a data file size of 1 MB, the simplest of the 9 programs considered in [105] required (7.7×10^{11}) instructions per bit. Other authors [84] used (2×10^{8}) instructions for per "task" but used much slower processor speeds than in [105]: a smartphone processor approximately 100-times slower than the faster on-board processor in [105] and a MCC server processor approximately 1000-times slower than the fastest server-side processor in [105].

The choice of scientific programs in [105] was made for practical reasons, i.e. using computational workload data from an advanced computing centre. The numerical simulations reported in [84] used much smaller numbers of instructions for image processing tools but also in applications for mathematical and scientific purposes. The pragmatic choice of the least computationally complex application from the data supplied by [105] was made to perform detailed quantitative analyses of the offloading process assuming that an application was pre-existing on a server in the MEC network which could run at far higher processing speeds than on a MD.

Consequently, no programming code was to be included as code to be transmitted from the MD; the necessity to install an application of a MCC) server would not only increase the amount of data transmitted but also require an installation process of unknown duration.

The processor speed for any processor can be calculated but the required information is seldom available. The calculation for instructions per second given in [68] is: processor speed (IPS) = clock rate (MHz or GHz) /cycles per instruction where the clock rate is readily available but cycle per instruction (CPI) requires considerable knowledge of processor functions, in particular pipeline and cache CPI) values. Three examples discussed in [68] (Page 248) had CPI values in the range 1.2-2.4 cycles per instruction; a website source gives 2.5 cycles per instruction [1].

The authors of [75] used this approach with a small data file (420 KB) which required (1×10^{9}) cycles in processor of clock rate (4×10^{9}) Hz (cycles per s) but did not provide any justification or calculation for the number of cycles required. For energy use, an important linking factor was k_{m}, a "coefficient depending on the chip architecture", which was quoted to be (5×10^{-27}) [75]. The basis for this value was another source [117], which in turn referred to a third source [156]. The third source refers to the parameter as "the effective coefficient that depends on the chip architecture" but gives the value as (10^{-28}) and does not provide any calculation basis or reference to manufacturers' data.

All of the above highlights the significant problems faced when selecting quantitative parameters for use in numerical simulations and experimentation, standard procedures when testing out novel processes and system models in computation offloading [74].

Neither [87] nor [105] considered congested networks and their analyses used very different link speeds in MCC. However, both studies emphasized the importance of being able to perform detailed calculations using numbers of instructions in computational operations so as

[1] CPU Performance Evaluation: Cycle per instruction (CPI)):
`http://meseec.ce.rit.edu/eecc550-winter2011/550-12-6-2011.pdf`

to accurately identify times required for task completion either on a MD or by offloading. The calculations required for offloading were far more complex than local processing on the MD and in particular required knowledge of the "bottleneck" link speed for communication and data transfer between an MD and distant MCC servers.

The remainder of this Chapter will, firstly, use the basic mathematical model proposed by [105] and expand it to incorporate factors such as network congestion and link delays ("latency"); the analysis is then used as a platform with which to determine energy savings possible to MDs when both MDs and MEC servers form a heterogeneous network with varying processing speeds (Sections 3.3-3.4). The Chapter will then present a more generalized mathematical model to incorporate factors such as Central Processing Unit (CPU) workload to analyse network congestion in greater depth (Sections 3.5 - 3.7). Finally, the implications of the results for users and providers of offloading to MEC networks are discussed.

3.3 Theoretical analysis and quantitative models for offloading to a MEC network

The overall scheme of the data transfers from MDs to a MEC network are shown in Figure. 3.1. The users of MDs are assumed to include smartphones, tablet computers and laptops. Of these MDs, smartphones suffer severely from battery lifetime issues as well as limited on-board processing power. Tablet and laptop computers have much longer working battery lifetimes (when unplugged) but their users may still welcome access to the much greater computing power of MCC or MEC networks. In heterogeneous populations of MDs in a MEC network, processing capacities and remaining battery lifetimes are likely to show high variabilities.

3.3.1 Improved Execution Speed

In an MCC model [105] for offloading to result in a faster execution time for a task the following inequality was required:

$$\Gamma\left(\frac{1}{\alpha}-\frac{1}{\beta}\right)>\frac{X^{\mathrm{MD}}}{C^{\mathrm{MD}}} \tag{3.1}$$

where Γ is the link speed (bps), α is the execution rate of the local computing device in instructions per second (IPS), β is the execution rate of the server in instructions per second (IPS), X^{MD} is the data (bits) transferred over the MEC network and C^{MD} is the size of the computational job (instructions) and the units for both sides of the equation are bits per instruction. If the left-hand side exceeded the right-hand side, computation offloading was favorable, i.e. the achieved task execution was faster by offloading to the external MEC server.

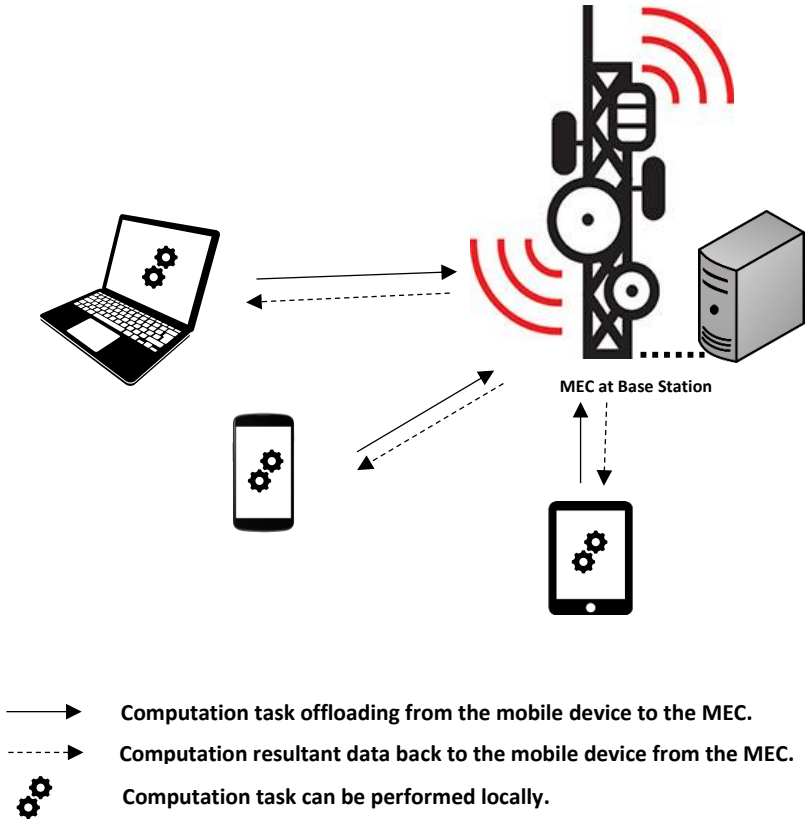


Figure 3.1: An illustrative diagram of a MEC system with multiple users. The arrows represents channels via which the mobile users can access the computational power of a MEC.

This inequality was a contraction of a more general inequality:

$$\Gamma\left(\frac{1}{\alpha}-\frac{1}{\beta}-\frac{H}{C^{\mathrm{MD}}}\right)>\frac{X^{\mathrm{MD}}}{C^{\mathrm{MD}}} \tag{3.2}$$

where $\frac{H}{C^{\mathrm{MD}}}$ was the time per instruction that degrades the performance of the offloading system as a result of communication problems if $H>0$. The authors of the study [105] equated H to zero and only considered an uncongested network but we have considered multi-user congestion in our analysis. The $\frac{X^{\mathrm{MD}}}{C^{\mathrm{MD}}}$ term in equations 3.1 and 3.2 refers to bits per instruction values computed for the 9 applications listed in Table 3.2, data from [105]. The $\frac{X^{\mathrm{MD}}}{C^{\mathrm{MD}}}$ term is inversely proportional to the computational complexity of the application.

[105] only considered two link speeds, 1 kbps and 1 Mbps for offloading. Here, we extend the range of link speeds up to 64 Mbps and include two types of communication delay: link access delays independent of the number of users and a user number-dependent reduction of the $\frac{1}{\beta}$ term in equation (3.1).

Table 3.1: On-board and server-side processor speeds used in the illustrative examples throughout the thesis [105].

Processor	**IPS**
MSP430	1.6×10^7
Apple A9	3.6×10^9
Intel Celeron	6.40×10^9
Xeon processor	1.40×10^{11}

Table 3.2: Computational requirements of 9 selected applications. The applications are taken from [105].

	Application Name	**Bits/Instruction**
1	`siesta`	5.29×10^{-5}
2	`charmm`	7.34×10^{-5}
3	`mdrun_mpi`	1.08×10^{-4}
4	`nwchem`	1.80×10^{-4}
5	`vasp_ncl`	2.86×10^{-4}
6	`cocmomc`	4.84×10^{-4}
7	`lmp_stampede`	9.53×10^{-4}
8	`namd2`	1.01×10^{-3}
9	`fvcom`	2.27×10^{-3}

3.3.2 Reduced Mobile Device Energy Usage

[90] presented an outline mathematical model for computing energy saving by offloading which relied on the inequality that the energy used by the mobile device was more than the energy of that mobile device in offloading; this can be written as:

$$\frac{C^{\mathrm{MD}} \times P^{\mathrm{MD}}}{\alpha} > \frac{C^{\mathrm{MD}} \times P^{\mathrm{idle}}}{\beta} + \frac{X^{\mathrm{MD}} \times (P^{\mathrm{send}} + P^{\mathrm{rec}})}{\Gamma} \tag{3.3}$$

where the power terms for the MD are taken from [90], β represents the processing speed of the server and α the processing speed of the MD ; other symbols have the definitions used for Equations (3.1) and (3.2). X^{MD} is considered to be the dominant contributor to any data exchange between the MD and MEC server, i.e. relatively little data is transmitted back to the MD. The authors of [90] quoted three values for power required by a MD: P^{MD} is the power required to compute a job on the MD (0.9 W), P^{idle} is the power required of the MD while idling (0.3 W) and P^{send} and P^{rec} are the powers required while transmitting and receiving information and were assumed to be equal (1.3 W). These values have been used in the calculations of the energy used by a mobile device computing locally or offloading to a MEC server.

If the left-hand side of equation (3.8) exceeds the right-hand side, the energy use by the

mobile device will be reduced by offloading to the MEC server. This will be advantageous to the user of the mobile device in, for example, extending the battery life.

Other studies have used different power ratings; the authors of [84] stated that the lack of detailed power rating information for commercially sourced processors greatly limited the ability of researchers to fully analyse power rating by MDs and used a P^{MD} value of (0.735 W) and a P^{idle} value of (0.096 W) but used an energy value per MB instead of P^{send} and P^{rec}. The numerical analyses in [87] used the following values: P^{MD} (0.4 W), P^{idle} (0.05 W) and P^{send} and P^{rec} (0.75 W) power rating of P^{MD} (0.6 W) (>75% CPU usage), P^{idle} 0.06-0.98 (W) in Wi-Fi mode and P^{send} and P^{rec} (1.4 W) at 1 Mbps in 3G were used in [116].

The authors of [100] state that transmission power is variable but do not provide any source for this statement or give any example of what conditions affect transmission power. However, any consistent set of power ratings will enable numerical simulations to provide a reliable estimate of percentage energy savings possible by offloading for a MD if data transfer and processing times and power ratings are combined in the calculations.

3.4 Effects of Various Parameters on Task Completion Time

Using the mathematical model described in Section 3.3. of this Chapter, a number of scenarios were analysed to explore the impact of changing parameter values on the offloading process and whether or not task completion times was reduced by offloading compared with local computation on the MD.

The motivation was to include the effects of multiple factors - on-board and MEC processor speeds, computational task complexity, a wide range of communication speeds, link access delay and the number of mobile users - on the success of offloading using task completion time as the sole criterion.

In addition, quantifying any reduction in energy use by MDs made possible by offloading to MEC servers was analysed with different processor speed combinations and varying communications link speeds.

3.4.1 Improved Execution Speed

Figure 3.2 shows the slower on-board processor (MSP430) with the slowest of the two server-side processors (Celeron, with a server-side:on-board processor speed ratio of 402:1). The calculated bits per instruction values were favoured offloading for shorter task completion times for all 9 applications at a link speed of 64 kbps or more. With even a low link speed of 1 kbps, the more computationally complex application siesta was offloaded.

In contrast, Figure 3.3 shows that the faster (A9) on-board processor required much faster link speeds to justify offloading (in order to achieve lower task completion time): 33 Mbps with the slower (Celeron), with a server-side processor speed ratio of 1.7:1) and 16.4 Mbps

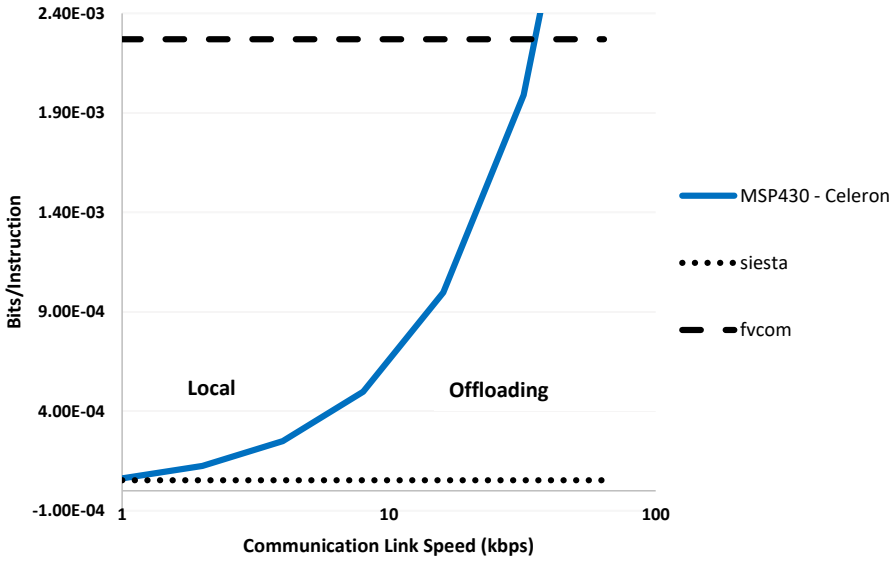


Figure 3.2: Effect of link speed on the offloading threshold for shorter task completion time: MSP430 on-board processor offloading to the Celeron server-side processor; siesta and fvcom are the most computationally complex applications and the least, respectively.

for the faster (Xeon) server-side processor speed ratio of 39:1). In general, the higher the bits per instruction value of an application, the higher was the minimum link speed required for a shorter task completion time to be possible by offloading. Table 3.3 enumerates the required link speed for all 9 applications in Table 3.2 with three combinations of mobile and server processors presented in Section 3.3.1.

Mobile devices with low-speed processors would, therefore, find offloading advantageous for shorter task completion time even when accessing wireless personal area networks with limited ranges and low link speeds 250 kbps), i.e. those specified in IEEE 802.15.4. Devices with faster on-board processors would benefit by offloading computations to MEC networks with link speeds comparable to current 4G [2] and WiFi networks [92].

3.4.2 Offloading from Mobile Device with Different Processor Speeds

A necessary corollary of the results presented with different combinations of processors with varying speeds is that, as the proportion of faster on-board processor mobile users in the user population increases, the success of offloading for faster task completion at a constant link speed decreases; this is because, with higher on-board processor speeds, the left-hand term in Equation 3.1 decreases.

If the user population in range of a MEC base station and server is composed of equal numbers of devices with on-board processor speeds covered by the range in Section 3.4.1,

[2] European Telecommunications Standards Institute (ETSI): `https://www.etsi.org/technologies/mobile/4g`.

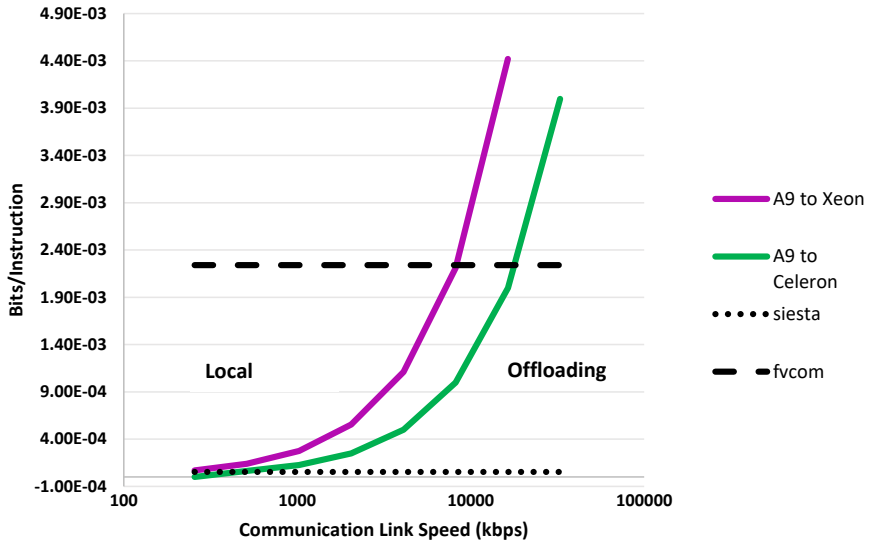


Figure 3.3: Effect of link speed on the offloading threshold for shorter task completion time: A9 on-board processor offloading to the Celeron or Xeon server-side processor; siesta and fvcom are the most computationally complex applications and the least, respectively.

Table 3.3: Minimum Link speed for offloading applications with different processor combinations for shorter task completion time

Application	**MSP430 to Celeron** (kbps)	**A9 to Celeron** (kbps)	**A9 to Xeon** (kbps)
`siesta`	0.8	433.6	195.9
`charmm`	1.2	601.6	271.9
`mdrun_mpi`	1.7	885.2	400.0
`nwchem`	2.9	1475.4	666.7
`vasp_ncl`	4.6	2344.3	1059.3
`cocmomc`	7.8	3967.2	1792.6
`lmp_stampede`	15.3	7811.5	3529.6
`namd2`	16.2	8278.7	3740.7
`fvcom`	36.4	18606.6	8407.4

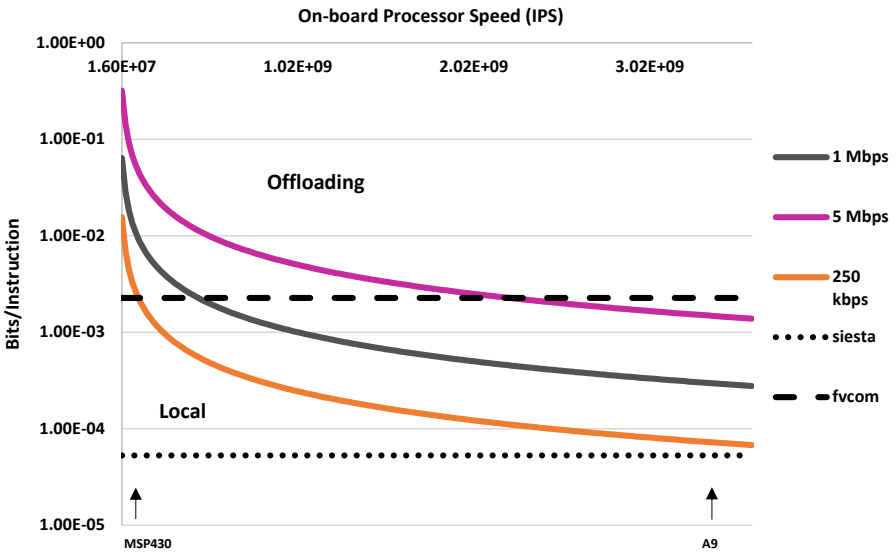


Figure 3.4: Effect of on-board processor speed on computed bits per instruction values required for offloading to a MEC server-side processor (Xeon) at different link speeds: e.g. siesta app is always offloaded.

approximately 2% found offloading for shorter task completion time is possible at a link speed of 250 kbps, 12% found offloading for shorter task completion time advantageous at a link speed of 1 Mbps whereas 64% benefited at a link speed of 5 Mbps, as shown in Figure 3.4. This analysis assumes a uniform distribution of processor speeds in mobile devices in the user population attempting to offload the application with the greatest computation complexity (the lowest $\frac{X^{\mathrm{MD}}}{C^{\mathrm{MD}}}$ value, see Equation 3.1 and Table 3.2) to a MEC server with the Xeon processor.

When applications with lower bits per instruction were considered, the percentage of mobile devices successfully offloading for shorter task completion time increased at a fixed link speed. If the link speed exceeded 8.4 Mbps (Table 3.3), all the mobile devices offloaded all the applications in (Table 3.2) for a shorter task completion time. Mobile devices with faster on-board processors than those considered would require faster link speeds when offloading to the Xeon server-side processor.

3.4.3 Link Access Delays

A positive $\frac{H}{C^{\mathrm{MD}}}$ term in Equation (3.2) adds a link access delay to the communication link between the mobile device and the MEC server and reduces the left-hand side of the Equation until eventually the inequality shown in Equation (3.2) fails and offloading does not result in short task completion times. Table 3.4 presents maximum computed link access delays for the 9 applications with three combinations of on-board and MEC server-side processors. With the MSP430 on-board processor in combination with any of the three server-side processors, offloading required progressively higher bandwidths until, at a link access delay factor exceeding

Table 3.4: Maximum Link access delay for offloading applications with different processor combinations for shorter completion time at 20 Mbps

Application	**MSP430 to Celeron** (ms)	**A9 to Celeron** (ms)	**A9 to Xeon** (ms)
siesta	62.3	0.12	0.27
charmm	62.3	0.12	0.27
mdrun_mpi	62.3	0.12	0.26
nwchem	62.3	0.11	0.26
vasp_ncl	62.3	0.11	0.26
cocmomc	63.3	0.10	0.25
lmp_stampede	62.3	0.07	0.22
namd2	62.2	0.07	0.22
fvcom	62.2	0.01	0.16

Table 3.5: Maximum number of mobile devices for offloading to shorten completion time with different processor combinations

Application	**MSP430 to Celeron** (250 kbps)	**A9 to Xeon** (20 Mbps)
siesta	400	36
charmm	399	36
mdrun_mpi	398	36
nwchem	396	36
vasp_ncl	394	35
cocmomc	388	34
lmp_stampede	376	30
namd2	375	30
fvcom	342	21

62.3 ms per 10^6 instructions at a link speed of 20 Mbps, offloading failed entirely to result in a short task completion time at any link speed, as shown in Table 3.4. The combination of the faster (A9) on board processor with the Celeron server-side processor did not tolerate link access delays factors greater than 0.12 ms per for 10^6 instructions, as shown in Table 3.4, while the A9/Xeon combination failed to offload at any link speed when link access delays factor exceed 0.27 ms per 10^6 instructions. Faster on-board processors therefore required a less interrupted and more seamless communication link in order to make offloading beneficial for shorter task completion times.

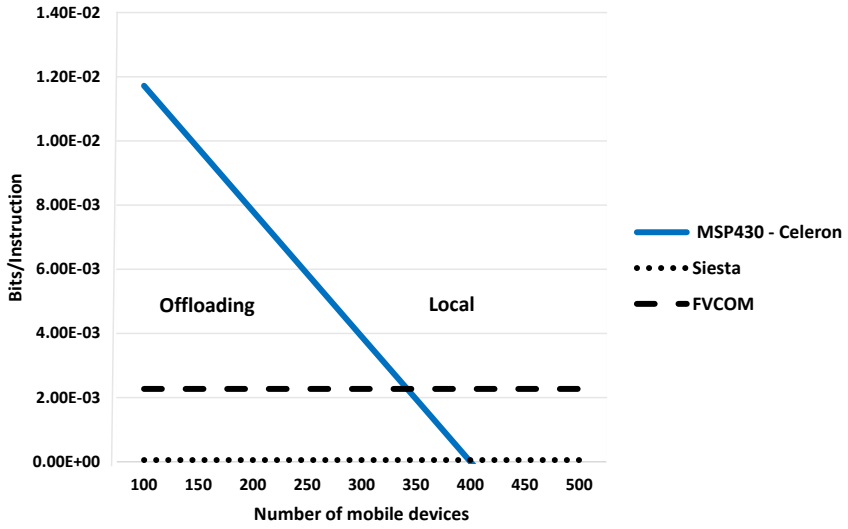


Figure 3.5: Effect of number of mobile users on the offloading for shorter task completion time: MSP430 on-board processor offloading to the Celeron server-side processor;siesta and fvcom are the most and least computationally complex applications, respectively (Table 3.2).

3.4.4 Offloading and Network Congestion

Ideally, any mobile device would have unimpeded access to the MEC server for offloading. When large numbers of users attempt to access the same MEC server, to avoid network overload, queuing and scheduling strategies have been proposed [102]. In the extreme case, an overloaded MEC server might also be able to share computational jobs with other servers [135].

To include an analysis of the effects of the number of mobile devices attempting to connect simultaneously to a MEC server, Equation (3.2) was modified as:

$$\Gamma\left(\frac{1}{\alpha}-\frac{D}{\beta}\right)>\frac{X^{\mathrm{MD}}}{C^{\mathrm{MD}}} \tag{3.4}$$

where D represents the number of users; this introduces a reduction in communication link speed depend on the number of users. In effect, overloading the MEC servers reduced the ratio of the server:mobile processor speeds. For the slower MSP430 processor with the slower Celeron server processor, a link speed of 250 kbps was sufficient to offloaded most of the applications for up to 400 users, as shown in Table 3.5. With the faster A9 processor with the faster Xeon server processor, even a link speed of 20 Mbps only offloaded much smaller numbers of mobile users.

Figure 3.5 shows that the most computationally complex application siesta was always offloaded for faster completion time from an on-board MSP430 processor to a MEC server (Celeron) until the number of mobile users exceeded 400 while the least computationally complex application (fvcom) was preferentially computed locally when the user number exceeded 342. Figure 3.6 shows that the most computationally complex application siesta was always offloaded for faster completion time from an on-board A9 processor to a MEC server

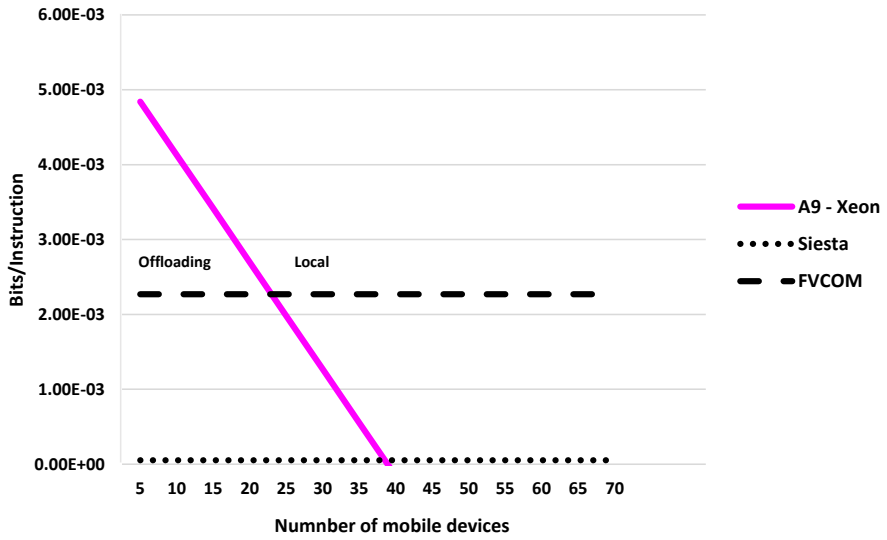


Figure 3.6: Effect of number of mobile users on the offloading for shorter task completion time: A9 on-board processor offloading to the Xeon server-side processor; siesta and fvcom are the most and least computationally complex applications, respectively (Table 3.2).

(Xeon) until the number of mobile users exceeds 36 while the least computationally complex application (fvcom) was preferentially computed locally when the user number exceeded 21. In general, the less computationally applications tolerated smaller maximum user numbers because they required higher differentials in the relative speeds of the server and on-board processors to achieve shorter task completion times (Table 3.5).

3.4.5 Energy Saving by Mobile Devices

Using Equation (3.8), combining the slower (MSP430) on-board processor with any of the three server-side processors resulted in major energy savings for the mobile device (up to 99%) at low link speeds (100-200 kbps) but the faster (A9) on-board processor required much faster link speeds for maximum energy savings (Figure 3.7). Even with a relatively low link speed of 1 Mbps, an energy saving of 80% was possible with the A9/Xeon combination. Calculations showed that combining the A9 processor with either of the two faster server-side processors could give energy savings for the mobile user exceeding 90% at high link speeds (50-100 Mbps). The metric used here for energy savings by the MD was the difference between the energy used for local processing minus the energy used by the MD in in transmitting information and in idling while the server processed the transmitted data as a percentage of the energy used for local processing.

Major savings were found to be possible with applications across the entire spectrum of complexity (Figure 3.8). With the lowest-complexity application, however, much faster link speeds were necessary: the lowest-complexity application required link speeds of 2 Mbps for energy savings while the highest-complexity application could show energy savings with link

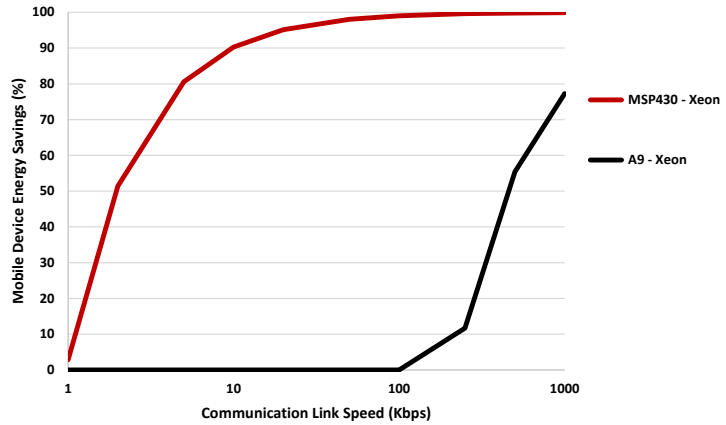


Figure 3.7: Effect of link speed on energy savings by mobile devices with slow and fast on-board processors offloading to the MEC server (Xeon).

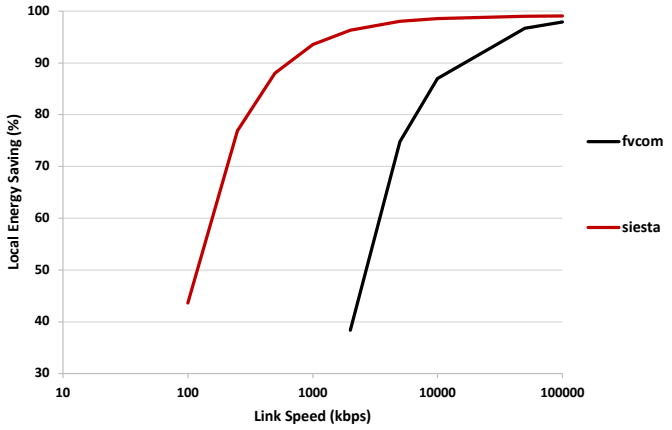


Figure 3.8: Local energy savings when an A9 on-board processor offloads highest-complexity (siesta) and lowest-complexity (fvcom) to a server-side Xeon processor.

speeds as low as 0.1 Mbps.

3.4.6 Summary of Key parameters affecting Offloading in a Heterogeneous MEC Network

The contributions made by the work described in Sections 3.4.1-3.4.5. Illustrate how a heterogeneous population of mobile devices with different MEC server-side processors can make decisions affecting offloading, specifically:

- Effects of different MD on-board processor speeds on the achieving successful offloading for shorter task completion time.
- Effects of different MEC server processor speeds on the achieving successful offloading for shorter task completion time.

- Effects of widely differing data transmission speeds to the MEC servers on the achieving successful offloading for shorter task completion time.

- Modelling how link access delay and reduced link speed caused by network congestion caused by widely differing numbers of users reduces the success of offloading in achieving faster task completion time.

- Energy savings achieved by mobile devices with various combinations of on-board and server-side processor speeds with different data transmission speeds to MEC networks.

The detailed numerical simulation data for how offloading can be beneficial in a MEC network with varying quantitative mobile user demand, heterogeneity in mobile device on-board and MEC processor speeds, computational task complexity, communication speeds, link access delays and mobile device user numbers.

More computationally complex applications are offloaded preferentially (especially with the higher server:on-board processor speed ratios) while low link speeds and any delays caused by network delays or excessive user numbers degrade any advantages in reduced task completion times offered by offloading. Additionally, significant savings in energy usage by mobile devices are guaranteed except at very low link speeds.

3.5 Offloading Model to Reduce Task Completion Time and Local Energy

As discussed in Chapter 2, a major perceived limitation of MCC) is that of link delays in communication that is required between MD and physically remote data centres. This has been an argument used in favour of Edge Computing but the potential numbers of users of MDs introduces the complication of localised overloading of MEC servers [135].

The authors of [135] proposed two mechanisms to alleviate problems to users attempting to offload caused by overloaded MEC servers: either a MEC server shares its task burden with nearby MEC servers or MDs can themselves act as relay nodes to connect the mobile users originally connected to a non-responsive MEC server to another MEC server not in direct contact with the non-responsive server.

A more direct analysis of server overload, however, can use server CPU workload as the primary determinant. In the extreme case, an overloaded server CPU could result in a Denial-of-Service to users of MDs attempting to offload; in less extreme scenarios, response times and round-trip times (from MD to MEC server to MD) would increase.

3.5.1 Problem Formulation

Let $u_{j,c}$ be the binary variable that models the offloading of a job j on a MEC c , respectively. The binary variable is defined as follows:

$$u_{j,c} = \begin{cases} 1 & \text{if job } j \text{ is offloaded to } c, \\ 0 & \text{otherwise} \end{cases} \tag{3.5}$$

The offloading decision-making strategy deals with determining, for a given computation task j, whether to compute it locally on an MD or leverage the computing facilities offered by a MEC server c. In other words, the computational offloading decision is to determine the set: $\{u_{j,c} : j \in J, c \in C\}$, that models the decision on each job that needs to be processed. In the following subsections, mathematical relations are derived that model the dynamics of the computational offloading.

3.5.2 Computational processing time on a mobile device

Considering a mobile device 'MD', let X_j^{MD} denote the computational data (in bits) to be processed. Let λ_i denote the complexity of the application that processed the data (in bits per instruction). The total numbers of instructions, C_j^{MD}, are calculated using the following equation:

$$\text{C}_j^{\text{MD}} = \frac{X_j^{\text{MD}}}{\lambda_i} \tag{3.6}$$

Let α_i be the on-board processor speed of a MD (in instructions per second) and $(1 - \frac{L_i^{\text{MD}}}{100})$ be the CPU workload on a MD. The time to compute the job on MD i is given as follows:

$$T_i^{\text{MD}} = \frac{\sum_{j \in J} X_j^{\text{MD}} u_{j,c}}{(1 - \frac{L_i^{\text{MD}}}{100}) \times \alpha_i} \tag{3.7}$$

Equation (3.7) provides a relationship between the completion time, computational data, MD load and the processing speed of the device. From this equation, we note that the completion time is directly proportional to the computational data and mobile device loading. In contrast, completion time is inversely proportional to the processing speed of the device.

3.5.3 Local Energy Consumption

Processing a computational task on an MD will require a certain amount of energy. Let P_i^{MD} be the power required of the embedded processor on MD i. The energy consumption can be quantified as follows:

$$E_i^{\text{MD}} = P_i^{\text{MD}} \times T_i^{\text{MD}} \tag{3.8}$$

where T_i^{MD} is obtained from the solution of Equation (3.7).

3.5.3.1 Computational processing time on a MEC

Let X_j^{MEC} denote the computational data (in bits) as the size of input data that needs to be processed from an application that is running on a MEC at λ_c (in bits per instruction). Let β_c be the on-board processor speed of MEC c (in instructions per second) and $(1-\frac{L_c^{\mathrm{MEC}}}{100})$ be the server-side processor workload of MEC c. The computational time to process a job on a MEC server is given as follows:

$$T_c^{\mathrm{MEC}} = \frac{\sum_{j\in J} X_j^{\mathrm{MEC}} u_{j,c}}{(1-\frac{L_c^{\mathrm{MEC}}}{100})\beta_c \lambda_c} \tag{3.9}$$

Let γ_c^{UL} be the up-link speed (in bits/second). The following equation gives the time to send the job over the link.

$$T_c^{\mathrm{UL}} = \frac{\sum_{j\in J} u_{j,c} X_j^{\mathrm{MD}}}{\gamma_c^{\mathrm{UL}}} \tag{3.10}$$

Let γ_c^{DL} be the downlink speed (in bits/second). The receiving time of the processed data can be calculated as follows:

$$T_c^{\mathrm{DL}} = \frac{\sum_{j\in J} \Pi X_j^{\mathrm{MEC}}}{\gamma_c^{\mathrm{DL}}} \tag{3.11}$$

where Π $(0 \le \Pi \le 1)$ is defined as the proportion of data size reduction after a job is processed. Furthermore, we assume that the links between the MD and MEC are symmetric, which means MD can send and receive data to and from MEC at the same rate without the loss of generality i.e. $\gamma^{\mathrm{UL}} = \gamma^{\mathrm{DL}} = \Gamma$.

Equations. 3.9, 3.10 and 3.11 can be represented as:

$$T_c^{\mathrm{Total}} = \underbrace{\frac{\sum_{j\in J} X_j^{\mathrm{MEC}} u_{j,c}}{(1-\frac{L_c^{\mathrm{MEC}}}{100})\beta_c \lambda_c}}_{\text{MEC Processing Time}} + \underbrace{\frac{\sum_{j\in J} u_{j,c} X_j^{\mathrm{MD}}}{\gamma_c^{\mathrm{UL}}}}_{\text{Transmission Time}} + \underbrace{\frac{\sum_{j\in J} \Pi X_j^{\mathrm{MEC}}}{\gamma_c^{\mathrm{DL}}}}_{\text{Receiving Time}} \tag{3.12}$$

Table 3.6: A list of parameters used in numerical experiments; MD and MEC processing speeds taken from from [105]

Entity	Parameter	Value	Unit
Jobs Size	X^{MD}	1	MB
MD	α	3.60×10^9	IPS
MEC	β	1.40×10^{11}	IPS
Application 1	C^{MD}	3.7×10^9	Ins/MB
Application 2	C^{MD}	3.7×10^8	Ins/MB
Network	Γ	20	Mbps

3.5.4 Offload Energy Consumption

Here we are only concerned with the energy consumption of the mobile device. Let P^{send} and P^{rec} denote the power required of the MD to send and receive the request for offloading the job (in W) respectively. The total energy consumption for this step is given as:

$$\left(E_{i,c}^{\text{UL}}, E_{i,c}^{\text{DL}}\right) = \left(P_i^{\text{send}} \times T_{i,c}^{\text{UL}}, P_i^{\text{rec}} \times T_{i,c}^{\text{DL}}\right) \tag{3.13}$$

Let P^{idle} denote the power rating of the MD (in W) when it is in the idle state and is waiting to receive the solution of the computational task back from the MEC. The energy consumption of the idle state is given as follows:

$$E_i^{\text{idle}} = P_i^{\text{idle}} \times T_c^{\text{MEC}} \tag{3.14}$$

The total energy consumption of processing all jobs on a mobile device i are given as follows:

$$\begin{aligned} E_i^{\text{Total}} = & \underbrace{P_i^{\text{send}} \times T_{i,c}^{\text{UL}}}_{\text{Transmit energy consumption}} + \underbrace{P_i^{\text{idle}} \times T_c^{\text{MEC}}}_{\text{Idling energy consumption}} \\ & + \underbrace{P_i^{\text{rec}} \times T_{i,c}^{\text{DL}}}_{\text{Receiving energy consumption}} \end{aligned} \tag{3.15}$$

3.6 Numerical Results

This section demonstrates the use of the mathematical model presented in Section 3.5.1 in numerical simulations with two applications whose parameters are presented in Table 3.6. The values used for Application 1 are taken from [105]. Application 2 had 10-fold reduction in the numbers of instructions generated, following the ratio proposed in [84] for MCC to distinguish between the high- and low-complexity applications considered for offloading.

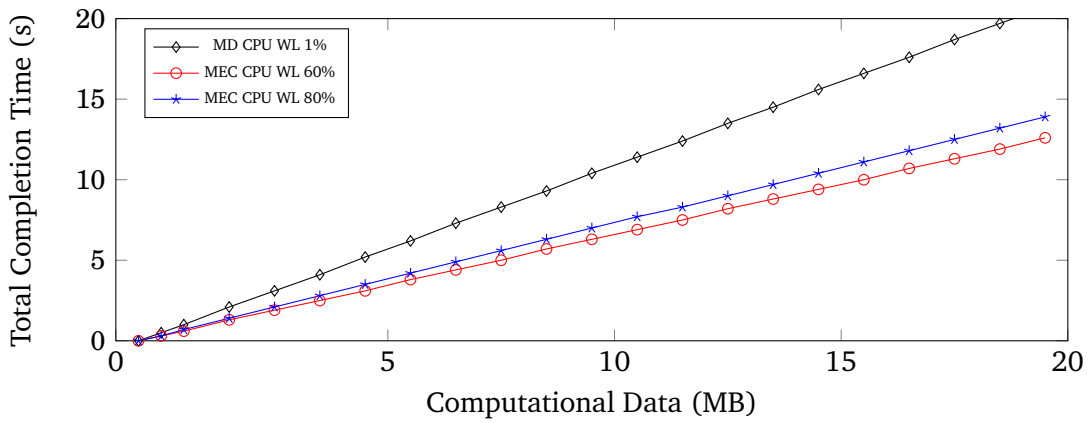


Figure 3.9: Effect of computational data size on task completion time with the higher complexity application 1 at a 20 Mbps communication link speed to/from a MEC server.

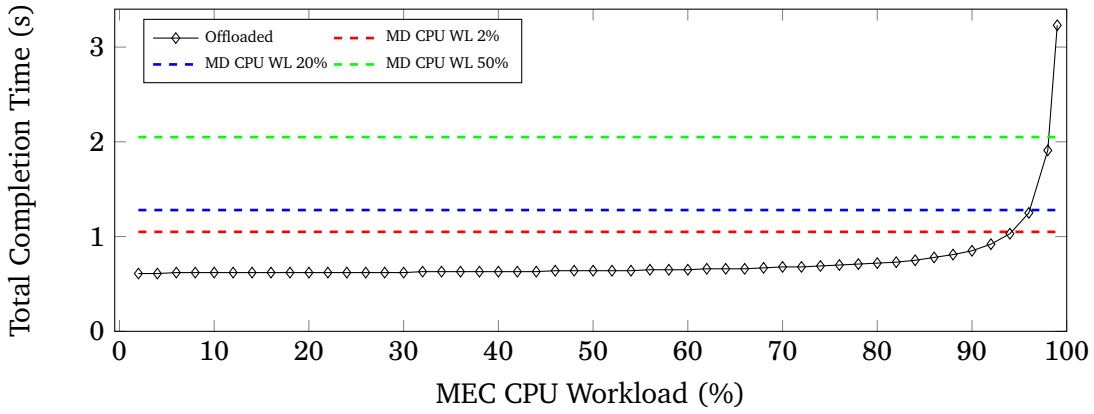
The authors of [90] quote three values for power rating by a mobile device: P^{MD} is the power rating of a mobile device while computing (0.9 W), P^{idle} is the power rating of a device while idling (0.3 W) and P^{Send} and P^{rec} are the power rating of a device while transmitting and receiving information (1.3 W). These values have been used for calculating the energy used by a MD computing locally or offloading to a MEC server. Furthermore, the value of $\Pi = 0.4$ is assumed in Equation (3.11), which means that the data returned from a MEC server is 60% less than the data which is sent to the server. The following subsections investigate the impact of increasing job data size, MEC workload, MD workload and link speed on the overall computational time and energy usage by the MD.

3.6.1 The impact of increasing job data size on the completion time

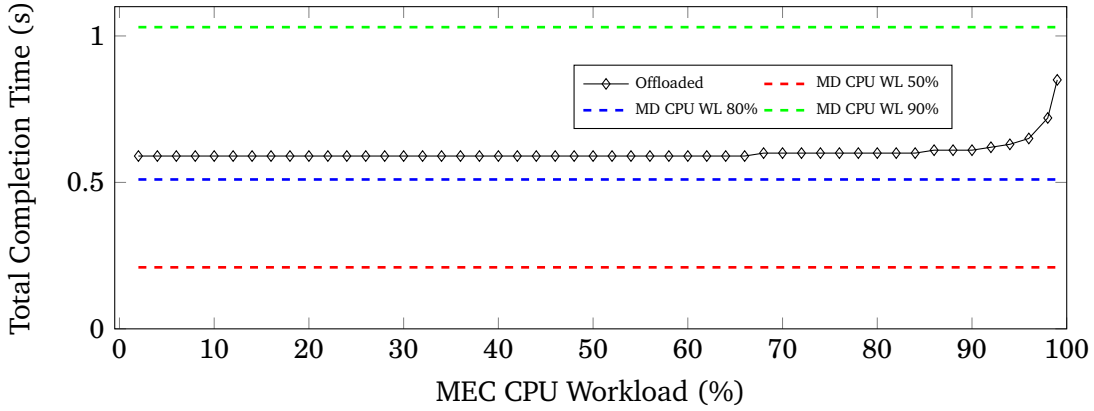
Job size and completion time have a linear relation as shown in Equations (3.7, 3.9). Figure 3.9 shows the effect of increasing job (from 1 MB to 20 MB) size on the total completion when computing locally or offloading files for Application 1. Three cases are plotted: 1% MD CPU loading, 60% MEC CPU loading and 80% MEC CPU loading. Each case showed a linear increase in the total completion time. Even with the very low (1%) MD loading, the local job completion time was longer than offloading the task to the MEC server at any job size and at either server CPU workload.

3.6.2 The impact of MEC workload on the completion time

Figure 3.10 presents the results with the two applications when the MEC CPU workload was increased from 1% to 99%. Figure 3.10(a) shows that the total task completion time with the higher complexity Application 1 increased greatly as the MEC server CPU workload approached



(a) Application 1: Offloaded and local computational times at MD CPU workloads of 2%, 20% and 50%.



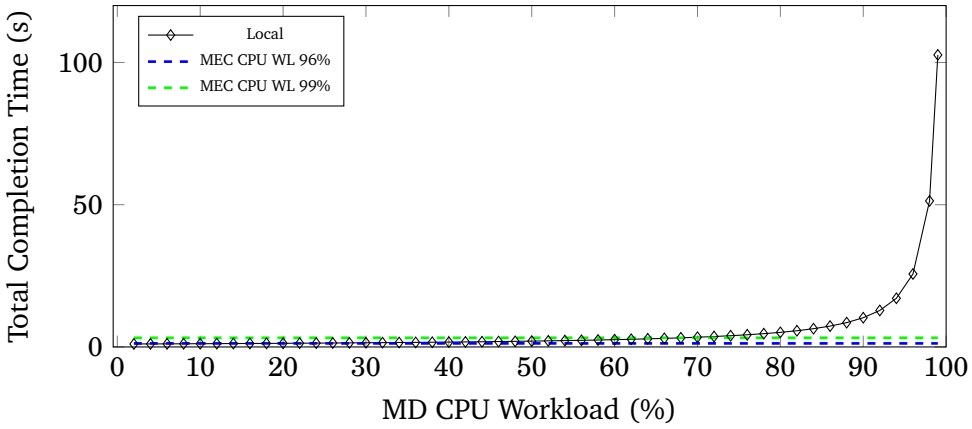
(b) Application 2: Offloaded and local computational times at MD CPU workloads of 50%, 80% and 90%.

Figure 3.10: Effect of varying MEC server CPU workload on task completion time for 1 MB data file offloaded at 20 Mbps connection link speed or processed locally at selected MD CPU workloads.

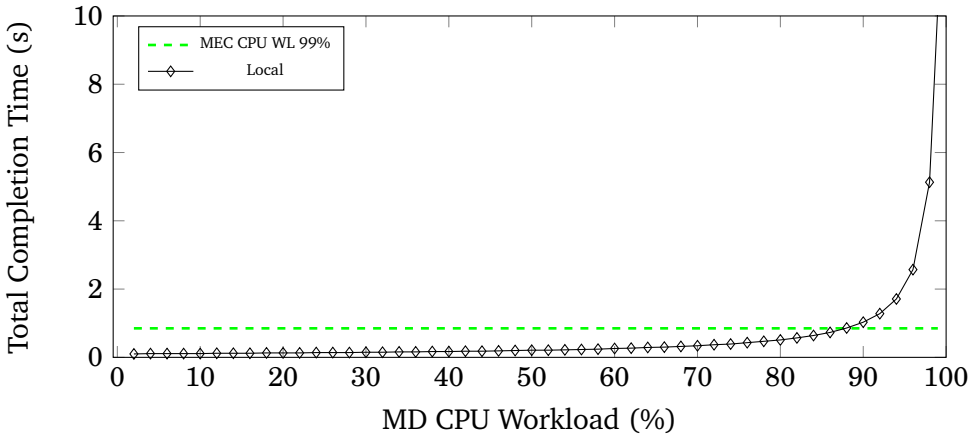
100%. Nevertheless, offloading could result in a shorter task completion time even at a >90% MEC CPU workload if the CPU workload on the MD processor was >50%. Figure 3.10(b) shows that, whatever the MEC server CPU workload, local computation was faster with the lower complexity Application 2 until the MD CPU workload became high.

3.6.3 The impact of MD workload on the completion time

Figure 3.11 presents the results with the two applications with the MD CPU varying up to 99%. Figure 3.11(a) shows that local computation was faster at low MD CPU workloads (<20%) but at higher MD CPU workloads offloading was beneficial for reducing task completion time at a MEC server CPU workload of 96%; even at 99% server CPU workload, offloading was beneficial if the MD CPU workload exceeded 70%. Figure 3.11(b) shows that local computation was faster with the lower complexity Application 2 than offloading to a very high CPU workload server



(a) Application 1: Offloaded and local computational times at MEC server CPU workloads of 96 and 99%.



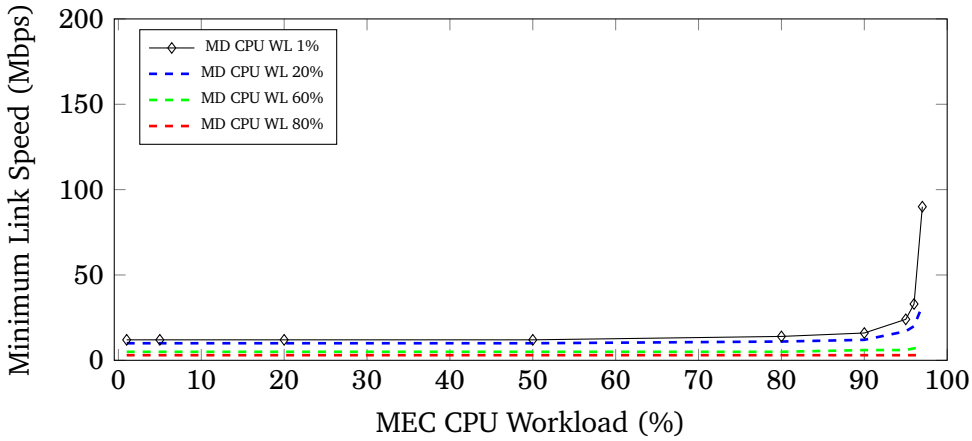
(b) Application 2: Offloaded and local computational times at MEC server CPU workload of 99%.

Figure 3.11: Effect of varying of MD server CPU workload on task completion time for a 1 MB data file offloaded at 20 Mbps connection link speed or processed locally at selected MEC server CPU workloads.

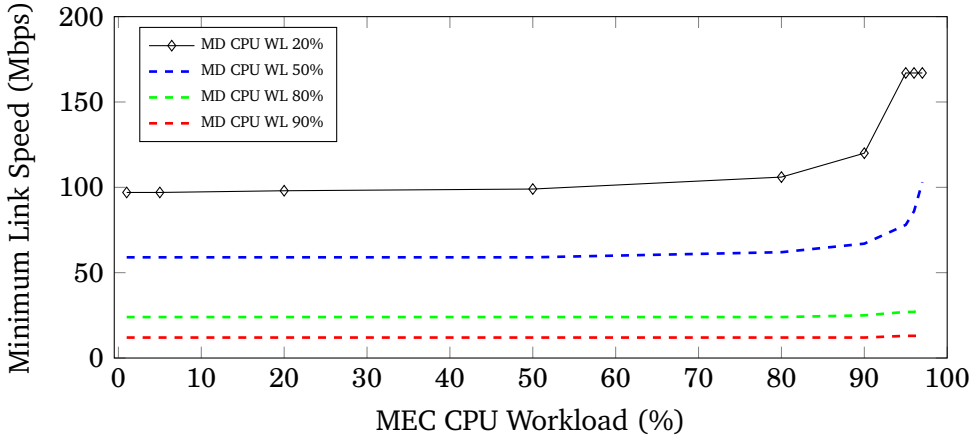
until the MD CPU workload approached 90%.

3.6.4 The impact of link speed on offloading decision

The higher the MD processor CPU workload, the lower was the minimum communication link speed required for shorter total task completion time by offloading; this is shown in Figure 3.12(a). With the much smaller local computation demands required for Application 2, minimum communication link speeds required for shorter total task completion time by offloading were much higher than for Application 1; this is shown in Figure 3.12(b). At high MD CPU workloads, link speeds were compatible with 4G wireless networks but 5G range speeds were required if local computation was performed at low MD CPU workloads.



(a) Application 1: Minimum link speed required for shorter completion time by offloading at different MD CPU workloads.

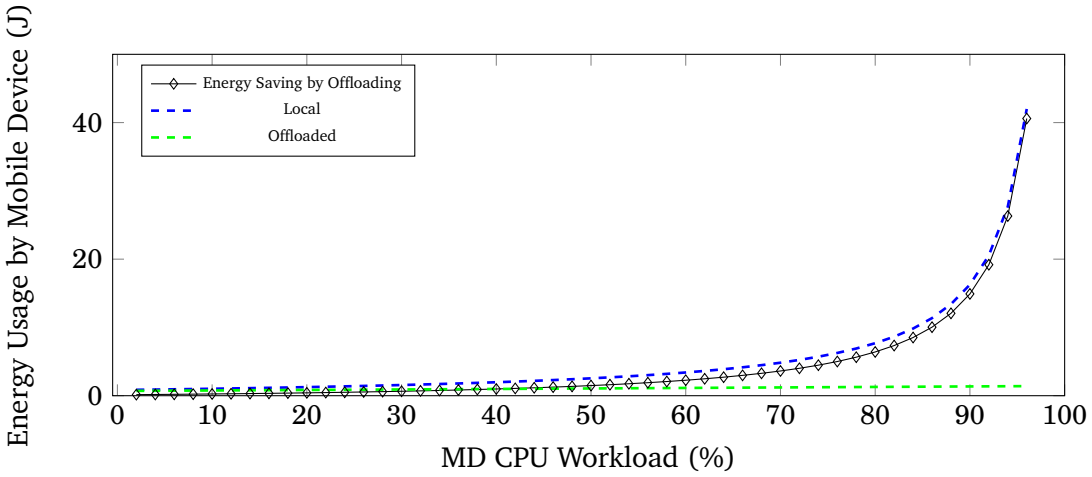


(b) Application 2: Minimum link speed required for shorter completion time by offloading at different MD CPU workloads.

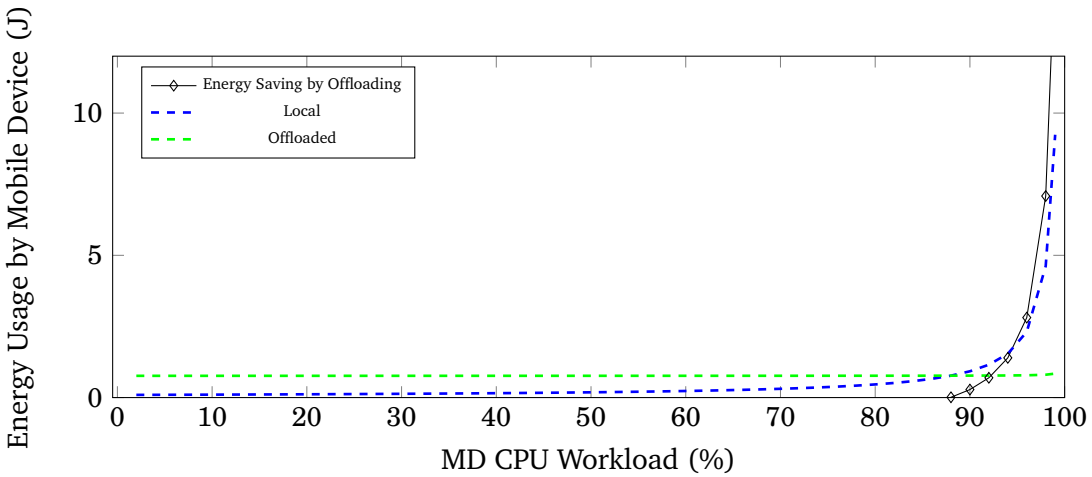
Figure 3.12: Effect of varying MEC server CPU workload on the minimum link speed required for shorter task completion time with a 1 MB data at selected MD CPU workloads.

3.6.5 Energy Usage by a MD

Figure 3.13(a) shows that energy usage for local processing by the MD increased as the MD CPU workload increased with the higher complexity Application 1. In contrast, MD energy usage while offloading increased only very little when very high MD CPU workloads were reached. The energy saving for the MD occurred at all MD CPU workloads but increased greatly as MD CPU workloads increased and reached nearly 90% of local energy use when the CPU workload reached 90%. With the lower complexity Application 2, however, no energy savings were possible for the MD by offloading until the MD CPU workload approached 90%, as shown in Figure 3.13(b). This was because the low complexity of the application resulted in very short completion times when computed locally and communication times with the MEC network



(a) Application 1: Energy use by a mobile device computing locally or offloading to a MEC server.



(b) Application 2: Energy use by a mobile device computing locally or offloading to a MEC server.

Figure 3.13: Effect of varying CPU workloads on energy consumption by an MD for a 1 MB data processed locally or offloaded at 20 Mbps.

caused total task completion times by offloading to be longer than local processing.

3.7 Discussion

The advantages of computation offloading to a MEC server for the users of MDs for shorter task completion times are critically dependent on the interaction of four independent variables: on-board processor CPU workload, server-side CPU workload, communication link speed and task complexity.

As an illustration, relatively slow on-board processors can advantageously (for task completion time) offload to MEC servers at very low link speeds provide by low-bandwidth networks

because the ratio of server-side processor speed to on-board processor speed is high. However, a MEC network should have the flexibility to adapt to widely fluctuating numbers of mobile devices seeking connections with different processor speeds and differing task complexities in an environment where available bandwidth could decrease rapidly. In addition, MEC service suppliers will increasingly be required to provide the greatest processor speed edge over currently available mobile devices together with the highest economically practical bandwidth for data transmission and return to ensure maximal user Quality of Experience (QoE).

MDs are increasingly engineered with high processing powers and this increased computational power of MDs puts limits on how much benefit can be achieved in terms of task completion time by offloading. With a higher computational task complexity, only very high server-side CPU workloads prevent offloading having shorter task completion times and even this factor is overcome as the on-board processor CPU workload increases. Energy savings for the MD are important even at very high MEC server CPU workloads. However, with a ten-fold lower computational task complexity, data transmission time is the dominant factor in rejecting offloading but higher link speeds eliminate any advantage of a high-power MD processor.

Interruptions and delays in the ability of mobile devices to access MEC servers ("link access delays") greatly reduce the ability of offloading to offer benefits for task completion times. This problem is magnified if a higher-speed on-board processor is used in the mobile device. The load on a MEC network caused by increasing numbers of mobile users can erode any task completion time advantage by offloading as the differential between the MEC server and on-board processor speeds decreases.

The energy savings mobile devices by offloading to a MEC server are major even at low or modest available bandwidths in a 4G network. In practice, with fast server-side processors and high bandwidths, the default option for the users of mobile devices could be to offload primarily or solely for energy (battery lifetime) savings even if execution times were not improved by offloading because of the very marked effect on energy use by the mobile device. This not only increases the demand pressure to offload to the MEC system on the supplier side but also necessitates (on the user side) a decision-making process in which both task execution time and energy saving factors can be assessed.

Energy usage by the MD is not reduced by offloading until the MD CPU workload is very high. For a MEC network, therefore, ease of use and the QoE for mobile users and subscribers can only be established if the network functions smoothly and efficiently. For this, high link speeds for data transmission and reception, the highest possible ratio of server-side to on-board processor speeds and the avoidance of overuse of the server CPU are essential. A congested MEC network will disappoint users of MDs with high on-board processor speeds searching for faster task completion times, although energy use reduction will be paramount for some users with low MD battery charge.

Any decision-making process for offloading must be able to compute advantages of task

completion time and energy savings in a dynamic environment where widely fluctuating user numbers are to be expected in, for example, city centres. Similarly, the provider of Edge Computing facilities must build in sufficient flexibility to cope with peak demand without overloading the network or to link servers to back-up servers in larger and responsive network architectures.

Finally, the results demonstrate that as 5G wireless technologies introduce faster connection speeds, more medium- to low-intensity computational tasks will be able to be offloaded to reduce task completion time and energy usage by mobile devices. This will increase CPU demands in MEC servers and lead to potential overloading, which will require attention to capacity factors in MEC networks.

The research work described in this Chapter was designed to answer Research Questions 1-6 (Chapter 1, Section 1.3.2). One of the main conclusions is that factors affecting the success of offloading tasks from MDs are highly interconnected. As demonstrated by the results presented in this Chapter, it can be seen that the advantages of computation offloading to a MEC server for the users of MDs for shorter task completion times are critically dependent on the interaction of five network and user variables: communication link speed, the increases in processor speed represented by the MEC server over the MD, task complexity, on-board processor CPU workload and server-side CPU workload. In contrast, the size of the data file to be transmitted from the MD is neutral for offloading versus local processing while link access delays are important and network congestion manifests itself as problems in both excessive server CPU workloads and reduced bandwidth.

CHAPTER 4

A HEURISTIC APPROACH TO OPTIMIZING OFFLOADING SCHEDULES IN HETEROGENEOUS MEC NETWORKS

4.1 Introduction

The major conclusions from the work presented in Chapter 3 was that most jobs could be offloaded to reduce computation time and energy usage by a MD unless link speeds were extremely slow or if the ratio of processor speed in a MEC server to processor speed in the MD was critically low as the result of server overload and CPU workloads approaching 100%.

Given this optimistic result, the next logical question is how to schedule multiple jobs for offloading from single or cluster of MDs to MEC servers with differing computing powers over links of variable connection speeds. An example of multiple jobs for offloading is that of successive frames in a video file to be processed by facial recognition software. At the network-wide level, this resolves into the problem of resource allocation if large numbers of users of MDs are attempting to offload jobs simultaneously and is the topic of much research in Cloud Computing for hardware resources [58].

From an extensive survey of published material, the authors of [58] concluded that effective cloud resource scheduling can minimize the makespan of the workflow while reducing execution time and computation time of workloads in Cloud Computing. The word “makespan” is of key importance in this context because it defines the least-time scheduling option when multiple options are possible. Makespan analysis is widely practised in studies of logistics and supply chains where parallel processing of mechanical tasks is the default scenario; an example is [21] which provides a formal definition of makespan as: the time at which machines complete processing of the last job.

For the user of an MD attempting to execute multiple jobs this may involve parallel processing

using the MD as well as the available MEC servers. The precise balance of the parallel processing will be determined by the parameters of the MD and MEC server connection: processor speeds, link speeds, job sizes and CPU usages.

Key research question: Can the minimum global time for the offloading of multiple computational jobs from an individual MD (as a decentralized form of resource allocation) or from multiple MDs in a heterogeneous MEC network (as a centralised form of resource allocation) be identified?

This Chapter makes the following two major contributions:

- A proposed use of scheduling to optimise the offloading of multiple jobs from an MD which is verified by linear programming and direct spreadsheet calculations for a heterogeneous MEC network with multiple servers.

- Novel heuristic algorithms to rapidly achieve near-optimal solutions to offloading scenarios where very numbers of schedules are possible in a heterogeneous MEC network.

This work was published in: R. Singh, S. Armour, A. Khan, M. Sooriyabandara and G. Oikonomou, "Heuristic Approaches for Computational Offloading in Multi-Access Edge Computing Networks", IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, 2020.

The remainder of this Chapter is arranged as follows: the concept of a least-time schedule is investigated in a relatively small network of one MD and up to 3 MEC servers with up to 5 jobs for offloading by manual calculation verified by a linear programming optimization approach; a heuristic is then developed for larger numbers of MDs and total job numbers to obviate the need for proprietary software and provide a faster means of identifying least-time schedules close to global optimum times as deduced by linear programming optimization; finally, the outcomes are discussed in relation to users of MDs offloading to heterogeneous MEC networks can assess time advantages presented by offloading schedules and the implications for the providers of MEC networks faced with potential high use of offloading and consequent server overloading.

4.2 A Mathematical model

Let us consider a system comprising of MDs denoted as $i = \{1,\ldots,m\}$. Let J_i denote the set of all jobs on a mobile device i. Also, let $c = \{1,\ldots,n\}$ denote MECs at a local base station represent the set of MEC servers available for the data requests from the given set of mobile users. Note that every MD is independent of the other and computation tasks offloaded on the MEC servers

are processed on first in first out basis. Let $u_{j,c}$ be the binary variable that models the offloading of a job j on MEC c, respectively. The binary variable is defined as follows:

$$u_{j,c} = \begin{cases} 1 & \text{if job } j \text{ is offloaded to } c, \\ 0 & \text{otherwise} \end{cases} \tag{4.1}$$

where $J_i \cap J_{i'} = \phi \; \forall \;\; i, \;\; i' \in M \;\; i \neq i'$.

Let A_i represents the allocation matrix of a MD i. The matrix A_i consists of binary variables in each row and column and represents the offloading decision as described by Equation 4.1. The allocation matrix is given as follows:

$$A_i = \begin{Bmatrix} u_{j_1,c_1} & u_{j_1,c_2} & \dots & u_{j_1,c_n} \\ u_{j_2,c_1} & u_{j_2,c_2} & \dots & u_{j_2,c_n} \\ \vdots & \vdots & & \vdots \\ \vdots & \vdots & & \vdots \\ \vdots & \vdots & & \vdots \\ u_{j_{mi},c_1} & u_{j_{mi},c_2} & \dots & u_{j_{mi},c_n} \end{Bmatrix}$$

Note that the sum of a row (and a column) is always less than or equal to 1. When the sum of a row is equal to zero, that means the job is processed locally on a mobile device.

4.2.1 Constraints for local computation

Let X_j^{MD} denote the computational data (in bits) as the size of input data that needs to be processed from an application that is running on MD i. The total numbers of instructions, C_i^{MD}, can be calculated as:

$$C_j^{\text{MD}} = \frac{X_j^{\text{MD}}}{\lambda_i} \tag{4.2}$$

where λ_i (bits per instruction) represents the complexity of the job at hand.

Let α_i be the on-board processor speed in (instructions per second). The completion time of a computational task on the MD i, T_i^{MD}, is defined as follows:

$$T_i^{\text{MD}} = \frac{\sum_{j \in Ji} C_j^{\text{MD}} (1 - \sum_{c \in C} u_{j,c})}{\alpha_i} \tag{4.3}$$

Let D_i^{MD} denote the total amount of data processed on the MD i. The total amount of data processed on the mobile device depends on the offloading decisions and therefore it is defined as follows as a function of the offloading binary decision variables.

$$D_i^{\text{MD}} = \sum_{j \in J_i} X_j^{\text{MD}} (1 - \sum_{c \in C} u_{j,c}) \tag{4.4}$$

If a job is offloaded, it is offloaded to a unique MEC server c. Therefore, the sum $\sum_{c \in C} u_{j,c}$ returns 1 for all offloaded jobs and hence the data for such jobs is not included in calculation of data processed locally in Equation 4.4.

4.2.2 Constraints for transmission data

Let γ^{UL} denote the speed of uplink (in bits per second). The total processing time $T_{i,c,j}$ of transmitting data from MD i to MEC c is given as follows:

$$T_{i,c,j} = \frac{\sum_{j \in J_i} u_{j,c} X_j^{\mathrm{MD}}}{\gamma^{\mathrm{UL}}} \tag{4.5}$$

All the data on the MD i which need to be processed is equal to sum of the data that is processed on the mobile device i locally and the data that flows over the link γ^{UL}.

$$\sum_{j \in J_i} X_j^{\mathrm{MD}} = D_i^{MD} + \sum F_{i,c} \tag{4.6}$$

The transmission links that connects mobile devices and MEC servers have finite capacities. In order to implement the transmission capacity constraints, first we need to quantify the amount of data that can flow on a given let. Let $F_{i,c}$ denote the data flow over the link between the MD i and the MEC c. This data flow depends on the number of jobs that are offloaded from the MD i to MEC server c over the link that connects them, and the equation modelling this is given as follows:

$$F_{i,c} = \sum_{j \in J_i} X_j^{\mathrm{MD}} u_{j,c} \tag{4.7}$$

A constraint on the flow $F_{i,c,}$ is implemented as follows:

$$0 \leq F_{i,c} \leq S_{i,c}^{\mathrm{Max}} \tag{4.8}$$

where $S_{i,c}^{\mathrm{Max}}$ is the transmission capacity of the link (i,c).

4.2.3 Constraints for MEC computation

We consider that there are $\{c = 1, \ldots, n\}$ MEC servers deployed at the local cellular network. Let X_c^{MEC} denote the computational data (in bits) as the size of input data that need to be processed from an application that is running on a MEC c at λ_c (in bits per instruction). The total numbers of instructions, C_c^{MEC} on a MEC c can be calculated as:

$$C_c^{\mathrm{MEC}} = \frac{X_c^{\mathrm{MEC}}}{\lambda_c} \tag{4.9}$$

The processing time of the task on the MEC server can be formulated from C_c^{MEC}, the computational task size (in instructions) and β_c, MEC processor speed (in instructions per

second). The offloading time of computational task on a MEC server, T_c^{MEC}, can be calculated as:

$$T_c^{\text{MEC}} = \frac{\sum_{j \in J} C_c^{\text{MEC}} u_{j,c}}{\beta_c} \tag{4.10}$$

Let us denote the downlink speed as γ^{DL} (bits per second). We assume that the total data size is reduced by a factor of Π. The total reception time of all the offloaded jobs is given as follows:

$$T_{c,i,j} = \frac{\sum_{j \in J_i} \Pi \times X_c^{\text{MEC}}}{\gamma^{\text{DL}}} \tag{4.11}$$

Adding the Equations. (4.5),(4.10) and (4.11), the total computational time for all offloaded jobs is given as follows:

$$T_c^{\text{MEC}} = \underbrace{\frac{\sum_{j \in J_i} u_{j,c} X_j^{\text{MD}}}{\gamma^{\text{UL}}}}_{\text{Transmission Time}} + \underbrace{\frac{\sum_{j \in J} C_c^{\text{MEC}} u_{j,c}}{\beta_c}}_{\text{MEC processing time}} + \underbrace{\frac{\sum_{j \in J_i} \Pi \times u_{j,c} X_c^{\text{MEC}}}{\gamma^{\text{DL}}}}_{\text{Receiving Time}} \tag{4.12}$$

Furthermore, a binary variable $u_{j,c}$ is constrained to respect the following constraint:

$$\sum_{j \in J_i} u_{j,c} \leq 1 \quad \forall \; i \in M, c \in C \tag{4.13}$$

When the job is processed locally, the right-hand side of the equation is equal to zero. When the task is offloaded on a server, the constraint ensures that it is not offloaded on more than one MEC server.

Let D_c^{MEC} denote the total amount of data that processed on a MEC c. The amount of data processed on a MEC depends on the number of jobs offloaded from the given number of mobile devices. The following equation determines the total computational data that is process of a MEC server c:

$$D_c^{\text{MEC}} = \sum_{j \in J_i} X_j^{\text{MD}} u_{j,c} \tag{4.14}$$

4.2.4 Objective Function and Overall Formulation

A scheduling optimisation problem is considered in which all the jobs are assigned to MECs at particular times and only one job can be processed at a given time on any individual MECs and evaluate that MD i given m_i jobs of varying processing times, which need to be scheduled on MECs with varying processing power and different link access speeds.

The overall objective of the optimisation problem is to minimise the total time needed to execute a given number of jobs. Mathematically this objective function is formulated as follows:

$$T^{\text{Total}} = \max\left\{\underbrace{\max\{T_i : i \in M\}}_{\text{Local Time}}, \underbrace{\max\{T_c : c \in C\} + \sum_{(i,c,j)} (2-\Pi)T_{i,c,j}}_{\text{MEC Processing Time}}\right\} \tag{4.15}$$

where T_i is the total jobs processing time on a MD i. There are three components in the equation. The first component is the local computation time of the jobs. The second component is the computation time of MEC servers and the third component is the transmission and reception time.

This problem can be formulated as a mixed integer linear programming (MILP), in which the objective function and the constraints are linear. The optimization mathematical model has been developed in Pyomo [1] using a solver from IBM called CPLEX [2]. It is not a recommendation, of course, that MDs utilize the IBM software programme; the use of CPLEX was solely to offer a theoretical solution for a global scheduling optimum. This approach was designed to investigate the use of makespan analysis in scheduling for offloading jobs from an MD.

4.3 Numerical Testing

The proposed linear programming model described in the earlier section is tested on two cases. As noted earlier, the model needs information about a range of parameters. These parameters are taken from a publication [105]. The complexity of an application which is defined by λ_m is assumed to take the value of (2.27×10^{-3}) instructions per bit. The following two processor speeds of MEC are considered are: (for on-board device) Apple A9 $(3.6 \times 10^9$ IPS) and (for server-side processor) the Intel Xeon processor $(1.40 \times 10^{11}$ IPS).

4.3.1 Illustrative example of a user-run optimization program – 81 scheduling options

In this section, we present numerical results to verify our analysis and validate the performance of the proposed model; the outcome of the optimization model was verified by manual spreadsheet calculations. In the first case, one mobile device sought to offload 4 jobs to one or both MECs. The parameters chosen for numerical simulation are shown in Table 4.1. The number of scheduling options was given by $(n+1)^m$, where n represents the number of MECs and m represents the number of jobs; in this case the total number of distinct scheduling options was

[1] Python PYOMO formulating, solving, and analyzing optimization models: `http://www.pyomo.org/about`

[2] IBM CPLEX Linear programming problem solver: `https://www.ibm.com/pt-en/products/ilog-cplex-optimization-studio`

Table 4.1: Parameters values used for illustrative example (Case 1)

Entity	Parameter	Value	Unit
Jobs Size	X^{MD}	1 - 4	MB
MD	α	3.60×10^9	IPS
MEC1	β_1	1.40×10^{11}	IPS
MEC2	β_2	1.40×10^{11}	IPS
	MD - MEC 1	15	Mbps
Network	MD - MEC 2	28	Mbps

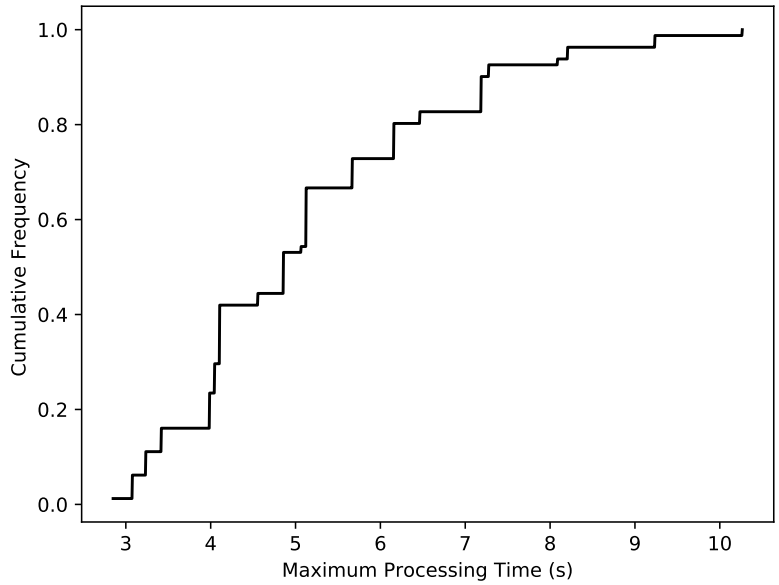


Figure 4.1: Cumulative Frequency Distribution plot of the maximum times for 81 scheduling options.

81. The full list of schedules used for 1 MD offloading up to 4 jobs to 2 MEC servers is contained in the Appendix, Table A.1.

The linear optimization program identified a unique solution of 2.85 (s) in which three jobs were offloaded (one to MEC 1 and 2 to MEC 2) with the fourth job performed locally on the MD i. The optimal time was 28% of that required for all jobs to be performed locally on the MD i i.e. 3.6 times faster.

Fig. 4.1 shows the cumulative distribution plot of the schedule times. Only 5% of the 81 scheduling options are within 10% of the optimal time, suggesting that there is significant scope to achieve a very bad outcome if jobs are not allocated well. The shortest schedule with all 4

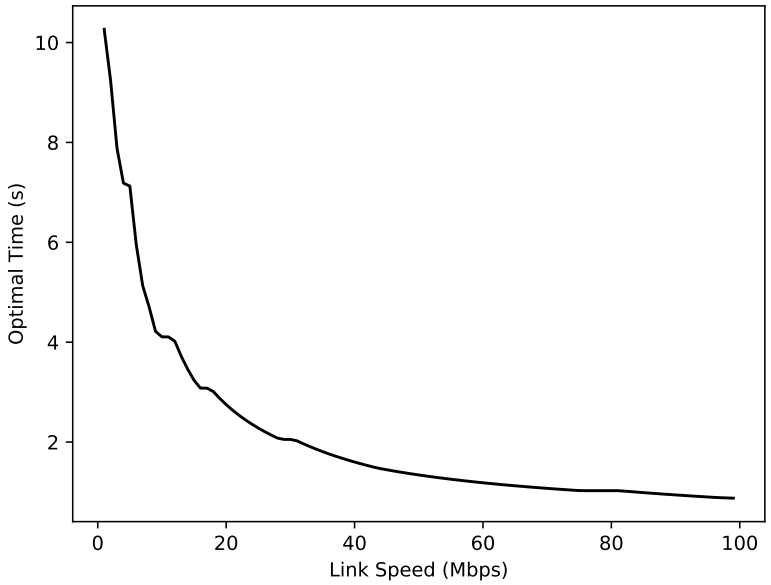


Figure 4.2: Effect of link speeds on optimal time that are submitted by the mobile devices.

jobs offloaded required a 20% longer completion time than the optimum which also suggests that a simple "always offload" decision process can be significantly sub-optimal.

Further analysis investigated the effects of total job size and link speed on the success of offloading jobs to the MEC network. Progressive increases in the total job size from 10 MB to 34 MB did not result in complete offloading. Increasing the link speed, however, had a more pronounced effect. Fig. 4.2 shows how the optimal time decreased with increasing link speed.

Table 4.2: Scheduling Options vs Link Speed

Link speeds increase (%)	Optimal Time (s)	Schedule option	Schedule Time for all jobs offloaded (s)	Schedule within 10% of the Optimum	Schedule within 25% of the Optimum	Schedule within 50% of the Optimum
0	2.85	6	3.42	5	15	41
10	2.64	6	3.16	2	15	28
20	2.46	6	2.95	0	6	28
30	2.43	6	2.77	2	6	28
40	2.34	5	2.62	2	11	37
50	2.19	5	2.48	1	11	35
75	1.90	26	2.21	1	10	21
100	1.68	26	2.01	0	10	21
225	1.02	68/73	1.02	5	1	20

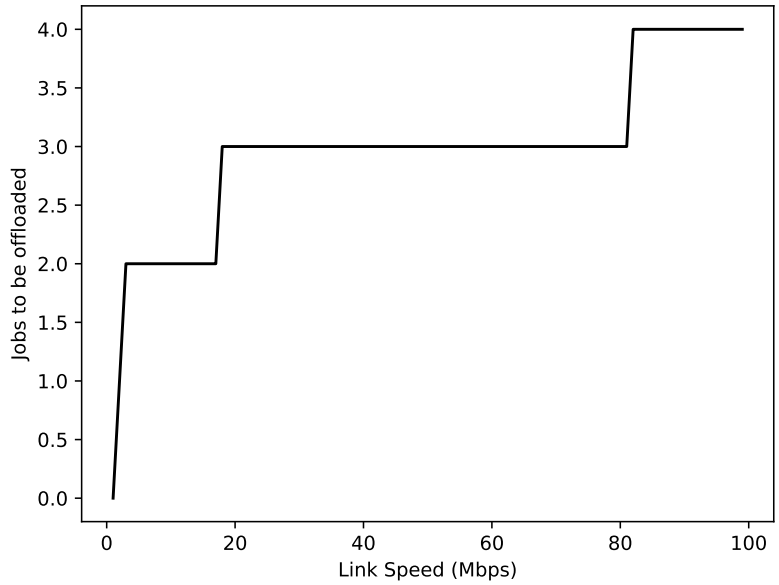


Figure 4.3: Jobs offloaded for the 81 scheduling options from 1 MD to 2 MEC servers.

Figure 4.3 does not show the fine granularity of the results. If the link speeds used in Table 4.1 are increased by 10% increments, the least-time schedule option changes twice while still offloading 3 of the 4 jobs (Table 4.21). Only by increasing from 100% to 225% of the original link speeds do 2 scheduling options with all 4 jobs offloaded become the least-time (optimal) solutions. Until the speeds have increased by this factor, a time penalty is incurred if all 4 jobs are offloaded, ranging from 12-20% longer than the optimal time. In general, the higher the link speeds, the more isolated the optimal solution becomes, i.e. fewer other scheduling options give times within 10%, 25% or 50% of the optimal (Table 4.21).

The effects of increasing the job sizes are very different (Table 4.3). The total job size represented by (Table 4.1) is 10 MB. If this is increased in 4 MB increments, although the least-time scheduling option changes, the same 3 jobs are offloaded. The interpretation is that transmission and receiving times during offloading greatly outweigh the faster processing time on the MEC servers, by factors of 2000-13000. As the total job size increases, the optimal time increases but the time penalty incurred by offloading all 4 jobs also increases.

Intuitively, if the CPU workload of MEC servers becomes too high, the optimum schedule for offloading multiple jobs from a single MD would be expected to shift to less jobs being offloaded and more being processed locally (Chapter 3, Section 3.6). Numerical analysis showed, however, that the impact of CPU workload began to be apparent at only moderate workloads (Table 4.4). By 40% CPU usage, the optimum schedule changed from a unique schedule to two different schedules sharing the optimum time but having 3 or 2 jobs offloaded. This result requires a policy to be pre-existing on the MD to choose either to offload the greater number of jobs or the

Table 4.3: Scheduling Options vs Job (MB)

Job Size (MB)	Total Job Size (MB)	Optimal Time (s)	Schedule Number	Schedule Time for all jobs offloaded (s)	Schedule within 10% of the Optimum	Schedule within 25% of the Optimum	Schedule within 50% of the Optimum
1-4	10	2.85	6	3.42	5	15	41
2-5	14	3.42	10	4.86	0	6	27
3-6	18	4.56	10	6.47	0	9	23
4-7	22	5.70	10	7.41	5	7	23
5-8	26	6.84	10	8.90	5	14	23
6-9	30	7.98	10	10.25	5	14	23
7-10	34	9.12	10	12.41	5	14	23

Table 4.4: Effect of MEC Server CPU Workload on Optimum Schedule Selection (81 Distinct Schedule Options)

CPU Usage (%)	Optimal Time	Option Number	Jobs Offloaded
0	2.85	6	3
20	2.98	6	3
30	3.05	6	3
40	3.08	13,42	3,2
80	3.84	18,43	3,2
90	4.19	32,51	3,2
94	4.89	32,51	3,2
95	5.13	59	2
96	5.77	39	2
97	6.20	46	2
98	6.31	46	2
99	8.21	64	1

larger total MB offloaded or to select the optimum by reference to local energy use or financial cost of accessing the offloading service (Chapter 5). By 95% CPU usage, a unique optimum schedule is again identified but with only 2 jobs offloaded and by 99% CPU usage only a single job is offloaded in the optimum-time schedule (Table 4.4).

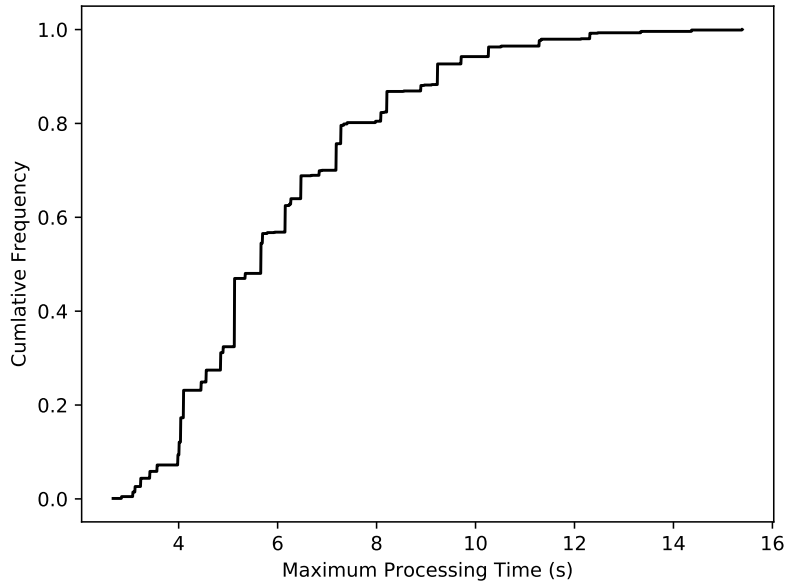


Figure 4.4: Cumulative Frequency Distribution plot of the maximum times for 1024 scheduling options

Table 4.5: Parameters Selected for Experimentation (Case 2)

Entity	Parameter	Value	Unit
Jobs Size	X^{MD}	1 - 4	MB
MD	α_i	3.60×10^9	IPS
MEC1	β_1	1.40×10^{11}	IPS
MEC2	β_2	1.40×10^{11}	IPS
MEC3	β_3	3.68×10^{10}	IPS
Network	MD - MEC 1	15	Mbps
	MD - MEC 2	28	Mbps
	MD - MEC 3	25	Mbps

Table 4.6: Effect of MEC Server CPU Workload on Optimum Schedule Selection (1024 Distinct Schedule Options)

CPU Workload (%)	Optimum Schedule	Optimum Time	Jobs Offloaded
0	124	2.67	4
20	124	2.71	4
30	124	2.74	4
40	124	2.78	4
50	124	2.83	4
60	124	2.91	4
70	68	3.08	2
80	68	3.08	2
90	68	3.42	2
95	68	4.74	2
96	68	5.40	2
97	457	6.16	2
98	457	6.16	2
99	457	9.18	2

4.3.2 Illustrative example of a user-run optimization program – 1024 scheduling options

The full list of schedules used for 1 MD offloading up to 5 jobs to 3 MEC servers is contained in in the Appendix, Table A.2. The parameters chosen for numerical simulation are shown in Table 4.5. Of the 1024 distinct scheduling options, a unique least-time option (no 124) was identified by both direct calculation and by application of the CPLEX program.

Figure 4.4 shows the CDF plot for the results. The least-time option was relatively isolated with only 0.4% of the other schedule options within 10% of the optimum time, 4.3% within 25% and 12.1% within 50%; these percentages are markedly smaller than found with the 81-option analysis (Table 4.6).

The optimum schedule time was 17.4% of the time required for the MD to process all 5 jobs but only 4 jobs were offloaded (Job 1 with a 2 MB file size was processed locally). The shortest schedule with all 5 jobs offloaded had a 16.7% longer time then the optimum.

As observed with the 81-option analysis, increasing the total job size (up 10-fold larger total MB) had no effect on the choice of optimum-time schedule option but increasing the link speeds by 100% resulted in two schedule options (no. 844 with Job 3 processed locally and no. 892 with all 5 jobs offloaded) sharing the optimum time.

As the MEC server CPU workload increased, the same optimum schedule was maintained until the workload reached 70% when a major shift occurred and the new optimum schedule only offloaded 2 of the 5 jobs (Table 4.6). At very high CPU workloads (97-99%), the optimum schedule changed again but still showed 2 jobs being offloaded.

4.4 The Heuristic Approach and the Problem of Scalability

Using the formula for the number of scheduling options given in Section 4.3.1 as $(1+n)^m$, the number of distinct schedules for a single MD offloading m jobs to n MEC servers rapidly increases; for example if $m = 10$ and $n = 4$, the number of distinct schedules becomes (9.8×10^6).

With massive numbers of possible schedules possible for relatively small numbers of users attempting to offload multiple jobs to a MEC network, the need arises for a generic and fast solution to achieving non-optimal but close-to-optimum scheduling solutions. Heuristic approaches provide such solutions and will form the topic explored in the remainder of this Chapter. A relevant example of this approach is that of [27] which analyzed the matching if user demand to MEC infrastructure in a large-scale simulations that covered an entire city (Milan, Italy) or a region (Trentino, Italy) using a publicly accessible telecommunications data set; "demand" was not precisely defined in [27] but the text did not specify "offloading" in the activities.

4.4.1 Methodology

To validate and to provide a comparison benchmark for results generated by the heuristic algorithms, linear programming optimization was performed using CPLEX. Using this mathematical model, we compute the optimum job allocation. Sub-optimal job allocations generated by the heuristic algorithms are then compared to this theoretical optimal allocation to assess the performance of the heuristic under different combinations of MDs and MEC servers. A heuristic algorithm can be modified to include user-selected features, for example, to introduce run-time constraints or fine-tune initial solutions by considering additional parameters [26]. Different algorithms were, therefore, developed to analyze how selected features affected the achieving of near-optimal solutions.

4.5 Heuristic Algorithms for Computation Offloading

This section presents a heuristic algorithm that is aimed at finding a near-optimal scheduling solution for a given number of job from MDs to a given set of MECs. The scheduling algorithm then proposes a solution for solving these jobs while minimizing the overall computational time.

Fig. 4.5 presents a flowchart of the proposed heuristic method. J represents the set of all jobs. The two key decision-making steps in the algorithm:

- For each job, whether to offload to a MEC or compute locally on the MD?
- If the decision from the above step is to offload, to which MEC that job is offloaded.

A total of nine policies (3 options each of the above two decision questions) are tested. These versions differed for the allocation of an offloading probability for an individual job on

Table 4.7: Notations used in the section.

Symbol	Definition
L_{c}	Loading on MEC c in bits
P_{j}^{i}	Probability of job j on MD i
O^{DD}	Offloading decision based on data size
O^{DFP}	Offloading decision based on fixed probability
O^{DPD}	Offloading decision based on probability distribution
M^{R}	Random allocation of MECs
M^{J}	MECs allocation based on job size
M^{T}	MECs allocation based on computational time

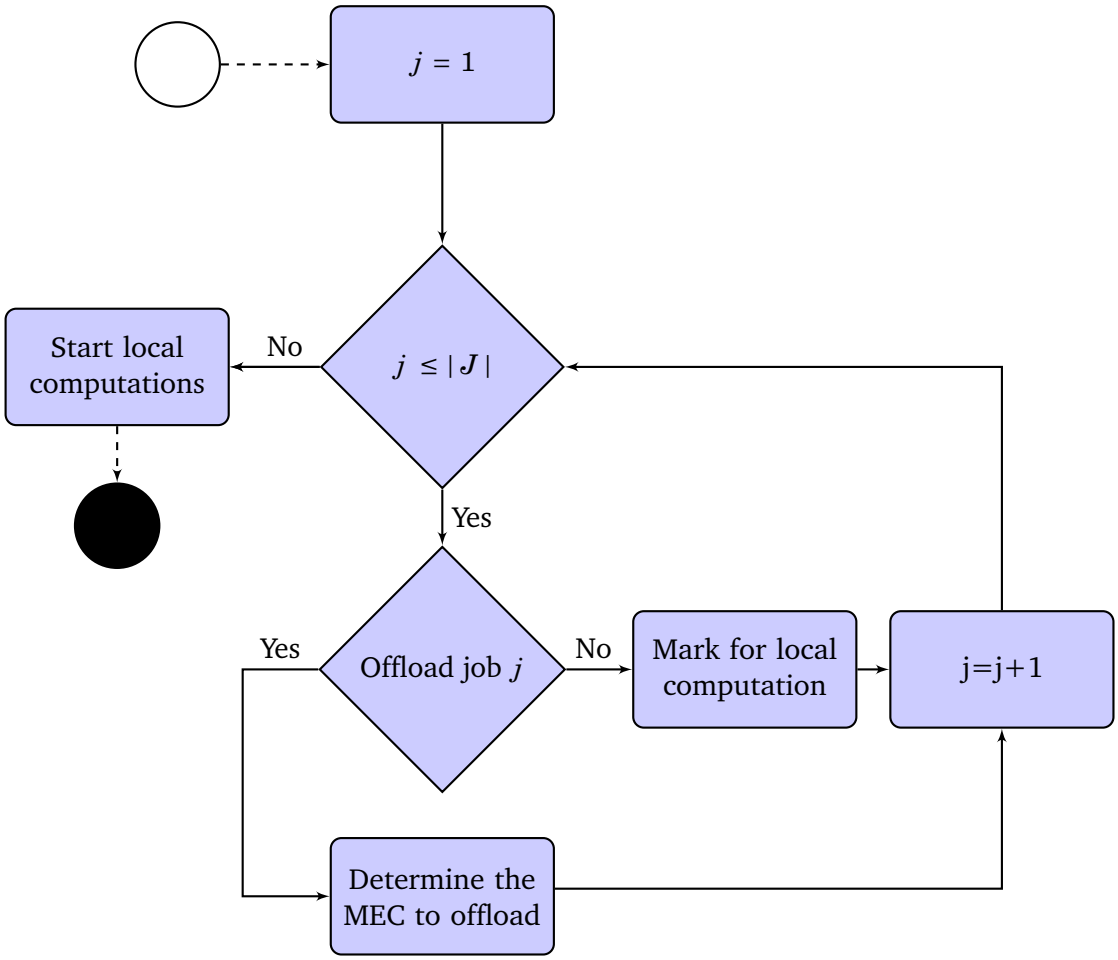


Figure 4.5: Flowchart of the heuristic algorithm for computation offloading.

each MD as shown by ("$j \leq |J|$") in the flowchart) and the allocation of a MEC server in the network to process an individual job as (indicated by "Determine the MEC to offload" in the flowchart). The detail regarding these policies is provided in the following subsections.

4.5.1 The offloading decision

Three different strategies were adopted for allocating the offloading probability of individual jobs on a MD. Generate a random number between 0 and 1 and if that number is less than P_J^i then offload. If not, execute on the MD.

4.5.1.1 Probability calculations for job offloading based on data file size ("$O^{\mathbf{DD}}$")

The algorithm follows the formula of job probability. The algorithm uses the following method to determine the offloading probability of individual jobs:

$$P_j^i = \frac{J_i}{\sum_{j \in J_i}} \tag{4.16}$$

where J_i is a job on MD i, $\sum_{j \in J_i}$ is a total data of all the jobs on MD i. The motivation here was to bias the algorithm to preferentially offload the larger job sizes, which would be most advantageous to reduce completion time on the faster server processors.

4.5.1.2 Offloading based on a fixed probability ("$O^{\mathbf{DFP}}$")

Each job has the same probability of being offloaded regardless of data size; in our experiments, we used a fixed probability of $P_j^i = 0.5$. The motivation here was to test how giving each job the same possibility would impact on the total tasks offloaded by the algorithm.

4.5.1.3 Offloading based on a known probability distribution ("$O^{\mathbf{DPD}}$")

In this policy, the probability of offloading is kept fixed for each job on the individually mobile device. The probability may be obtained by pre-solving job allocation solution of the local sub-problem on each mobile device. A further weighting is added that includes the number of MDs and MECs a system to influence the probability. The main aim of this policy is that a mobile device may have a preference for offloading a certain number of jobs to the MEC server. The resulting probability can be derived as follows:

$$P_j^i = \frac{O_i^J}{T_i^J} \times \frac{T^{\text{MEC}}}{T^{\text{MDs}}} \tag{4.17}$$

where O_i^J is the total number of jobs offloaded, T_i^J is the total number of jobs on the MD i, respectively. T^{MEC} denotes the total number of MECs and $T^{\text{MD}s}$ is the total number of MDs.

In the case when the number of MDs is less than the number of MEC servers, the formula in Equation (4.17) is simplified as follows:

$$P_j^i = \frac{O_i^J}{T_i^J} \tag{4.18}$$

4.5.2 MEC allocation of offloaded jobs

4.5.2.1 Random allocation of MEC ("M^{R}")

After a job offload decision is made, the job is allocated to a random MEC server $\{c = 1,\ldots,n\}$ in the network without knowledge of how busy the MEC is. All MECs are equally likely to be selected.

4.5.2.2 Offloading based on the job size ("M^{J}")

In this offloading policy, the jobs are assigned to MECs based on their current loading. This policy ensures that the next offloading job is assigned to the MEC that has the least loading. Mathematically, the offloading is achieved using the following equation:

$$MEC^{\mathrm{L,Offload}} = \min\{L_1, L_2, \ldots, L_{\mathrm{n}}\} \tag{4.19}$$

where L_c is the loading on MEC c, and is defined as follows:

$$L_c = \sum_{j \in J} X_{j,c} u_{j,c}$$

4.5.2.3 Minimum MEC computational time ("M^{T}")

After a job offload decision is made, the job is allocated to the MEC server, which is calculated to complete job processing quickest. This policy requires knowledge of a MEC processing capability to calculate the minimum computational time.

$$MEC^{\mathrm{T,Offload}} = \min\{T_1, T_2, \ldots, T_{\mathrm{n}}\} \tag{4.20}$$

where T_c is the computational time on MEC c, and is defined as follows:

$$T_c = X_c^{\mathrm{MEC}} \times (\frac{1}{\beta_c \lambda_c})$$

Table 4.8: Parameters Selected for Experimentation (Case 1)

Entity	Parameter	Value	Unit
Jobs Size	X^{MD}	2 - 9	MB
MD1	α_1	3.60×10^9	IPS
MEC1	β_1	1.40×10^{11}	IPS
MEC2	β_2	1.40×10^{11}	IPS
MEC3	β_3	3.68×10^{10}	IPS
	MD - MEC 1	15	Mbps
Network	MD - MEC 2	28	Mbps
	MD - MEC 3	25	Mbps

4.6 Numerical Testing of Heuristic Algorithms for Computation Offloading

In this section, the performance of the heuristics under systems consisting of various numbers of MDs, jobs and MECs is investigated. This explored the scalability of the algorithms and how the performance of methods for assigning the probabilities of offloading individual jobs varied with increasing job numbers and total data sizes. A value of (2.27×10^{-3}) instructions per bit again [105] was again chosen for the numerical simulations. Each heuristic algorithm was allowed 100 iterations to reach a least-time solution for comparison with CPLEX solutions for the individual cases considered below.

4.6.1 Offloading from a single MD (20 Jobs)

The parameters for this simulation are included in Table 4.8. Fig. 4.6 shows combined results from 20 jobs on the single MD in the form of a cumulative distribution; the least-time solutions are normalised, i.e. expressed relative to the optimal solution from CPLEX. The algorithm was assumed to be run on the MD or (on a one-to-one basis) server-side.

Offloading based on a fixed probability ("O^{DFP}") and offloading based on probability distribution ("O^{DPD}") with three different mechanisms for server allocation (random ("M^{R}"), job size ("M^{J}") or minimum remaining computational time ("M^{T}") performed far better than calculating the probability for job offloading based on data file size ("O^{DD}"). The best version of the algorithm with offloading based on probability distribution with minimum remaining MEC computation time reached the least-time schedules within 1% of the optimum time as identified by linear programming. The worst least-time solutions were all reached by offloading based on data file size ("O^{DD}"), from 153% to 157% longer than the CPLEX optimum time, 10.7 (s) (Table 4.9).

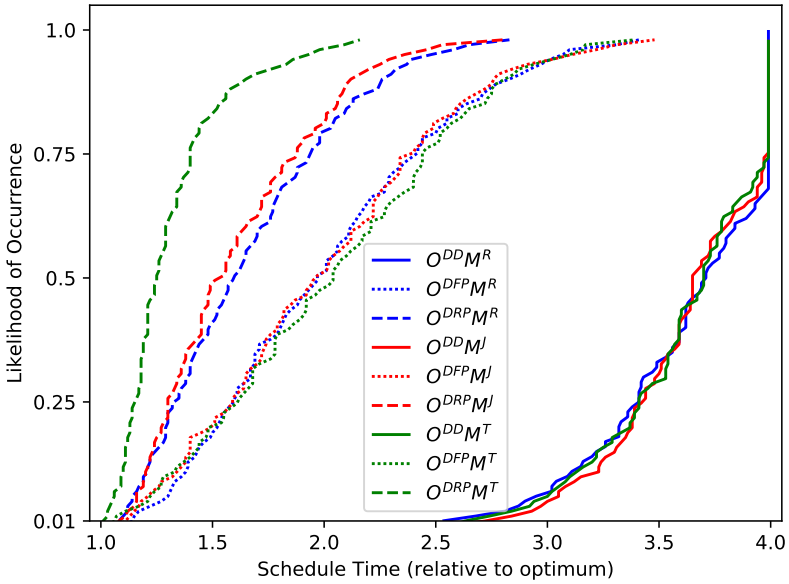


Figure 4.6: Cumulative Distribution Function graphs for heuristic algorithm outputs of schedule times for Case 1 (1 MD, 3 MEC servers, 20 jobs).

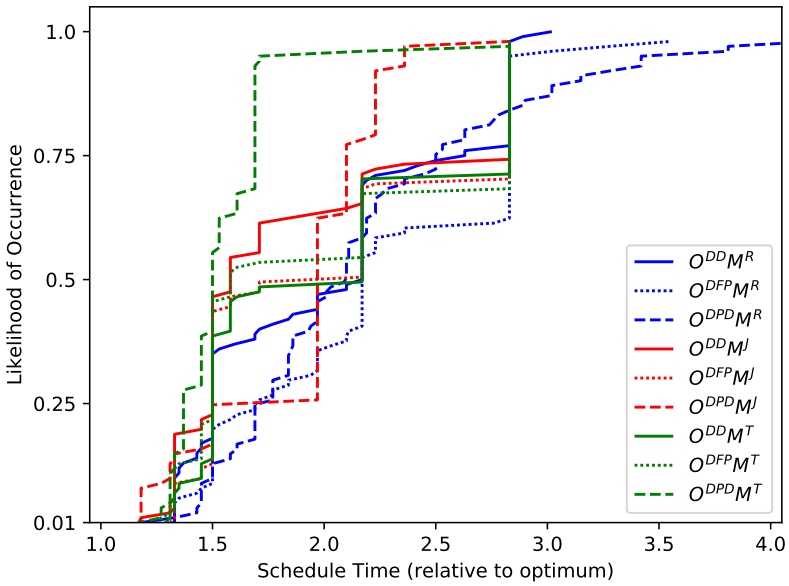


Figure 4.7: Cumulative Distribution Function graphs for heuristic algorithm outputs of schedule times for Case 2 (4 MDs, 3 MEC servers, 8 jobs).

Table 4.9: Least-time Schedules Identified by Heuristic Algorithms for 1 MD Offloading up to 20 Jobs to 3 MEC Servers

Algorithm	Least-Time Schedule (s)	Least-Time Schedule (normalised to CPLEX)
$O^{\mathrm{DPD}}/M^{\mathrm{T}}$	10.7	1.00
$O^{\mathrm{DPD}}/M^{\mathrm{J}}$	11.0	1.02
$O^{\mathrm{DFP}}/M^{\mathrm{T}}$	11.0	1.03
$O^{\mathrm{DFP}}/M^{\mathrm{J}}$	11.5	1.07
$O^{\mathrm{DPD}}/M^{\mathrm{J}}$	11.7	1.08
$O^{\mathrm{DFP}}/M^{\mathrm{R}}$	11.9	1.11
$O^{\mathrm{DD}}/M^{\mathrm{R}}$	27.9	2.53
$O^{\mathrm{DD}}/M^{\mathrm{J}}$	27.9	2.54
$O^{\mathrm{DD}}/M^{\mathrm{T}}$	28.1	2.57

Table 4.10: Parameters Selected for Experimentation (Case 2)

Entity	Parameter	Value	Unit
Jobs	X^{MD}	2 - 9	MB
MDs	α_{1-4}	3.60×10^{9}	IPS
MEC1	β_1	1.40×10^{11}	IPS
MEC2	β_2	1.40×10^{11}	IPS
MEC3	β_3	3.68×10^{10}	IPS
Network	MDs - MEC 1	15	Mbps
	MDs - MEC 2	25	Mbps
	MDs - MEC 3	28	Mbps

4.6.2 Offloading from multiple MDs (8 jobs)

The parameters for this simulation are included in Table 4.10 with a small number of jobs distributed across 4 MDs, offloading based on a fixed probability ("O^{DFP}"), offloading based on probability distribution ("O^{DPD}") and offloading based on data file size ("O^{DD}") could all reach very similar least-time solutions: 1.17 × CPLEX optimum ("O^{DPD}") and 1.18 × CPLEX optimum ("O^{DFP} and O^{DD}"); the worst least-time solution was 33% longer than the CPLEX optimum time, 6.2 s (Table 4.11). The CDF plots (Figure 4.7) showed the superiority of ("O^{DPD}") combined with minimum remaining computational time (M^{T}).

4.6.3 Offloading from multiple MDs (72 jobs)

The parameters for this network simulation are included in Table 4.12. In this simulation, 10 MDs attempted to offload a total of 72 jobs to 3 MEC servers. Fig. 4.8 shows that the three different approaches in the algorithms yielded different best schedule times but that offloading based on probability distribution ("O^{DPD}") with minimum remaining MEC computation time

Table 4.11: Least-time Schedules Identified by Heuristic Algorithms (Case 2) for 4 MDs Offloading up to 8 Jobs to 3 MEC Servers

Algorithm	Least-Time Schedule (s)	Least-Time Schedule (normalised to CPLEX)
$O^{\mathrm{DPD}}/M^{\mathrm{T}}$	7.2	1.17
$O^{\mathrm{DPD}}/M^{\mathrm{J}}$	7.2	1.17
$O^{\mathrm{DFP}}/M^{\mathrm{T}}$	7.3	1.18
$O^{\mathrm{DFP}}/M\mathrm{J}$	7.3	1.18
$O^{\mathrm{DD}}/M^{\mathrm{J}}$	7.3	1.18
$O^{\mathrm{DD}}/M^{\mathrm{R}}$	7.3	1.18
$O^{\mathrm{DFP}}/M^{\mathrm{R}}$	7.4	1.21
$O^{\mathrm{DD}}/M^{\mathrm{T}}$	7.8	1.27
$O^{\mathrm{DPD}}/M^{\mathrm{R}}$	8.2	1.33

Table 4.12: Parameters Selected for Experimentation (Case 3)

Entity	Parameter	Value	Unit
Jobs Size	X^{MD}	1 - 7	MB
MD	α	3.60×10^{9}	IPS
MEC1	β_1	1.40×10^{11}	IPS
MEC2	β_2	3.68×10^{10}	IPS
MEC3	β_3	1.40×10^{11}	IPS
Network	MDs - MEC 1	15	Mbps
	MDs - MEC 2	28	Mbps
	MDs - MEC 3	25	Mbps

("M^{T}") could reach a schedule time only 13% greater than the optimum value as identified by linear programming. The variation in performance between the nine versions of the algorithm was less pronounced than in the scenario of case 1 and all schedules were within 55% longer times than the optimum as deduced by linear programming.

With the 9-fold larger number of jobs compared to Section 4.6.2, the variation in least-time schedule solutions found by the different forms of the heuristic algorithm was greater with the worst least-time solution 54% longer than the CPLEX optimum time, 15.4 s (Table 4.13).

4.6.4 Offloading from multiple MDs (115 Jobs)

The parameters for this larger network simulation are included in Table 4.14. In this simulation, 13 MDs attempted to offload a total of 115 jobs to 3 MEC servers. Fig. 4.9 shows that calculating the probability for job offloading based on data file size (O^{DD}) was much inferior to offloading based on a fixed probability ("O^{DFP}") and offloading based on probability distribution ("O^{DPD}"). Offloading based on probability distribution ("O^{DPD}") with minimum remaining MEC computation time ("M^{T}") could reach a schedule time within 20% of the optimum value as identified by

Table 4.13: Least-time Schedules Identified by Heuristic Algorithms for (Case 3) 10 MDs Offloading up to 72 Jobs to 3 MEC Servers

Algorithm	Least-Time Schedule (s)	Least-Time Schedule (normalised to CPLEX)
$O^{\mathrm{DPD}}/M^{\mathrm{T}}$	17.5	1.13
$O^{\mathrm{DFP}}/M^{\mathrm{T}}$	18.6	1.21
$O^{\mathrm{DPD}}/M^{\mathrm{R}}$	19.5	1.27
$O^{\mathrm{DD}}/M^{\mathrm{R}}$	19.8	1.29
$O^{\mathrm{DFP}}/M^{\mathrm{J}}$	21.0	1.37
$O^{\mathrm{DPD}}/M^{\mathrm{J}}$	22.6	1.47
$O^{\mathrm{DD}}/M^{\mathrm{T}}$	22.6	1.47
$O^{\mathrm{DD}}/M^{\mathrm{J}}$	23.6	1.54
$O^{\mathrm{DFP}}/M^{\mathrm{R}}$	23.6	1.54

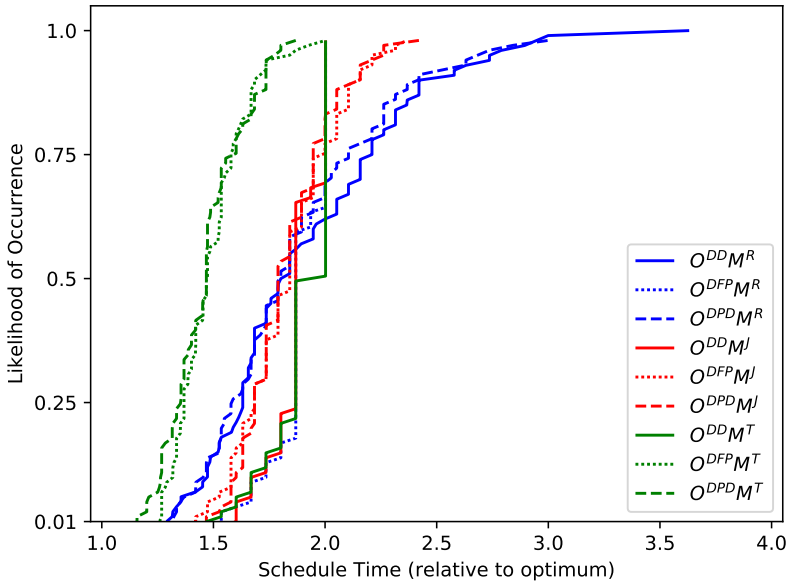


Figure 4.8: Cumulative Distribution Function graphs for heuristic algorithm outputs of schedule times for Case 3 (10 MDs, 3 MEC servers, 72 jobs).

linear programming.

With the largest number of jobs considered, the variation in least-time schedule solutions found by the different forms of the heuristic algorithm was again greater with the worst least-time solution 160% longer than the CPLEX optimum time, 25.7 s (Table 4.16).

Table 4.14: Parameters Selected for Experimentation (Case 4)

Entity	Parameter	Value	Unit
Jobs Size	X^{MD}	1 - 6	MB
MD	α_i	3.60×10^9	IPS
MEC1	β_1	1.40×10^{11}	IPS
MEC2	β_2	3.68×10^{10}	IPS
MEC3	β_3	3.68×10^{10}	IPS
Network	MDs - MEC 1	15	Mbps
	MDs - MEC 2	28	Mbps
	MDs - MEC 3	25	Mbps

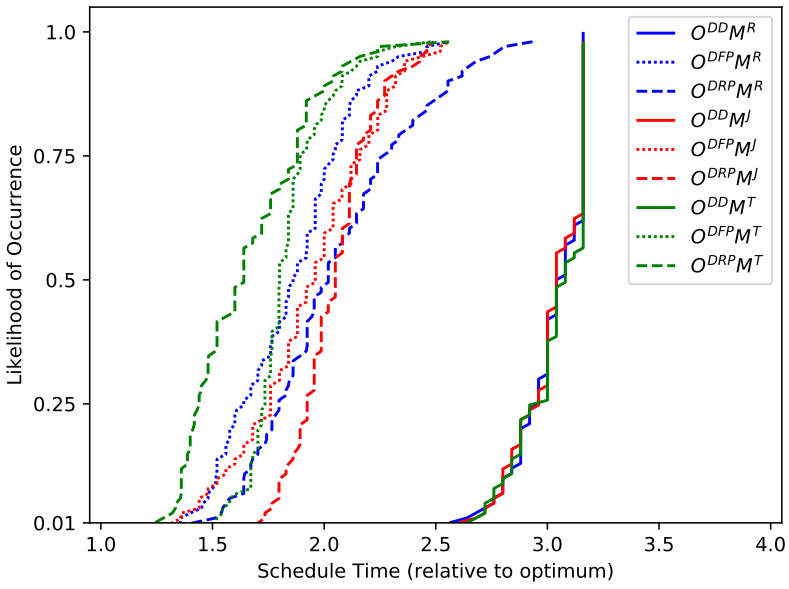


Figure 4.9: Cumulative Distribution Function graphs for heuristic algorithm outputs of schedule times for Case 4 (13 MDs, 3 MEC servers, 115 jobs).

Table 4.15: Least-time Schedules Identified by Heuristic Algorithms Case 4 for 13 MDs Offloading up to 115 Jobs to 3 MEC Servers

Algorithm	Least-Time Schedule (s)	Least-Time Schedule (relative to CPLEX)
$O^{\mathrm{DPD}}/M^{\mathrm{T}}$	30.3	1.18
$O^{\mathrm{DFP}}/M^{\mathrm{R}}$	30.8	1.20
$O^{\mathrm{DD}}/M^{\mathrm{R}}$	31.8	1.24
$O^{\mathrm{DD}}/M^{\mathrm{T}}$	34.2	1.33
$O^{\mathrm{DPD}}/M^{\mathrm{R}}$	38.0	1.48
$O^{\mathrm{DFP}}/M^{\mathrm{T}}$	43.7	1.70
$O^{\mathrm{DD}}/M^{\mathrm{J}}$	63.6	2.48
$O^{\mathrm{DPD}}/M^{\mathrm{J}}$	65.7	2.56
$O^{\mathrm{DFP}}/M^{\mathrm{J}}$	66.7	2.60

Table 4.16: Analysis of a highly heterogeneous MEC network. The diversity in processing speeds of all MDs and MECs were introduced to examine the performance of our heuristic algorithm.

Parameter	Time (s)
CPLEX optimum	33.8
O^{DFP}/M^{T} Least-Time Schedule	40.4
SD	0.59
CI 95%	0.12
CI 99%	0.15

Table 4.17: Statistics for scheduling identify by heuristic algorithms for case 4

Heuristic Algorithms	CPLEX Optimum Schedule Time (s)	Mean Least Schedule Time (s)	95% Confidence Interval (s)	99% Confidence Interval (s)	Least-time: Optimum Ratio
O^{DFP}/M^{T}	25.20	30.75	±1.10	±1.45	1.22
O^{DRP}/M^{T}	25.20	31.33	±2.50	±3.25	1.24
O^{DRP}/M^{R}	25.20	34.80	±0.99	±1.30	1.38
O^{DFP}/M^{R}	25.20	34.93	±1.70	±2.23	1.39
O^{DRP}/M^{J}	25.20	35.59	±1.71	±2.23	1.41
O^{DRP}/M^{T}	25.20	35.64	±2.30	±3.03	1.41
O^{DD}/M^{T}	25.20	64.33	±3.73	±4.91	2.55
O^{DD}/M^{J}	25.20	66.38	±1.77	±2.33	2.63
O^{DD}/M^{R}	25.20	66.38	±2.42	±3.18	2.63

4.6.5 Analysis of offloading in a highly heterogeneous MEC network

Case 4 (Section 4.6.4) was modified to generate a greater degree of heterogeneity in the MEC network. The results of this experiment are presented in Table 4.16), and the method of experiment is described as follow:

- All the 13 MDs had different processor speeds that covered a 25-fold range;
- The MEC servers had processor speeds that covered a 7.6-fold range;
- The link speeds to the servers had a broader (3.5-fold) range of values;
- The maximum job size was increased to 10 MB.

The best-performing heuristic algorithm (O^{DFP}/M^{T}) was used to identify near-optimum least-schedule times and was, as previously, compared with CPLEX. The results from three separate runs with the algorithm are contained in Table 4.17.

The heuristic algorithm achieved a least-time schedule within 20% (mean value 19.4%) of the CPLEX optimum. The CPLEX value was clearly outside the 95% and 99% confidence intervals calculated from the heuristic algorithm data.

The relative performances of CPLEX and the heuristic algorithm were very similar to results obtained with a much narrower range of parameter values in Case 4 (Section 4.6.4).

Examination of the outcomes of the CPLEX optimum solution showed that no jobs were offloaded from the 5 MDs with the fastest on-board processors. This illustrates a potential practical difficulty with central resource allocation in a MEC network, i.e., that faster-processor MDs may "subsidise" slower-processor MDs because jobs will be preferentially from slower-processor MDs who will contain the makespan schedule times. Service Level agreements may require, therefore, at least one job to be offloaded from all MDs simultaneously requesting offloading.

4.6.6 Detailed Comparison of Heuristic and CPLEX Outcomes

4.6.6.1 Total jobs and job sizes offloaded

Since optimal-time solutions are associated with parallel processing, offloading all jobs from a MD incurs a time penalty (as discussed in Sections 4.3.1-4.3.2). The longer schedule times identified by the best heuristic approach were longer than times identified by CPLEX and, therefore, an analysis was made of how the two procedures differed in the total numbers of jobs offloaded and the total data sizes of the associated files (Table 4.16).

Compared with CPLEX optima, the best-performing heuristic algorithm offloaded both more jobs and a greater total data size. For CPLEX, the mean job size offloaded was 3.59 MB; for algorithm ("O^{DPD}, M^{T}"), the mean job size offloaded was 2.82 MB. As demonstrated in

Table 4.18: Comparison between CPLEX and the Best Performing Heuristic Algorithm for Job Numbers and Total Data Offloaded

MDs	MECs	MD jobs	CPLEX jobs offloaded	CPLEX data offloaded (MB)	$O^{\mathrm{DPD}}/M^{\mathrm{T}}$ jobs offloaded	$O^{\mathrm{DPD}}/M^{\mathrm{T}}$ jobs offloaded (MB)
1	2	20	11	39	15	42
10	3	72	24	74	36	111
13	3	115	29	120	57	147

Sections 4.3.1-4.3.2 and with relatively small numbers of schedule options, offloading all the jobs from the MD incurred a time penalty, i.e. the schedule times were longer than optima identified by CPLEX and verified by direct calculation. In the scenarios summarized in Table 4.16, the heuristic algorithm identified longer least-time schedules than CPLEX but this allowed those least-time solutions to include more offloaded jobs. Although mean job data sizes decreased by approximately 21%, the increase in job numbers offloaded more than compensated for this effect.

4.6.6.2 Statistical analysis of heuristic algorithm performance

Statistical data for the performance of the heuristic algorithms in Case 4 are collected in Table 4.16. Each of the 9 algorithms was run three times, each with 100 iterations. From the results, means and confidence intervals were calculated.

The outcomes show that even with 99% confidence intervals the differences between the increased least schedule times and the optimum were all considerably greater than the confidence intervals, ranging from 1.9-fold for O^{DRP}/M^{T} to 15.7-fold for O^{DD}/M^{J}. This confirms that the algorithms can identify reproducible least-time schedules within 100 iterations.

The effect of increasing the number of iterations on the identified least-time schedule was carried out with the data for the three worst-performing algorithms (a data set which contained the largest 95% and 99% confidence intervals). Increasing the number of iterations from 30 to 40 clearly reduced the manual time identified but the trend continued (although much more slowly) at up to 10 iterations. The variability (as indicated by the standard deviations) did not become smaller as the number of iterations increased, indicating that each iteration was random event and was no influenced by previous iterations . Incorporation of Artificial Intelligence or Machine Learning into the algorithm approach could guide the process to more rapidly identify solutions closer to the optimum.

The effect of increasing the number of iterations on the identified least-time schedule was carried out with the data for the three worst-performing algorithms (a data set which contained

the largest 95% and 99% confidence intervals, Table 4.19). Increasing the number of iterations from 30 to 40 clearly reduced the manual time identified but the trend continued (although much more slowly) at up to 100 iterations.

The variability (as indicated by the standard deviations) did not become smaller as the number of iterations increased, indicating that each iteration was random event and was not influenced by previous iterations.

Incorporation of Artificial Intelligence or Machine Learning into the algorithm approach could guide the process to more rapidly identify solutions closer to the optimum.

Table 4.19: Effect of increasing the number of algorithm iterations on the numerical value of the least-schedule time using O^{DD}/M^{T}, O^{DD}/M^{J} and O^{DD}/M^{R} in Case 4.

Iterations	Least-Time Schedule (s)	Standard Deviation (s)
20	68.89	3.88
30	68.66	3.80
40	66.27	3.96
60	66.04	3.89
80	66.04	3.76
100	65.70	3.76

4.6.6.3 Program run times

As expected, the run time for the CPLEX program increased greatly with increasing number of schedules options (Table 4.20). In contrast, the run time for the best-performing algorithm ("O^{DPD}, M^{T}") were much shorter and the discrepancy between the CPLEX and ("O^{DPD}, M^{T}"), run times increased greatly with increasing number of schedules options. With the largest numbers of MDs and jobs, the heuristic algorithm had a run time only 0.2% that of CPLEX (Table 4.20).

Table 4.20: Comparison between CPLEX and the Best Performing Heuristic Algorithm for Run Times

MDs	MECs	MD jobs	CPLEX run time (s)	$O^{\mathrm{DPD}}/M^{\mathrm{T}}$ run time (s)	$O^{\mathrm{DPD}}/M^{\mathrm{T}}$ (%) of CPLEX
1	2	20	1.23	0.16	13.0
10	3	72	3.36	0.26	7.7
13	3	115	342	0.60	0.2

4.7 Analysis of an Impaired MEC Network

4.7.1 Case 4: 13 MDs, 3 MEC servers, 115 jobs

The analyses presented in Sections 4.4-4.6 assumed a fully functional MEC network with no problems, i.e., very low server CPU workloads, accessibility to all servers and maximum link speeds (15-25 Mbps). The best-performing heuristic algorithm, O^{DFP}/M^{T} (Table 4.17), was subsequently used in three scenarios where Case 4 was modified to impair the functioning of the network:

1. Only two of the three servers were accessible, indicative of link unreliability
2. The link speeds were reduced by 50%, indicative of network congestion
3. The server CPU workloads were high, indicative of server overload (95% server MEC #1, 97% server MEC #2, and 99% server MEC #3).

The results are collected in Table 4.20. Three separate runs with the algorithm were used to set a statistical baseline. Compared with the fully functional network:

- Link unreliability increased the identified minimum schedule time and decreased the total number of jobs offloaded and the total data size of jobs offloaded outside the 99% confidence interval but the median offloaded job size was unaffected.
- Network congestion increased the identified minimum schedule time and decreased the total number of jobs offloaded and the total data size of jobs offloaded outside the 99% confidence interval but the mean offloaded job size was unaffected.
- Server overload increased the identified minimum schedule time and decreased the total number of jobs offloaded, the total data size of jobs offloaded and the mean offloaded job size outside the 99% confidence interval.

4.7.2 1 MD, 3 MEC servers, 5 jobs

This much smaller scenario was run (from the MD side assuming an on-board optimisation program) to investigate simultaneous variation in MD and MEC server workloads and in link speeds but with all servers accessible:

- MD CPU workload randomly varied in the range 20-80%
- Server CPU workloads randomly varied in the range 80-99%
- Link speeds random varied from 0-100% of the baseline values (15 Mbps for MEC #1, 25 Mbps for MEC #2 and 28 Mbps for MEC #3).

Table 4.21: The impact of component outages and CPU workload on job offloading and computational times.

Scenarios	Minimum Time (s)	Jobs Offloaded	Data Offloaded (MB)	Mean Offload (MB)
Full Network	25.7	33	129	3.9
Outage of MEC#1	30.8	29	113	3.9
Outage of MEC#2	31.8	30	105	3.5
Outage of MEC#3	30.8	25	96	3.8
50% link speed:	33.9	23	90	3.9
95-99% CPU workload	50.1	16	40	2.5

The results and their statistical analysis are contained in Tables 4.22 and 4.23.

In the 49 randomly chosen parameters, all five jobs were offloaded in four instances, all with high MD CPU workloads compared with the baseline (MD CPU 20%). The median number of jobs offloaded, however, remained unchanged (at 4) despite minimum schedule times being up to four-times higher than the baseline.

The results demonstrated that parallel processing was an important feature of the offloading process, i.e., the MD processor was involved in most optimum schedules in parallel with the network servers. This gives a stability to the offloading process in terms of jobs offloaded despite serious impairments to the MEC network's functioning.

4.8 Discussion of Numerical Result Outcomes

This Chapter was concerned with multiple jobs to be offloaded from single or multiple MDs in heterogeneous MEC networks and addressed Research Questions 7 and 8 (Chapter 1, Section 1.3.2).

With a single MD and relatively small numbers of MEC servers and jobs for offloading, the major conclusion was that the optimum schedule (verified by both linear programming and manual spreadsheet calculations) used parallel processing with the whole of the processor capabilities accessed (including the MD). This was an important conclusion because it showed that, when only total task completion time was the criterion, offloading all jobs from the MD incurred a time penalty because the processing ability of the MD was ignored.

With increasing MEC server CPU workload, the effect of network congestion became important for the optimum solution time and for the numbers of jobs offloaded. This reduced efficiency of offloading was apparent at server CPU workloads as low as 70% and reinforces the

conclusion as drawn in Chapter 3 that overloading MEC serves with excessive numbers of users will degrade offloading efficiency and the advantages of offloading.

With increasing numbers of MDs, jobs and MEC servers, the numbers of distinct offloading schedules rapidly become very large and heuristic approaches offer the only practical means of rapidly approaching least-time solutions without resorting to proprietary software. The heuristic framework presented in this Chapter found non-optimal time-based solutions for the computational offloading of tasks from one or more mobile devices to one or more MEC. In total, nine approaches were devised, which differed in how the offloading probability for an individual job was calculated and in which MEC server was then selected for offloading.

Numerical experiments showed that the solution obtained by this heuristic approach was between 1% and 20% longer than the global optimal solution obtained by linear programming with CPLEX. It was observed that the best solutions are obtained by using a known probability distribution for offloading decision ("O^{DPD}") and choosing a MEC with minimum solution time ("M^{T}"); this conclusion became better defined with increasing numbers of MDs, MEC servers and jobs to be offloaded. The reason for this best combination may reside in the second component ("M^{T}"): because a least-time schedule is being sought by the algorithm, only the ("M^{T}") option directly included a minimum time in the assessment. However, different combinations of the "O" and "M" components could closely approach the (O^{DPD})/("M^{T}") least-time solution and further work is required to conclusively identify the explanation for the relative performances of different heuristic algorithms.

More conclusive, however, was the result that the heuristic algorithm approach out-performed linear optimisation for the numbers of jobs offloaded and the total data offloaded. The explanation for this could have been that, because longer schedule times were selected, this allowed more jobs to be offloaded, i.e. the conclusion reached from analysis of much smaller numbers of possible schedules. Alternatively, the mechanisms of the algorithms activity selected larger numbers of smaller jobs. Users of MDs may prefer larger numbers of jobs to be offloaded by the MEC service.

The heuristic algorithms developed in this research can be run on individual MDs for identify least-time schedules if there is a "trust relationship" between the MEC network and the individual subscriber such that the MD can access the full knowledge required to perform the calculations (including MEC server link speed and CPU workload). If this access is not granted – for example, in an ad hoc relationship between the user of a MD and a MEC network – the decision making must be performed by a server-side program that can access the MD processor speed, the number of jobs for offloading and their job sizes. When multiple MDs attempt to offloaded simultaneously, however, a network resource allocator may be invoked to maximise the efficient (but limited) resources of the network. This latter case involved "time slicing": at time zero, a number of MDs attempt to offloaded simultaneously; the allocation procedure allocates jobs to be offloaded and does not resurvey MDs and jobs until the first batch

of offloaded jobs are completed – at that time, the allocation process can recommence with a second batch of MDs. For the network, the priority is to minimise the time any one batch of MDs requires to process jobs so that the next batch can be accommodated. The network resource allocator may, furthermore, have a maximum number of jobs and/or users at any one time to more equably share out offloading time among multiple users (all of whom will, probably, be accessing the MEC service on a subscription basis).

While it is anticipated that MEC networks will rely on multiple servers and have back-up processing power available in case of network overloads, providers of MEC services may not always be able to accurately predict total user demand and the proposed "time slicing" approach may help to avoid network overload.

In general, users of MDs with faster on-board processors will expect to offload fewer jobs if total completion time is the sole criterion. However, as link speeds increase with the further development of 5G networks, the advantages of offloading to users of MDs will significantly increase because data transfer times in 3G and 4G networks greatly exceed server processing times.

Impaired MEC networks with problems with link unreliability, network congestion or network server overload will offload fewer jobs from MDs and this will be a major practical challenge for commercial MEC network service providers. Highly heterogeneous MEC networks in which MDs have very variable on-board processor speeds can be analysed rapidly by the heuristic algorithms developed in the work for this Thesis; the use of either optimisation programs or algorithms for centralised resource management when multiple MDs attempt to offload simultaneously, however, may experience practical issues because jobs from MDs with slower on-board processors distort the offloading schedules and extension of critical parameters to include, for example, energy lifetimes in MDs may be preferable to relying solely on total processing time minimisation.

Table 4.22: Effect of random changes in CPU workload and link speed with 5 offloaded jobs

MD CPU (%)	MEC1 CPU (%)	MEC2 CPU (%)	MEC3 CPU (%)	MEC1 LS (Mbps)	MEC2 LS (Mbps)	MEC3 LS (Mbps)	Schedule	Time (s)	Relative	Jobs Offloaded
20	80	80	80	15.0	25.0	28.0	168	3.66	1.0	4
59	94	93	82	3.8	7.6	26.4	224	7.04	1.9	4
66	90	80	84	8.8	20.1	27.1	460	4.78	1.3	4
49	86	88	83	2.2	19.0	10.6	799	7.27	2.0	4
72	81	86	84	7.6	10.7	18.0	712	6.72	1.8	4
31	86	94	86	14.4	6.2	1.3	526	7.11	1.9	3
20	91	80	80	6.9	7.9	6.5	765	7.75	2.1	4
55	84	83	89	1.4	16.8	13.8	207	7.64	2.1	3
78	92	94	93	9.9	0.1	12.0	272	9.47	2.6	4
37	81	80	80	3.5	10.3	23.8	144	5.64	1.5	3
66	95	87	87	13.8	7.9	4.4	218	9.04	2.5	3
43	87	84	83	0.3	11.2	16.8	204	6.82	1.9	3
35	80	92	82	7.9	0.6	6.4	14	9.48	2.6	2
47	87	94	87	12.2	9.3	18.8	808	5.81	1.6	4
34	87	86	88	12.3	0.9	17.3	372	6.22	1.7	4
22	87	82	82	15.3	4.6	14.0	528	6.20	1.7	3
73	87	89	93	7.9	3.0	24.8	232	7.60	2.1	4
69	94	92	95	10.8	0.9	8.3	798	9.93	2.7	4
21	91	89	83	9.0	15.3	25.8	404	5.20	1.4	4
51	93	86	91	7.8	17.8	22.1	124	5.67	1.5	4
38	88	87	82	6.4	20.1	9.5	79	6.79	1.9	3
31	94	91	84	8.5	16.5	23.8	80	5.92	1.6	3
33	90	87	85	6.0	11.4	14.8	436	6.13	1.7	4
35	80	94	94	13.4	17.2	3.3	526	6.32	1.7	3
23	90	89	83	11.9	13.3	20.6	168	5.00	1.4	4
22	84	88	87	0.9	12.1	1.2	549	10.53	2.9	3
21	89	81	93	8.5	18.4	23.1	76	4.86	1.3	3
46	93	95	92	1.4	5.1	8.0	528	8.94	2.4	3
31	86	94	94	14.8	16.5	15.5	766	4.91	1.3	5
59	93	95	81	2.6	15.1	23.9	544	7.51	2.1	4
36	80	83	88	10.3	3.9	10.8	994	6.54	1.8	4
30	94	81	82	14.9	8.4	17.4	168	4.90	1.3	4
30	86	92	94	12.1	19.1	7.2	526	5.87	1.6	3
45	89	90	85	3.9	22.2	2.6	799	9.35	2.6	4
47	86	84	80	2.0	2.2	24.9	800	6.63	1.8	4
50	88	88	85	1.1	13.3	9.8	528	8.21	2.2	3
37	83	80	91	10.9	16.6	2.2	380	6.17	1.7	5
50	91	83	84	8.6	24.7	17.0	124	5.13	1.4	4
26	91	91	85	23.4	0.2	0.7	65	16.65	4.5	1
29	83	90	88	2.3	20.8	15.5	528	5.78	1.6	3
71	85	87	89	6.6	23.6	9.1	111	7.62	2.1	4
60	84	94	84	5.2	3.0	3.4	526	12.13	3.3	3
46	81	85	85	9.7	22.8	10.5	79	5.93	1.6	3
60	85	90	91	8.0	19.3	11.7	92	6.59	1.8	4
21	82	87	89	5.7	12.2	10.9	765	6.50	1.8	4
64	92	82	80	14.8	8.9	8.5	766	6.07	1.7	5
47	85	83	85	7.3	17.7	25.5	220	5.09	1.4	4
47	81	95	87	7.8	18.5	7.2	143	8.24	2.3	3
71	85	94	92	12.5	22.2	7.3	158	7.73	2.1	4
46	95	93	95	13.7	8.2	24.3	568	6.07	1.7	4
74	86	83	82	4.3	2.7	1.8	526	15.79	4.3	3

Table 4.23: Statistical analysis of random changes in CPU workload and link speed with 5 offloaded jobs

	MD CPU (%)	MEC1 CPU (%)	MEC2 CPU (%)	MEC3 CPU (%)	MEC1 LS (Mbps)	MEC2 LS (Mbps)	MEC3 LS (Mbps)	Optimum Schedule Time (s)	Relative Optimum	Jobs Offloaded
Median:	56.0	87.0	92.0	90.5	7.6	12.2	14.1	7.3	2.0	4.0
Mean:	49.1	88.4	90.5	90.6	8.2	12.9	13.7	8.0	2.2	3.7
STDEV:	18.76	5.23	5.73	5.64	4.35	7.43	7.24	2.16	0.59	0.68
95% CI:	5.25	1.46	1.60	1.58	1.22	2.08	2.03	0.60	0.17	0.19
99% CI:	6.90	1.92	2.11	2.07	1.60	2.73	2.67	0.79	0.22	0.25

CHAPTER 5

MULTI-OBJECTIVE OPTIMISATION FRAMEWORK FOR COMPUTATIONAL OFFLOADING

5.1 Introduction

In Chapter 3, the analysis of computation offloading was focused on identifying the operating conditions and parameters that gave advantages in task completion time for individual jobs on a MD and that minimized local energy usage on the MD. In Chapter 4, the analysis was extended to find heuristic least-time solutions for offloading multiple jobs from single or multiple MDs.

Several studies have claimed major savings in both task completion time and energy usage by the MD offloading to both MCC and MEC environments [115, 157, 96, 162, 61, 143]. In such studies, task completion time and local energy use were treated as independent variables in the context of finding either the shortest total computation times or minimal energy usage by the MD. In this context, an ambitious and a challenging question is: can time and energy savings be simultaneously maximized in a MEC network? A conclusion from the work presented in Chapter 4 was that least-time schedules for offloading multiple jobs preferentially uses parallel processing with both MD and MEC server processors rather than offloading all jobs, especially when link speeds are not sufficiently high; this implies a trade-off between achieving minimum task completion time and minimizing local energy usage by the MD is unavoidable. This has been described as the "divergent goals" problem [149].

Forming a link between time and energy parameters is straightforward if two weighting factors are used, the sum of the weighting factors being fixed. This is explained in more detail in Section 5.2.1. Multi-objective analysis of the offloading process simultaneously incorporated energy consumption, execution delay and the price cost of offloading in what was (in 2017) known as Mobile Edge Computing [100]. In this study, execution delay was a consequence of a

queuing process for multiple MDs and local energy consumption decreases as the probability of a job being offloaded increased. The authors of [100] aimed to find a global optimum that minimized an objective function that integrated all three parameters, each multiplied by an appropriate weighting factor. Only one set of weighting factors was used in numerical analyses in [100] and the sole evidence for such a global optimization was a minimization with respect to transmission power (expressed, presumably as relative values) of the MD; however, no explanation was given as to what factors influenced the transmission power of the MD or if this was in any way modifiable by the user of the MD.

Key Research Question: Can time, energy and other factors be combined to identify a multi-factorial optimal solution for offloading multiple jobs from a single MD or from multiple MDs in a heterogeneous MEC network? This question combines Research Questions 9 and 10 (Chapter 1, Section 1.3.2)

The main contributions made by work presented in this Chapter:

- Heuristic algorithms were developed to incorporate time and local energy use using weighting factors to achieve sub-optimal solutions close to optimal solutions provided by linear optimization using CPLEX.

- Investigation of varying weighting factors for time and energy showed that time and local energy savings could be selected by the MD user but that no global optimum solution could be identified, i.e. only trade-offs between the two factors were possible.

- Mathematical approaches were developed to incorporate time, energy and economic (price) cost factors into a tri-factorial analysis and preliminary studies were performed to identify successful combinatorial methodologies.

A part of the research findings presented in this chapter are published to the 20th IEEE International Conference on Scalable Computing and Communications: R. Singh, S. Armour, A. Khan, M. Sooriyabandara and G. Oikonomou, “Towards Multi- Criteria Heuristic Optimization for Computational Offloading in Multi-Access Edge Computing”.

The remainder of this Chapter will, firstly, use the mathematical models presented in Sections 3.5.2-3.5.4 and Section 4.2 to combine calculations of time and energy for the 81-schedule and 1024-schedule scenarios discussed in Chapter 4 and to use these computations to validate a program designed in CPLEX for linear optimization; secondly, heuristic approaches are then presented to find comparable solutions to the use of CPLEX with larger numbers of jobs offloaded from multiple MDs in heterogeneous MEC networks; thirdly, this approach is expanded to incorporate cost in a subscription MEC system to explore achievable optimisation strategies.

5.2 Basic System Model

The mathematical models presented in Sections 3.5.2-3.5.4 and in Section 4.2 and were used to perform calculations of total task completion time and local (MD) energy use for the multiple-schedule scenarios Appendix Tables A.1-A.2.

5.2.1 Overall Problem Formulation

The overall objective of our problem is to minimise the computational time and energy consumption across all the jobs on the given mobile devices.

5.2.2 Computational Time

The overall computational time of processing the jobs is given as follows:

$$T^{\text{Total}} = \max\left\{ \underbrace{\max\{T_i : i \in M\}}_{\text{Local Time}}, \underbrace{\max\{T_c : c \in C\} + \sum_{(i,c,j)} (2-\Pi) T_{i,c,j}}_{\text{MEC Processing Time}} \right\} \tag{5.1}$$

where T_i is the total jobs processing time on a MD i. There are three components in the equation. The first component is the local computation time of the jobs. The second component is the computation time of MEC servers and the third component is the transmission and reception time.

5.2.3 Computational Energy

The overall computational energy consumption of all mobile devices to process all the jobs is given as follows:

$$\begin{aligned} E^{\text{Total}} = & \underbrace{\sum_{i \in M} P_i^{MD} T_i}_{\text{Mobile energy consumption}} + \underbrace{\sum_{i \in M} P_i^{\text{idle}} \max\{0, \max\{T_c; c \in C\}\}}_{\text{Idling energy consumption}} \\ & + \underbrace{\sum_{(i,c,j)} (2-\pi) T_{i,c,j} P_i^{\text{Send,Rec}}}_{\text{Transmission and receiving energy consumption}} \end{aligned} \tag{5.2}$$

There are three components in Equation 5.2. The first component defines the energy consumption while the jobs are being processed on MEC servers and the MDs are in idle state. The second component defines the energy consumption due to transmission and receiving of data. We ignore the computational energy consumption of MEC servers. The third component is the energy consumption on the mobile side.

5.2.4 Multi-objective optimization formulation

The weighted multi-objective function is given as follows:

$$\min \ (w_t \times \frac{T^{\text{Total}}}{T_{\max}} + w_e \times \frac{E^{\text{Total}}}{E_{\max}}) \tag{5.3}$$

where w_t and w_e are the weightings on computational time and computational energy consumption, respectively; additionally, $w_t + w_e = 1$. Further, $T_{\max}$ and $E_{\max}$ are the expected worse case computational times and energy, respectively. The worst case options can be determined by solving all the jobs locally (for time) and (for energy) either when tasks are performed locally or when using the slowest MEC link and/or processor speeds, depending on the numerical values selected. We assume that the weighting factor can be adjusted according to MD users' needs or by the MEC service provider at the local cell station when a "cluster" of MD users attempt to connect simultaneously.

To correct for large numerical discrepancies in the ranges of absolute values taken by the different variables, a Bias Correction Coefficient (BCC) $\eta = \frac{E_{\max}}{T_{\max}}$ was introduced in the objective function as follows:

$$\min \ (w_t \times \frac{\eta T^{\text{Total}}}{T_{\max}} + w_e \times \frac{E^{\text{Total}}}{\eta E_{\max}}) \tag{5.4}$$

5.3 Numerical Results

The performance of the mathematical model presented in Section 5.2 is tested on two small-scale networks. In order to evaluate the performance of heuristic algorithm, exhaustive search for all possible solutions were conducted for these two networks. The performance of the proposed model on the two chosen networks is described in detail in the following subsections.

5.3.1 A MD with 4 jobs offloading on 2 MEC servers

This is the same example that was used in the previous chapter. The parameters used in this example are provided in Table 4.1.

We have tested the following three options for the objective function:

- un-normalised time and energy components in the objective function i.e. removing $T_{\max}$ and $E_{\max}$ from Equation (5.3)
- normalising time and energy components in the objective function with the worst case solution i.e. Equation (5.3)
- BCC-normalised time and energy components: the objective function is defined by Equation (5.3)

Table 5.1: Optimal schedules based on minimal Weighted Scores for 1 MD offloading up to 4 jobs to 2 MEC servers (un-normalised times and energies)

Time Weighting Factor	Energy Weighting Factor	Schedule option	Time	Energy	Weighted Score	Jobs Offloaded
0	1	3	5.70	6.41	6.41	4
0.1	0.9	3	5.70	6.41	6.34	4
0.2	0.8	3	5.70	6.41	6.27	4
0.3	0.7	3	5.70	6.41	6.20	4
0.4	0.6	19,41	3.99	7.26	5.95	3,2
0.5	0.5	6	2.85	8.13	5.49	3
0.6	0.4	6	2.85	8.13	4.96	3
0.7	0.3	6	2.85	8.13	4.43	3
0.8	0.2	6	2.85	8.13	3.91	3
0.9	0.1	6	2.85	8.13	3.38	3
1	0	6	2.85	8.13	2.85	3

Table 5.2: Optimal schedules based on minimal Weighted Scores for 1 MD offloading up to 4 jobs to 2 MEC servers (normalised times and energies)

Time Weighting Factor	Energy Weighting Factor	Schedule option	Normalised Time	Normalised Energy	Weighted Score	Jobs Offloaded
0	1	3	0.56	0.63	0.63	4
0.1	0.9	3	0.56	0.63	0.62	4
0.2	0.8	3	0.56	0.63	0.61	4
0.3	0.7	3	0.56	0.63	0.61	4
0.4	0.6	19,41	0.39	0.71	0.58	3,2
0.5	0.5	6	0.28	0.79	0.54	3
0.6	0.4	6	0.28	0.79	0.48	3
0.7	0.3	6	0.28	0.79	0.43	3
0.8	0.2	6	0.28	0.79	0.38	3
0.9	0.1	6	0.28	0.79	0.33	3
1	0	6	0.28	0.79	0.28	3

The aforementioned three objective function choices were investigated. The results are presented in Tables 5.1-5.3. The three sets of data show identical trends in Weighted Score with a decrease as the weighting factor for time increased (and the weighting factor for MD energy decreased).

As the weighting factor for time increased, the optimum time decreased proportionally more rapidly than the local energy term increased. At the least value for weighting factor for time, one extra job was offloaded but this impacted on the local energy less than the advantage of

Table 5.3: Optimal schedules based on minimal Weighted Scores for 1 MD offloading up to 4 jobs to 2 MEC servers (normalised times and energies) with BCC

Time Weighting Factor	Energy Weighting Factor	Schedule option	Normalised Time	Normalised Energy	Weighted Score	Jobs Offloaded
0	1	3	0.56	0.63	0.63	4
0.1	0.9	3	0.56	0.63	0.62	4
0.2	0.8	3	0.56	0.63	0.61	4
0.3	0.7	3	0.56	0.63	0.61	4
0.4	0.6	19,41	0.39	0.71	0.58	3,2
0.5	0.5	6	0.28	0.79	0.54	3
0.6	0.4	6	0.28	0.79	0.49	3
0.7	0.3	6	0.28	0.79	0.43	3
0.8	0.2	6	0.28	0.79	0.38	3
0.9	0.1	6	0.28	0.79	0.33	3
1	0	6	0.28	0.79	0.28	3

using parallel processing with all three processors in the network when a high weighting factor for time was applied.

The transition from schedule 3 to schedule 6 occurred when weighting factor for time reached 0.4 when two schedules gave the same Weighted Score values (to 10 decimal places) even though different numbers of jobs were offloaded. Use of CPLEX validated these solutions. With the small number of schedules involved and the overlapping numerical ranges for time and energy, unnormalised, normalised and normalised with BCC all gave the same schedule options and overall trends in Weighted Scores as weighting factors changed.

5.3.2 A MD with 5 jobs offloading on 3 MEC servers

This is the same example that was used in the previous chapter and the parameters used in this example are provided in Table 4.5.

Similarly, in this example the three options for the objective function choice were used. The results are presented in Tables 5.4-5.6. The three sets of data show similar trends in Weighted Score with a decrease as the weighting factor for time increased (and the weighting factor for MD energy decreased).

As the weighting factor for time increased, the optimum time decreased proportionally more rapidly than the local energy term increased. At the least value for weighting factor for time, one extra job was offloaded but this impacted on the local energy less than the advantage of using parallel processing with all four processors in the network when a high time weighting factor was applied. The transition from schedule 1024 occurred when the weighting factor for time reached 0.2 when three schedules gave the same Weighted Score values (to 10 decimal

Table 5.4: Optimal schedules based on minimal Weighted Scores for 1 MD offloading up to 5 jobs to 3 MEC servers (un-normalised times and energies)

Time Weighting Factor	Energy Weighting Factor	Schedule option	Normalised Time	Normalised Energy	Weighted Score	Jobs Offloaded
0.1	0.9	1024	6.69	8.30	8.21	5
0.2	0.8	688,764,1007	4.01	8.82	7.86	5, 5, 5
0.3	0.7	688,764,1007	4.01	8.82	7.38	5, 5, 5
0.4	0.6	688,764,1007	4.01	8.82	6.90	5, 5, 5
0.5	0.5	716	3.57	9.19	6.38	4
0.6	0.4	207,812,831	3.12	9.85	5.81	3, 4, 4
0.7	0.3	124	2.67	10.81	5.11	4
0.8	0.2	124	2.67	10.81	4.30	4
0.9	0.1	124	2.67	10.81	3.49	4
1.0	0.0	124	2.67	10.81	2.67	4

Table 5.5: Optimal schedules based on minimal Weighted Scores for 1 MD offloading up to 5 jobs to 3 MEC servers (normalised times and energies)

Time Weighting Factor	Energy Weighting Factor	Schedule option	Normalised Time	Normalised Energy	Weighted Score	Jobs Offloaded
0.1	0.9	1024	0.43	0.48	0.48	5
0.2	0.8	688, 764, 1007	0.26	0.51	0.46	5, 5, 5
0.3	0.7	688, 764, 1007	0.26	0.51	0.44	5, 5, 5
0.4	0.6	688, 764, 1007	0.26	0.51	0.41	5, 5, 5
0.5	0.5	716	0.23	0.53	0.38	4
0.6	0.4	207, 812, 831	0.20	0.57	0.35	3, 4, 4
0.7	0.3	124	0.17	0.63	0.31	4
0.8	0.2	124	0.17	0.63	0.26	4
0.9	0.1	124	0.17	0.63	0.22	4
1.0	0.0	124	0.17	0.63	0.17	4

places).

A second transition occurred when the weighting factor for time reached 0.5 to a single optimum schedule; this change occurred earlier, however, when the BCC was applied, at a weighting factor for time of 0.4.

Use of CPLEX validated these solutions. In both cases with relatively small numbers of schedule options, the choice of weighting factors was crucial. As the weighting factor for time

Table 5.6: Optimal schedules based on minimal Weighted Scores for 1 MD offloading up to 5 jobs to 3 MEC servers (normalised times and energies) with BCC

Time Weighting Factor	Energy Weighting Factor	Schedule option	Time	Energy	Weighted Score	Jobs Offloaded
0.1	0.9	1024	6.69	8.30	0.44	5
0.2	0.8	688,764,1007	4.01	8.82	0.42	5, 5, 5
0.3	0.7	688,764,1007	4.01	8.82	0.41	5, 5, 5
0.4	0.6	716	3.57	9.19	0.39	4
0.5	0.5	716	3.57	9.19	0.37	4
0.6	0.4	207,812,831	3.12	9.85	0.34	3, 4, 4
0.7	0.3	124	2.67	10.81	0.30	4
0.8	0.2	124	2.67	10.81	0.27	4
0.9	0.1	124	2.67	10.81	0.23	4
1.0	0.0	124	2.67	10.81	0.19	4

was increased, the Weighted Score continuously decreased but no maximum or minimum was apparent. The effect on the Weighted Score was mediated by a greater proportional change in schedule time than local energy and applying data normalisation or the BCC factor did not change this.

With the proposed definition of the objective function, the user of a MD can choose a weighting for energy and time components according to their preference. From the results, a clear trade-off is apparent between energy consumption and computational time as the weightings change.

5.4 Development of Heuristic Algorithms

As the numbers of jobs to be offloaded increase, the numbers of possible schedules rise rapidly, as discussed in Chapter 4. Heuristic algorithms were developed to incorporate time and energy factors.

The two most successful heuristic algorithms presented in [144] and Chapter 4 were extended here to include both time and MD energy factors. The generic heuristic algorithm has two stages of decision making: (a) whether to offload a job or not, (b) if a job is offloaded, which MEC to offload it to. The first decision of whether to offload or not is governed by probabilities and we have consider two options for these probabilities: offloading based on a single fixed probability ("O^{DFP}") and offloading based on a known probability distribution ("O^{DPD}"). These steps were each combined with three policies for offloading to specific MEC servers; two polices were described in [144]: offloading based on offloading based the job size ("M^{J}") and offloading based on the minimum remaining MEC computational time ("M^{T}"). Further, a new MEC offloading option is constructed on the fastest link MEC connection ("M^{W}").

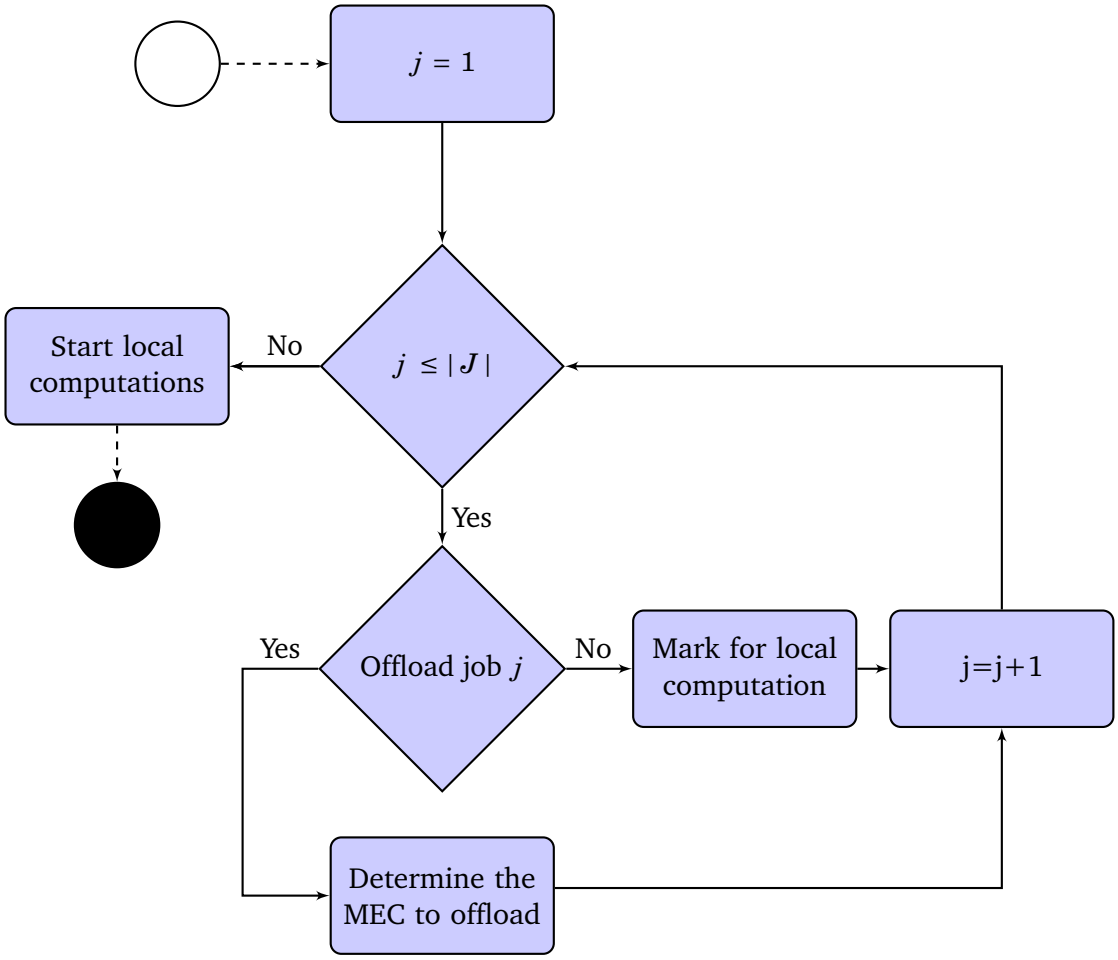


Figure 5.1: Flowchart of the heuristic algorithm for computation offloading.

The motivation is to offload a job from an individual MD to the MEC with the most agile bandwidth connection to reduce the minimum score. As was demonstrated in Chapter 3, the faster the link speed between an MD and a MEC server, the shorter is the time required for data transmission and reception and the less MD energy is expended in sending and receiving data.

Figure 5.1 is identical to Fig 4.5 presented in Chapter 4 and is included in this chapter again for convenience.

In addition to the two approaches described above and, an additional heuristic offloading approach, "(O^{GS})", is defined as follows.

1. A solution is obtained by one of the two heuristic approaches and scores for all the jobs are calculated

2. Allocation of the n^{fix} jobs with minimum scores is fixed and step 1 is repeated.

3. Step (1) and (2) are repeated until all the jobs have been allocated.

Table 5.7: Parameters used for numerical simulations in Cases 1,2 & 3

Entity	**Parameter**	**Value**	**Unit**
Jobs Size	X^{MD}	2-9 (1), 1-7 (2), 1-6 (3)	MB
MDs	α_i	3.60×10^9	IPS
MEC1	β_1	1.40×10^{11}	IPS
MEC2	β_2	1.40×10^{11} (1),	IPS
		3.68×10^{10} (2 & 3)	
MEC3	β_3	1.40×10^{11} (2)	IPS
		3.68×10^{10} (1 & 3)	
Network	MDs - MECs	15-28	Mbps

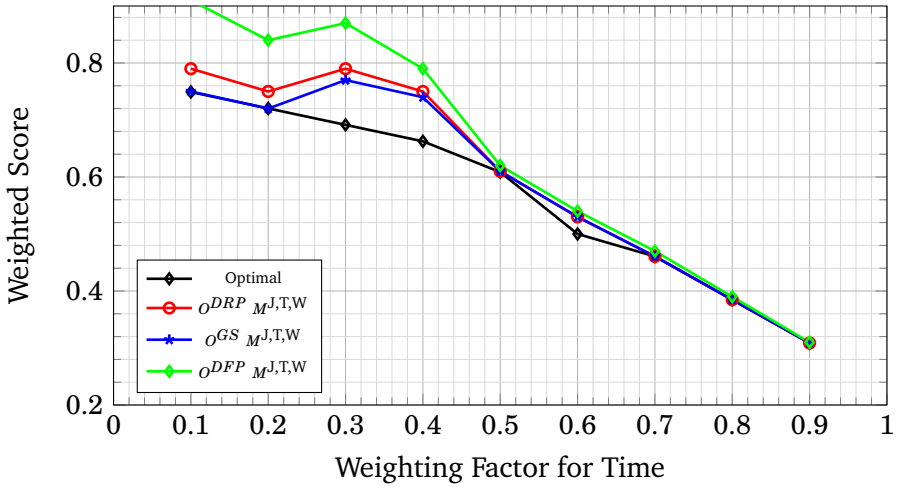


Figure 5.2: Performance of the proposed heuristic approaches in MEC network with 1MD and 20 Jobs. The solution for each heuristic approach was the best solution from 100 runs.

Each job's score is calculated based on its time and energy and divided by the local worst time and energy. This strategy is a guided search, as the score reflects the computational load of mobile devices and MEC servers.

5.4.1 Numerical Results

The numerical values for the parameters used in simulations are presented in Table 5.7 using processor speeds and power ratings from [105] and [90]. Three cases were considered: one with jobs offloaded from a single MD and two cases where multiple MDs attempted to offload; all three cases were in heterogeneous MEC networks.

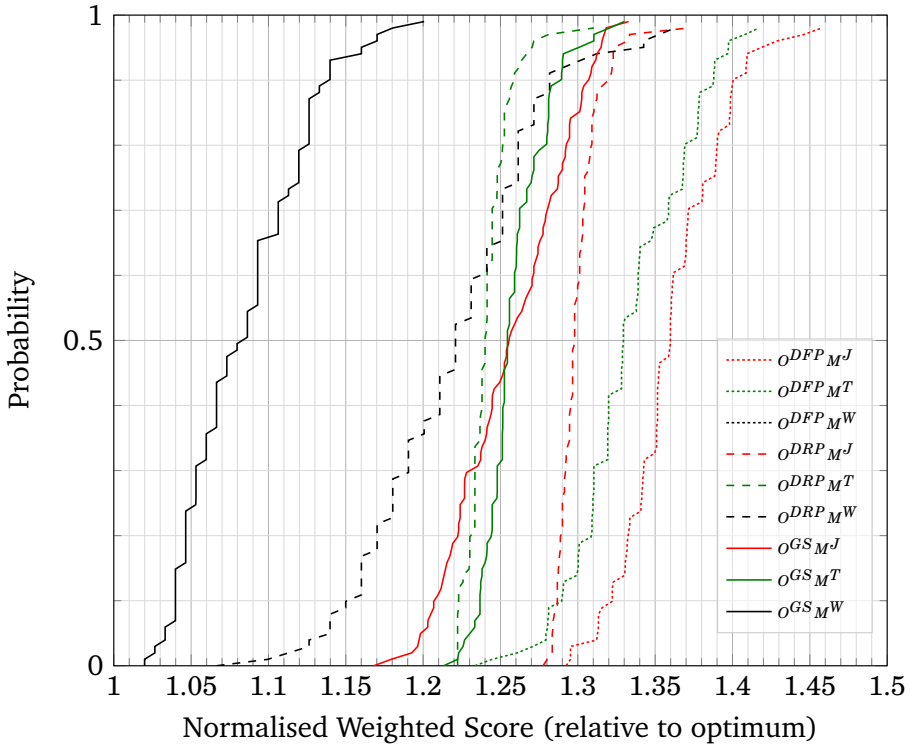


Figure 5.3: Cumulative Distribution Function graphs for heuristic algorithm outputs of schedule times and energy for Case 1 with $w_t : 0.1$ $w_e : 0.9$ (1 MD, 2 MEC servers, 20 jobs).

5.4.2 Case 1: Offloading from a single MD

In this scenario, heuristic algorithms were assumed to run on a single MD that has 20 jobs and is connected to 2 MEC servers. Figure 5.2 presents the minimum Weighted Score as a function of the weighting factor for time. Because the weighting factors for time and local energy are linked, i.e. their sum is equal to 1, as the weighting factor for the time was increased, the weighting factor for local energy. Figure 5.2 presents the optimal solution from Linear Programming and three heuristic approaches. As the time weighting was increased, all three heuristic approaches were able to closely approximate the optimal solution within 100 runs.

Cumulative Distribution Frequency plots with a low weighting factor for time (0.1) are shown in Figure 5.3. The closest approximation to the optimal value (2% greater Weighted Score) was achieved by the ("$O^{\mathrm{GS}}, M^{\mathrm{W}}$") algorithm. The ("$O^{\mathrm{DRP}}, M^{\mathrm{W}}$") algorithm was 7% greater than the optimal least Weighted Score. The ("$O^{\mathrm{DFP}}, M^{\mathrm{W}}$") algorithm gave a poor match to the optimal solution and is not included in Figure 5.3.

Cumulative Distribution Frequency plots with equal weighting factors for time and local energy are shown in Figure 5.4. Six algorithms gave close approximations to the optimal value (up to 4 % greater Weighted Score). The ("$O^{\mathrm{GS}}, M^{\mathrm{W}}$") and ("$O^{\mathrm{DRP}}, M^{\mathrm{W}}$") algorithms were only 1% greater than the optimal least Weighted Score. The ("$O^{\mathrm{DFP}}, M^{\mathrm{W}}$")algorithm again gave a

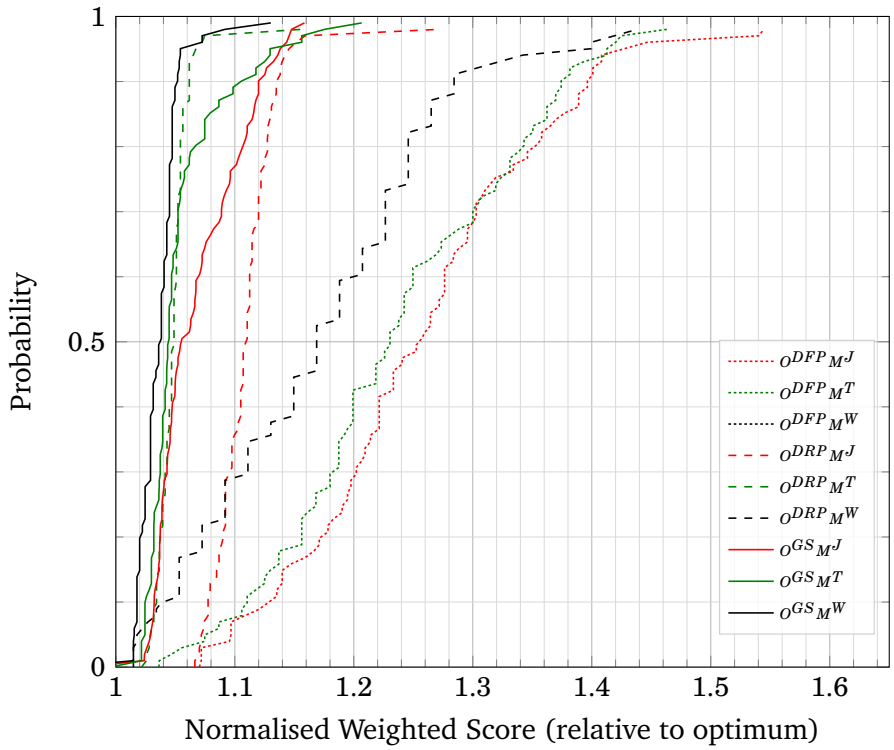


Figure 5.4: Cumulative Distribution Function graphs for heuristic algorithm outputs of schedule times and energy for Case 1 with $w_t : 0.5$ $w_e : 0.5$ (1 MD, 2 MEC servers, 20 jobs).

poor match to the optimal solution and is not included in Figure 5.5.

Cumulative Distribution Frequency plots with a high weighting factor for time (0.9) are shown in Figure 5.5. Four algorithms gave close approximations to the optimal value (up to 4% greater Weighted Score). The ("$O^{\mathrm{GS}}, M^{\mathrm{T}}$") and ("$O^{\mathrm{DRP}}, M^{\mathrm{T}}$") algorithms equalled the optimal least Weighted Score within 100 runs. The ("$O^{\mathrm{DFP}}, M^{\mathrm{W}}$") algorithm again gave a poor match to the optimal solution and is not included in Figure 5.5.

5.4.3 Case 2: Offloading from 10 MDs

In this scenario, heuristic algorithms were assumed to run on a centralized resource-allocator with a client cluster of MDs attempting to offload a maximum of 72 jobs. The minimum weighted score, normalised and using the BCC, computed by linear programming with CPLEX, again decreased continuously as the weighting factor for time increased (Figure 5.6).

Cumulative Distribution Frequency plots with a low weighting factor for time (0.1) are shown in Figure 5.7. The closest approximation to the optimal value (11% greater Weighted Score) was achieved by the ("$O^{\mathrm{GS}}, M^{\mathrm{W}}$")algorithm. Seven algorithms were within 17% higher than the optimal least Weighted Score.

Cumulative Distribution Frequency plots with equal weighting factors for time and local

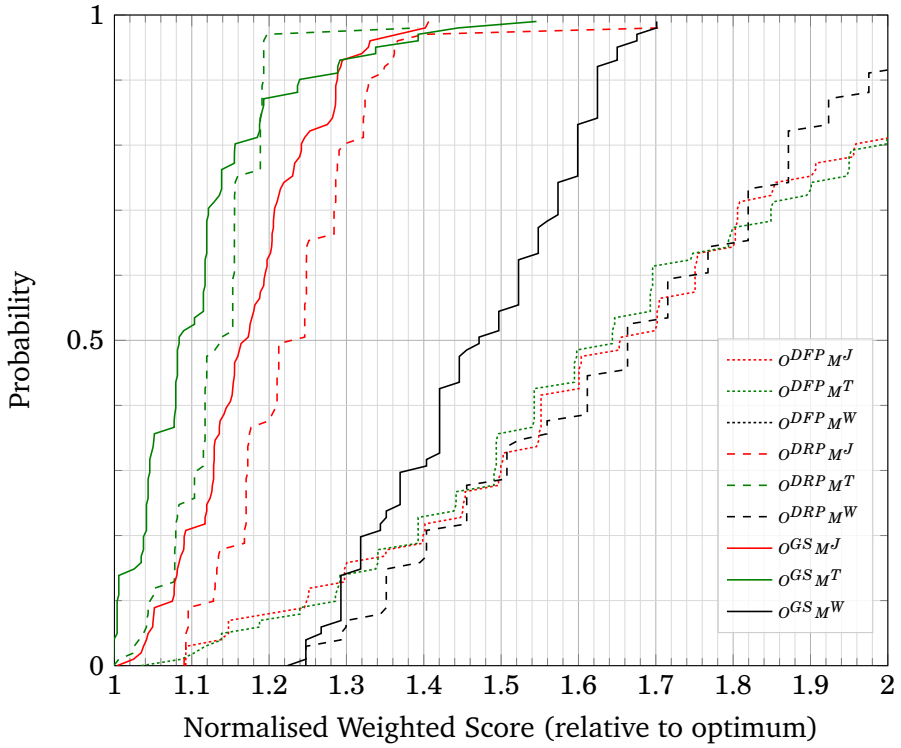


Figure 5.5: Cumulative Distribution Function graphs for heuristic algorithm outputs of schedule times and energy for Case 1 with $w_t : 0.9$ $w_e : 0.1$ (1 MD, 2 MEC servers, 20 jobs).

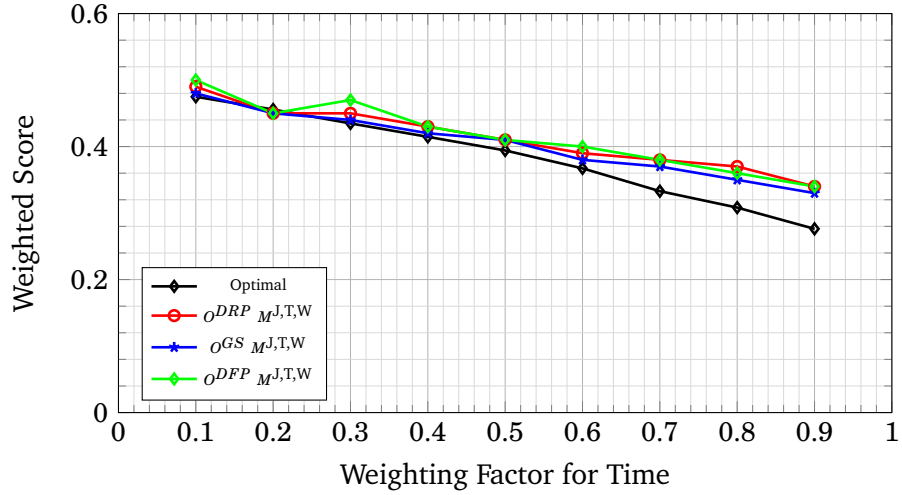


Figure 5.6: Performance of the proposed heuristic approaches in MEC network with 10MD and 72 Jobs. The solution for each heuristic approach was the best solution from 100 runs.

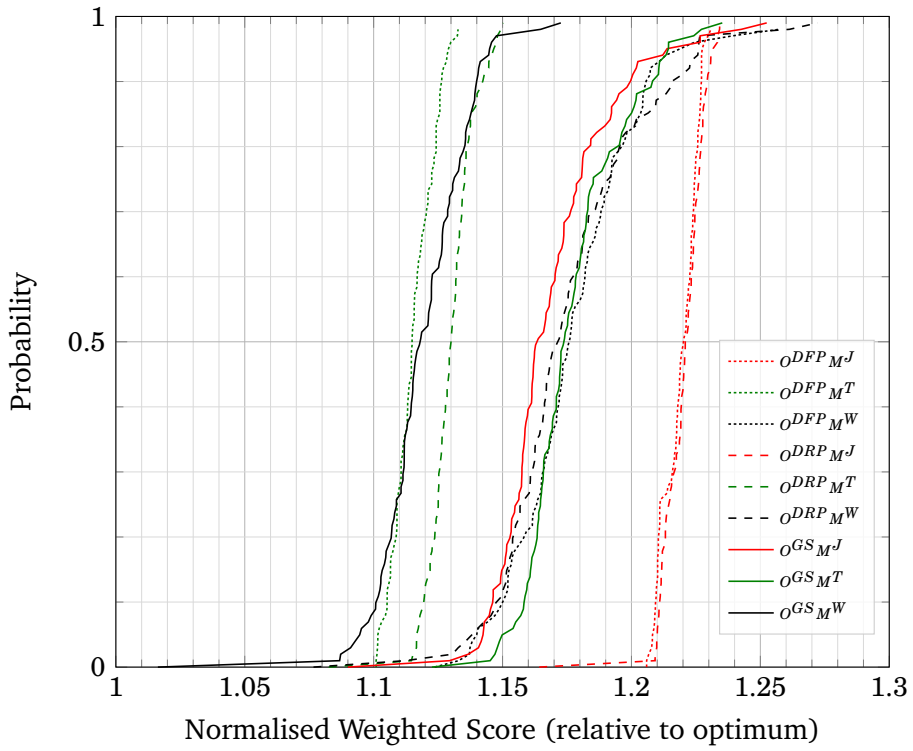


Figure 5.7: Cumulative Distribution Function graphs for heuristic algorithm outputs of schedule times and energy for Case 2 with $w_t : 0.1$ $w_e : 0.9$ (10 MDs, 3 MEC servers, 72 jobs).

energy are shown in Figure 5.8. Five algorithms gave close approximations to the optimal value (up to 8% greater Weighted Score). The ("O^{DRP},M^{W}") algorithm was only 4% greater than the optimal least Weighted Score.

Cumulative Distribution Frequency plots with a high weighting factor for time (0.9) are shown in Figure 5.9. The algorithms could not give very close approximations to the optimal value but the ("O^{DRP},M^{T}") algorithm was 22% higher the optimal least Weighted Score within 100 runs. Averaging least Weighted Scores over all 9 algorithms, the O^{GS} group were the best, followed by ("O^{DRP} and then O^{DFP})".

5.4.4 Case 3: Offloading from 13 MDs

In this scenario, heuristic algorithms were assumed to run on a centralized resource-allocator with a client cluster of MDs that have 115 jobs for processing. The minimum weighted score, normalised and using the BCC, computed by linear programming with CPLEX, again decreased continuously as the weighting factor for time increased (Figure 5.10).

At the lowest weighting factor for time, the closest match to the minimum weighted score was that generated by ("O^{GS}, M^{W}") (+7%). At the highest weighting factor for time, the algorithms could not so closely approach the minimum weighted score; the closest match was

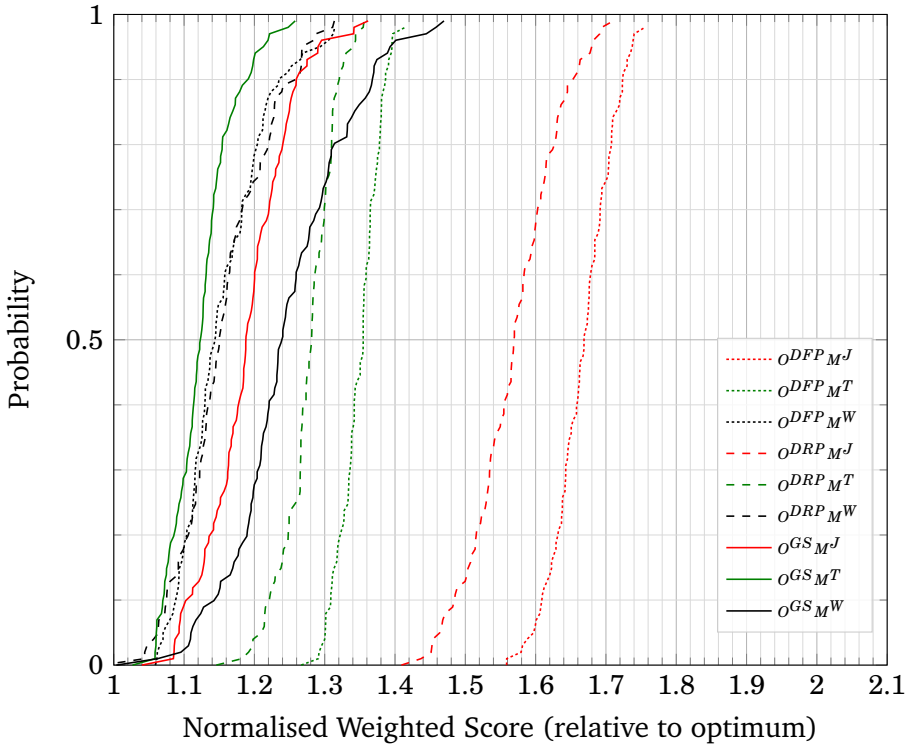


Figure 5.8: Cumulative Distribution Function graphs for heuristic algorithm outputs of schedule times and energy for Case 2 with $w_t : 0.5$ $w_e : 0.5$ (10 MDs, 3 MEC servers, 72 jobs).

that generated by ("O^{DFP}, M^{W}") (+26%).

Cumulative Distribution Frequency plots with a low weighting factor for time (0.1) are shown in Figure 5.11. The closest approximation to the optimal value (7% greater Weighted Score) was achieved by the ("O^{GS},M^{W}") algorithm. The ("O^{DRP},M^{W}") algorithm was 10% higher than the optimal least Weighted Score.

Cumulative Distribution Frequency plots with equal weighting factors for time and local energy are shown in Figure 5.12. All 3 algorithms based on ("O^{GS}") gave close approximations to the optimal value (1-2% greater Weighted Score).

Cumulative Distribution Frequency plots with a high weighting factor for time (0.9) are shown in Figure 5.13. The algorithms could not give very close approximations to the optimal value but the ("O^{GS},M^{J}") algorithm was 26% higher the optimal least Weighted Score within 100 runs. The next best algorithm was ("O^{GS}, M^{T}") (+29%).

5.5 Discussion of Results with Heuristic Algorithms

In general, the new algorithm ("O^{GS}") performed better than algorithms with job probabilities in achieving offloading algorithms in achieving least values for Weighted Scores which were closest to the optimal values identified by CPLEX.

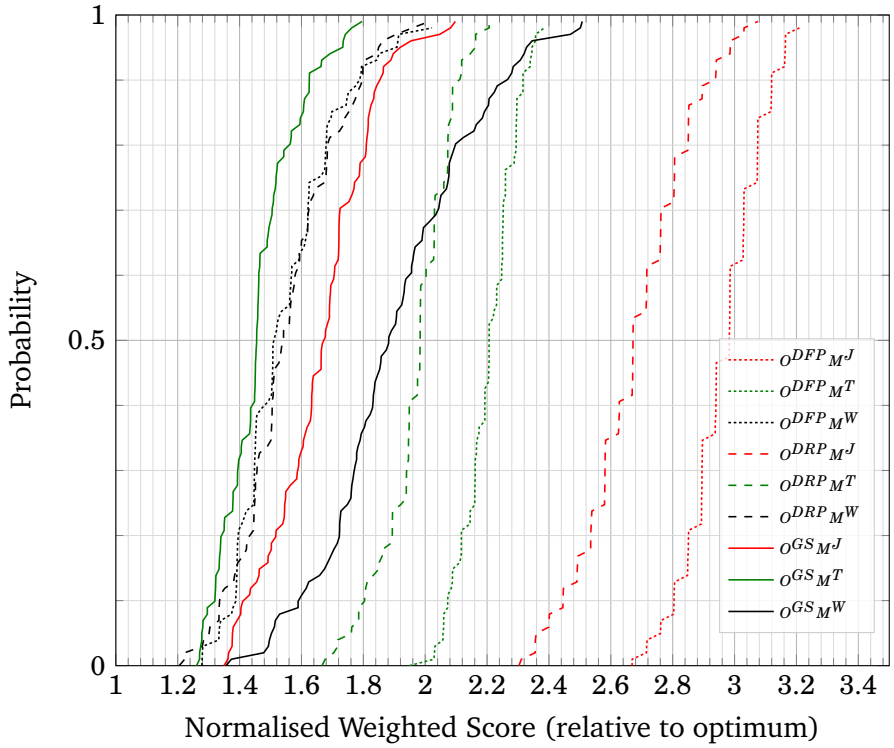


Figure 5.9: Cumulative Distribution Function graphs for heuristic algorithm outputs of schedule times and energy for Case 2 with w_t : 0.9 w_e : 0.1 (10 MDs, 3 MEC servers, 72 jobs).

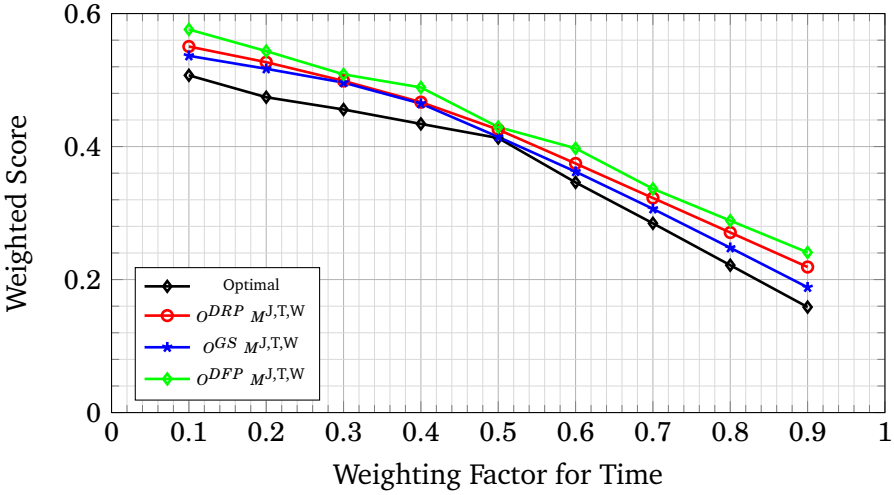


Figure 5.10: Performance of the proposed heuristic approaches in MEC network with 13 MDs and 20 Jobs. The solution for each heuristic approach was the best solution from 100 runs.)

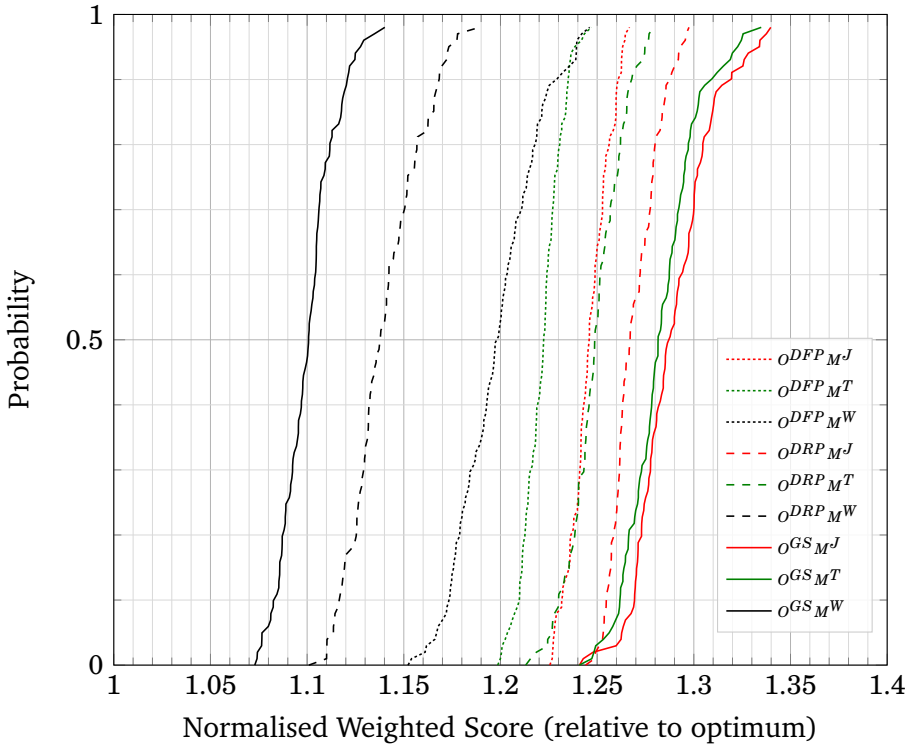


Figure 5.11: Cumulative Distribution Function graphs for heuristic algorithm outputs of schedule times and energy for Case 3 with $w_t : 0.1$ $w_e : 0.9$ (13 MDs, 3 MEC servers, 115 jobs).

The probable explanation for this difference in relative performance was that the guided-search algorithms were focused on identifying minimum Weighted Scores for each iteration with jobs yet to be allocated to MEC servers.

Changing the weighting factors, however, greatly affected the results because the time and energy factors were linked, especially if the connection speed to a MEC server was the important criterion for allocating a job to a MEC server. Connection speeds dominate total task completion times and, therefore, local energy use if data require longer times to be transmitted and received at constant power ratings for MDs transmitting and receiving data.

The heuristic approach has been successful with an individual MD and with multiple MDs but the detailed results in the three cases simulated numerically differed. With a single MD, the algorithms matched the CPLEX results better at higher weighting factors for time while this trend was not apparent with multiple MDs. Further numerical simulations with larger numbers of iterations could demonstrate the consistency of trends with changing weighting factors and explore how random and statistical effects determine overall outcomes of employing heuristic algorithms.

In general, low time weighting factors resulted in higher minimum weighted scores; greater emphasis on energy use encouraged more jobs to be offloading and this caused queuing at MEC servers which significantly increased task completion times. In contrast, higher emphasis on

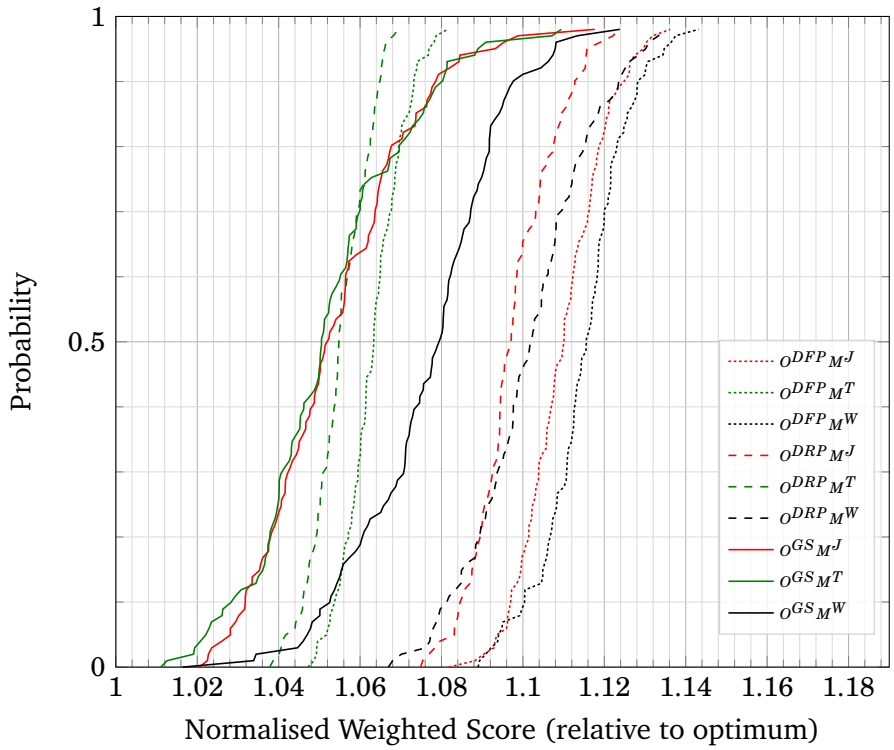


Figure 5.12: Cumulative Distribution Function graphs for heuristic algorithm outputs of schedule times and energy for Case 3 with $w_t : 0.5$ $w_e : 0.5$ (13 MDs, 3 MEC servers, 115 jobs).

time saving resulted in more parallel processing, using the full computational resources of the MEC network.

For the user of a MD, the choice between time and local energy use will be determined by the individual circumstances; for example, low battery charge will favour low MD energy use. Conversely, for a network with multiple devices attempting to connect, time is more favourable a parameter for centralised resource allocation because this will reduce time occupancy on the servers.

An important conclusion from the numerical simulations was that the choice of weighting factors is crucial. Total task completion times and local energy use are both dependent on the selection of weighting factors but there is no optimal solution which combines both. An arbitrary choice of weighting factors, for example in [100], does not allow an individual user to tailor solutions that meet immediate needs (time savings or MD energy savings).

For both an individual user and a network resource allocator, the choice of the best heuristic algorithm will depend on the parameter of higher value (total task completion time or local energy use) and from a suite of algorithms can be built into devices to adapt to changing user demands.

The results demonstrate that the choice of heuristic algorithm depends on the choice of weightings on energy and time components in the objective function. We have seen that

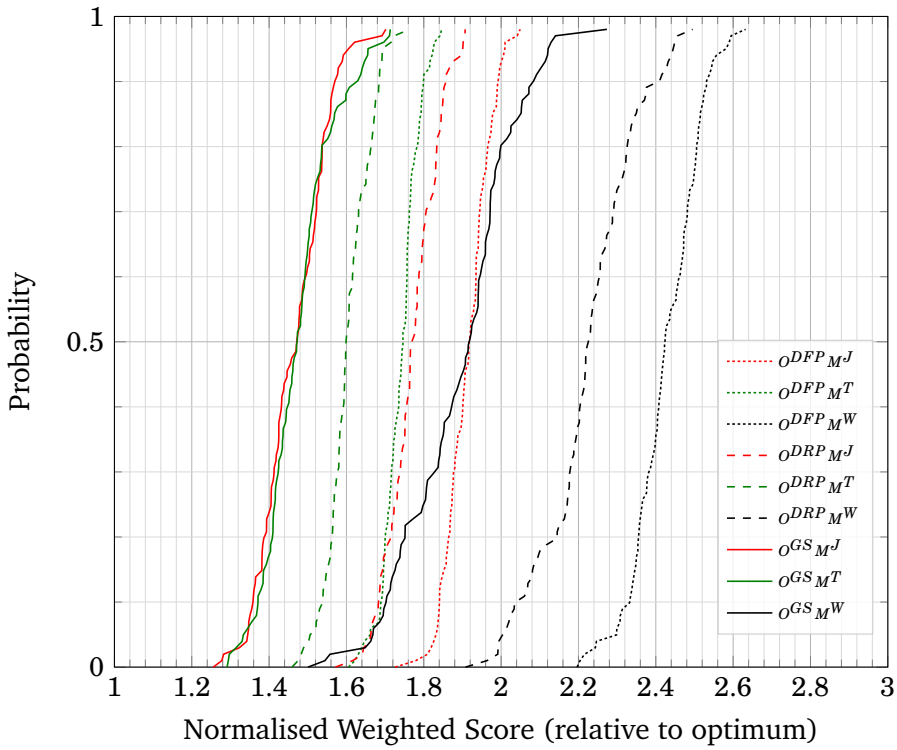


Figure 5.13: Cumulative Distribution Function graphs for heuristic algorithm outputs of schedule times and energy for Case 3 with w_t : 0.9 w_e : 0.1 (13 MDs, 3 MEC servers, 115 jobs).

("O^{GS},M^T") performs better when weighting for time is higher and ("O^{GS},M^W") performs better otherwise.

5.6 Extending Optimisation to Incorporate Economic Cost Factors

In the MEC networks described in Chapter 3 and 4 and in Sections 5.1-5.5. the implicit assumption has been made that users subscribe for premium services with Service Level Agreements (SLA). The cost of offloading can be modelled on a relative basis with different servers having different costs because of their link speeds and processor speeds, which together offer users of MDs faster or slower total task completion times and different local energy savings. As an example of the range of economic costs incurred by users, photography software services in Cloud Computing differ approximately two-fold between entry and premium offerings [1]. Google's G Suite Cloud Computing service has a "Business" plan costing twice that of a "Basic" plan. Subscription-based offloading to a MEC network can, therefore, be anticipated to have a similar range of price plans [2] For offloading, the size of the job offloaded is likely to be the

[1] Adobe Creative Cloud:
`https://www.adobe.com/uk/creativecloud/plans.html?promoid=NV3KR7S1&mv=other`

[2] Google GSUITE :
`https://gsuite.google.com/pricing.html).`

determining factor for a subscriber. The cost of the offloading process can be simply modelled as a unit price per MB × job size.

Placing the emphasis on cost minimization, i.e. a high weighting factor for cost relative to time and local energy, poses an immediate problem in that maximum cost reduction implies restricted or prohibited offloading. Cost has, therefore, an inverse relationship to local energy savings: whereas a high weighting factor for energy favours the offloading of most or all jobs from an MD, a high weighting factor for cost inhibits offloading. Task completion time, as discussed in Chapters 3 and 4, aims at maximising parallel processing using all the CPU resources available in the multi-server MEC network. A preliminary analysis of the interaction of the three factors (time, local energy and cost) has been performed using the single MD/multiple MEC server cases used in Sections 5.3.1-5.3.2.

5.6.1 Generalising model using to multiple components

So far, minimisation of time and energy have been considered. Other components in the objective function can also be introduced in a similar manner. The following equation is a generalised formulation of a multi-objective function consisting of N components.

$$\min \sum_{i=1}^{N} w_i C_i \tag{5.5}$$

where w_i is the weighting factor of the function component C_i, respectively. The sum of the weighting factors is constrained to be 1, i.e. $\sum_{i=1}^{N} w_i = 1$.
With cost incorporated into the analysis, the Weighted Sum is the sum of three components:

$$w_t \times T + w_e \times E + w_c \times C \tag{5.6}$$

where T, E and C represent time, energy and cost terms normalised to the maximum values.

5.6.2 Time-Energy-Cost Performance Analysis of the 81-schedule Scenario: 1 MD Offloading up to 4 Jobs to 2 MEC Servers

Of the two MEC servers included in this network, one has a higher link while the other has a higher processor speed. Since is link speed the dominant factor in determining task completion time and local energy use in offloading, the server (MEC 2) with the 25 Mbps link speed is given a unit cost factor of 1.5 while MEC 1 (15 Mbps) is given the unit cost factor of 1.0. To offload all 4 jobs (10 MB), the cost to the user is 10 units when offloading to MEC1 and 15 units when offloading to MEC 2.

Using normalised data (parameter/maximum parameter), the effects of introducing costs and cost weighting factors were investigated by sequentially increasing the cost weighting factor (with the sum of the three weighting factors = 1). Table 5.8 presents the outcomes using a cost weighting factor of 0.1 (the value selected in [100]). At time weighting factors of 0.2 and

0.3, two different schedules had the same Weighted Scores. As the weighting factor for time increased, the Weighted Score decreased, the local energy increased while the time and cost decreased. Table 5.9 uses a cost weighting factor of 0.2. The outcomes are very similar to those in Table 5.6, although the cost values are more erratic. Table 5.10 uses a cost weighting factor of 0.3. With a zero-value weighting factor for time, no jobs were offloaded. As the weighting factor for time increased, jobs were offloaded (2 or 3 out of 4) and the Weighted Score decreased. Local energy use showed a weak minimum at weighting factors for time of 0.2 and 0.3.

Table 5.8: Optimal schedules based on minimal Weighted Scores for 1 MD offloading up to 4 jobs to 2 MEC servers (normalised times, energies and costs); wc=0.1.

Time Weighting Factor	Energy Weighting Factor	Cost Weighting Factor	Schedule option	Time	Energy	Cost	Weighted Score	Jobs Offloaded
0	0.9	0.1	3	0.56	0.627	1.00	0.664	4
0.1	0.8	0.1	3	0.56	0.627	1.00	0.657	4
0.2	0.7	0.1	19,41	0.39	0.710	0.70	0.645	3,2
0.3	0.6	0.1	19,41	0.39	0.710	0.70	0.612	3,2
0.4	0.5	0.1	17	0.33	0.747	0.67	0.574	3
0.5	0.4	0.1	6	0.28	0.795	0.70	0.527	3
0.6	0.3	0.1	6	0.28	0.795	0.70	0.475	3
0.7	0.2	0.1	6	0.28	0.795	0.70	0.423	3
0.8	0.1	0.1	6	0.28	0.795	0.70	0.372	3
0.9	0	0.1	6	0.28	0.795	0.70	0.320	3

Table 5.9: Optimal schedules based on minimal Weighted Scores for 1 MD offloading up to 4 jobs to 2 MEC servers (normalised times, energies and costs); wc=0.2.

Time Weighting Factor	Energy Weighting Factor	Cost Weighting Factor	Schedule option	Time	Energy	Cost	Weighted Score	Jobs Offloaded
0	0.8	0.2	3	0.56	0.627	1.00	0.701	4
0.1	0.7	0.2	19,41	0.39	0.710	0.70	0.676	3,2
0.2	0.6	0.2	35,49	0.40	0.737	0.60	0.642	3,2
0.3	0.5	0.2	17,35,49	0.33	0.747	0.67	0.607	3,3,2
0.4	0.4	0.2	14	0.30	0.785	0.63	0.561	3
0.5	0.3	0.2	14	0.30	0.785	0.63	0.512	3
0.6	0.2	0.2	14	0.30	0.785	0.63	0.464	3
0.7	0.1	0.2	13,42	0.30	0.823	0.60	0.412	3,2
0.8	0	0.2	13,42	0.30	0.823	0.60	0.360	3,2

Table 5.11 uses a cost weighting factor of 0.4. No jobs were offloaded until the weighting factor for time reached 0.3. A poorly defined minimum energy and a poorly defined maximum for the Weighted Score occurred at a time weighting factor of 0.3. Table 5.12 uses a cost weighting factor of 0.5. No jobs were offloaded until the weighting factor for time reached 0.4. Poorly defined maxima for the Weighted Score and local energy occurred at a time weighting factor of 0.3 and 0.4, respectively. These results show that incorporating cost into the analysis did not result in a minimal value for Weighted Score. Maximal values for Weighted Score were,

Table 5.10: Optimal schedules based on minimal Weighted Scores for 1 MD offloading up to 4 jobs to 2 MEC servers (normalised times, energies and costs); wc=0.3.

Time Weighting Factor	Energy Weighting Factor	Cost Weighting Factor	Schedule option	Time	Energy	Cost	Weighted Score	Jobs Offloaded
0	0.7	0.3	1	1.00	0.903	0.00	0.632	0
0.1	0.6	0.3	35	0.40	0.737	0.60	0.629	3
0.2	0.5	0.3	35,49	0.40	0.737	0.60	0.629	3,2
0.3	0.4	0.3	14,49	0.30	0.785	0.63	0.594	3,2
0.4	0.3	0.3	14	0.30	0.785	0.63	0.545	3
0.5	0.2	0.3	13,42	0.30	0.823	0.60	0.495	3,2
0.6	0.1	0.3	13,42	0.30	0.823	0.60	0.442	3,2
0.7	0	0.3	13,42	0.30	0.823	0.60	0.390	3,2

Table 5.11: Optimal schedules based on minimal Weighted Scores for 1 MD offloading up to 4 jobs to 2 MEC servers (normalised times, energies and costs); wc=0.4.

Time Weighting Factor	Energy Weighting Factor	Cost Weighting Factor	Schedule option	Time	Energy	Cost	Weighted Score	Jobs Offloaded
0	0.6	0.4	1	1.00	0.903	0.00	0.542	0
0.1	0.5	0.4	1	1.00	0.903	0.00	0.561	0
0.2	0.4	0.4	1	1.00	0.903	0.00	0.561	0
0.3	0.3	0.4	53	0.70	0.820	0.27	0.563	2
0.4	0.2	0.4	29	0.40	0.926	0.43	0.518	3
0.5	0.1	0.4	29	0.40	0.926	0.43	0.466	3
0.6	0	0.4	29	0.40	0.926	0.43	0.413	3

Table 5.12: Optimal schedules based on minimal Weighted Scores for 1 MD offloading up to 4 jobs to 2 MEC servers (normalised times, energies and costs); wc=0.5.

Time Weighting Factor	Energy Weighting Factor	Cost Weighting Factor	Schedule option	Time	Energy	Cost	Weighted Score	Jobs Offloaded
0	0.5	0.5	1	1.00	0.903	0.00	0.452	0
0.1	0.4	0.5	1	1.00	0.903	0.00	0.471	0
0.2	0.3	0.5	1	1.00	0.903	0.00	0.471	0
0.3	0.2	0.5	1	1.00	0.903	0.00	0.481	0
0.4	0.1	0.5	36	0.50	0.950	0.33	0.462	3
0.5	0	0.5	29,36	0.40	0.926	0.43	0.417	3,3

however, apparent with weighting factors for cost of 0.4 or 0.5. Minimal values for Weighted Score always occurred at the highest possible values for the time weighting factor. With cost weighting factors above 0.2, the maximum number of jobs offloaded clearly decreased.

5.6.3 Time-Energy-Cost Performance Analysis of the 1024-schedule Scenario: 1 MD Offloading up to 5 Jobs to 3 MEC Servers

The corresponding outcomes for 1 MD Offloading up to 5 Jobs to 3 MEC Servers are presented in Tables 5.13-5.17. At weighting factors for cost of 0.1 and 0.2, Weighted Scores decreased as time weighting factors increased. At higher cost weighting factors, however, cost and energy values and jobs offloaded became erratic at low time weighting factors. Poorly defined maxima for Weighted Scores were identifiable at cost weighting factors of 0.3-0.5.

Table 5.13: Optimal schedules based on minimal Weighted Scores for 1 MD offloading up to 5 jobs to 3 MEC servers (normalised times, energies and costs); wc=0.1.

Time Weighting Factor	Energy Weighting Factor	Cost Weighting Factor	Schedule option	Time	Energy	Cost	Weighted Score	Jobs Offloaded
0.1	0.8	0.1	688,764,1007	0.43	0.481	1.00	0.528	5
0.2	0.7	0.1	1023	0.29	0.505	0.92	0.503	5
0.3	0.6	0.1	716	0.23	0.533	0.83	0.472	4
0.4	0.5	0.1	207,812,831	0.20	0.571	0.72	0.439	3, 4, 4
0.5	0.4	0.1	207,812,831	0.20	0.571	0.72	0.402	3, 4, 4
0.6	0.3	0.1	124	0.17	0.626	0.70	0.362	4
0.7	0.2	0.1	124	0.17	0.626	0.70	0.317	4
0.8	0.1	0.1	124	0.17	0.627	0.70	0.271	4

Table 5.14: Table 5.13: Optimal schedules based on minimal Weighted Scores for 1 MD offloading up to 5 jobs to 3 MEC servers (normalised times, energies and costs); wc=0.2.

Time Weighting Factor	Energy Weighting Factor	Cost Weighting Factor	Schedule option	Time	Energy	Cost	Weighted Score	Jobs Offloaded
0.1	0.7	0.2	688,764,1007	0.26	0.510	0.90	0.563	5
0.2	0.6	0.2	1023	0.29	0.505	0.92	0.544	5
0.3	0.5	0.2	207,812,831	0.20	0.568	0.72	0.488	3, 4, 4
0.4	0.4	0.2	207,812,831	0.20	0.568	0.72	0.452	3, 4, 4
0.5	0.3	0.2	207,812,831	0.20	0.568	0.72	0.415	3, 4, 4
0.6	0.2	0.2	124	0.17	0.626	0.70	0.369	4
0.7	0.1	0.2	124	0.17	0.626	0.70	0.324	4
0.8	0	0.2	124	0.17	0.627	0.70	0.279	4

5.7 Conclusions

This Chapter extended the heuristic approaches described for task completion time only in Chapter 4 to include local (MD) energy savings and a preliminary analysis of the economic cost of accessing offloading services on a subscription or other basis. This work addressed Research Questions 9 and 10 (Chapter 1, Section 1.3.2).

Table 5.15: Optimal schedules based on minimal Weighted Scores for 1 MD offloading up to 5 jobs to 3 MEC servers (normalised times, energies and costs); wc=0.3.

Time Weighting Factor	Energy Weighting Factor	Cost Weighting Factor	Schedule option	Time	Energy	Cost	Weighted Score	Jobs Offloaded
0.1	0.6	0.3	99	0.40	0.686	0.35	0.557	3
0.2	0.5	0.3	1023	0.29	0.505	0.92	0.585	5
0.3	0.4	0.3	207,812,831	0.20	0.568	0.72	0.503	3, 4, 4
0.4	0.3	0.3	79,287	0.20	0.653	0.62	0.461	3, 4
0.5	0.2	0.3	79,287	0.26	0.628	0.62	0.441	3, 4
0.6	0.1	0.3	199,823	0.21	0.681	0.58	0.369	3, 4
0.7	0	0.3	199,823	0.21	0.681	0.58	0.322	3, 4

Table 5.16: Optimal schedules based on minimal Weighted Scores for 1 MD offloading up to 5 jobs to 3 MEC servers (normalised times, energies and costs); wc=0.4.

Time Weighting Factor	Energy Weighting Factor	Cost Weighting Factor	Schedule option	Time	Energy	Cost	Weighted Score	Jobs Offloaded
0.05	0.55	0.4	1	1.00	0.510	0.00	0.331	0
0.1	0.6	0.4	1024	0.43	0.481	1.00	0.732	5
0.2	0.4	0.4	99	0.40	0.686	0.35	0.494	3
0.3	0.3	0.4	99	0.40	0.686	0.35	0.466	3
0.4	0.2	0.4	99	0.20	0.568	0.35	0.335	3
0.5	0.1	0.4	355,522,674	0.27	0.735	0.47	0.393	4, 3, 4
0.55	0.05	0.4	355,522,674	0.27	0.735	0.47	0.370	4, 3, 4
0.6	0	0.4	355,522,674	0.27	0.735	0.47	0.347	4, 3, 4

Table 5.17: Optimal schedules based on minimal Weighted Scores for 1 MD offloading up to 5 jobs to 3 MEC servers (normalised times, energies and costs); wc=0.5.

Time Weighting Factor	Energy Weighting Factor	Cost Weighting Factor	Schedule option	Time	Energy	Cost	Weighted Score	Jobs Offloaded
0.05	0.45	0.5	1	1.00	0.804	0.00	0.412	0
0.1	0.4	0.5	1023	0.29	0.505	0.92	0.689	5
0.2	0.3	0.5	1	1.00	0.804	0.00	0.441	0
0.3	0.2	0.5	99	0.40	0.686	0.35	0.432	3
0.4	0.1	0.5	99	0.40	0.686	0.35	0.404	3
0.45	0.05	0.5	99	0.40	0.686	0.35	0.389	3
0.5	0	0.5	26	0.33	0.774	0.40	0.367	3

The heuristic approach has been successful with an individual MD and with multiple MDs but the detailed results in the three cases simulated numerically differed. With a single MD, the algorithms matched the CPLEX results better at higher weighting factors for time while this trend was not apparent with multiple MDs. Further numerical simulations with larger numbers of iterations could demonstrate the consistency of trends with changing weighting factors and explore how random and statistical effects determine overall outcomes of employing heuristic algorithms.

In general, low time weighting factors resulted in higher minimum weighted scores; greater emphasis on energy use encouraged more jobs to be offloading and this caused queuing at MEC servers which significantly increased task completion times. In contrast, higher emphasis on time saving resulted in more parallel processing, using the full computational resources of the MEC network.

For the user of a MD, the choice between time and local energy use will be determined by the individual circumstances; for example, low battery charge will favour low MD energy use. Conversely, for a network with multiple devices attempting to connect, time is more favourable a parameter for centralised resource allocation because this will reduce time occupancy on the servers.

An important conclusion from the numerical simulations was that the choice of weighting factors is crucial. Total task completion times and local energy use are both dependent on the selection of weighting factors but there is no optimal solution which combines both. An arbitrary choice of weighting factors, for example in [84], does not allow an individual user to tailor solutions that meet immediate needs (time savings or MD energy savings). For both an individual user and a network resource allocator, the choice of the best heuristic algorithm will depend on the parameter of higher value (total task completion time or local energy use) and from a suite of algorithms can be built into devices to adapt to changing user demands. Machine Learning can be deployed to use historical data from individual MD users or network resource allocators to automate decision making to make the offloading process more efficient and more responsive in the face of rapidly changing user numbers and network overloading.

When the analysis presented in this Chapter was extended to include a preliminary analysis of the economic cost with three weighting factors included for time, local energy and economic cost, no minimal Weighting Scores were apparent. Time, energy and cost can, therefore, be viewed as parameters that cannot be combined into global optima by this approach; instead, trade-offs were identified which would prove useful for the users of MDs, when task completion time, local energy usage or cost to the user become important relative to each other and can awarded variable weighting factors by the user.

For example, low battery charge will probably always emphasise energy and require a high energy weighting factor and the offloading of all or most jobs; this has an associated economic cost but may not result in the shortest task completion times. Conversely, when battery charge is not an issue, an individual user can make a selection between offloading or local processing; this is a choice based on cost grounds.

A possible implementation for users of MDs is a linked triple sliding scale on screen which is user-determined and would work within the limitation that the sum of the three weighting factors must be constant.

CHAPTER 6

CONCLUSIONS AND IMPLICATIONS FOR FUTURE WORK

6.1 Conclusions

Chapter 1 reviewed the present status of Multi-access Edge Computing (MEC) and noted the persistent confusion over nomenclature and its growing interactions with 5G technologies. Still in a pre-deployment state, MEC suffers from major economic and commercial uncertainties. As the authors of the reference most cited in Chapter 1 [129] conclude, an "approach that provides a reasonable immediate return on investment while also representing a significant long-term opportunity" has not yet been identified.

Chapter 2 presented an in-depth analysis of how the various proposal for Edge Computing evolved and advances the view that, semantically, "Multi-access Edge Computing" can be viewed as Edge Computing, with various deployment strategies from single-enterprise to regional levels. Nevertheless, the unifying factor is the access to servers which are much physically closer than those in distant consolidated data centres at which extra latency is added because of delays in job processing. In Figure 6.1., the analyses presented in this thesis focus on the relationship between MDs and the Edge Nodes, minimising latency and other delays inherent in multi-stage transfer of information; this will be particularly valuable for MD applications which are highly latency-sensitive.

Chapter 3 analysed the considerations in a heterogeneous MEC network that determine the advantages and the success (or not) of the offloading strategy from MDs viewed in its most basic form: one MD attempting to offload one task (job) to a single MEC server over a wireless link. Major advantages in total task completion time and local energy saving over values computed for local processing can be demonstrated but multiple parameters must be considered: on-board and server-side processing speeds, link communication speeds, task computational complexity, network congestion and latency and the CPU workloads of the MD and MEC server.

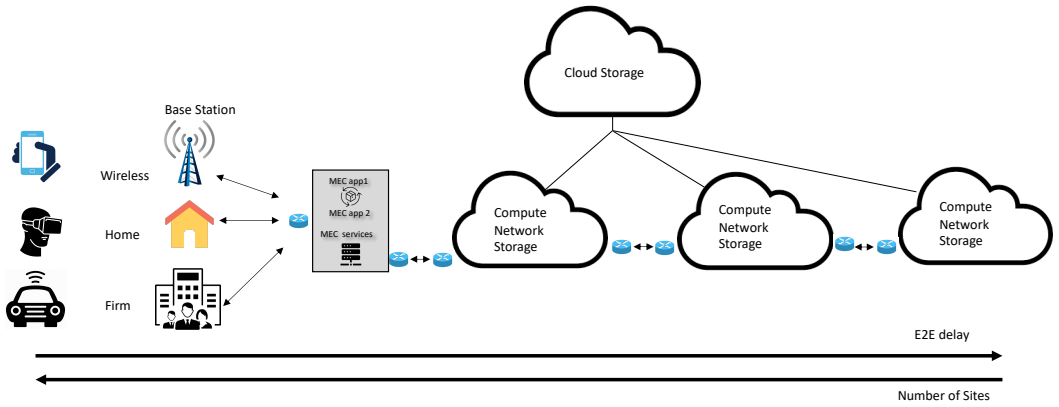


Figure 6.1: Interfaces between Edge Computing and Cloud Computing (redrawn from [129].

Chapter 4 extended the analysis to consider multiple jobs to be either offloaded or processed locally on the MD in networks with increasing numbers of MEC servers. The approach employed is that of assessing possible schedules of operations (i.e. offloading and local processing) from which an optimal strategy to minimise total task completion time can be computed. Using only task completion time as the criterion, multiple parallel processing, using all the processing capacity in the network, was found to be essential for optimal solutions.

Chapter 5 then incorporated local energy savings into the analysis with heuristic algorithms extended to incorporate both time and local energy factors; results from numerical simulations showed clearly that only trade-offs between the two goals could be achieved; this was because, using weighting factors to balance the different aims of minimising total task completion time and local energy use, no independent variables could be included – mathematically, the two weighting values are inextricably linked and act as mutual constraints. Even adding in economic cost as a third consideration failed to identify “global” minima, although weak maxima were apparent

6.2 Implications of the Research Outcomes

6.2.1 Offloading decision-making programmes

To the question “Which computational tasks benefit most from offloading?”, the answer is complex because of the effects of several distinct parameters. Task complexity is a crucial determinant but only when combined with connection speed. Other parameters are, however, important:

- On-board and server-side processor speeds
- MD and MEC server CPU workloads

- Latency or access delays
- Network load

The file size of the data to be offloaded is not in itself important but will affect the benefits to be gained in terms of shorter task completion times and local (MD) energy savings. If the full list of information and parametric values is accessible to a single MD accessing the MEC network, a computational programme can be written to assess whether or not an individual task would benefit from offloading in terms of task completion time and reduced local energy use.

If the totality of information is not directly accessible, a centralised resource allocator in the MEC network would take on the role of offload decision maker. This would be essential for offloading to save MD battery charge because offloading is not guaranteed to result in MD energy savings if link speeds are low and the data transmission/reception cycle is lengthy. Access to and the actions of such a centralised decision maker would be included in the Service Level Agreement between MD users and a commercial MEC network.

6.2.2 Heuristic Optimisation of Offloading Schedules for Shorter Task Completion Times

Offloading multiple jobs from a single MD to multiple MEC servers can generate very large numbers of possible schedules. Linear optimisation can identify unique optimum solutions but to avoid time-consuming analyses, heuristic approaches were adopted.

Heuristic algorithms were constructed which could closely (sometimes, within 1%) match the times of optimum solutions for offloading multiple jobs from an individual MD to multiple MEC servers and for multiple MDs to offload jobs to multiple MEC servers. The functioning of distributed heuristic algorithms assumes that full knowledge of network parameters can be accessed by individual MDs; if full knowledge of the parameters is reserved to a centralised resource allocator in the network, communication between MDs and the network will provide to an individual MD user the offloading decisions in an near-optimal schedule.

The advantage of a centralised offloading decision maker is that network status – in particular, the number of users per server and server CPU workloads – is constantly monitored and offloading demands and network traffic can be balanced to avoid severe network congestion or to access back-up MEC servers or cloud servers. Commercial MEC network providers will probably impose fair-use policies in Service Level Agreements to fine-tune offloading demands and network traffic to prevent network congestion.

6.2.3 Multi-Factor Heuristic Optimisation of Offloading to Minimise Time, Energy and Cost

Combining time and local energy introduces a new relationship into the MEC network. The most efficient solution for individual MD users is for them to set the weighting factors to reflect

their circumstances (low battery charge or time demands). The weighting factors, therefore, become parameters communicated by the MD to the MEC network and, as in Section 6.2.2, the network's status with respect to traffic and offloading requests can be used to accept or modify identified schedules chosen by heuristic algorithms.

6.3 Aspects of MEC offloading not considered in this thesis

As discussed in Section 2.8, several authors have proposed features of the offloading process that were not incorporated into the mathematical modelling of Chapter 3-5. The resulted presented in Chapter 3-5 have, nevertheless, implications for offloading processes with significant differences from the processes from individual or multiple MDs described in Chapter 3-5.

Task partitioning: this form of offloading divides the task into sub-tasks, only some of which are offloaded. The results of Chapter 4 indicate that where multiple tasks are being offloaded, the optimum solution is to include the on-board processor to work in parallel with the MEC servers. This parallel processing can be simply extended to imply that task partitioning would also be beneficial for total task completion time; if, however, only the least computational complex sub-tasks were to be performed locally, the impact on total task completion time would be minimal. The overall effect would be highly dependent on how the task is partitioned [98].

Application migration: the program code for highly specialised applications could be transferred from the MD to the MEC servers or Cloud-to-Edge transfer of program code might be required. The results of Chapter 3 imply that either form of application migration would greatly extend total task completion time and reduce any advantage of offloading.

Data caching: if a user seeks to repeat processing of previously processed data files, stored data would, on the basis of results presented in Chapter 3, greatly reduce total task completion time because data transfer times generally outweigh MEC server processing times, especially if link communication speeds are slow. The main issues here are data security and user privacy and strict protocols would need to be incorporated into the contractual relationship between and service provider.

Multiple-hop networks: transfer of data from server to server and, in the extreme case, MEC network to Cloud data centre may be required in case of sever overload. The results from Chapter 3 imply an erosion of the advantages of offloading (in terms of both total task completion time and MD energy use) if longer times to receive data are incurred and MDs spend longer times in "idling" mode before data is received back from the MEC network.

Resource allocation and management strategies in MEC networks: Chapters 4 and 5 assumed an "instant" response to any requestions from single MDs (where the optimal scheduling is identified locally) or from multiple MDs simultaneously (where the optimum schedule is identified centrally). Other management protocols might impose queuing and network-based analysis of the types of tasks to be offloaded might, in the extreme, "quarantine"

some task to improve work flow inside the network server (s); for example, lengthy processing times for some user requests could be postponed to maximise server use but avoid excessive CPU workloads. This type of micro-management would be expected to impose longer total task completion times but is a stochastic problem because of the essentially unpredictable nature of network traffic in different time periods.

Non-subscription access to MEC services: if only "guest" users interact with a MEC network on an ad hoc basis, dynamic pricing would result in the maximisation of revenues for the service provider. Under some circumstances, however, periods of low demand would enable users to access services on a more economic basis. From Chapter 5, this would translate into a reduced cost and encourage users to shift attention to time and energy to identify best solutions for offloading schedules.

Offloaded data for applications beyond facial recognition of digital images: other forms of Visual Analytics (for example, object recognition) and Augmented Reality, which have been combined in an interactive perception application [121], are examples to which the results presented in this Thesis could be extended. While smartphones are highly unlikely to be involved in Big Data analytics, high-end tablet computers could seek to offload large data sets (100 MB – 1 GB) for analysis by applications other than spreadsheet (for example, Excel) or statistical programs (for example, SigmaPlot. In such cases, the analyses presented in Chapters 3-5 would be directly applicable for total task completion times; battery lifetimes are generally considered much longer in tablet devices than in older laptop computers and smartphones and energy use is less likely to be a major consideration.

6.4 Future Work

In this section, I will outline my possible future research in MEC networks and then briefly consider work in related fields.

6.4.1 The impact of 5G Technologies on MEC Networks

A key parameter identified in Chapters 3-5 for successful offloading to a MEC network was the link speed at which MDs transmit and receive data from MEC servers. As 5G technologies increase communications speeds, less computationally complex jobs would be open for offloading with shorter task completion times and reduced local energy use.

To illustrate this effect, Figure Figure 3.8 can provide a baseline, where a low-complexity application requires a much faster link speed to achieve very high energy savings for the MD. If – and has almost certainly already occurred – both the on-board and server-side processor speeds have increased in the telecommunications and IT industries from the values quoted in 2017 [105], the demands for faster link speeds become important for offloading.

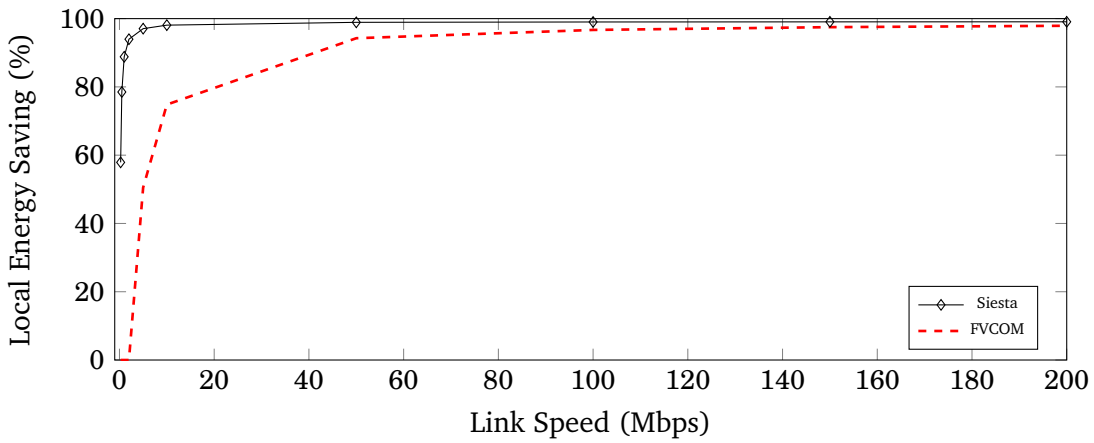


Figure 6.2: Local energy savings by offloading with higher- and lower-applications with two-fold increases in processor speeds.

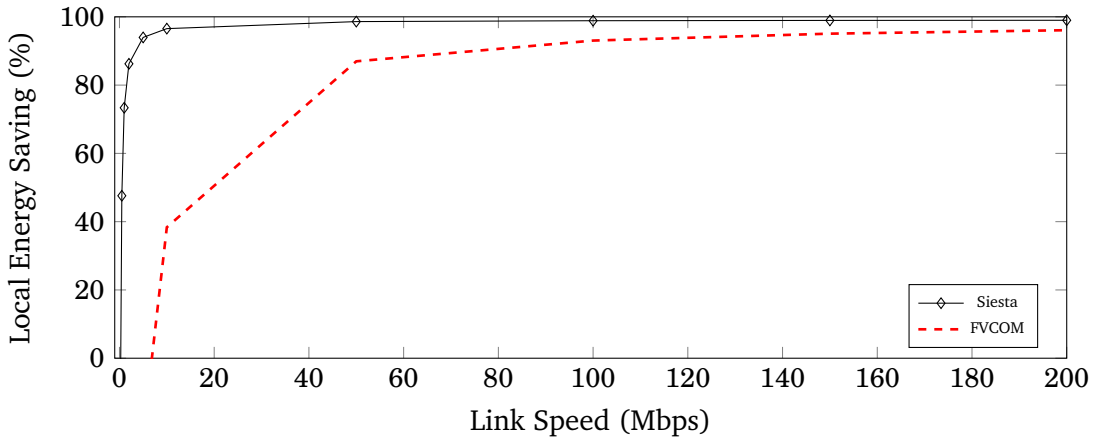


Figure 6.3: Local energy savings by offloading with higher- and lower-applications with five-fold increases in processor speeds.

Figure 6.2 updates Figure 3.8 to twice the processor speeds used earlier. For full energy savings by offloading, link speeds approaching 10 Mbps are required. Figure 6.3 further increases processor speeds to five-fold higher than in the baseline scenario and link speeds exceeding 10 Mbps are required to achieve maximum energy savings.

The introduction of 5G technologies, therefore, allows increasingly powerful MDs with faster on-board processors to benefit from offloading. Conversely, the introduction of MDs with increasingly powerful processors will require 5G link speeds to find advantages in task completion time and achieve the greatest local energy savings. This is a demonstration of the links between 5G and MEC systems deployment.

As more jobs and tasks benefit from offloading, however, the threat of network overloading and congestion increases and the pressure on MEC service providers to meet peak demand periods becomes more intense. Back-up MEC servers are one solution; resorting to transferring

jobs without critical latency requirements to cloud servers is another (Figure 6.1).

Consequently, a key management task in deployed MEC networks will be in the filtering of offloaded jobs into time-critical, latency-intolerant and a third category where neither time nor latency is crucial for the offloading process. Future work could focus on identifying informational clues associated with tasks to be offloaded which to be used to make this categorisation by a network resource allocator.

6.4.2 Green Energy Price Costing for Offloading

Chapter 3 and 5 considered only the energy usage by individual MDs. To understand the full energy requirement for the offloading process, however, requires to compute more than the transmission, idling and receiving phases (plus display and storage events) in the MD. MEC systems will probably have clusters of servers housed in much smaller aggregates than are customary for distant consolidated date centres and will not, therefore, be able to increase energy efficiency to the same extent.

Nevertheless, experimental investigation is possible to define the energy requirements for discrete job processing in a MEC server. If the totality of energy usage during offloading from the MD to the MEC system is less than the energy required for local processing on the MD, the opportunity arises from Green Tariffs to encourage energy efficiency.

A further strategy for the providers of MEC services would be to offer reduced prices (for other than time-critical offloaded jobs) during off-peak periods to reduce network congestion and increased power consumption by high-workload servers or to divert suitable tasks to cloud servers (Figure 6.1). This implies a continuing but evolving relationship between Edge Computing and Cloud Computing with co-operation (or joint ownership) between service providers to maximise the Quality of Experience of MD users.

6.4.3 Possible paths of future research

The examples given in Section 6.4.1-6.4.2 are straightforward extensions of the work presented in Chapters 3 4.

More radical research areas include those of software-defined batteries for mobile devices (i.e., the fusion of multiple battery chemistries which are differentially adapted to different tasks and power demands) to challenge the physical resource-poor nature of smartphones [10], the adoption of blockchain decentralized data management frameworks for MEC [161] and Software-Defined Networking, which is claimed to improve network management and facilitate network evolution [88], each of which individually could greatly affect the relationships between MDs and MEC networks.

APPENDIX

Table A.1: Schedule Options for Offloading with 81 Distinct Schedules

Option	Job 1	Job 2	Job 3	Job 4
1	J1-0	J2-0	J3-0	J4-0
2	J1-C1	J2-C1	J3-C1	J4-C1
3	J1-C2	J2-C2	J3-C2	J4-C2
4	J1-0	J2-C1	J3-C1	J4-C1
5	J1-0	J2-C1	J3-C1	J4-C2
6	J1-0	J2-C1	J3-C2	J4-C2
7	J1-0	J2-C1	J3-C2	J4-C1
8	J1-0	J2-C2	J3-C1	J4-C1
9	J1-0	J2-C2	J3-C1	J4-C2
10	J1-0	J2-C2	J3-C2	J4-C1
11	J1-0	J2-C2	J3-C2	J4-C2
12	J1-C1	J2-0	J3-C1	J4-C1
13	J1-C1	J2-0	J3-C1	J4-C2
14	J1-C1	J2-0	J3-C2	J4-C2
15	J1-C1	J2-0	J3-C2	J4-C1
16	J1-C2	J2-0	J3-C1	J4-C1
17	J1-C2	J2-0	J3-C1	J4-C2
18	J1-C2	J2-0	J3-C2	J4-C1
19	J1-C2	J2-0	J3-C2	J4-C2
20	J1-C1	J2-C1	J3-0	J4-C1
21	J1-C1	J2-C1	J3-0	J4-C2
22	J1-C1	J2-C2	J3-0	J4-C2
23	J1-C1	J2-C2	J3-0	J4-C1
24	J1-C2	J2-C1	J3-0	J4-C1
25	J1-C2	J2-C1	J3-0	J4-C2
26	J1-C2	J2-C2	J3-0	J4-C1
27	J1-C2	J2-C2	J3-0	J4-C2
28	J1-C1	J2-C1	J3-C1	J4-0
29	J1-C1	J2-C1	J3-C2	J4-0
30	J1-C1	J2-C2	J3-C2	J4-0
31	J1-C1	J2-C2	J3-C1	J4-0
32	J1-C2	J2-C1	J3-C1	J4-0
33	J1-C2	J2-C1	J3-C2	J4-0
34	J1-C2	J2-C2	J3-C1	J4-0
35	J1-C2	J2-C2	J3-C2	J4-0
36	J1-0	J2-0	J3-C1	J4-C1
37	J1-0	J2-0	J3-C2	J4-C1
38	J1-0	J2-0	J3-C1	J4-C2
39	J1-0	J2-0	J3-C2	J4-C1
40	J1-0	J2-C1	J3-0	J4-C1
41	J1-0	J2-C2	J3-0	J4-C2
42	J1-0	J2-C1	J3-0	J4-C2
43	J1-0	J2-C2	J3-0	J4-C1
44	J1-0	J2-C1	J3-C1	J4-0
45	J1-0	J2-C2	J3-C2	J4-0
46	J1-0	J2-C1	J3-C2	J4-0
47	J1-0	J2-C2	J3-C1	J4-0
48	J1-C1	J2-0	J3-0	J4-C1
49	J1-C2	J2-0	J3-0	J4-C2
50	J1-C1	J2-0	J3-0	J4-C2
51	J1-C2	J2-0	J3-0	J4-C1
52	J1-C1	J2-0	J3-C1	J4-0
53	J1-C2	J2-0	J3-C2	J4-0
54	J1-C1	J2-0	J3-C2	J4-0
55	J1-C2	J2-0	J3-C1	J4-0
56	J1-C1	J2-C1	J3-0	J4-0
57	J1-C2	J2-C2	J3-0	J4-0
58	J1-C1	J2-C2	J3-0	J4-0
59	J1-C2	J2-C1	J3-0	J4-0
60	J1-0	J2-0	J3-0	J4-C1
61	J1-0	J2-0	J3-0	J4-C2
62	J1-0	J2-0	J3-C1	J4-0
63	J1-0	J2-0	J3-C2	J4-0
64	J1-C1	J2-0	J3-0	J4-0
65	J1-C2	J2-0	J3-0	J4-0
66	J1-0	J2-C1	J3-0	J4-0
67	J1-0	J2-C2	J3-0	J4-0
68	J1-C1	J2-C1	J3-C2	J4-C2
69	J1-C1	J2-C2	J3-C1	J4-C2
70	J1-C2	J2-C1	J3-C1	J4-C2
71	J1-C2	J2-C1	J3-C2	J4-C1
72	J1-C1	J2-C2	J3-C2	J4-C1
73	J1-C2	J2-C2	J3-C1	J4-C1
74	J1-C1	J2-C1	J3-C1	J4-C2
75	J1-C1	J2-C1	J3-C2	J4-C1
76	J1-C2	J2-C1	J3-C1	J4-C1
77	J1-C1	J2-C2	J3-C1	J4-C1
78	J1-C2	J2-C2	J3-C2	J4-C1
79	J1-C2	J2-C2	J3-C1	J4-C2
80	J1-C2	J2-C1	J3-C2	J4-C2
81	J1-C1	J2-C2	J3-C2	J4-C2

Table A.2: Schedule Options for Offloading with 1024 Distinct Schedules

Option	Job 1	Job 2	Job 3	Job 4	Job 5
1	J1-0	J2-0	J3-0	J4-0	J5-0
2	J1-0	J2-0	J3-0	J4-0	J5-C1
3	J1-0	J2-0	J3-0	J4-0	J5-C2
4	J1-0	J2-0	J3-0	J4-0	J5-C3
5	J1-0	J2-0	J3-0	J4-C1	J5-0
6	J1-0	J2-0	J3-0	J4-C1	J5-C1
7	J1-0	J2-0	J3-0	J4-C1	J5-C2
8	J1-0	J2-0	J3-0	J4-C1	J5-C3
9	J1-0	J2-0	J3-0	J4-C2	J5-0
10	J1-0	J2-0	J3-0	J4-C2	J5-C1
11	J1-0	J2-0	J3-0	J4-C2	J5-C2
12	J1-0	J2-0	J3-0	J4-C2	J5-C3
13	J1-0	J2-0	J3-0	J4-C3	J5-0
14	J1-0	J2-0	J3-0	J4-C3	J5-C1
15	J1-0	J2-0	J3-0	J4-C3	J5-C2
16	J1-0	J2-0	J3-0	J4-C3	J5-C3
17	J1-0	J2-0	J3-C1	J4-0	J5-0
18	J1-0	J2-0	J3-C1	J4-0	J5-C1
19	J1-0	J2-0	J3-C1	J4-0	J5-C2
20	J1-0	J2-0	J3-C1	J4-0	J5-C3
21	J1-0	J2-0	J3-C1	J4-C1	J5-0
22	J1-0	J2-0	J3-C1	J4-C1	J5-C1
23	J1-0	J2-0	J3-C1	J4-C1	J5-C2
24	J1-0	J2-0	J3-C1	J4-C1	J5-C3
25	J1-0	J2-0	J3-C1	J4-C2	J5-0
26	J1-0	J2-0	J3-C1	J4-C2	J5-C1
27	J1-0	J2-0	J3-C1	J4-C2	J5-C2
28	J1-0	J2-0	J3-C1	J4-C2	J5-C3
29	J1-0	J2-0	J3-C1	J4-C3	J5-0
30	J1-0	J2-0	J3-C1	J4-C3	J5-C1
31	J1-0	J2-0	J3-C1	J4-C3	J5-C2
32	J1-0	J2-0	J3-C1	J4-C3	J5-C3
33	J1-0	J2-0	J3-C2	J4-0	J5-0
34	J1-0	J2-0	J3-C2	J4-0	J5-C1
35	J1-0	J2-0	J3-C2	J4-0	J5-C2
36	J1-0	J2-0	J3-C2	J4-0	J5-C3
37	J1-0	J2-0	J3-C2	J4-C1	J5-0
38	J1-0	J2-0	J3-C2	J4-C1	J5-C1
39	J1-0	J2-0	J3-C2	J4-C1	J5-C2
40	J1-0	J2-0	J3-C2	J4-C1	J5-C3
41	J1-0	J2-0	J3-C2	J4-C2	J5-0
42	J1-0	J2-0	J3-C2	J4-C2	J5-C1
43	J1-0	J2-0	J3-C2	J4-C2	J5-C2
44	J1-0	J2-0	J3-C2	J4-C2	J5-C3
45	J1-0	J2-0	J3-C2	J4-C3	J5-0
46	J1-0	J2-0	J3-C2	J4-C3	J5-C1
47	J1-0	J2-0	J3-C2	J4-C3	J5-C2
48	J1-0	J2-0	J3-C2	J4-C3	J5-C3
49	J1-0	J2-0	J3-C3	J4-0	J5-0
50	J1-0	J2-0	J3-C3	J4-0	J5-C1
51	J1-0	J2-0	J3-C3	J4-0	J5-C2
52	J1-0	J2-0	J3-C3	J4-0	J5-C3
53	J1-0	J2-0	J3-C3	J4-C1	J5-0
54	J1-0	J2-0	J3-C3	J4-C1	J5-C1
55	J1-0	J2-0	J3-C3	J4-C1	J5-C2
56	J1-0	J2-0	J3-C3	J4-C1	J5-C3
57	J1-0	J2-0	J3-C3	J4-C2	J5-0
58	J1-0	J2-0	J3-C3	J4-C2	J5-C1
59	J1-0	J2-0	J3-C3	J4-C2	J5-C2
60	J1-0	J2-0	J3-C3	J4-C2	J5-C3
61	J1-0	J2-0	J3-C3	J4-C3	J5-0
62	J1-0	J2-0	J3-C3	J4-C3	J5-C1
63	J1-0	J2-0	J3-C3	J4-C3	J5-C2
64	J1-0	J2-0	J3-C3	J4-C3	J5-C3
65	J1-0	J2-C1	J3-0	J4-0	J5-0
66	J1-0	J2-C1	J3-0	J4-0	J5-C1
67	J1-0	J2-C1	J3-0	J4-0	J5-C2
68	J1-0	J2-C1	J3-0	J4-0	J5-C3
69	J1-0	J2-C1	J3-0	J4-C1	J5-0
70	J1-0	J2-C1	J3-0	J4-C1	J5-C1
71	J1-0	J2-C1	J3-0	J4-C1	J5-C2
72	J1-0	J2-C1	J3-0	J4-C1	J5-C3
73	J1-0	J2-C1	J3-0	J4-C2	J5-0
74	J1-0	J2-C1	J3-0	J4-C2	J5-C1
75	J1-0	J2-C1	J3-0	J4-C2	J5-C2
76	J1-0	J2-C1	J3-0	J4-C2	J5-C3
77	J1-0	J2-C1	J3-0	J4-C3	J5-0
78	J1-0	J2-C1	J3-0	J4-C3	J5-C1
79	J1-0	J2-C1	J3-0	J4-C3	J5-C2
80	J1-0	J2-C1	J3-0	J4-C3	J5-C3
81	J1-0	J2-C1	J3-C1	J4-0	J5-0
82	J1-0	J2-C1	J3-C1	J4-0	J5-C1
83	J1-0	J2-C1	J3-C1	J4-0	J5-C2
84	J1-0	J2-C1	J3-C1	J4-0	J5-C3
85	J1-0	J2-C1	J3-C1	J4-C1	J5-0
86	J1-0	J2-C1	J3-C1	J4-C1	J5-C1
87	J1-0	J2-C1	J3-C1	J4-C1	J5-C2

Option	Job 1	Job 2	Job 3	Job 4	Job 5
88	J1-0	J2-C1	J3-C1	J4-C1	J5-C3
89	J1-0	J2-C1	J3-C1	J4-C2	J5-0
90	J1-0	J2-C1	J3-C1	J4-C2	J5-C1
91	J1-0	J2-C1	J3-C1	J4-C2	J5-C2
92	J1-0	J2-C1	J3-C1	J4-C2	J5-C3
93	J1-0	J2-C1	J3-C1	J4-C3	J5-0
94	J1-0	J2-C1	J3-C1	J4-C3	J5-C1
95	J1-0	J2-C1	J3-C1	J4-C3	J5-C2
96	J1-0	J2-C1	J3-C1	J4-C3	J5-C3
97	J1-0	J2-C1	J3-C2	J4-0	J5-0
98	J1-0	J2-C1	J3-C2	J4-0	J5-C1
99	J1-0	J2-C1	J3-C2	J4-0	J5-C2
100	J1-0	J2-C1	J3-C2	J4-0	J5-C3
101	J1-0	J2-C1	J3-C2	J4-C1	J5-0
102	J1-0	J2-C1	J3-C2	J4-C1	J5-C1
103	J1-0	J2-C1	J3-C2	J4-C1	J5-C2
104	J1-0	J2-C1	J3-C2	J4-C1	J5-C3
105	J1-0	J2-C1	J3-C2	J4-C2	J5-0
106	J1-0	J2-C1	J3-C2	J4-C2	J5-C1
107	J1-0	J2-C1	J3-C2	J4-C2	J5-C2
108	J1-0	J2-C1	J3-C2	J4-C2	J5-C3
109	J1-0	J2-C1	J3-C2	J4-C3	J5-0
110	J1-0	J2-C1	J3-C2	J4-C3	J5-C1
111	J1-0	J2-C1	J3-C2	J4-C3	J5-C2
112	J1-0	J2-C1	J3-C2	J4-C3	J5-C3
113	J1-0	J2-C1	J3-C3	J4-0	J5-0
114	J1-0	J2-C1	J3-C3	J4-0	J5-C1
115	J1-0	J2-C1	J3-C3	J4-0	J5-C2
116	J1-0	J2-C1	J3-C3	J4-0	J5-C3
117	J1-0	J2-C1	J3-C3	J4-C1	J5-0
118	J1-0	J2-C1	J3-C3	J4-C1	J5-C1
119	J1-0	J2-C1	J3-C3	J4-C1	J5-C2
120	J1-0	J2-C1	J3-C3	J4-C1	J5-C3
121	J1-0	J2-C1	J3-C3	J4-C2	J5-0
122	J1-0	J2-C1	J3-C3	J4-C2	J5-C1
123	J1-0	J2-C1	J3-C3	J4-C2	J5-C2
124	J1-0	J2-C1	J3-C3	J4-C2	J5-C3
125	J1-0	J2-C1	J3-C3	J4-C3	J5-0
126	J1-0	J2-C1	J3-C3	J4-C3	J5-C1
127	J1-0	J2-C1	J3-C3	J4-C3	J5-C2
128	J1-0	J2-C1	J3-C3	J4-C3	J5-C3
129	J1-0	J2-C2	J3-0	J4-0	J5-0
130	J1-0	J2-C2	J3-0	J4-0	J5-C1
131	J1-0	J2-C2	J3-0	J4-0	J5-C2
132	J1-0	J2-C2	J3-0	J4-0	J5-C3
133	J1-0	J2-C2	J3-0	J4-C1	J5-0
134	J1-0	J2-C2	J3-0	J4-C1	J5-C1
135	J1-0	J2-C2	J3-0	J4-C1	J5-C2
136	J1-0	J2-C2	J3-0	J4-C1	J5-C3
137	J1-0	J2-C2	J3-0	J4-C2	J5-0
138	J1-0	J2-C2	J3-0	J4-C2	J5-C1
139	J1-0	J2-C2	J3-0	J4-C2	J5-C2
140	J1-0	J2-C2	J3-0	J4-C2	J5-C3
141	J1-0	J2-C2	J3-0	J4-C3	J5-0
142	J1-0	J2-C2	J3-0	J4-C3	J5-C1
143	J1-0	J2-C2	J3-0	J4-C3	J5-C2
144	J1-0	J2-C2	J3-0	J4-C3	J5-C3
145	J1-0	J2-C2	J3-C1	J4-0	J5-0
146	J1-0	J2-C2	J3-C1	J4-0	J5-C1
147	J1-0	J2-C2	J3-C1	J4-0	J5-C2
148	J1-0	J2-C2	J3-C1	J4-0	J5-C3
149	J1-0	J2-C2	J3-C1	J4-C1	J5-0
150	J1-0	J2-C2	J3-C1	J4-C1	J5-C1
151	J1-0	J2-C2	J3-C1	J4-C1	J5-C2
152	J1-0	J2-C2	J3-C1	J4-C1	J5-C3
153	J1-0	J2-C2	J3-C1	J4-C2	J5-0
154	J1-0	J2-C2	J3-C1	J4-C2	J5-C1
155	J1-0	J2-C2	J3-C1	J4-C2	J5-C2
156	J1-0	J2-C2	J3-C1	J4-C2	J5-C3
157	J1-0	J2-C2	J3-C1	J4-C3	J5-0
158	J1-0	J2-C2	J3-C1	J4-C3	J5-C1
159	J1-0	J2-C2	J3-C1	J4-C3	J5-C2
160	J1-0	J2-C2	J3-C1	J4-C3	J5-C3
161	J1-0	J2-C2	J3-C2	J4-0	J5-0
162	J1-0	J2-C2	J3-C2	J4-0	J5-C1
163	J1-0	J2-C2	J3-C2	J4-0	J5-C2
164	J1-0	J2-C2	J3-C2	J4-0	J5-C3
165	J1-0	J2-C2	J3-C2	J4-C1	J5-0
166	J1-0	J2-C2	J3-C2	J4-C1	J5-C1
167	J1-0	J2-C2	J3-C2	J4-C1	J5-C2
168	J1-0	J2-C2	J3-C2	J4-C1	J5-C3
169	J1-0	J2-C2	J3-C2	J4-C2	J5-0
170	J1-0	J2-C2	J3-C2	J4-C2	J5-C1
171	J1-0	J2-C2	J3-C2	J4-C2	J5-C2
172	J1-0	J2-C2	J3-C2	J4-C2	J5-C3
173	J1-0	J2-C2	J3-C2	J4-C3	J5-0
174	J1-0	J2-C2	J3-C2	J4-C3	J5-C1
175	J1-0	J2-C2	J3-C2	J4-C3	J5-C2
176	J1-0	J2-C2	J3-C2	J4-C3	J5-C3
177	J1-0	J2-C2	J3-C3	J4-0	J5-0

Option	Job 1	Job 2	Job 3	Job 4	Job 5
178	J1-0	J2-C2	J3-C3	J4-0	J5-C1
179	J1-0	J2-C2	J3-C3	J4-0	J5-C2
180	J1-0	J2-C2	J3-C3	J4-0	J5-C3
181	J1-0	J2-C2	J3-C3	J4-C1	J5-0
182	J1-0	J2-C2	J3-C3	J4-C1	J5-C1
183	J1-0	J2-C2	J3-C3	J4-C1	J5-C2
184	J1-0	J2-C2	J3-C3	J4-C1	J5-C3
185	J1-0	J2-C2	J3-C3	J4-C2	J5-0
186	J1-0	J2-C2	J3-C3	J4-C2	J5-C1
187	J1-0	J2-C2	J3-C3	J4-C2	J5-C2
188	J1-0	J2-C2	J3-C3	J4-C2	J5-C3
189	J1-0	J2-C2	J3-C3	J4-C3	J5-0
190	J1-0	J2-C2	J3-C3	J4-C3	J5-C1
191	J1-0	J2-C2	J3-C3	J4-C3	J5-C2
192	J1-0	J2-C2	J3-C3	J4-C3	J5-C3
193	J1-0	J2-C3	J3-0	J4-0	J5-0
194	J1-0	J2-C3	J3-0	J4-0	J5-C1
195	J1-0	J2-C3	J3-0	J4-0	J5-C2
196	J1-0	J2-C3	J3-0	J4-0	J5-C3
197	J1-0	J2-C3	J3-0	J4-C1	J5-0
198	J1-0	J2-C3	J3-0	J4-C1	J5-C1
199	J1-0	J2-C3	J3-0	J4-C1	J5-C2
200	J1-0	J2-C3	J3-0	J4-C1	J5-C3
201	J1-0	J2-C3	J3-0	J4-C2	J5-0
202	J1-0	J2-C3	J3-0	J4-C2	J5-C1
203	J1-0	J2-C3	J3-0	J4-C2	J5-C2
204	J1-0	J2-C3	J3-0	J4-C2	J5-C3
205	J1-0	J2-C3	J3-0	J4-C3	J5-0
206	J1-0	J2-C3	J3-0	J4-C3	J5-C1
207	J1-0	J2-C3	J3-0	J4-C3	J5-C2
208	J1-0	J2-C3	J3-0	J4-C3	J5-C3
209	J1-0	J2-C3	J3-C1	J4-0	J5-0
210	J1-0	J2-C3	J3-C1	J4-0	J5-C1
211	J1-0	J2-C3	J3-C1	J4-0	J5-C2
212	J1-0	J2-C3	J3-C1	J4-0	J5-C3
213	J1-0	J2-C3	J3-C1	J4-C1	J5-0
214	J1-0	J2-C3	J3-C1	J4-C1	J5-C1
215	J1-0	J2-C3	J3-C1	J4-C1	J5-C2
216	J1-0	J2-C3	J3-C1	J4-C1	J5-C3
217	J1-0	J2-C3	J3-C1	J4-C2	J5-0
218	J1-0	J2-C3	J3-C1	J4-C2	J5-C1
219	J1-0	J2-C3	J3-C1	J4-C2	J5-C2
220	J1-0	J2-C3	J3-C1	J4-C2	J5-C3
221	J1-0	J2-C3	J3-C1	J4-C3	J5-0
222	J1-0	J2-C3	J3-C1	J4-C3	J5-C1
223	J1-0	J2-C3	J3-C1	J4-C3	J5-C2
224	J1-0	J2-C3	J3-C1	J4-C3	J5-C3
225	J1-0	J2-C3	J3-C2	J4-0	J5-0
226	J1-0	J2-C3	J3-C2	J4-0	J5-C1
227	J1-0	J2-C3	J3-C2	J4-0	J5-C2
228	J1-0	J2-C3	J3-C2	J4-0	J5-C3
229	J1-0	J2-C3	J3-C2	J4-C1	J5-0
230	J1-0	J2-C3	J3-C2	J4-C1	J5-C1
231	J1-0	J2-C3	J3-C2	J4-C1	J5-C2
232	J1-0	J2-C3	J3-C2	J4-C1	J5-C3
233	J1-0	J2-C3	J3-C2	J4-C2	J5-0
234	J1-0	J2-C3	J3-C2	J4-C2	J5-C1
235	J1-0	J2-C3	J3-C2	J4-C2	J5-C2
236	J1-0	J2-C3	J3-C2	J4-C2	J5-C3
237	J1-0	J2-C3	J3-C2	J4-C3	J5-0
238	J1-0	J2-C3	J3-C2	J4-C3	J5-C1
239	J1-0	J2-C3	J3-C2	J4-C3	J5-C2
240	J1-0	J2-C3	J3-C2	J4-C3	J5-C3
241	J1-0	J2-C3	J3-C3	J4-0	J5-0
242	J1-0	J2-C3	J3-C3	J4-0	J5-C1
243	J1-0	J2-C3	J3-C3	J4-0	J5-C2
244	J1-0	J2-C3	J3-C3	J4-0	J5-C3
245	J1-0	J2-C3	J3-C3	J4-C1	J5-0
246	J1-0	J2-C3	J3-C3	J4-C1	J5-C1
247	J1-0	J2-C3	J3-C3	J4-C1	J5-C2
248	J1-0	J2-C3	J3-C3	J4-C1	J5-C3
249	J1-0	J2-C3	J3-C3	J4-C2	J5-0
250	J1-0	J2-C3	J3-C3	J4-C2	J5-C1
251	J1-0	J2-C3	J3-C3	J4-C2	J5-C2
252	J1-0	J2-C3	J3-C3	J4-C2	J5-C3
253	J1-0	J2-C3	J3-C3	J4-C3	J5-0
254	J1-0	J2-C3	J3-C3	J4-C3	J5-C1
255	J1-0	J2-C3	J3-C3	J4-C3	J5-C2
256	J1-0	J2-C3	J3-C3	J4-C3	J5-C3
257	J1-C1	J2-0	J3-0	J4-0	J5-0
258	J1-C1	J2-0	J3-0	J4-0	J5-C1
259	J1-C1	J2-0	J3-0	J4-0	J5-C2
260	J1-C1	J2-0	J3-0	J4-0	J5-C3
261	J1-C1	J2-0	J3-0	J4-C1	J5-0
262	J1-C1	J2-0	J3-0	J4-C1	J5-C1
263	J1-C1	J2-0	J3-0	J4-C1	J5-C2
264	J1-C1	J2-0	J3-0	J4-C1	J5-C3
265	J1-C1	J2-0	J3-0	J4-C2	J5-0
266	J1-C1	J2-0	J3-0	J4-C2	J5-C1
267	J1-C1	J2-0	J3-0	J4-C2	J5-C2

Option	Job 1	Job 2	Job 3	Job 4	Job 5
268	J1-C1	J2-0	J3-0	J4-C2	J5-C3
269	J1-C1	J2-0	J3-0	J4-C3	J5-0
270	J1-C1	J2-0	J3-0	J4-C3	J5-C1
271	J1-C1	J2-0	J3-0	J4-C3	J5-C2
272	J1-C1	J2-0	J3-0	J4-C3	J5-C3
273	J1-C1	J2-0	J3-C1	J4-0	J5-0
274	J1-C1	J2-0	J3-C1	J4-0	J5-C1
275	J1-C1	J2-0	J3-C1	J4-0	J5-C2
276	J1-C1	J2-0	J3-C1	J4-0	J5-C3
277	J1-C1	J2-0	J3-C1	J4-C1	J5-0
278	J1-C1	J2-0	J3-C1	J4-C1	J5-C1
279	J1-C1	J2-0	J3-C1	J4-C1	J5-C2
280	J1-C1	J2-0	J3-C1	J4-C1	J5-C3
281	J1-C1	J2-0	J3-C1	J4-C2	J5-0
282	J1-C1	J2-0	J3-C1	J4-C2	J5-C1
283	J1-C1	J2-0	J3-C1	J4-C2	J5-C2
284	J1-C1	J2-0	J3-C1	J4-C2	J5-C3
285	J1-C1	J2-0	J3-C1	J4-C3	J5-0
286	J1-C1	J2-0	J3-C1	J4-C3	J5-C1
287	J1-C1	J2-0	J3-C1	J4-C3	J5-C2
288	J1-C1	J2-0	J3-C1	J4-C3	J5-C3
289	J1-C1	J2-0	J3-C2	J4-0	J5-0
290	J1-C1	J2-0	J3-C2	J4-0	J5-C1
291	J1-C1	J2-0	J3-C2	J4-0	J5-C2
292	J1-C1	J2-0	J3-C2	J4-0	J5-C3
293	J1-C1	J2-0	J3-C2	J4-C1	J5-0
294	J1-C1	J2-0	J3-C2	J4-C1	J5-C1
295	J1-C1	J2-0	J3-C2	J4-C1	J5-C2
296	J1-C1	J2-0	J3-C2	J4-C1	J5-C3
297	J1-C1	J2-0	J3-C2	J4-C2	J5-0
298	J1-C1	J2-0	J3-C2	J4-C2	J5-C1
299	J1-C1	J2-0	J3-C2	J4-C2	J5-C2
300	J1-C1	J2-0	J3-C2	J4-C2	J5-C3
301	J1-C1	J2-0	J3-C2	J4-C3	J5-0
302	J1-C1	J2-0	J3-C2	J4-C3	J5-C1
303	J1-C1	J2-0	J3-C2	J4-C3	J5-C2
304	J1-C1	J2-0	J3-C2	J4-C3	J5-C3
305	J1-C1	J2-0	J3-C3	J4-0	J5-0
306	J1-C1	J2-0	J3-C3	J4-0	J5-C1
307	J1-C1	J2-0	J3-C3	J4-0	J5-C2
308	J1-C1	J2-0	J3-C3	J4-0	J5-C3
309	J1-C1	J2-0	J3-C3	J4-C1	J5-0
310	J1-C1	J2-0	J3-C3	J4-C1	J5-C1
311	J1-C1	J2-0	J3-C3	J4-C1	J5-C2
312	J1-C1	J2-0	J3-C3	J4-C1	J5-C3
313	J1-C1	J2-0	J3-C3	J4-C2	J5-0
314	J1-C1	J2-0	J3-C3	J4-C2	J5-C1
315	J1-C1	J2-0	J3-C3	J4-C2	J5-C2
316	J1-C1	J2-0	J3-C3	J4-C2	J5-C3
317	J1-C1	J2-0	J3-C3	J4-C3	J5-0
318	J1-C1	J2-0	J3-C3	J4-C3	J5-C1
319	J1-C1	J2-0	J3-C3	J4-C3	J5-C2
320	J1-C1	J2-0	J3-C3	J4-C3	J5-C3
321	J1-C1	J2-C1	J3-0	J4-0	J5-0
322	J1-C1	J2-C1	J3-0	J4-0	J5-C1
323	J1-C1	J2-C1	J3-0	J4-0	J5-C2
324	J1-C1	J2-C1	J3-0	J4-0	J5-C3
325	J1-C1	J2-C1	J3-0	J4-C1	J5-0
326	J1-C1	J2-C1	J3-0	J4-C1	J5-C1
327	J1-C1	J2-C1	J3-0	J4-C1	J5-C2
328	J1-C1	J2-C1	J3-0	J4-C1	J5-C3
329	J1-C1	J2-C1	J3-0	J4-C2	J5-0
330	J1-C1	J2-C1	J3-0	J4-C2	J5-C1
331	J1-C1	J2-C1	J3-0	J4-C2	J5-C2
332	J1-C1	J2-C1	J3-0	J4-C2	J5-C3
333	J1-C1	J2-C1	J3-0	J4-C3	J5-0
334	J1-C1	J2-C1	J3-0	J4-C3	J5-C1
335	J1-C1	J2-C1	J3-0	J4-C3	J5-C2
336	J1-C1	J2-C1	J3-0	J4-C3	J5-C3
337	J1-C1	J2-C1	J3-C1	J4-0	J5-0
338	J1-C1	J2-C1	J3-C1	J4-0	J5-C1
339	J1-C1	J2-C1	J3-C1	J4-0	J5-C2
340	J1-C1	J2-C1	J3-C1	J4-0	J5-C3
341	J1-C1	J2-C1	J3-C1	J4-C1	J5-0
342	J1-C1	J2-C1	J3-C1	J4-C1	J5-C1
343	J1-C1	J2-C1	J3-C1	J4-C1	J5-C2
344	J1-C1	J2-C1	J3-C1	J4-C1	J5-C3
345	J1-C1	J2-C1	J3-C1	J4-C2	J5-0
346	J1-C1	J2-C1	J3-C1	J4-C2	J5-C1
347	J1-C1	J2-C1	J3-C1	J4-C2	J5-C2
348	J1-C1	J2-C1	J3-C1	J4-C2	J5-C3
349	J1-C1	J2-C1	J3-C1	J4-C3	J5-0
350	J1-C1	J2-C1	J3-C1	J4-C3	J5-C1
351	J1-C1	J2-C1	J3-C1	J4-C3	J5-C2
352	J1-C1	J2-C1	J3-C1	J4-C3	J5-C3
353	J1-C1	J2-C1	J3-C2	J4-0	J5-0
354	J1-C1	J2-C1	J3-C2	J4-0	J5-C1
355	J1-C1	J2-C1	J3-C2	J4-0	J5-C2
356	J1-C1	J2-C1	J3-C2	J4-0	J5-C3
357	J1-C1	J2-C1	J3-C2	J4-C1	J5-0

Option	Job 1	Job 2	Job 3	Job 4	Job 5
358	J1-C1	J2-C1	J3-C2	J4-C1	J5-C1
359	J1-C1	J2-C1	J3-C2	J4-C1	J5-C2
360	J1-C1	J2-C1	J3-C2	J4-C1	J5-C3
361	J1-C1	J2-C1	J3-C2	J4-C2	J5-0
362	J1-C1	J2-C1	J3-C2	J4-C2	J5-C1
363	J1-C1	J2-C1	J3-C2	J4-C2	J5-C2
364	J1-C1	J2-C1	J3-C2	J4-C2	J5-C3
365	J1-C1	J2-C1	J3-C2	J4-C3	J5-0
366	J1-C1	J2-C1	J3-C2	J4-C3	J5-C1
367	J1-C1	J2-C1	J3-C2	J4-C3	J5-C2
368	J1-C1	J2-C1	J3-C2	J4-C3	J5-C3
369	J1-C1	J2-C1	J3-C3	J4-0	J5-0
370	J1-C1	J2-C1	J3-C3	J4-0	J5-C1
371	J1-C1	J2-C1	J3-C3	J4-0	J5-C2
372	J1-C1	J2-C1	J3-C3	J4-0	J5-C3
373	J1-C1	J2-C1	J3-C3	J4-C1	J5-0
374	J1-C1	J2-C1	J3-C3	J4-C1	J5-C1
375	J1-C1	J2-C1	J3-C3	J4-C1	J5-C2
376	J1-C1	J2-C1	J3-C3	J4-C1	J5-C3
377	J1-C1	J2-C1	J3-C3	J4-C2	J5-0
378	J1-C1	J2-C1	J3-C3	J4-C2	J5-C1
379	J1-C1	J2-C1	J3-C3	J4-C2	J5-C2
380	J1-C1	J2-C1	J3-C3	J4-C2	J5-C3
381	J1-C1	J2-C1	J3-C3	J4-C3	J5-0
382	J1-C1	J2-C1	J3-C3	J4-C3	J5-C1
383	J1-C1	J2-C1	J3-C3	J4-C3	J5-C2
384	J1-C1	J2-C1	J3-C3	J4-C3	J5-C3
385	J1-C1	J2-C2	J3-0	J4-0	J5-0
386	J1-C1	J2-C2	J3-0	J4-0	J5-C1
387	J1-C1	J2-C2	J3-0	J4-0	J5-C2
388	J1-C1	J2-C2	J3-0	J4-0	J5-C3
389	J1-C1	J2-C2	J3-0	J4-C1	J5-0
390	J1-C1	J2-C2	J3-0	J4-C1	J5-C1
391	J1-C1	J2-C2	J3-0	J4-C1	J5-C2
392	J1-C1	J2-C2	J3-0	J4-C1	J5-C3
393	J1-C1	J2-C2	J3-0	J4-C2	J5-0
394	J1-C1	J2-C2	J3-0	J4-C2	J5-C1
395	J1-C1	J2-C2	J3-0	J4-C2	J5-C2
396	J1-C1	J2-C2	J3-0	J4-C2	J5-C3
397	J1-C1	J2-C2	J3-0	J4-C3	J5-0
398	J1-C1	J2-C2	J3-0	J4-C3	J5-C1
399	J1-C1	J2-C2	J3-0	J4-C3	J5-C2
400	J1-C1	J2-C2	J3-0	J4-C3	J5-C3
401	J1-C1	J2-C2	J3-C1	J4-0	J5-0
402	J1-C1	J2-C2	J3-C1	J4-0	J5-C1
403	J1-C1	J2-C2	J3-C1	J4-0	J5-C2
404	J1-C1	J2-C2	J3-C1	J4-0	J5-C3
405	J1-C1	J2-C2	J3-C1	J4-C1	J5-0
406	J1-C1	J2-C2	J3-C1	J4-C1	J5-C1
407	J1-C1	J2-C2	J3-C1	J4-C1	J5-C2
408	J1-C1	J2-C2	J3-C1	J4-C1	J5-C3
409	J1-C1	J2-C2	J3-C1	J4-C2	J5-0
410	J1-C1	J2-C2	J3-C1	J4-C2	J5-C1
411	J1-C1	J2-C2	J3-C1	J4-C2	J5-C2
412	J1-C1	J2-C2	J3-C1	J4-C2	J5-C3
413	J1-C1	J2-C2	J3-C1	J4-C3	J5-0
414	J1-C1	J2-C2	J3-C1	J4-C3	J5-C1
415	J1-C1	J2-C2	J3-C1	J4-C3	J5-C2
416	J1-C1	J2-C2	J3-C1	J4-C3	J5-C3
417	J1-C1	J2-C2	J3-C2	J4-0	J5-0
418	J1-C1	J2-C2	J3-C2	J4-0	J5-C1
419	J1-C1	J2-C2	J3-C2	J4-0	J5-C2
420	J1-C1	J2-C2	J3-C2	J4-0	J5-C3
421	J1-C1	J2-C2	J3-C2	J4-C1	J5-0
422	J1-C1	J2-C2	J3-C2	J4-C1	J5-C1
423	J1-C1	J2-C2	J3-C2	J4-C1	J5-C2
424	J1-C1	J2-C2	J3-C2	J4-C1	J5-C3
425	J1-C1	J2-C2	J3-C2	J4-C2	J5-0
426	J1-C1	J2-C2	J3-C2	J4-C2	J5-C1
427	J1-C1	J2-C2	J3-C2	J4-C2	J5-C2
428	J1-C1	J2-C2	J3-C2	J4-C2	J5-C3
429	J1-C1	J2-C2	J3-C2	J4-C3	J5-0
430	J1-C1	J2-C2	J3-C2	J4-C3	J5-C1
431	J1-C1	J2-C2	J3-C2	J4-C3	J5-C2
432	J1-C1	J2-C2	J3-C2	J4-C3	J5-C3
433	J1-C1	J2-C2	J3-C3	J4-0	J5-0
434	J1-C1	J2-C2	J3-C3	J4-0	J5-C1
435	J1-C1	J2-C2	J3-C3	J4-0	J5-C2
436	J1-C1	J2-C2	J3-C3	J4-0	J5-C3
437	J1-C1	J2-C2	J3-C3	J4-C1	J5-0
438	J1-C1	J2-C2	J3-C3	J4-C1	J5-C1
439	J1-C1	J2-C2	J3-C3	J4-C1	J5-C2
440	J1-C1	J2-C2	J3-C3	J4-C1	J5-C3
441	J1-C1	J2-C2	J3-C3	J4-C2	J5-0
442	J1-C1	J2-C2	J3-C3	J4-C2	J5-C1
443	J1-C1	J2-C2	J3-C3	J4-C2	J5-C2
444	J1-C1	J2-C2	J3-C3	J4-C2	J5-C3
445	J1-C1	J2-C2	J3-C3	J4-C3	J5-0
446	J1-C1	J2-C2	J3-C3	J4-C3	J5-C1
447	J1-C1	J2-C2	J3-C3	J4-C3	J5-C2

Option	Job 1	Job 2	Job 3	Job 4	Job 5
448	J1-C1	J2-C2	J3-C3	J4-C3	J5-C3
449	J1-C1	J2-C3	J3-0	J4-0	J5-0
450	J1-C1	J2-C3	J3-0	J4-0	J5-C1
451	J1-C1	J2-C3	J3-0	J4-0	J5-C2
452	J1-C1	J2-C3	J3-0	J4-0	J5-C3
453	J1-C1	J2-C3	J3-0	J4-C1	J5-0
454	J1-C1	J2-C3	J3-0	J4-C1	J5-C1
455	J1-C1	J2-C3	J3-0	J4-C1	J5-C2
456	J1-C1	J2-C3	J3-0	J4-C1	J5-C3
457	J1-C1	J2-C3	J3-0	J4-C2	J5-0
458	J1-C1	J2-C3	J3-0	J4-C2	J5-C1
459	J1-C1	J2-C3	J3-0	J4-C2	J5-C2
460	J1-C1	J2-C3	J3-0	J4-C2	J5-C3
461	J1-C1	J2-C3	J3-0	J4-C3	J5-0
462	J1-C1	J2-C3	J3-0	J4-C3	J5-C1
463	J1-C1	J2-C3	J3-0	J4-C3	J5-C2
464	J1-C1	J2-C3	J3-0	J4-C3	J5-C3
465	J1-C1	J2-C3	J3-C1	J4-0	J5-0
466	J1-C1	J2-C3	J3-C1	J4-0	J5-C1
467	J1-C1	J2-C3	J3-C1	J4-0	J5-C2
468	J1-C1	J2-C3	J3-C1	J4-0	J5-C3
469	J1-C1	J2-C3	J3-C1	J4-C1	J5-0
470	J1-C1	J2-C3	J3-C1	J4-C1	J5-C1
471	J1-C1	J2-C3	J3-C1	J4-C1	J5-C2
472	J1-C1	J2-C3	J3-C1	J4-C1	J5-C3
473	J1-C1	J2-C3	J3-C1	J4-C2	J5-0
474	J1-C1	J2-C3	J3-C1	J4-C2	J5-C1
475	J1-C1	J2-C3	J3-C1	J4-C2	J5-C2
476	J1-C1	J2-C3	J3-C1	J4-C2	J5-C3
477	J1-C1	J2-C3	J3-C1	J4-C3	J5-0
478	J1-C1	J2-C3	J3-C1	J4-C3	J5-C1
479	J1-C1	J2-C3	J3-C1	J4-C3	J5-C2
480	J1-C1	J2-C3	J3-C1	J4-C3	J5-C3
481	J1-C1	J2-C3	J3-C2	J4-0	J5-0
482	J1-C1	J2-C3	J3-C2	J4-0	J5-C1
483	J1-C1	J2-C3	J3-C2	J4-0	J5-C2
484	J1-C1	J2-C3	J3-C2	J4-0	J5-C3
485	J1-C1	J2-C3	J3-C2	J4-C1	J5-0
486	J1-C1	J2-C3	J3-C2	J4-C1	J5-C1
487	J1-C1	J2-C3	J3-C2	J4-C1	J5-C2
488	J1-C1	J2-C3	J3-C2	J4-C1	J5-C3
489	J1-C1	J2-C3	J3-C2	J4-C2	J5-0
490	J1-C1	J2-C3	J3-C2	J4-C2	J5-C1
491	J1-C1	J2-C3	J3-C2	J4-C2	J5-C2
492	J1-C1	J2-C3	J3-C2	J4-C2	J5-C3
493	J1-C1	J2-C3	J3-C2	J4-C3	J5-0
494	J1-C1	J2-C3	J3-C2	J4-C3	J5-C1
495	J1-C1	J2-C3	J3-C2	J4-C3	J5-C2
496	J1-C1	J2-C3	J3-C2	J4-C3	J5-C3
497	J1-C1	J2-C3	J3-C3	J4-0	J5-0
498	J1-C1	J2-C3	J3-C3	J4-0	J5-C1
499	J1-C1	J2-C3	J3-C3	J4-0	J5-C2
500	J1-C1	J2-C3	J3-C3	J4-0	J5-C3
501	J1-C1	J2-C3	J3-C3	J4-C1	J5-0
502	J1-C1	J2-C3	J3-C3	J4-C1	J5-C1
503	J1-C1	J2-C3	J3-C3	J4-C1	J5-C2
504	J1-C1	J2-C3	J3-C3	J4-C1	J5-C3
505	J1-C1	J2-C3	J3-C3	J4-C2	J5-0
506	J1-C1	J2-C3	J3-C3	J4-C2	J5-C1
507	J1-C1	J2-C3	J3-C3	J4-C2	J5-C2
508	J1-C1	J2-C3	J3-C3	J4-C2	J5-C3
509	J1-C1	J2-C3	J3-C3	J4-C3	J5-0
510	J1-C1	J2-C3	J3-C3	J4-C3	J5-C1
511	J1-C1	J2-C3	J3-C3	J4-C3	J5-C2
512	J1-C1	J2-C3	J3-C3	J4-C3	J5-C3
513	J1-C2	J2-0	J3-0	J4-0	J5-0
514	J1-C2	J2-0	J3-0	J4-0	J5-C1
515	J1-C2	J2-0	J3-0	J4-0	J5-C2
516	J1-C2	J2-0	J3-0	J4-0	J5-C3
517	J1-C2	J2-0	J3-0	J4-C1	J5-0
518	J1-C2	J2-0	J3-0	J4-C1	J5-C1
519	J1-C2	J2-0	J3-0	J4-C1	J5-C2
520	J1-C2	J2-0	J3-0	J4-C1	J5-C3
521	J1-C2	J2-0	J3-0	J4-C2	J5-0
522	J1-C2	J2-0	J3-0	J4-C2	J5-C1
523	J1-C2	J2-0	J3-0	J4-C2	J5-C2
524	J1-C2	J2-0	J3-0	J4-C2	J5-C3
525	J1-C2	J2-0	J3-0	J4-C3	J5-0
526	J1-C2	J2-0	J3-0	J4-C3	J5-C1
527	J1-C2	J2-0	J3-0	J4-C3	J5-C2
528	J1-C2	J2-0	J3-0	J4-C3	J5-C3
529	J1-C2	J2-0	J3-C1	J4-0	J5-0
530	J1-C2	J2-0	J3-C1	J4-0	J5-C1
531	J1-C2	J2-0	J3-C1	J4-0	J5-C2
532	J1-C2	J2-0	J3-C1	J4-0	J5-C3
533	J1-C2	J2-0	J3-C1	J4-C1	J5-0
534	J1-C2	J2-0	J3-C1	J4-C1	J5-C1
535	J1-C2	J2-0	J3-C1	J4-C1	J5-C2
536	J1-C2	J2-0	J3-C1	J4-C1	J5-C3
537	J1-C2	J2-0	J3-C1	J4-C2	J5-0

Option	Job 1	Job 2	Job 3	Job 4	Job 5
538	J1-C2	J2-0	J3-C1	J4-C2	J5-C1
539	J1-C2	J2-0	J3-C1	J4-C2	J5-C2
540	J1-C2	J2-0	J3-C1	J4-C2	J5-C3
541	J1-C2	J2-0	J3-C1	J4-C3	J5-0
542	J1-C2	J2-0	J3-C1	J4-C3	J5-C1
543	J1-C2	J2-0	J3-C1	J4-C3	J5-C2
544	J1-C2	J2-0	J3-C1	J4-C3	J5-C3
545	J1-C2	J2-0	J3-C2	J4-0	J5-0
546	J1-C2	J2-0	J3-C2	J4-0	J5-C1
547	J1-C2	J2-0	J3-C2	J4-0	J5-C2
548	J1-C2	J2-0	J3-C2	J4-0	J5-C3
549	J1-C2	J2-0	J3-C2	J4-C1	J5-0
550	J1-C2	J2-0	J3-C2	J4-C1	J5-C1
551	J1-C2	J2-0	J3-C2	J4-C1	J5-C2
552	J1-C2	J2-0	J3-C2	J4-C1	J5-C3
553	J1-C2	J2-0	J3-C2	J4-C2	J5-0
554	J1-C2	J2-0	J3-C2	J4-C2	J5-C1
555	J1-C2	J2-0	J3-C2	J4-C2	J5-C2
556	J1-C2	J2-0	J3-C2	J4-C2	J5-C3
557	J1-C2	J2-0	J3-C2	J4-C3	J5-0
558	J1-C2	J2-0	J3-C2	J4-C3	J5-C1
559	J1-C2	J2-0	J3-C2	J4-C3	J5-C2
560	J1-C2	J2-0	J3-C2	J4-C3	J5-C3
561	J1-C2	J2-0	J3-C3	J4-0	J5-0
562	J1-C2	J2-0	J3-C3	J4-0	J5-C1
563	J1-C2	J2-0	J3-C3	J4-0	J5-C2
564	J1-C2	J2-0	J3-C3	J4-0	J5-C3
565	J1-C2	J2-0	J3-C3	J4-C1	J5-0
566	J1-C2	J2-0	J3-C3	J4-C1	J5-C1
567	J1-C2	J2-0	J3-C3	J4-C1	J5-C2
568	J1-C2	J2-0	J3-C3	J4-C1	J5-C3
569	J1-C2	J2-0	J3-C3	J4-C2	J5-0
570	J1-C2	J2-0	J3-C3	J4-C2	J5-C1
571	J1-C2	J2-0	J3-C3	J4-C2	J5-C2
572	J1-C2	J2-0	J3-C3	J4-C2	J5-C3
573	J1-C2	J2-0	J3-C3	J4-C3	J5-0
574	J1-C2	J2-0	J3-C3	J4-C3	J5-C1
575	J1-C2	J2-0	J3-C3	J4-C3	J5-C2
576	J1-C2	J2-0	J3-C3	J4-C3	J5-C3
577	J1-C2	J2-C1	J3-0	J4-0	J5-0
578	J1-C2	J2-C1	J3-0	J4-0	J5-C1
579	J1-C2	J2-C1	J3-0	J4-0	J5-C2
580	J1-C2	J2-C1	J3-0	J4-0	J5-C3
581	J1-C2	J2-C1	J3-0	J4-C1	J5-0
582	J1-C2	J2-C1	J3-0	J4-C1	J5-C1
583	J1-C2	J2-C1	J3-0	J4-C1	J5-C2
584	J1-C2	J2-C1	J3-0	J4-C1	J5-C3
585	J1-C2	J2-C1	J3-0	J4-C2	J5-0
586	J1-C2	J2-C1	J3-0	J4-C2	J5-C1
587	J1-C2	J2-C1	J3-0	J4-C2	J5-C2
588	J1-C2	J2-C1	J3-0	J4-C2	J5-C3
589	J1-C2	J2-C1	J3-0	J4-C3	J5-0
590	J1-C2	J2-C1	J3-0	J4-C3	J5-C1
591	J1-C2	J2-C1	J3-0	J4-C3	J5-C2
592	J1-C2	J2-C1	J3-0	J4-C3	J5-C3
593	J1-C2	J2-C1	J3-C1	J4-0	J5-0
594	J1-C2	J2-C1	J3-C1	J4-0	J5-C1
595	J1-C2	J2-C1	J3-C1	J4-0	J5-C2
596	J1-C2	J2-C1	J3-C1	J4-0	J5-C3
597	J1-C2	J2-C1	J3-C1	J4-C1	J5-0
598	J1-C2	J2-C1	J3-C1	J4-C1	J5-C1
599	J1-C2	J2-C1	J3-C1	J4-C1	J5-C2
600	J1-C2	J2-C1	J3-C1	J4-C1	J5-C3
601	J1-C2	J2-C1	J3-C1	J4-C2	J5-0
602	J1-C2	J2-C1	J3-C1	J4-C2	J5-C1
603	J1-C2	J2-C1	J3-C1	J4-C2	J5-C2
604	J1-C2	J2-C1	J3-C1	J4-C2	J5-C3
605	J1-C2	J2-C1	J3-C1	J4-C3	J5-0
606	J1-C2	J2-C1	J3-C1	J4-C3	J5-C1
607	J1-C2	J2-C1	J3-C1	J4-C3	J5-C2
608	J1-C2	J2-C1	J3-C1	J4-C3	J5-C3
609	J1-C2	J2-C1	J3-C2	J4-0	J5-0
610	J1-C2	J2-C1	J3-C2	J4-0	J5-C1
611	J1-C2	J2-C1	J3-C2	J4-0	J5-C2
612	J1-C2	J2-C1	J3-C2	J4-0	J5-C3
613	J1-C2	J2-C1	J3-C2	J4-C1	J5-0
614	J1-C2	J2-C1	J3-C2	J4-C1	J5-C1
615	J1-C2	J2-C1	J3-C2	J4-C1	J5-C2
616	J1-C2	J2-C1	J3-C2	J4-C1	J5-C3
617	J1-C2	J2-C1	J3-C2	J4-C2	J5-0
618	J1-C2	J2-C1	J3-C2	J4-C2	J5-C1
619	J1-C2	J2-C1	J3-C2	J4-C2	J5-C2
620	J1-C2	J2-C1	J3-C2	J4-C2	J5-C3
621	J1-C2	J2-C1	J3-C2	J4-C3	J5-0
622	J1-C2	J2-C1	J3-C2	J4-C3	J5-C1
623	J1-C2	J2-C1	J3-C2	J4-C3	J5-C2
624	J1-C2	J2-C1	J3-C2	J4-C3	J5-C3
625	J1-C2	J2-C1	J3-C3	J4-0	J5-0
626	J1-C2	J2-C1	J3-C3	J4-0	J5-C1
627	J1-C2	J2-C1	J3-C3	J4-0	J5-C2

Option	Job 1	Job 2	Job 3	Job 4	Job 5
628	J1-C2	J2-C1	J3-C3	J4-0	J5-C3
629	J1-C2	J2-C1	J3-C3	J4-C1	J5-0
630	J1-C2	J2-C1	J3-C3	J4-C1	J5-C1
631	J1-C2	J2-C1	J3-C3	J4-C1	J5-C2
632	J1-C2	J2-C1	J3-C3	J4-C1	J5-C3
633	J1-C2	J2-C1	J3-C3	J4-C2	J5-0
634	J1-C2	J2-C1	J3-C3	J4-C2	J5-C1
635	J1-C2	J2-C1	J3-C3	J4-C2	J5-C2
636	J1-C2	J2-C1	J3-C3	J4-C2	J5-C3
637	J1-C2	J2-C1	J3-C3	J4-C3	J5-0
638	J1-C2	J2-C1	J3-C3	J4-C3	J5-C1
639	J1-C2	J2-C1	J3-C3	J4-C3	J5-C2
640	J1-C2	J2-C1	J3-C3	J4-C3	J5-C3
641	J1-C2	J2-C2	J3-0	J4-0	J5-0
642	J1-C2	J2-C2	J3-0	J4-0	J5-C1
643	J1-C2	J2-C2	J3-0	J4-0	J5-C2
644	J1-C2	J2-C2	J3-0	J4-0	J5-C3
645	J1-C2	J2-C2	J3-0	J4-C1	J5-0
646	J1-C2	J2-C2	J3-0	J4-C1	J5-C1
647	J1-C2	J2-C2	J3-0	J4-C1	J5-C2
648	J1-C2	J2-C2	J3-0	J4-C1	J5-C3
649	J1-C2	J2-C2	J3-0	J4-C2	J5-0
650	J1-C2	J2-C2	J3-0	J4-C2	J5-C1
651	J1-C2	J2-C2	J3-0	J4-C2	J5-C2
652	J1-C2	J2-C2	J3-0	J4-C2	J5-C3
653	J1-C2	J2-C2	J3-0	J4-C3	J5-0
654	J1-C2	J2-C2	J3-0	J4-C3	J5-C1
655	J1-C2	J2-C2	J3-0	J4-C3	J5-C2
656	J1-C2	J2-C2	J3-0	J4-C3	J5-C3
657	J1-C2	J2-C2	J3-C1	J4-0	J5-0
658	J1-C2	J2-C2	J3-C1	J4-0	J5-C1
659	J1-C2	J2-C2	J3-C1	J4-0	J5-C2
660	J1-C2	J2-C2	J3-C1	J4-0	J5-C3
661	J1-C2	J2-C2	J3-C1	J4-C1	J5-0
662	J1-C2	J2-C2	J3-C1	J4-C1	J5-C1
663	J1-C2	J2-C2	J3-C1	J4-C1	J5-C2
664	J1-C2	J2-C2	J3-C1	J4-C1	J5-C3
665	J1-C2	J2-C2	J3-C1	J4-C2	J5-0
666	J1-C2	J2-C2	J3-C1	J4-C2	J5-C1
667	J1-C2	J2-C2	J3-C1	J4-C2	J5-C2
668	J1-C2	J2-C2	J3-C1	J4-C2	J5-C3
669	J1-C2	J2-C2	J3-C1	J4-C3	J5-0
670	J1-C2	J2-C2	J3-C1	J4-C3	J5-C1
671	J1-C2	J2-C2	J3-C1	J4-C3	J5-C2
672	J1-C2	J2-C2	J3-C1	J4-C3	J5-C3
673	J1-C2	J2-C2	J3-C2	J4-0	J5-0
674	J1-C2	J2-C2	J3-C2	J4-0	J5-C1
675	J1-C2	J2-C2	J3-C2	J4-0	J5-C2
676	J1-C2	J2-C2	J3-C2	J4-0	J5-C3
677	J1-C2	J2-C2	J3-C2	J4-C1	J5-0
678	J1-C2	J2-C2	J3-C2	J4-C1	J5-C1
679	J1-C2	J2-C2	J3-C2	J4-C1	J5-C2
680	J1-C2	J2-C2	J3-C2	J4-C1	J5-C3
681	J1-C2	J2-C2	J3-C2	J4-C2	J5-0
682	J1-C2	J2-C2	J3-C2	J4-C2	J5-C1
683	J1-C2	J2-C2	J3-C2	J4-C2	J5-C2
684	J1-C2	J2-C2	J3-C2	J4-C2	J5-C3
685	J1-C2	J2-C2	J3-C2	J4-C3	J5-0
686	J1-C2	J2-C2	J3-C2	J4-C3	J5-C1
687	J1-C2	J2-C2	J3-C2	J4-C3	J5-C2
688	J1-C2	J2-C2	J3-C2	J4-C3	J5-C3
689	J1-C2	J2-C2	J3-C3	J4-0	J5-0
690	J1-C2	J2-C2	J3-C3	J4-0	J5-C1
691	J1-C2	J2-C2	J3-C3	J4-0	J5-C2
692	J1-C2	J2-C2	J3-C3	J4-0	J5-C3
693	J1-C2	J2-C2	J3-C3	J4-C1	J5-0
694	J1-C2	J2-C2	J3-C3	J4-C1	J5-C1
695	J1-C2	J2-C2	J3-C3	J4-C1	J5-C2
696	J1-C2	J2-C2	J3-C3	J4-C1	J5-C3
697	J1-C2	J2-C2	J3-C3	J4-C2	J5-0
698	J1-C2	J2-C2	J3-C3	J4-C2	J5-C1
699	J1-C2	J2-C2	J3-C3	J4-C2	J5-C2
700	J1-C2	J2-C2	J3-C3	J4-C2	J5-C3
701	J1-C2	J2-C2	J3-C3	J4-C3	J5-0
702	J1-C2	J2-C2	J3-C3	J4-C3	J5-C1
703	J1-C2	J2-C2	J3-C3	J4-C3	J5-C2
704	J1-C2	J2-C2	J3-C3	J4-C3	J5-C3
705	J1-C2	J2-C3	J3-0	J4-0	J5-0
706	J1-C2	J2-C3	J3-0	J4-0	J5-C1
707	J1-C2	J2-C3	J3-0	J4-0	J5-C2
708	J1-C2	J2-C3	J3-0	J4-0	J5-C3
709	J1-C2	J2-C3	J3-0	J4-C1	J5-0
710	J1-C2	J2-C3	J3-0	J4-C1	J5-C1
711	J1-C2	J2-C3	J3-0	J4-C1	J5-C2
712	J1-C2	J2-C3	J3-0	J4-C1	J5-C3
713	J1-C2	J2-C3	J3-0	J4-C2	J5-0
714	J1-C2	J2-C3	J3-0	J4-C2	J5-C1
715	J1-C2	J2-C3	J3-0	J4-C2	J5-C2
716	J1-C2	J2-C3	J3-0	J4-C2	J5-C3
717	J1-C2	J2-C3	J3-0	J4-C3	J5-0

Option	Job 1	Job 2	Job 3	Job 4	Job 5
718	J1-C2	J2-C3	J3-0	J4-C3	J5-C1
719	J1-C2	J2-C3	J3-0	J4-C3	J5-C2
720	J1-C2	J2-C3	J3-0	J4-C3	J5-C3
721	J1-C2	J2-C3	J3-C1	J4-0	J5-0
722	J1-C2	J2-C3	J3-C1	J4-0	J5-C1
723	J1-C2	J2-C3	J3-C1	J4-0	J5-C2
724	J1-C2	J2-C3	J3-C1	J4-0	J5-C3
725	J1-C2	J2-C3	J3-C1	J4-C1	J5-0
726	J1-C2	J2-C3	J3-C1	J4-C1	J5-C1
727	J1-C2	J2-C3	J3-C1	J4-C1	J5-C2
728	J1-C2	J2-C3	J3-C1	J4-C1	J5-C3
729	J1-C2	J2-C3	J3-C1	J4-C2	J5-0
730	J1-C2	J2-C3	J3-C1	J4-C2	J5-C1
731	J1-C2	J2-C3	J3-C1	J4-C2	J5-C2
732	J1-C2	J2-C3	J3-C1	J4-C2	J5-C3
733	J1-C2	J2-C3	J3-C1	J4-C3	J5-0
734	J1-C2	J2-C3	J3-C1	J4-C3	J5-C1
735	J1-C2	J2-C3	J3-C1	J4-C3	J5-C2
736	J1-C2	J2-C3	J3-C1	J4-C3	J5-C3
737	J1-C2	J2-C3	J3-C2	J4-0	J5-0
738	J1-C2	J2-C3	J3-C2	J4-0	J5-C1
739	J1-C2	J2-C3	J3-C2	J4-0	J5-C2
740	J1-C2	J2-C3	J3-C2	J4-0	J5-C3
741	J1-C2	J2-C3	J3-C2	J4-C1	J5-0
742	J1-C2	J2-C3	J3-C2	J4-C1	J5-C1
743	J1-C2	J2-C3	J3-C2	J4-C1	J5-C2
744	J1-C2	J2-C3	J3-C2	J4-C1	J5-C3
745	J1-C2	J2-C3	J3-C2	J4-C2	J5-0
746	J1-C2	J2-C3	J3-C2	J4-C2	J5-C1
747	J1-C2	J2-C3	J3-C2	J4-C2	J5-C2
748	J1-C2	J2-C3	J3-C2	J4-C2	J5-C3
749	J1-C2	J2-C3	J3-C2	J4-C3	J5-0
750	J1-C2	J2-C3	J3-C2	J4-C3	J5-C1
751	J1-C2	J2-C3	J3-C2	J4-C3	J5-C2
752	J1-C2	J2-C3	J3-C2	J4-C3	J5-C3
753	J1-C2	J2-C3	J3-C3	J4-0	J5-0
754	J1-C2	J2-C3	J3-C3	J4-0	J5-C1
755	J1-C2	J2-C3	J3-C3	J4-0	J5-C2
756	J1-C2	J2-C3	J3-C3	J4-0	J5-C3
757	J1-C2	J2-C3	J3-C3	J4-C1	J5-0
758	J1-C2	J2-C3	J3-C3	J4-C1	J5-C1
759	J1-C2	J2-C3	J3-C3	J4-C1	J5-C2
760	J1-C2	J2-C3	J3-C3	J4-C1	J5-C3
761	J1-C2	J2-C3	J3-C3	J4-C2	J5-0
762	J1-C2	J2-C3	J3-C3	J4-C2	J5-C1
763	J1-C2	J2-C3	J3-C3	J4-C2	J5-C2
764	J1-C2	J2-C3	J3-C3	J4-C2	J5-C3
765	J1-C2	J2-C3	J3-C3	J4-C3	J5-0
766	J1-C2	J2-C3	J3-C3	J4-C3	J5-C1
767	J1-C2	J2-C3	J3-C3	J4-C3	J5-C2
768	J1-C2	J2-C3	J3-C3	J4-C3	J5-C3
769	J1-C3	J2-0	J3-0	J4-0	J5-0
770	J1-C3	J2-0	J3-0	J4-0	J5-C1
771	J1-C3	J2-0	J3-0	J4-0	J5-C2
772	J1-C3	J2-0	J3-0	J4-0	J5-C3
773	J1-C3	J2-0	J3-0	J4-C1	J5-0
774	J1-C3	J2-0	J3-0	J4-C1	J5-C1
775	J1-C3	J2-0	J3-0	J4-C1	J5-C2
776	J1-C3	J2-0	J3-0	J4-C1	J5-C3
777	J1-C3	J2-0	J3-0	J4-C2	J5-0
778	J1-C3	J2-0	J3-0	J4-C2	J5-C1
779	J1-C3	J2-0	J3-0	J4-C2	J5-C2
780	J1-C3	J2-0	J3-0	J4-C2	J5-C3
781	J1-C3	J2-0	J3-0	J4-C3	J5-0
782	J1-C3	J2-0	J3-0	J4-C3	J5-C1
783	J1-C3	J2-0	J3-0	J4-C3	J5-C2
784	J1-C3	J2-0	J3-0	J4-C3	J5-C3
785	J1-C3	J2-0	J3-C1	J4-0	J5-0
786	J1-C3	J2-0	J3-C1	J4-0	J5-C1
787	J1-C3	J2-0	J3-C1	J4-0	J5-C2
788	J1-C3	J2-0	J3-C1	J4-0	J5-C3
789	J1-C3	J2-0	J3-C1	J4-C1	J5-0
790	J1-C3	J2-0	J3-C1	J4-C1	J5-C1
791	J1-C3	J2-0	J3-C1	J4-C1	J5-C2
792	J1-C3	J2-0	J3-C1	J4-C1	J5-C3
793	J1-C3	J2-0	J3-C1	J4-C2	J5-0
794	J1-C3	J2-0	J3-C1	J4-C2	J5-C1
795	J1-C3	J2-0	J3-C1	J4-C2	J5-C2
796	J1-C3	J2-0	J3-C1	J4-C2	J5-C3
797	J1-C3	J2-0	J3-C1	J4-C3	J5-0
798	J1-C3	J2-0	J3-C1	J4-C3	J5-C1
799	J1-C3	J2-0	J3-C1	J4-C3	J5-C2
800	J1-C3	J2-0	J3-C1	J4-C3	J5-C3
801	J1-C3	J2-0	J3-C2	J4-0	J5-0
802	J1-C3	J2-0	J3-C2	J4-0	J5-C1
803	J1-C3	J2-0	J3-C2	J4-0	J5-C2
804	J1-C3	J2-0	J3-C2	J4-0	J5-C3
805	J1-C3	J2-0	J3-C2	J4-C1	J5-0
806	J1-C3	J2-0	J3-C2	J4-C1	J5-C1
807	J1-C3	J2-0	J3-C2	J4-C1	J5-C2

Option	Job 1	Job 2	Job 3	Job 4	Job 5
808	J1-C3	J2-0	J3-C2	J4-C1	J5-C3
809	J1-C3	J2-0	J3-C2	J4-C2	J5-0
810	J1-C3	J2-0	J3-C2	J4-C2	J5-C1
811	J1-C3	J2-0	J3-C2	J4-C2	J5-C2
812	J1-C3	J2-0	J3-C2	J4-C2	J5-C3
813	J1-C3	J2-0	J3-C2	J4-C3	J5-0
814	J1-C3	J2-0	J3-C2	J4-C3	J5-C1
815	J1-C3	J2-0	J3-C2	J4-C3	J5-C2
816	J1-C3	J2-0	J3-C2	J4-C3	J5-C3
817	J1-C3	J2-0	J3-C3	J4-0	J5-0
818	J1-C3	J2-0	J3-C3	J4-0	J5-C1
819	J1-C3	J2-0	J3-C3	J4-0	J5-C2
820	J1-C3	J2-0	J3-C3	J4-0	J5-C3
821	J1-C3	J2-0	J3-C3	J4-C1	J5-0
822	J1-C3	J2-0	J3-C3	J4-C1	J5-C1
823	J1-C3	J2-0	J3-C3	J4-C1	J5-C2
824	J1-C3	J2-0	J3-C3	J4-C1	J5-C3
825	J1-C3	J2-0	J3-C3	J4-C2	J5-0
826	J1-C3	J2-0	J3-C3	J4-C2	J5-C1
827	J1-C3	J2-0	J3-C3	J4-C2	J5-C2
828	J1-C3	J2-0	J3-C3	J4-C2	J5-C3
829	J1-C3	J2-0	J3-C3	J4-C3	J5-0
830	J1-C3	J2-0	J3-C3	J4-C3	J5-C1
831	J1-C3	J2-0	J3-C3	J4-C3	J5-C2
832	J1-C3	J2-0	J3-C3	J4-C3	J5-C3
833	J1-C3	J2-C1	J3-0	J4-0	J5-0
834	J1-C3	J2-C1	J3-0	J4-0	J5-C1
835	J1-C3	J2-C1	J3-0	J4-0	J5-C2
836	J1-C3	J2-C1	J3-0	J4-0	J5-C3
837	J1-C3	J2-C1	J3-0	J4-C1	J5-0
838	J1-C3	J2-C1	J3-0	J4-C1	J5-C1
839	J1-C3	J2-C1	J3-0	J4-C1	J5-C2
840	J1-C3	J2-C1	J3-0	J4-C1	J5-C3
841	J1-C3	J2-C1	J3-0	J4-C2	J5-0
842	J1-C3	J2-C1	J3-0	J4-C2	J5-C1
843	J1-C3	J2-C1	J3-0	J4-C2	J5-C2
844	J1-C3	J2-C1	J3-0	J4-C2	J5-C3
845	J1-C3	J2-C1	J3-0	J4-C3	J5-0
846	J1-C3	J2-C1	J3-0	J4-C3	J5-C1
847	J1-C3	J2-C1	J3-0	J4-C3	J5-C2
848	J1-C3	J2-C1	J3-0	J4-C3	J5-C3
849	J1-C3	J2-C1	J3-C1	J4-0	J5-0
850	J1-C3	J2-C1	J3-C1	J4-0	J5-C1
851	J1-C3	J2-C1	J3-C1	J4-0	J5-C2
852	J1-C3	J2-C1	J3-C1	J4-0	J5-C3
853	J1-C3	J2-C1	J3-C1	J4-C1	J5-0
854	J1-C3	J2-C1	J3-C1	J4-C1	J5-C1
855	J1-C3	J2-C1	J3-C1	J4-C1	J5-C2
856	J1-C3	J2-C1	J3-C1	J4-C1	J5-C3
857	J1-C3	J2-C1	J3-C1	J4-C2	J5-0
858	J1-C3	J2-C1	J3-C1	J4-C2	J5-C1
859	J1-C3	J2-C1	J3-C1	J4-C2	J5-C2
860	J1-C3	J2-C1	J3-C1	J4-C2	J5-C3
861	J1-C3	J2-C1	J3-C1	J4-C3	J5-0
862	J1-C3	J2-C1	J3-C1	J4-C3	J5-C1
863	J1-C3	J2-C1	J3-C1	J4-C3	J5-C2
864	J1-C3	J2-C1	J3-C1	J4-C3	J5-C3
865	J1-C3	J2-C1	J3-C2	J4-0	J5-0
866	J1-C3	J2-C1	J3-C2	J4-0	J5-C1
867	J1-C3	J2-C1	J3-C2	J4-0	J5-C2
868	J1-C3	J2-C1	J3-C2	J4-0	J5-C3
869	J1-C3	J2-C1	J3-C2	J4-C1	J5-0
870	J1-C3	J2-C1	J3-C2	J4-C1	J5-C1
871	J1-C3	J2-C1	J3-C2	J4-C1	J5-C2
872	J1-C3	J2-C1	J3-C2	J4-C1	J5-C3
873	J1-C3	J2-C1	J3-C2	J4-C2	J5-0
874	J1-C3	J2-C1	J3-C2	J4-C2	J5-C1
875	J1-C3	J2-C1	J3-C2	J4-C2	J5-C2
876	J1-C3	J2-C1	J3-C2	J4-C2	J5-C3
877	J1-C3	J2-C1	J3-C2	J4-C3	J5-0
878	J1-C3	J2-C1	J3-C2	J4-C3	J5-C1
879	J1-C3	J2-C1	J3-C2	J4-C3	J5-C2
880	J1-C3	J2-C1	J3-C2	J4-C3	J5-C3
881	J1-C3	J2-C1	J3-C3	J4-0	J5-0
882	J1-C3	J2-C1	J3-C3	J4-0	J5-C1
883	J1-C3	J2-C1	J3-C3	J4-0	J5-C2
884	J1-C3	J2-C1	J3-C3	J4-0	J5-C3
885	J1-C3	J2-C1	J3-C3	J4-C1	J5-0
886	J1-C3	J2-C1	J3-C3	J4-C1	J5-C1
887	J1-C3	J2-C1	J3-C3	J4-C1	J5-C2
888	J1-C3	J2-C1	J3-C3	J4-C1	J5-C3
889	J1-C3	J2-C1	J3-C3	J4-C2	J5-0
890	J1-C3	J2-C1	J3-C3	J4-C2	J5-C1
891	J1-C3	J2-C1	J3-C3	J4-C2	J5-C2
892	J1-C3	J2-C1	J3-C3	J4-C2	J5-C3
893	J1-C3	J2-C1	J3-C3	J4-C3	J5-0
894	J1-C3	J2-C1	J3-C3	J4-C3	J5-C1
895	J1-C3	J2-C1	J3-C3	J4-C3	J5-C2
896	J1-C3	J2-C1	J3-C3	J4-C3	J5-C3
897	J1-C3	J2-C2	J3-0	J4-0	J5-0

Option	Job 1	Job 2	Job 3	Job 4	Job 5
898	J1-C3	J2-C2	J3-0	J4-0	J5-C1
899	J1-C3	J2-C2	J3-0	J4-0	J5-C2
900	J1-C3	J2-C2	J3-0	J4-0	J5-C3
901	J1-C3	J2-C2	J3-0	J4-C1	J5-0
902	J1-C3	J2-C2	J3-0	J4-C1	J5-C1
903	J1-C3	J2-C2	J3-0	J4-C1	J5-C2
904	J1-C3	J2-C2	J3-0	J4-C1	J5-C3
905	J1-C3	J2-C2	J3-0	J4-C2	J5-0
906	J1-C3	J2-C2	J3-0	J4-C2	J5-C1
907	J1-C3	J2-C2	J3-0	J4-C2	J5-C2
908	J1-C3	J2-C2	J3-0	J4-C2	J5-C3
909	J1-C3	J2-C2	J3-0	J4-C3	J5-0
910	J1-C3	J2-C2	J3-0	J4-C3	J5-C1
911	J1-C3	J2-C2	J3-0	J4-C3	J5-C2
912	J1-C3	J2-C2	J3-0	J4-C3	J5-C3
913	J1-C3	J2-C2	J3-C1	J4-0	J5-0
914	J1-C3	J2-C2	J3-C1	J4-0	J5-C1
915	J1-C3	J2-C2	J3-C1	J4-0	J5-C2
916	J1-C3	J2-C2	J3-C1	J4-0	J5-C3
917	J1-C3	J2-C2	J3-C1	J4-C1	J5-0
918	J1-C3	J2-C2	J3-C1	J4-C1	J5-C1
919	J1-C3	J2-C2	J3-C1	J4-C1	J5-C2
920	J1-C3	J2-C2	J3-C1	J4-C1	J5-C3
921	J1-C3	J2-C2	J3-C1	J4-C2	J5-0
922	J1-C3	J2-C2	J3-C1	J4-C2	J5-C1
923	J1-C3	J2-C2	J3-C1	J4-C2	J5-C2
924	J1-C3	J2-C2	J3-C1	J4-C2	J5-C3
925	J1-C3	J2-C2	J3-C1	J4-C3	J5-0
926	J1-C3	J2-C2	J3-C1	J4-C3	J5-C1
927	J1-C3	J2-C2	J3-C1	J4-C3	J5-C2
928	J1-C3	J2-C2	J3-C1	J4-C3	J5-C3
929	J1-C3	J2-C2	J3-C2	J4-0	J5-0
930	J1-C3	J2-C2	J3-C2	J4-0	J5-C1
931	J1-C3	J2-C2	J3-C2	J4-0	J5-C2
932	J1-C3	J2-C2	J3-C2	J4-0	J5-C3
933	J1-C3	J2-C2	J3-C2	J4-C1	J5-0
934	J1-C3	J2-C2	J3-C2	J4-C1	J5-C1
935	J1-C3	J2-C2	J3-C2	J4-C1	J5-C2
936	J1-C3	J2-C2	J3-C2	J4-C1	J5-C3
937	J1-C3	J2-C2	J3-C2	J4-C2	J5-0
938	J1-C3	J2-C2	J3-C2	J4-C2	J5-C1
939	J1-C3	J2-C2	J3-C2	J4-C2	J5-C2
940	J1-C3	J2-C2	J3-C2	J4-C2	J5-C3
941	J1-C3	J2-C2	J3-C2	J4-C3	J5-0
942	J1-C3	J2-C2	J3-C2	J4-C3	J5-C1
943	J1-C3	J2-C2	J3-C2	J4-C3	J5-C2
944	J1-C3	J2-C2	J3-C2	J4-C3	J5-C3
945	J1-C3	J2-C2	J3-C3	J4-0	J5-0
946	J1-C3	J2-C2	J3-C3	J4-0	J5-C1
947	J1-C3	J2-C2	J3-C3	J4-0	J5-C2
948	J1-C3	J2-C2	J3-C3	J4-0	J5-C3
949	J1-C3	J2-C2	J3-C3	J4-C1	J5-0
950	J1-C3	J2-C2	J3-C3	J4-C1	J5-C1
951	J1-C3	J2-C2	J3-C3	J4-C1	J5-C2
952	J1-C3	J2-C2	J3-C3	J4-C1	J5-C3
953	J1-C3	J2-C2	J3-C3	J4-C2	J5-0
954	J1-C3	J2-C2	J3-C3	J4-C2	J5-C1
955	J1-C3	J2-C2	J3-C3	J4-C2	J5-C2
956	J1-C3	J2-C2	J3-C3	J4-C2	J5-C3
957	J1-C3	J2-C2	J3-C3	J4-C3	J5-0
958	J1-C3	J2-C2	J3-C3	J4-C3	J5-C1
959	J1-C3	J2-C2	J3-C3	J4-C3	J5-C2
960	J1-C3	J2-C2	J3-C3	J4-C3	J5-C3
961	J1-C3	J2-C3	J3-0	J4-0	J5-0
962	J1-C3	J2-C3	J3-0	J4-0	J5-C1
963	J1-C3	J2-C3	J3-0	J4-0	J5-C2
964	J1-C3	J2-C3	J3-0	J4-0	J5-C3
965	J1-C3	J2-C3	J3-0	J4-C1	J5-0
966	J1-C3	J2-C3	J3-0	J4-C1	J5-C1
967	J1-C3	J2-C3	J3-0	J4-C1	J5-C2
968	J1-C3	J2-C3	J3-0	J4-C1	J5-C3
969	J1-C3	J2-C3	J3-0	J4-C2	J5-0
970	J1-C3	J2-C3	J3-0	J4-C2	J5-C1
971	J1-C3	J2-C3	J3-0	J4-C2	J5-C2
972	J1-C3	J2-C3	J3-0	J4-C2	J5-C3
973	J1-C3	J2-C3	J3-0	J4-C3	J5-0
974	J1-C3	J2-C3	J3-0	J4-C3	J5-C1
975	J1-C3	J2-C3	J3-0	J4-C3	J5-C2
976	J1-C3	J2-C3	J3-0	J4-C3	J5-C3
977	J1-C3	J2-C3	J3-C1	J4-0	J5-0
978	J1-C3	J2-C3	J3-C1	J4-0	J5-C1
979	J1-C3	J2-C3	J3-C1	J4-0	J5-C2
980	J1-C3	J2-C3	J3-C1	J4-0	J5-C3
981	J1-C3	J2-C3	J3-C1	J4-C1	J5-0
982	J1-C3	J2-C3	J3-C1	J4-C1	J5-C1
983	J1-C3	J2-C3	J3-C1	J4-C1	J5-C2
984	J1-C3	J2-C3	J3-C1	J4-C1	J5-C3
985	J1-C3	J2-C3	J3-C1	J4-C2	J5-0
986	J1-C3	J2-C3	J3-C1	J4-C2	J5-C1
987	J1-C3	J2-C3	J3-C1	J4-C2	J5-C2

Option	Job 1	Job 2	Job 3	Job 4	Job 5
988	J1-C3	J2-C3	J3-C1	J4-C2	J5-C3
989	J1-C3	J2-C3	J3-C1	J4-C3	J5-0
990	J1-C3	J2-C3	J3-C1	J4-C3	J5-C1
991	J1-C3	J2-C3	J3-C1	J4-C3	J5-C2
992	J1-C3	J2-C3	J3-C1	J4-C3	J5-C3
993	J1-C3	J2-C3	J3-C2	J4-0	J5-0
994	J1-C3	J2-C3	J3-C2	J4-0	J5-C1
995	J1-C3	J2-C3	J3-C2	J4-0	J5-C2
996	J1-C3	J2-C3	J3-C2	J4-0	J5-C3
997	J1-C3	J2-C3	J3-C2	J4-C1	J5-0
998	J1-C3	J2-C3	J3-C2	J4-C1	J5-C1
999	J1-C3	J2-C3	J3-C2	J4-C1	J5-C2
1000	J1-C3	J2-C3	J3-C2	J4-C1	J5-C3
1001	J1-C3	J2-C3	J3-C2	J4-C2	J5-0
1002	J1-C3	J2-C3	J3-C2	J4-C2	J5-C1
1003	J1-C3	J2-C3	J3-C2	J4-C2	J5-C2
1004	J1-C3	J2-C3	J3-C2	J4-C2	J5-C3
1005	J1-C3	J2-C3	J3-C2	J4-C3	J5-0
1006	J1-C3	J2-C3	J3-C2	J4-C3	J5-C1
1007	J1-C3	J2-C3	J3-C2	J4-C3	J5-C2
1008	J1-C3	J2-C3	J3-C2	J4-C3	J5-C3
1009	J1-C3	J2-C3	J3-C3	J4-0	J5-0
1010	J1-C3	J2-C3	J3-C3	J4-0	J5-C1
1011	J1-C3	J2-C3	J3-C3	J4-0	J5-C2
1012	J1-C3	J2-C3	J3-C3	J4-0	J5-C3
1013	J1-C3	J2-C3	J3-C3	J4-C1	J5-0
1014	J1-C3	J2-C3	J3-C3	J4-C1	J5-C1
1015	J1-C3	J2-C3	J3-C3	J4-C1	J5-C2
1016	J1-C3	J2-C3	J3-C3	J4-C1	J5-C3
1017	J1-C3	J2-C3	J3-C3	J4-C2	J5-0
1018	J1-C3	J2-C3	J3-C3	J4-C2	J5-C1
1019	J1-C3	J2-C3	J3-C3	J4-C2	J5-C2
1020	J1-C3	J2-C3	J3-C3	J4-C2	J5-C3
1021	J1-C3	J2-C3	J3-C3	J4-C3	J5-0
1022	J1-C3	J2-C3	J3-C3	J4-C3	J5-C1
1023	J1-C3	J2-C3	J3-C3	J4-C3	J5-C2
1024	J1-C3	J2-C3	J3-C3	J4-C3	J5-C3

BIBLIOGRAPHY

[1] 5GAMERICAS, *Understanding information centric networking and mobile edge computing*. https://www.5gamericas.org/understanding-information-centric, 2016. (Date accessed: 01-10-2020).

[2] I. ABDULLAHI, S. ARIF, AND S. HASSAN, *Ubiquitous Shift with Information Centric Network Caching using Fog Computing*, in : Phon-Amnuaisuk S, Au T (eds) Computational Intelligence in Information Systems, vol. 331, Springer, 2015, pp. 327–335.

[3] H. AHLEHAGH AND S. DEY, *Video-aware scheduling and caching in the radio access network*, IEEE/ACM Transactions on Networking, 22 (2014), pp. 1444–1462.

[4] B. AHLGREN, C. DANNEWITZ, C. IMBRENDA, D. KUTSCHER, AND B. OHLMAN, *A survey of information-centric networking*, in IEEE Communications Magazine, 50 (2012), pp. 26–36.

[5] M. AHMAD, M. B. AMIN, S. HUSSAIN, B. H. KANG, T. CHEONG, AND S. LEE, *Health fog: A novel framework for health and wellness applications*, The Journal of Supercomputing, 72 (2016), pp. 3677–3695.

[6] M. G. R. ALAM, Y. K. TUN, AND C. S. HONG, *Multi-agent and reinforcement learning based code offloading in mobile fog*, in International Conference on Information Networking (ICOIN), IEEE, 2016, pp. 285–290.

[7] H. A. ALAMEDDINE, S. SHARAFEDDINE, S. SEBBAH, S. AYOUBI, AND C. ASSI, *Dynamic Task Offloading and Scheduling for Low-Latency IoT Services in Multi-Access Edge Computing*, IEEE Journal on Selected Areas in Communications, 37 (2019), pp. 668–682.

[8] L. ALE, N. ZHANG, H. WU, D. CHEN, AND T. HAN, *Online proactive caching in mobile edge computing using bidirectional deep recurrent neural network*, IEEE Internet of Things Journal, 6 (2019), pp. 5520–5530.

[9] R. T. AZUMA, *Augmented reality: Approaches and technical challenges*, Fundamentals of Wearable Computers and Augmented Reality, W.Barfield and T. Caudell, eds., (2001), pp. 27–63.

[10] A. BADAM, R. CHANDRA, J. DUTRA, A. FERRESE, S. HODGES, P. HU, J. MEINERSHAGEN, T. MOSCIBRODA, B. PRIYANTHA, AND E. SKIANI, *Software defined batteries*, in Proceedings of the 25th Symposium on Operating Systems Principles, 2015, pp. 215–229.

[11] B. BAEK, J. LEE, Y. PENG, AND S. PARK, *Three dynamic pricing schemes for resource allocation of edge computing for iot environment*, IEEE Internet of Things Journal, 7 (2020), pp. 4292–4303.

[12] G. BAHL, *Emergence of micro data center (cloudlets/edges) for mobile computing*. `https://tinyurl.com/yxvzn83l`, 2015.
(Date accessed: 01-10-2020).

[13] BALTIMORE–WASHINGTON INTERNATIONAL AIRPORT, *Bwi marshall's smart park technology: It's all about the lights!*
`https://tinyurl.com/y2jywfel`, 2015.
(Date accessed: 01-10-2020).

[14] M. T. BECK, M. WERNER, S. FELD, AND S. SCHIMPER, *Mobile edge computing: A taxonomy*, in in Proc. of the Sixth International Conference on Advances in Future Internet, Citeseer, 2014, pp. 48–54.

[15] S. BHATTACHARYYA, *Research on edge computing: A detailed study*, International Journal of Information Technology (IJIT), 2 (2016), pp. 9–13.

[16] L. F. BITTENCOURT, M. M. LOPES, I. PETRI, AND O. F. RANA, *Towards virtual machine migration in fog computing*, in P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC), 2015 10th International Conference on, IEEE, 2015, pp. 1–8.

[17] F. BONOMI, R. MILITO, P. NATARAJAN, AND J. ZHU, *Fog computing: A platform for internet of things and analytics*, in Bessie N., Dobre C. (eds) Big Data and Internet of Things: A Roadmap for Smart Environments, Springer, 2014, pp. 169–186.

[18] F. BONOMI, R. MILITO, J. ZHU, AND S. ADDEPALLI, *Fog computing and its role in the internet of things*, in Proceedings of the first edition of the MCC workshop on Mobile cloud computing, ACM, 2012, pp. 13–16.

[19] E. BORCOCI, *Fog computing, mobile edge computing, cloudlets - which one?*
`https://www.iaria.org/conferences2016/filesICSNC16/Softnet2016_Tutorial_Fog-MEC-Cloudlets-E.Borcoci-v1.1.pdf`, 2016.
(Date accessed: 01-10-2020).

[20] G. BROWN, *Mobile edge computing use cases and deployment options*.
`https://www.juniper.net/assets/us/en/local/pdf/whitepapers/2000642-en.pdf`, 2016.

(Date accessed: 01-10-2020).

[21] L. CAO, J. HAO, AND D. JIANG, *Two parallel machines scheduling with two-vehicle job delivery to minimize makespan*, Complexity, (2020), pp. 1–6.

[22] Y. CAO, S. CHEN, P. HOU, AND D. BROWN, *Fast: A fog computing assisted distributed analytics system to monitor fall for stroke mitigation*, in Networking, Architecture and Storage (NAS), IEEE International Conference on, IEEE, 2015, pp. 2–11.

[23] S. CARLINI, *The drivers and benefits of edge computing*.
`https://tinyurl.com/yxkdaopc`, 2016.
(Date accessed: 01-10-2020).

[24] P. CARNELLI, J. YEH, M. SOORIYABANDARA, AND A. KHAN, *Parkus: A novel vehicle parking detection system*, in In Proceedings of the Twenty-Ninth Innovative Applications of Artificial Intelligence (IAAI-17), 2017, pp. 4650–4656.

[25] M. CARROLL, A. VAN DER MERWE, AND P. KOTZE, *Secure cloud computing: Benefits, risks and controls*, in Information Security South Africa (ISSA), 2011, IEEE, 2011, pp. 1–9.

[26] K. CASTILLO-VILLAR, R. GONZALEZ RAMIREZ, P. MIRANDA, AND N. SMITH, *A heuristic procedure for a ship routing and scheduling problem with variable speed and discretized time windows*, Mathematical Problems in Engineering, 2014 (2014), pp. 1–13.
doi: `10.1155/2014/750232`.

[27] A. CESELLI, M. FIORE, M. PREMOLI, AND S. SECCI, *Optimized assignment patterns in mobile edge cloud networks*, Computers & Operations Research, 106 (2019), pp. 246–259.

[28] A. CESELLI, M. PREMOLI, AND S. SECCI, *Cloudlet network design optimization*, in Proc IFIP Networking Conference, 2015, pp. 1–9.

[29] S. CHEN, L. LI, Z. CHEN, AND S. LI, *Dynamic pricing for smart mobile edge computing: A reinforcement learning approach*, IEEE Wireless Communications Letters, (2020).

[30] X. CHEN, L. JIAO, W. LI, AND X. FU, *Efficient multi-user computation offloading for mobile-edge cloud computing*, IEEE/ACM Transactions on Networking, 24 (2016), pp. 2795–2808.

[31] Y. CHEN, Z. LI, B. YANG, K. NAI, AND K. LI, *A stackelberg game approach to multiple resources allocation and pricing in mobile edge computing*, Future Generation Computer Systems, 108 (2020), pp. 273–287.

[32] M. CHIANG, *Fog networking: An overview on research opportunities*. http://www.princeton.edu/~chiangm/FogResearchOverview.pdf, 2015. (Date accessed: 01-10-2020).

[33] B.-G. CHUN, S. IHM, P. MANIATIS, M. NAIK, AND A. PATTI, *CloneCloud: elastic execution between mobile device and cloud*, in Proceedings of the sixth conference on Computer systems, ACM, 2011, pp. 301–314.

[34] K. CHURCH, A. G. GREENBERG, AND J. R. HAMILTON, *On delivering embarrassingly distributed cloud services.*, in HotNets, 2008, pp. 55–60.

[35] CISCO, *Fog computing and the internet of things: Extend the cloud to where the things are*. https://tinyurl.com/zktxmux, 2015. (Date accessed: 01-10-2020).

[36] CISCO, *Global cloud index: Forecast and methodology 2015 – 2020 (white paper)*. https://tinyurl.com/y5w275mv, 2015. (Date accessed: 01-10-2020).

[37] S. CLINCH, J. HARKES, A. FRIDAY, N. DAVIES, AND M. SATYANARAYANAN, *How close is close enough? understanding the role of cloudlets in supporting display appropriation by mobile users*, in IEEE International Conference on Pervasive Computing and Communications (PerCom), IEEE, 2012, pp. 122–127.

[38] P. CORCORAN AND S. K. DATTA, *Mobile-edge computing and the internet of things for consumers: Extending cloud computing and services to the edge of the network*, IEEE Consumer Electronics Magazine, 5 (2016), pp. 73–74.

[39] R. CRACIUNESCU, A. MIHOVSKA, M. MIHAYLOV, S. KYRIAZAKOS, R. PRASAD, AND S. HALUNGA, *Implementation of Fog computing for reliable E-health applications*, in 49th Asilomar Conference on Signals, Systems and Computers, IEEE, 2015, pp. 459–463.

[40] E. CUERVO, A. BALASUBRAMANIAN, D.-K. CHO, A. WOLMAN, S. SAROIU, R. CHANDRA, AND P. BAHL, *MAUI: making smartphones last longer with code offload*, in Proceedings of the 8th international conference on Mobile systems, applications, and services, ACM, 2010, pp. 49–62.

[41] N.-N. DAO, Y. LEE, S. CHO, E. KIM, K.-S. CHUNG, AND C. KEUM, *Multi-tier multi-access edge computing: The role for the fourth industrial revolution*, in 2017 International Conference on Information and Communication Technology Convergence (ICTC), IEEE, 2017, pp. 1280–1282.

[42] A. V. DASTJERDI, H. GUPTA, R. N. CALHEIROS, S. K. GHOSH, AND R. BUYYA, *Fog Computing: principles, architectures, and applications*, in Internet of Things, Eds. R.Buyya, A.V. Dastjerdi, (2016), pp. 61–75.

[43] S. K. DATTA, C. BONNET, AND J. HAERRI, *Fog Computing architecture to enable consumer centric Internet of Things services*, in IEEE International Symposium on Consumer Electronics (ISCE), IEEE, 2015, pp. 1–2.

[44] M. DAYARATHNA, Y. WEN, AND R. FAN, *Data center energy consumption modeling: A survey*, IEEE Communications Surveys & Tutorials, 18 (2015), pp. 732–794.

[45] S. DENG, H. ZHAO, W. FANG, J. YIN, S. DUSTDAR, AND A. Y. ZOMAYA, *Edge intelligence: The confluence of edge computing and artificial intelligence*, IEEE Internet of Things Journal, 7 (2020), pp. 7457–7469.

[46] S. DESHMUKH AND R. SHAH, *Computation offloading frameworks in mobile cloud computing: a survey*, in 2016 IEEE International Conference on Current Trends in Advanced Computing (ICCTAC), IEEE, 2016, pp. 1–5.

[47] H. T. DINH, C. LEE, D. NIYATO, AND P. WANG, *A survey of mobile cloud computing: architecture, applications, and approaches*, Wireless Communications and Mobile Computing, 13 (2013), pp. 1587–1611.

[48] H. DUBEY, J. YANG, N. CONSTANT, A. M. AMIRI, Q. YANG, AND K. MAKODIYA, *Fog data: Enhancing telehealth big data through fog computing*, in Proceedings of the ASE Big Data & Social Informatics, ACM, 2015, pp. 6–9.

[49] I. ELGENDY, W. ZHANG, Y. ZENG, H. HE, Y. TIAN, AND Y. YANG, *Efficient and secure multi-user multi-task computation offloading for mobile-edge computing in mobile IoT networks*, in IEEE Transactions on Network and Service Management,.
doi: 10.1109/TNSM.2020.3020249.

[50] W. EMFINGER, A. DUBEY, P. VOLGYESI, J. SALLAI, AND G. KARSAI, *Demo abstract: RIAPS—a resilient information architecture platform for edge computing*, in IEEE/ACM Symposium on Edge Computing (SEC), IEEE, 2016, pp. 119–120.

[51] M. EROL-KANTARCI AND S. SUKHMANI, *Caching and computing at the edge for mobile augmented reality and virtual reality (ar/vr) in 5g*, Ad Hoc Networks, (2018), pp. 169–177.

[52] ETSI, *Mobile-Edge Computing – Introductory Technical White Paper*.
https://tinyurl.com/gmybuhs, 2014.
(Date accessed: 01-10-2020).

[53] D. FESEHAYE, Y. GAO, K. NAHRSTEDT, AND G. WANG, *Impact of cloudlets on interactive mobile cloud applications*, in IEEE 16th International on Enterprise Distributed Object Computing Conference (EDOC), IEEE, 2012, pp. 123–132.

[54] F. F. FONSECA, L. MAMATAS, A. C. VIANA, S. L. CORREA, AND K. V. CARDOSO, *Personalized travel itineraries with multi-access edge computing touristic services*, in IEEE Global Communications Conference (GLOBECOM), IEEE, 2019, pp. 1–6.

[55] R. FRANK, *Understanding smart sensors*, Artech House, 2013.

[56] O. FRATU, C. PENA, R. CRACIUNESCU, AND S. HALUNGA, *Fog computing system for monitoring Mild Dementia and COPD patients-Romanian case study*, in 12th International Conference on Telecommunication in Modern Satellite, Cable and Broadcasting Services (TELSIKS), 2015, IEEE, 2015, pp. 123–128.

[57] P. GARCIA LOPEZ, A. MONTRESOR, D. EPEMA, A. DATTA, T. HIGASHINO, A. IAMNITCHI, M. BARCELLOS, P. FELBER, AND E. RIVIERE, *Edge-centric computing: Vision and challenges*, ACM SIGCOMM Computer Communication Review, 45 (2015), pp. 37–42.

[58] S. S. GILL AND I. CHANA, *A survey on resource scheduling in cloud computing: Issues and challenges*, Journal of Grid Computing, 14 (2016).

[59] F. GIUST, V. SCIANCALEPORE, D. SABELLA, M. C. FILIPPOU, S. MANGIANTE, W. FEATHERSTONE, AND D. MUNARETTO, *Multi-access edge computing: The driver behind the wheel of 5G-connected cars*, IEEE Communications Standards Magazine, 2 (2018), pp. 66–73.

[60] G. GRASSI, K. JAMIESON, P. BAHL, AND G. PAU, *Parkmaster: An in-vehicle, edge-based video analytics service for detecting open parking spaces in urban environments*, in Proceedings of the Second ACM/IEEE Symposium on Edge Computing, 2017, pp. 1–14.

[61] X. GU, L. JIN, N. ZHAO, AND G. ZHANG, *Energy-efficient computation offloading and transmit power allocation scheme for mobile edge computing*, Mobile Information Systems, 2019 (2019), pp. 1–9.
doi: 10.1155/2019/3613250.

[62] T. GUÉROUT, T. MONTEIL, G. DA COSTA, R. N. CALHEIROS, R. BUYYA, AND M. ALEXANDRU, *Energy-aware simulation with DVFS*, Simulation Modelling Practice and Theory, 39 (2013), pp. 76–91.

[63] H. GUPTA, A. V. DASTJERDI, S. K. GHOSH, AND R. BUYYA, *ifogsim: A toolkit for modeling and simulation of resource management techniques in internet of things, edge and fog computing environments*, Software: Practice and Experience, (2017), pp. 1275–1296.

[64] K. HA, P. PILLAI, W. RICHTER, Y. ABE, AND M. SATYANARAYANAN, *Just-in-time provisioning for cyber foraging*, in Proceeding of the 11th Annual International Conference on Mobile Systems, Applications, and Services, 2013, pp. 153–166.

[65] K. HABAK, M. AMMAR, K. A. HARRAS, AND E. ZEGURA, *Femto clouds: Leveraging mobile devices to provide cloud service at the edge*, in 8th IEEE International Conference on Cloud Computing (CLOUD), 2015, pp. 9–16.

[66] J. HAMMER, P. MOLL, AND H. HELLWAGNER, *Transparent access to 5G edge computing services*, in IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW), IEEE, 2019, pp. 895–898.

[67] M. A. HASSAN, M. XIAO, Q. WEI, AND S. CHEN, *Help your mobile applications with fog computing*, in 12th Annual IEEE International Conference on Sensing, Communication, and Networking-Workshops (SECON Workshops), IEEE, 2015, pp. 1–6.

[68] J. L. HENNESSY AND D. A. PATTERSON, *Computer architecture: A Quantitative Approach (5th edition)*, Morgan Kaufman, inc., San Francisco, 2011.

[69] K. HONG, D. LILLETHUN, U. RAMACHANDRAN, B. OTTENWÄLDER, AND B. KOLDEHOFE, *Mobile fog: A programming model for large-scale applications on the internet of things*, in Proceedings of the second ACM SIGCOMM workshop on Mobile cloud computing, ACM, 2013, pp. 15–20.

[70] T. HOU, G. FENG, S. QIN, AND W. JIANG, *Proactive content caching by exploiting transfer learning for mobile edge computing*, International Journal of Communication Systems, 31 (2018), p. e3706.

[71] X. HOU, Y. LI, M. CHEN, D. WU, D. JIN, AND S. CHEN, *Vehicular fog computing: A viewpoint of vehicles as the infrastructures*, IEEE Transactions on Vehicular Technology, 65 (2016), pp. 3860–3873.

[72] W. HU, Y. GAO, K. HA, J. WANG, B. AMOS, Z. CHEN, P. PILLAI, AND M. SATYANARAYANAN, *Quantifying the impact of edge computing on mobile applications*, in Proceedings of the 7th ACM SIGOPS Asia-Pacific Workshop on Systems, ACM, 2016, p. 5.

[73] Y. C. HU, M. PATEL, D. SABELLA, N. SPRECHER, AND V. YOUNG, *Mobile edge computing—a key technology towards 5G*.
`https://www.etsi.org/images/files/ETSIWhitePapers/etsi_wp11_mec_a_key_technology_towards_5g.pdf`, 2015.
(Date accessed: 01-10-2020).

[74] D. HUANG, P. WANG, AND D. NIYATO, *A dynamic offloading algorithm for mobile computing*, IEEE Transactions on Wireless Communications, 11 (2012), pp. 1991–1995.

[75] L. N. HUYNH, Q.-V. PHAM, T. D. NGUYEN, M. D. HOSSAIN, J. H. PARK, AND E.-N. HUH, *A study on computation offloading in mec systems using whale optimization algorithm*, in 14th International Conference on Ubiquitous Information Management and Communication (IMCOM), IEEE, 2020, pp. 1–4.

[76] E. IBUKUN AND O. DARAMOLA, *A systematic literature review of mobile cloud computing*, International Journal of Multimedia and Ubiquitous Engineering, 10 (2015), pp. 135–152.

[77] Y. JARARWEH, A. DOULAT, A. DARABSEH, M. ALSMIRAT, M. AL-AYYOUB, AND E. BENKHELIFA, *SDMEC:software defined system for mobile edge computing*, in IEEE International Conference on Cloud Engineering Workshop (IC2EW), IEEE, 2016, pp. 88–93.

[78] Y. JARARWEH, L. TAWALBEH, F. ABABNEH, AND F. DOSARI, *Resource efficient mobile computing using cloudlet infrastructure*, in 9th IEEE Conference on Mobile Ad-hoc and Sensor Networks (MSN), IEEE, 2013, pp. 373–377.

[79] P. P. JAYARAMAN, J. B. GOMES, H. L. NGUYEN, Z. S. ABDALLAH, S. KRISHNASWAMY, AND A. ZASLAVSKY, *CARDAP: A scalable energy-efficient context aware distributed mobile data analytics platform for the fog*, in East European Conference on Advances in Databases and Information Systems, Springer, 2014, pp. 192–206.

[80] C. JIANG, X. CHENG, H. GAO, X. ZHOU, AND J. WAN, *Toward computation offloading in edge computing: A survey*, IEEE Access, 7 (2019), pp. 131543–131558.

[81] C. JIANG, T. FAN, H. GAO, W. SHI, L. LIU, C. CÉRIN, AND J. WAN, *Energy aware edge computing: A survey*, Computer Communications, 151 (2020), pp. 556–580.

[82] S. JINGTAO, L. FUHONG, Z. XIANWEI, AND L. XING, *Steiner tree based optimal resource caching scheme in fog computing*, China Communications, 12 (2015), pp. 161–168.

[83] N. JONES AND D. CEARLEY, *Top 10 strategic technology trends for 2020: Empowered edge*. `https://www.gartner.com/doc/reprints?id=1-1Z4PHL0Q&ct=200529&st=sb`, 2020. (Date accessed: 21-02-2020).

[84] A. R. KHAN, M. OTHMAN, A. N. KHAN, J. SHUJA, AND S. MUSTAFA, *Computation offloading cost estimation in mobile cloud application models*, Wireless Personal Communications, 97 (2017), pp. 4897–4920.

[85] Y. KIM AND E.-N. HUH, *Edcrammer: An efficient caching rate-control algorithm for streaming data on resource-limited edge nodes*, Applied Sciences, 9 (2019), p. 2560.

[86] G. I. KLAS, *Fog computing and mobile edge cloud gain momentum*. `https://yucianga.info/?p=938`, 2015.

(Date accessed: 01-10-2020).

[87] D. KOVACHEV AND R. KLAMMA, *Framework for computation offloading in mobile cloud computing*, International Journal of Interactive Multimedia and Artificial Intelligence, 1 (2012), pp. 6–15.

[88] D. KREUTZ, F. M. RAMOS, P. E. VERISSIMO, C. E. ROTHENBERG, S. AZODOLMOLKY, AND S. UHLIG, *Software-defined networking: A comprehensive survey*, Proceedings of the IEEE, 103 (2014), pp. 14–76.

[89] K. KUMAR, J. LIU, Y.-H. LU, AND B. BHARGAVA, *A survey of computation offloading for mobile systems*, Mobile Networks and Applications, 18 (2013), pp. 129–140.

[90] K. KUMAR AND Y.-H. LU, *Cloud computing for mobile users: Can offloading computation save energy?*, IEEE Computer, 43 (2010), pp. 51–56.

[91] H. Q. LE, H. AL-SHATRI, AND A. KLEIN, *Efficient resource allocation in mobile-edge computation offloading: Completion time minimization*, in IEEE International Symposium on Information Theory (ISIT), IEEE, 2017, pp. 2513–2517.

[92] J.-S. LEE, Y.-W. SU, AND C.-C. SHEN, *A comparative study of wireless protocols: Bluetooth, uwb, zigbee, and wi-fi*, in 33rd Annual Conference of the Industrial Electronics Society (IECON), 2017, pp. 46–51.

[93] H. LI, S. CI, C. YANG, X. TAN, S. HOU, AND Z. WANG, *Moving to green edges: A cooperative MEC framework to reduce energy demand of clouds*, in IEEE Globecom Workshops (GC Wkshps), IEEE, 2019, pp. 1–6.

[94] H. LI AND F. FANG, *Joint resource allocation for NOMA-assisted MEC networks*, in IEEE International Conference on Communications Workshops (ICC Workshops), IEEE, 2020, pp. 1–6.

[95] H. LI, G. SHOU, Y. HU, AND Z. GUO, *Mobile edge computing: progress and challenges*, in 4th IEEE International Conference on Mobile Cloud Computing, Services, and Engineering (MobileCloud), IEEE, 2016, pp. 83–84.

[96] X. LI, Y. DANG, M. AAZAM, X. PENG, T. CHEN, AND C. CHEN, *Energy-efficient computation offloading in vehicular edge cloud computing*, IEEE Access, 8 (2020), pp. 37632–37644.

[97] Y. LI AND W. WANG, *Can mobile cloudlets support mobile applications?*, in IEEE INFOCOM Conference on Computer Communications, IEEE, 2014, pp. 1060–1068.

[98] L. LIN, X. LIAO, H. JIN, AND P. LI, *Computation offloading toward edge computing*, Proceedings of the IEEE, 107 (2019), pp. 1584–1607.

[99] J. LIU, G. SHOU, Y. LIU, Y. HU, AND Z. GUO, *Performance evaluation of integrated multi-access edge computing and fiber-wireless access networks*, IEEE Access, 6 (2018), pp. 30269–30279.

[100] L. LIU, Z. CHANG, X. GUO, AND T. RISTANIEMI, *Multi-objective optimization for computation offloading in mobile-edge computing*, in IEEE Symposium on Computers and Communications (ISCC), IEEE, 2017, pp. 832–837.

[101] T. H. LUAN, L. GAO, Z. LI, Y. XIANG, G. WEI, AND L. SUN, *Fog computing. focusing on users at the edge of Internet of Things*, arXiv preprint arXiv:1502.01815, (2015).

[102] X. LYU, W. NI, H. TIAN, R. P. LIU, X. WANG, G. B. GIANNAKIS, AND A. PAULRAJ, *Optimal schedule of mobile edge computing for internet of things using partial information*, IEEE Journal on Selected Areas in Communications, 35 (2017), pp. 2606–2615.

[103] F. MALANDRINO, C. F. CHIASSERINI, G. AVINO, M. MALINVERNO, AND S. KIRKPATRICK, *From megabits to CPU ticks: Enriching a demand trace in the age of MEC*, IEEE Transactions on Big Data, 6 (2020), pp. 43–50.

[104] F. MALANDRINO, S. KIRKPATRICK, AND C.-F. CHIASSERINI, *How close to the edge? delay/utilization trends in MEC*, in Proceedings of the ACM Workshop on Cloud-Assisted Networking, 2016, pp. 37–42.

[105] S. MELENDEZ AND M. P. MCGARRY, *Computation offloading decisions for reducing completion time*, in 14th IEEE Annual Consumer Communications & Networking Conference (CCNC), IEEE, 2017, pp. 160–164.

[106] R. A. NAJDI, T. G. SHABAN, M. J. MOURAD, AND S. H. KARAKI, *Hydrogen production and filling of fuel cell cars*, in 3rd International Conference on Advances in Computational Tools for Engineering Applications (ACTEA), IEEE, 2016, pp. 43–48.

[107] A. V. NATRAJ, *"Fog Computing" focusing on users at the edge of Internet of Things*, International Journal of Engineering Research Special 5, pp: 992 - 1128, (2016).

[108] Q.-H. NGUYEN, M. MOROLD, K. DAVID, AND F. DRESSLER, *Car-to-pedestrian communication with MEC-support for adaptive safety of vulnerable road users*, Computer Communications, 150 (2020), pp. 83–93.

[109] S. NINGNING, G. CHAO, A. XINGSHUO, AND Z. QIANG, *Fog computing dynamic load balancing mechanism based on graph repartitioning*, China Communications, 13 (2016), pp. 156–164.

[110] NSF EDGE COMPUTING WORKSHOP COMMITTEE, *Grand challenges in edge computing*. `http://iot.eng.wayne.edu/edge/goals.php`, 2016.

(Date accessed: 01-10-2020).

[111] S. NUNNA, A. KOUSARIDAS, M. IBRAHIM, M. DILLINGER, C. THUEMMLER, H. FEUSSNER, AND A. SCHNEIDER, *Enabling real-time context-aware collaboration through 5G and mobile edge computing*, in 12th International Conference on Information Technology-New Generations (ITNG), IEEE, 2015, pp. 601–605.

[112] F. Y. OKAY AND S. OZDEMIR, *A fog computing based smart grid model*, in International Symposium on Networks, Computers and Communications (ISNCC)„ IEEE, 2016, pp. 1–6.

[113] G. ORSINI, D. BADE, AND W. LAMERSDORF, *Computing at the mobile edge: Designing elastic android applications for computation offloading*, in 8th IFIP Wireless and Mobile Networking Conference (WMNC), IEEE, 2015, pp. 112–119.

[114] Z. PANG, L. SUN, Z. WANG, E. TIAN, AND S. YANG, *A survey of cloudlet based mobile computing*, in Cloud Computing and Big Data (CCBD), 2015 International Conference on, IEEE, 2015, pp. 268–275.

[115] A. PAWAR, V. JAGTAP, AND M. BHAMARE, *Time and energy saving through computation offloading with bandwidth consideration for mobile cloud computing*, in Proceedings of the Third International Symposium on Women in Computing and Informatics, 2015, pp. 527–532.

[116] G. P. PERRUCCI, F. H. P. FITZEK, AND J. WIDMER, *Survey on energy consumption entities on the smartphone platform*, in IEEE 73rd Vehicular Technology Conference (VTC Spring), 2011, pp. 1–6.

[117] Q.-V. PHAM, T. LEANH, N. H. TRAN, B. J. PARK, AND C. S. HONG, *Decentralized computation offloading and resource allocation for mobile-edge computing: A matching game approach*, IEEE Access, 6 (2018), pp. 75868–75885.

[118] P. PORAMBAGE, J. OKWUIBE, M. LIYANAGE, M. YLIANTTILA, AND T. TALEB, *Survey on multi-access edge computing for Internet of Things realization*, IEEE Communications Surveys & Tutorials, 20 (2018), pp. 2961–2991.

[119] J. PREDEN, J. KAUGERAND, E. SUURJAAK, S. ASTAPOV, L. MOTUS, AND R. PAHTMA, *Data to decision: pushing situational information needs to the edge of the network*, in IEEE International Inter-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support (CogSIMA), IEEE, 2015, pp. 158–164.

[120] J. P. QUERALTA, L. QINGQING, Z. ZOU, AND T. WESTERLUND, *Enhancing autonomy with blockchain and multi-acess edge computing in distributed robotic systems*, in The Fifth

International Conference on Fog and Mobile Edge Computing (FMEC). IEEE, 2020, pp. 180–187.

[121] M.-R. RA, A. SHETH, L. MUMMERT, P. PILLAI, D. WETHERALL, AND R. GOVINDAN, *Odessa: enabling interactive perception applications on mobile devices*, in Proceedings of the 9th international conference on Mobile systems, applications, and services, 2011, pp. 43–56.

[122] F. RAYAL, *A perspective on multi-access edge computing*.
https://tinyurl.com/yyeztzcc, 2017.
(Date accessed: 01-10-2020).

[123] S. RAZA, S. WANG, M. AHMED, AND M. R. ANWAR, *A survey on vehicular edge computing: architecture, applications, technical issues, and future directions*, Wireless Communications and Mobile Computing, Article ID 3159762, 19 pages (2019).
doi: 10.1155/2019/3159762.

[124] ETSI GS MEC 002, *Multi-access edge computing (MEC); phase 2: Use cases and requirements*.
https://www.etsi.org/deliver/etsi_gs/MEC/001_099/002/02.01.01_60/gs_MEC002v020101p.pdf, 2018.
(Date accessed: 01-09-2020).

[125] ETSI GS MEC 003, *Multi-access edge computing (MEC); framework and reference architecture*.
https://www.etsi.org/deliver/etsi_gs/MEC/001_099/003/02.01.01_60/gs_mec003v020101p.pdf, 2019.
(Date accessed: 05-10-2020).

[126] JONES. N, *How to stop data centres from gobbling up the world's electricity. Nature vol. 561, pp. 163-166*, 2018.

[127] M. LI, F. R. YU, P. SI, R. YANG, Z. WANG, AND Y. ZHANG, *UAV-assisted data transmission in blockchain-enabled m2m communications with mobile edge computing*, in IEEE Network, (2020).
doi: 10.1109/MNET.011.2000147.

[128] R. ROMAN, J. LOPEZ, AND M. MAMBO, *Mobile edge computing, fog et al.: A survey and analysis of security threats and challenges*, Future Generation Computer Systems.

[129] D. SABELLA, A. REZNIK, AND R. FRAZAO, *Multi-Access Edge Computing in Action*, CRC Press, 2020.

[130] D. SABELLA, A. VAILLANT, P. KUURE, U. RAUSCHENBACH, AND F. GIUST, *Mobile-edge computing architecture: The role of mec in the internet of things*, IEEE Consumer Electronics Magazine, 5 (2016), pp. 84–91.

[131] S. SAFAVAT, N. N. SAPAVATH, AND D. B. RAWAT, *Recent advances in mobile edge computing and content caching*, Digital Communications and Networks, 6 (2020), pp. 189–194.

[132] K. SAHARAN AND A. KUMAR, *Fog in comparison to cloud: A survey*, International Journal of Computer Applications, 122 (2015), pp. 10–12.

[133] L. SANCHEZ, L. MUÑOZ, J. A. GALACHE, P. SOTRES, J. R. SANTANA, V. GUTIERREZ, R. RAMDHANY, A. GLUHAK, S. KRCO, E. THEODORIDIS, ET AL., *SmartSantander: IoT experimentation over a smart city testbed*, Computer Networks, 61 (2014), pp. 217–238.

[134] S. SARKAR AND S. MISRA, *Theoretical modelling of fog computing: a green computing paradigm to support IoT applications*, IET Networks, 5 (2016), pp. 23–29.

[135] D. SATRIA, D. PARK, AND M. JO, *Recovery for overloaded mobile edge computing*, Future Generation Computer Systems, 70 (2017), pp. 138–147.

[136] M. SATYANARAYANAN, *Cloudlets: at the leading edge of cloud-mobile convergence*, in Proceedings of the 9th international ACM Sigsoft conference on Quality of software architectures, ACM, 2013, pp. 1–2.

[137] M. SATYANARAYANAN, P. BAHL, R. CACERES, AND N. DAVIES, *The case for VM-based cloudlets in mobile computing*, IEEE pervasive Computing, 8 (2009), pp. 14–23.

[138] M. SATYANARAYANAN, P. SIMOENS, Y. XIAO, P. PILLAI, Z. CHEN, K. HA, W. HU, AND B. AMOS, *Edge analytics in the internet of things*, IEEE Pervasive Computing, 14 (2015), pp. 24–31.

[139] R. SCHUSTER AND P. RAMCHANDRAN, *Open edge computing-from vision to reality-*. `https://tinyurl.com/yxqdk346`, 2016.
(Date accessed: 01-10-2020).

[140] U. SHAUKAT, E. AHMED, Z. ANWAR, AND F. XIA, *Cloudlet deployment in local wireless networks: Motivation, architectures, applications, and open challenges*, Journal of Network and Computer Applications, 62 (2016), pp. 18–40.

[141] Y. SHI, G. DING, H. WANG, H. E. ROMAN, AND S. LU, *The fog computing service for healthcare*, in In Proc 2nd International Symposium on Future Information and Communication Technologies for Ubiquitous HealthCare, IEEE, 2015, pp. 1–5.

[142] P. SIMOENS, L. VAN HERZEELE, F. VANDEPUTTE, AND L. VERMOESEN, *Challenges for orchestration and instance selection of composite services in distributed edge clouds*, in IFIP/IEEE International Symposium on Integrated Network Management (IM), 2015, pp. 1196–1201.

[143] R. SINGH, S. ARMOUR, A. KHAN, M. SOORIYABANDARA, AND G. OIKONOMOU, *The advantage of computation offloading in multi-access edge computing*, in 2019 Fourth International Conference on Fog and Mobile Edge Computing (FMEC), 2019, pp. 289–294.

[144] R. SINGH, S. ARMOUR, A. KHAN, M. SOORIYABANDARA, AND G. OIKONOMOU, *Heuristic approaches for computational offloading in multi-access edge computing networks*, in IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC): Track 4: Applications and Business, 2020, pp. 1–6.

[145] C. SONMEZ, A. OZGOVDE, AND C. ERSOY, *Edgecloudsim: An environment for performance evaluation of edge computing systems*, Transactions on Emerging Telecommunications Technologies, 29 (2018), p. e3493.

[146] K. SRINIVASAN AND N. K. AGRAWAL, *A study on M-CORD based architecture in traffic offloading for 5G-enabled multiaccess edge computing networks*, in IEEE International Conference on Applied System Invention (ICASI), IEEE, 2018, pp. 303–307.

[147] I. STOJMENOVIC AND S. WEN, *The fog computing paradigm: Scenarios and security issues*, in proc 2014 Federated Conference Computer Science and Information Systems, IEEE, 2014, pp. 1–8.

[148] C. SUN, J. ZHOU, X. ZHOU, X. ZHANG, AND W. WANG, *Deep learning enabled dynamic reactive video caching in mobile edge networks*, in 2018 IEEE International Conference on Communication Systems (ICCS), IEEE, 2018, pp. 280–285.

[149] J. A. SURADKAR AND R. BHARATI, *An effective computation offloading from mobile devices to cloud*, International Journal of Computer Science and Information Technologies, vol 7 (2016), pp. 1922–1927.

[150] R. SURYAWANSHI AND G. MANDLIK, *Focusing on mobile users at edge and internet of things using fog computing*, International Journal of Scientific Engineering and Technology Research, 4 (2015), pp. 3225–3231.

[151] A. T. TRAN AND R. C. PALACIOS, *A systematic literature review of fog computing*. `ojs.bibsys.no/index.php/Nokobit/article/download/312/286`, 2016. (Date accessed: 01-10-2020).

[152] C. VALLATI, A. VIRDIS, E. MINGOZZI, AND G. STEA, *Mobile-edge computing come home connecting things in future smart homes using lte device-to-device communications*, IEEE Consumer Electronics Magazine, 5 (2016), pp. 77–83.

[153] L. M. VAQUERO AND L. RODERO-MERINO, *Finding your way in the fog: Towards a comprehensive definition of fog computing*, ACM SIGCOMM Computer Communication Review, 44 (2014), pp. 27–32.

[154] B. VARGHESE, N. WANG, S. BARBHUIYA, P. KILPATRICK, AND D. S. NIKOLOPOULOS, *Challenges and opportunities in edge computing*, in Smart Cloud (SmartCloud), IEEE International Conference on, IEEE, 2016, pp. 20–26.

[155] T. VERBELEN, P. SIMOENS, F. DE TURCK, AND B. DHOEDT, *Cloudlets: Bringing the cloud to the mobile user*, in Proceedings of the third ACM workshop on Mobile cloud computing and services, ACM, 2012, pp. 29–36.

[156] C. WANG, F. R. YU, C. LIANG, Q. CHEN, AND L. TANG, *Joint computation offloading and interference management in wireless cellular networks with mobile edge computing*, IEEE Transactions on Vehicular Technology, 66 (2017), pp. 7432–7445.

[157] Q. WANG, S. GUO, J. LIU, AND Y. YANG, *Energy-efficient computation offloading and resource allocation for delay-sensitive mobile edge computing*, Sustainable Computing: Informatics and Systems, 21 (2019), pp. 154–164.

[158] S. WANG, X. ZHANG, Y. ZHANG, L. WANG, J. YANG, AND W. WANG, *A survey on mobile edge networks: Convergence of computing, caching and communications*, Ieee Access, 5 (2017), pp. 6757–6779.

[159] F. XIA, F. DING, J. LI, X. KONG, L. T. YANG, AND J. MA, *Phone2cloud: Exploiting computation offloading for energy saving on smartphones in mobile cloud computing*, Information Systems Frontiers, 16 (2014), pp. 95–111.

[160] H. XIANG, M. PENG, Y. CHENG, AND H.-H. CHEN, *Joint mode selection and resource allocation for downlink fog radio access networks supported d2d*, in 11th International Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness (QSHINE), IEEE, 2015, pp. 177–182.

[161] Z. XIONG, Y. ZHANG, D. NIYATO, P. WANG, AND Z. HAN, *When mobile blockchain meets edge computing*, IEEE Communications Magazine, 56 (2018), pp. 33–39.

[162] D. XU, Q. LI, AND H. ZHU, *Energy-saving computation offloading by joint data compression and resource allocation for mobile-edge computing*, IEEE Communications Letters, 23 (2019), pp. 704–707.

[163] X. YANG, Z. CHEN, K. LI, Y. SUN, AND H. ZHENG, *Optimal task scheduling in communication-constrained mobile edge computing systems for wireless virtual reality*, in 2017 23rd Asia-Pacific Conference on Communications (APCC), IEEE, 2017, pp. 1–6.

[164] Y. YAO, B. XIAO, W. WANG, G. YANG, X. ZHOU, AND Z. PENG, *Real-time cache-aided route planning based on mobile edge computing*, in IEEE Wireless Communications, (2020). doi: 10.1109/MWC.001.1900559.

[165] S. YI, Z. HAO, Z. QIN, AND Q. LI, *Fog computing: Platform and applications*, in Hot Topics in Web Systems and Technologies (HotWeb), 2015 Third IEEE Workshop on, IEEE, 2015, pp. 73–78.

[166] S. YI, Z. HAO, Q. ZHANG, Q. ZHANG, W. SHI, AND Q. LI, *Lavea: Latency-aware video analytics on edge computing platform*, in Proceedings of the Second ACM/IEEE Symposium on Edge Computing, 2017, pp. 1–13.

[167] S. YI, C. LI, AND Q. LI, *A survey of fog computing: concepts, applications and issues*, in Proceedings of the 2015 Workshop on Mobile Big Data, ACM, 2015, pp. 37–42.

[168] Z. YU, J. HU, G. MIN, H. LU, Z. ZHAO, H. WANG, AND N. GEORGALAS, *Federated learning based proactive content caching in edge computing*, in 2018 IEEE Global Communications Conference (GLOBECOM), IEEE, 2018, pp. 1–6.

[169] J. K. ZAO, T. T. GAN, C. K. YOU, S. J. R. MÉNDEZ, C. E. CHUNG, Y. TE WANG, T. MULLEN, AND T. P. JUNG, *Augmented brain computer interaction based on fog computing and linked data*, in International Conference on Intelligent Environments (IE), IEEE, 2014, pp. 374–377.

[170] L. ZENG, L. LI, L. DUAN, K. LU, Z. SHI, M. WANG, W. WU, AND P. LUO, *Distributed data mining: a survey*, Information Technology and Management, 13 (2012), pp. 403–409.

[171] W. ZHANG, Y. WEN, AND H.-H. CHEN, *Toward transcoding as a service: energy-efficient offloading policy for green mobile cloud*, IEEE Network, 28 (2014), pp. 67–73.