Investigating applications of optimization techniques in mobile cloud computing

Chase, Jonathan David

2017

Chase, J. D. (2017). Investigating applications of optimization techniques in mobile cloud computing. Doctoral thesis, Nanyang Technological University, Singapore.

<http://hdl.handle.net/10356/71542>

<https://doi.org/10.32657/10356/71542>

NANYANG TECHNOLOGICAL UNIVERSITY

Investigating Applications of Optimization Techniques in Mobile Cloud Computing

Jonathan David Chase

Supervisor: Prof. Dusit Niyato

School of Computer Science and Engineering

A thesis submitted to the Nanyang Technological University
in partial fulfillment of the requirement for the degree of
Doctor of Philosophy

2017

Acknowledgments

First and foremost, I would like to thank my supervisor, Prof. Dusit Niyato, for providing me with the opportunity to pursue a PhD at Nanyang Technological University. His direction, expertise, insight, and advice have been invaluable over the last few years. He has always been supportive and willing to spare time to discuss improvements to my research. It has been a privilege to work with such an accomplished and personable academic. I would also like to thank Prof. Ping Wang, Prof. Yonggang Wen, Dr Sivadon Chaisiri, Dr Zhiguang Cao, Dr Ryan Ko, Dr Hongliang Guo, and Rakpong Kaewpuang for their invaluable collaboration. In addition, I gratefully acknowledge the scholarship from Nanyang Technological University and the support of my colleagues.

No acknowledgements section would be complete without recognising my family, both in the UK, and my in-laws in Singapore. I am deeply thankful for my parents, who raised me with love and care, for my brother, who inspired me to do my best, and for my in-laws, for making a foreign country feel like a second home. Most of all I am grateful for my excellent wife Denise, without whom none of this would have been remotely possible. I would also like to thank my cat, Patches, who made many unsolicited contributions to this thesis. Last, but not least, I thank God, who has been endlessly good to me.

Contents

Acknowledgments	1
List of Figures	6
List of Tables	9
List of Abbreviations	10
Abstract	11
1 Introduction	13
1.1 Joint Virtual Machine and Bandwidth Allocation	16
1.1.1 Virtual Machine Provisioning	17
1.1.2 Bandwidth provisioning	19
1.2 Bring-Your-Own-Application	21
1.2.1 Mobile Application Migration	23
1.3 Joint Security and Cyber Insurance Provisioning	25
1.3.1 Cyber Insurance	27
1.4 Stochastic Optimization	29
1.5 Total Unimodularity	31
1.6 Partial Lagrange Multiplier Method	32
1.6.1 Partial Total Unimodularity of Constraint Matrix	33
1.6.2 Partial Lagrange Multiplier	33
1.6.3 Solution to the Partial Lagrange Multiplier	34
1.7 Key Contributions	36
1.7.1 Chapter 3	36
1.7.2 Chapter 4	37
1.7.3 Chapter 5	38
2 Literature Review	40
2.1 Virtual machine and bandwidth allocation in cloud computing environments	40
2.1.1 Virtual Machine Allocation	40

2.1.2	Bandwidth Allocation	44
2.1.3	Software Defined Networking	46
2.1.4	Uncertainty in Allocation	49
2.2	Bring-Your-Own-Device and Mobile Device Migration	53
2.2.1	Mobile Application Migration	53
2.2.2	Bring-Your-Own-Device	58
2.3	Cyber Insurance	62
2.3.1	Security-as-a-Service	63
2.3.2	Cyber Insurance	65
3	Joint Optimization of Resource Provisioning in Cloud Computing	72
3.1	System Model	72
3.1.1	Provisioning Scheme	74
3.1.2	Reservation Contracts	75
3.1.3	Uncertainty of Parameters	76
3.1.4	Provisioning Costs	77
3.2	Problem Formulation	77
3.2.1	Stochastic Optimization Formulation	78
3.2.2	Deterministic Equivalent Formulation	83
3.3	Scenario Tree Reduction	83
3.3.1	Tree Reduction	85
3.3.2	Tree Construction	87
3.4	Sensitivity Analysis	88
3.5	Performance Evaluation	93
3.5.1	Parameter Settings	93
3.5.2	Numerical Results	95
3.6	Conclusion	104
4	Bring-Your-Own-Application (BYOA): A Scalable Stochastic Mobile Application Migration Service	107
4.1	System Model	108
4.1.1	Uncertainty Parameters	110
4.2	Problem Formulation	111
4.2.1	Deterministic Equivalent Formulation	115
4.3	Partial Lagrange Multiplier Method	116
4.3.1	Total Unimodularity of Constraint Matrix	116

4.3.2	Partial Lagrange Multiplier	117
4.3.3	Solution to the Partial Lagrange Multiplier	118
4.4	Theoretical Performance Analysis	120
4.4.1	Convergence Analysis	120
4.4.2	Scalability Analysis	124
4.5	Sensitivity Analysis	125
4.6	Performance Evaluation	129
4.6.1	Parameter Settings	129
4.6.2	Numerical Results	130
4.6.3	Sensitivity Analysis Result	138
4.7	Conclusion	139
5	A Scalable Approach to Joint Cyber Insurance and Security-as-a-Service Provisioning in Cloud Computing	141
5.1	System Model	141
5.1.1	Security-as-a-Service Model	142
5.1.2	Cyber Insurance Model	143
5.1.3	Decision Stages	143
5.1.4	Uncertainty Parameters	145
5.1.5	Decision Variables	146
5.2	Problem Formulation	147
5.3	Partial Lagrange Multiplier Method	151
5.3.1	Total Unimodularity of Constraint Matrix	152
5.3.2	Partial Lagrange Multiplier	152
5.3.3	Solution to the Partial Lagrange Multiplier	154
5.4	Theoretical Performance Analysis	156
5.4.1	Convergence Analysis	156
5.4.2	Scalability Analysis	160
5.5	Sensitivity Analysis	161
5.6	Performance Evaluation	164
5.6.1	Parameter Settings	164
5.6.2	Numerical Results	165
5.7	Conclusion	176

6	Conclusions and Further Work	178
6.1	Conclusions	178
6.2	Future Work	181
6.2.1	Integrated System	181
6.2.2	Joint Virtual Machine and Bandwidth Allocation	182
6.2.3	Bring-Your-Own-Application	183
6.2.4	Joint Security-as-a-Service and Cyber Insurance Provisioning . .	184
6.3	Summary	184
	References	186
	Author's Publications	199

List of Figures

1.1	Typical SDN Architecture. The Data Plane is unchanged, with the key point of implementation being the interface between the Control and Data planes. Implementations such as OpenFlow address this interface.	18
1.2	Typical mobile application migration scenario. Middleware provides a common environment allowing the offloading of a whole VM or code segment to either a cloud data center, or local equipment. Typically, computation is paused on the mobile device and resumed when remote computation is complete and the resulting VM or program state is returned.	23
1.3	Overview of the techniques used in each chapter. Each chapter addresses a resource allocation problem using stochastic optimization, followed by a runtime improvement technique and a sensitivity analysis.	39
3.1	Layout of network with regions, cloud providers, and clients (users) labeled.	73
3.2	Possible scenario tree layouts.	84
3.3	Probability distribution of VM demand test data. Each curve represents a different VM demand scenario.	95
3.4	Variation of cost with VM reservation.	96
3.5	Variation of cost with bandwidth reservation.	97
3.6	Variation of cost with VM and bandwidth reservation.	98
3.7	Comparison of joint and separate provisioning solutions when provider CPU capacity varies.	99
3.8	Comparison of joint and separate provisioning solutions when South gateway router bandwidth capacity varies.	99
3.9	Magnified view of changed probability distributions to highlight shape changes.	100
3.10	Effect of varying VM demands on network router bandwidth allocation across the network.	101

3.11	Effect of price change on bandwidth allocation to gateway router between network regions 3 and 1.	102
3.12	Total cost of different solution methods under different demand probability distributions.	103
3.13	Precise cost coefficient thresholds triggering VM reservation changes. .	104
3.14	Full price of contract $L2$ in each network region, with the price needed for the contract to become useful.	105
4.1	Illustration of multi-location system model and intra-location equipment allocation.	109
4.2	Illustration of user allocation at time stages and uncertainty scenarios for an individual location.	111
4.3	Impact of equipment availability on lost users.	131
4.4	Impact of battery levels on lost users.	132
4.5	Comparison of greedy approach with stochastic optimization with variation in user duration.	133
4.6	Benefits of migration on total profit vs non-migratory method.	133
4.7	Behaviour of migration as equipment quantity varies.	134
4.8	Effect of migration probability on user allocation.	135
4.9	Probability of user movement based on trace data.	136
4.10	Effect of price variation on user allocation.	137
4.11	Comparison of execution time of B&B algorithm vs partial Lagrange multiplier method.	137
4.12	Comparison of convergence rates with different stepsize calculation methods.	138
4.13	Comparison sensitivity analysis results for different parameters.	140
5.1	Illustration of interaction between system components.	142
5.2	In each time period, there are three provisioning stages. This figure illustrates an example case with 3 time periods, however this can be freely expanded to n time periods.	144
5.3	Impact of prepaid SecaaS services on service utilization and total cost.	166
5.4	Impact of packet damages on security provisioning measured in unprocessed packets. As damages increase in cost, it becomes more costly to leave packets unprocessed, and so security provisioning is increased to ensure more packets are processed.	167

5.5	Impact of packet indemnity on security provisioning measured in unprocessed packets. Higher indemnity provides an incentive to increase security provisioning so that packets are not left unprocessed.	168
5.6	Comparison of alternative solution methods with varying attack probability distributions.	169
5.7	Exploration of the relationship between security and insurance provisioning under budget constraints.	170
5.8	Exploration of the effect security capacity has on total cost.	171
5.9	Demonstration of the benefits of joint optimization.	172
5.10	Comparison of execution time and memory usage of partial Lagrange multiplier and branch-and-bound (B&B) methods as problem size increases.	173
5.11	Example case of speed of convergence of the partial Lagrange algorithm to the optimal value.	174
5.12	Cost breakpoints of the variable $X_{1,1,1,1,1}^\dagger$, comparing analytical and experimental approaches to sensitivity analysis.	175
5.13	Cost breakpoints of the variable $X_{1,1,2,1,1}^\dagger$, for the full-size problem showing the choices made between plans and providers.	176
6.1	Larger mobile cloud computing context, where the user chooses between cloud offloading, with optimized VM and bandwidth allocation, and offloading to local equipment while mobile. Security and insurance services are provisioned to cover their offloading in each case.	182

List of Tables

2.1	Summary of works addressing problems in virtual machine and bandwidth allocation	51
2.2	Summary of works addressing problems in mobile application offloading and Bring-Your-Own-Device	61
2.3	Summary of works addressing problems in Security-as-a-Service and cyber insurance	70
3.1	List of Key Notations for Chapter 3.	74
3.3	Cost per Mbps of bandwidth provisioning through network provider routers	94
3.4	Impact of scenario reduction on optimization performance	100
3.2	VM costs for each cloud provider and provisioning phase.	106
4.1	List of Key Notations for Chapter 4.	112
5.1	List of Key Notations for Chapter 5.	148

List of Abbreviations

API	Application Programming Interface
B&B	Branch-and-Bound algorithm
BYOA	Bring-Your-Own-Application
BYOD	Bring-Your-Own-Device
CSP	Cloud Service Provider
IaaS	Infrastructure-as-a-Service
IMP	Insurance Management Process
IPM	Interior Point Method
IPS/IDS	Intrusion Prevention/Detection System
ISP	Internet Service Provider
LAN	Local Area Network
LP	Linear Program
MAM	Mobile Application Management
MDM	Mobile Device Management
MIM	Mobile Information Management
MIP	Mixed Integer Program
QoS	Quality of Service
SaaS	Software-as-a-Service
SDN	Software Defined Network
SECaaS	Security-as-a-Service
SLA	Service Level Agreement
SME	Small and Medium Enterprise
SMP	Subscription Management Process
TU	Total Unimodularity
VLAN	Virtual Local Area Network
VM	Virtual Machine
VoD	Video-on-Demand
VPC	Virtual Private Cloud
VPN	Virtual Private Network
WAN	Wide Area Network

Abstract

Cloud computing, and particularly mobile cloud computing, contains a great deal of uncertainty. User requirements are changeable, and environmental factors, such as network delay and bandwidth availability, as well as changing prices, make efficient allocation of resources challenging. Stochastic optimization is a technique designed to solve optimization problems, such as for resource allocation, in a way that accounts for the uncertainty of future parameters. It is a flexible and powerful technique, and in this thesis I demonstrate three key areas where it can be used with great efficacy.

The two key resources in mobile cloud computing are computation and bandwidth. The problem of migrating computation to the cloud through virtual machines (VMs) has been addressed in prior work, as has the problem of routing bandwidth through a network effectively. However, to provision resources in the cloud, these two must be allocated in concert, as they are interdependent. Without sufficient bandwidth, efficient computation will be wasted, and a clear network path is worthless if the destination lacks computing power. Many cloud services (e.g., Amazon) offer a reservation option for VMs that can be used to reduce costs if uncertainty is properly accounted for. Likewise, the recent development of Software Defined Networking (SDN) allows the creation of flexible, multi-tenant networks allowing bandwidth reservation. In this thesis I apply stochastic optimization to the problem of jointly provisioning VMs and bandwidth despite uncertain user demand and price information and show results to demonstrate the importance of joint allocation.

Much of research in mobile cloud computing focuses on the problem of energy efficiency in mobile devices. In addition to offloading to the cloud, the rise of the Bring-Your-Own-Device (BYOD) paradigm has inspired research into offloading to local equipment (such as a desktop PC). Users who require low latency or a larger screen may prefer not to offload to the cloud, so I propose a model where users can travel to a location (such as a library) and offload their computation to local equipment. Due to a variety of factors, such as user battery levels, number of users, and application requirements, deterministic methods are inadequate. I apply stochastic optimization to the problem to maximize the number of users able to access equipment.

The third major issue in cloud computing is the problem of computer security. Mobile users offloading to the cloud, or to local equipment, require reassurance that their data and communications will be secure. Thus, Security-as-a-Service (SECaaS) has arisen to provide security in the cloud without taxing the user’s processing resources. However, no security is perfect, and a successful cyber attack is inevitable. Cyber insurance provides compensation in the event of a successful attack. In this thesis I consider a joint provisioning of cyber insurance and SECaaS using stochastic optimization to reserve security services and purchase insurance policies, allowing for uncertainty in attack probabilities, attack success and pricing. I find that cyber insurance can be used to encourage security investment.

This thesis contributes to the field of mobile cloud computing in two major ways. The first is the application of stochastic optimization to three vital and related areas in computation offloading, bandwidth provisioning, and cyber security and insurance. In each case I provide numerical results based on parameters derived from real usage data, such as cluster usage traces, mobile user movements, and honeypot attack data. These numerical results show the effectiveness of stochastic optimization as it accounts for uncertainty in a way that is necessary for a realistic model of cloud computing. Secondly, I provide performance optimizations and sensitivity analyses for each problem area. As some decision variables are integer values, I employ two methods (a scenario tree reduction method and a partial Lagrange multiplier algorithm) to significantly improve the solution runtime and show their effectiveness through analytical and numerical results. I formulate two linear programs to perform the sensitivity analysis, giving precise values for the sensitivity range of parameter values.

In summary, in this thesis I focus on the application of stochastic optimization to key areas in mobile cloud computing. Existing research either fails to account for uncertainty, or fails to consider interdependent resources in a suitably rich joint approach. To the best of my knowledge, this is the first work to consider joint VM and bandwidth provisioning under uncertainty, to propose the local equipment offloading model, and to apply multi-stage recourse and real test data and sensitivity analysis to joint SECaaS and cyber insurance provisioning. Thus each contribution introduces novel system models, with solution improvements, analysis, and test data that has not been considered until now.

Chapter 1

Introduction

The history of computer science can be broadly characterized as a cycle of centralization and de-centralization. The earliest computers were too large and expensive to be personally owned, and followed a client-server model with users accessing powerful mainframes through connected terminals. With the rapid increase of the computational power of computers in conjunction with the decrease in component size, the focus was on the establishment of personal computing. This trend has continued into smart mobile devices, with powerful devices that can be kept in a pocket. However, the battery limitations of mobile devices, and increasing computational expectations have led to users again looking to offload computation to powerful centralized processing locations. With the wide coverage of high bandwidth Internet, a new model has arisen to enable the re-centralization of computation. Instead of connecting to a terminal on-site, users now connect over the Internet, with access to services that run the whole gamut of computing potential - from infrastructure, through to services and application suites. This paradigm of outsourcing computing needs to remote data centers has been dubbed ‘cloud’ computing, with computing and storage taking place ‘up in the cloud’ in a way that is opaque to the user. One of the key uses of cloud computing is for access to flexible computing resources without the cost of investing in physical infrastructure and without draining a mobile battery. This is often in the form of virtual machines (VMs) that are instantiated on a data center’s servers. The provisioning of VMs and other resources, such as bandwidth, is a key topic of current research, as the cost incurred by the user can increase dramatically if the resources allocated to them are excessive. VM offloading is particularly useful for mobile users and

two key questions in cloud research are, therefore, how to assign VMs and bandwidth to physical devices, and when and where to offload mobile computation. I address these questions of resource allocation in this thesis, examining the most cost effective way to allocate cloud resources to users, and providers, according to demand and the problem constraints. As sensitive data is increasingly stored in the cloud, both the servers hosting the data and the Internet connections that transport it are vulnerable to attack. Thus, security services (called Security-as-a-Service, or SECaaS), are also offered in the cloud. These services can be provisioned in the same manner as VMs, to protect data from external attack. However, cyber security is never infallible, and both users and service providers must anticipate the financial loss that can result from a security breach. As a result, the market of cyber insurance has grown to provide insurance for damages resulting from cyber attacks, where the data may not have a literal financial value, and the costs and risk levels are difficult to determine accurately. Thus, to be comprehensively protected when migrating computation to the cloud, or even to local computing equipment, a user must purchase both security services and insurance policies.

Provisioning resources in a cloud computing context involves a great deal of uncertainty. In this thesis I employ a technique called ‘stochastic optimization’, which calculates resource allocation through the formulation of an optimization problem, with some parameters behaving as random variables that can take a range of values. The purpose of this thesis is to examine the ways this technique can be applied to the problem areas outlined above, as each problem area addresses a different key issue in mobile cloud computing, namely performance guarantees, battery life constraints, and security. First, I consider the problem of a user offloading VMs to the cloud, using a stochastic optimization problem that aims to reduce cost to the user through effective allocation of bandwidth and VMs to service providers. Secondly, I consider a mobile user who wishes to offload computation, not to the cloud, but to a nearby piece of computing equipment, such as a desktop computer. For mobile users who wish to take advantage of, for example, a larger screen, offloading to the cloud (as in the first problem) may not be the best option. Rather, they may prefer to visit a location,

such as a library or coffee shop, and offload to a nearby computer. Stochastic optimization can be used to allocate users to equipment, and point them to alternative locations if the local equipment is in use. Thus, these two problems can be regarded as two tiers in the scenario of a mobile user offloading their computing activity to the cloud, with the mobile user able to choose which of the two offloading options is preferable. Finally, I apply stochastic optimization to the problem of cloud security, a key issue when offloading from a mobile device. When offloading to a remote location, users must be confident that their data and device remain secure, and provisioning security services in the cloud increases this security. Using stochastic optimization, I consider how users can optimally provision protection for their remotely executed applications, and whether the availability of cyber insurance policies can encourage improved security provisioning. For each of the three problems, I apply techniques to improve the runtime performance of the stochastic optimization solution, as all three problems use integer variables. Integer problems are NP-hard [1] and general solution methods have exponential complexity [2], and thus computation time increases rapidly with the problem size, until optimal solution calculation becomes intractable. I also apply an analytical sensitivity analysis technique to examine the behaviour of the solution, investigating how various parameters, such as price, may be altered and how these alterations influence the optimal solution. In performing these analyses, this thesis will show that stochastic optimization can be usefully applied to improve resource allocation in each of these areas, and therefore can be applied in all areas of mobile cloud computing to improve performance, battery life, and cyber security.

In this chapter I provide more detailed introductions to each problem case in Sections 1.1-1.3, outlining some key technologies and explaining the key design choices that must be made when designing a system model in these areas. In Sections 1.4-1.6 I provide detailed introductions to Stochastic Optimization and the partial Lagrange multiplier method, for the benefit of readers who are unfamiliar with these techniques. The mathematical detail given here provides the basis for scalability proofs given later in the thesis. Additionally, the reader will see that stochastic optimization proves to be a powerful technique for reducing costs (and increasing profits) in mobile cloud

computing, and can be applied to benefit both the user and the service provider. At the end of this chapter I list the key contributions of this thesis, highlighting the unique contributions made by this work, and demonstrating why I believe this work is a valuable addition to the field of mobile cloud computing.

1.1 Joint Virtual Machine and Bandwidth Allocation

In recent years, cloud computing has increased dramatically in popularity. The Infrastructure-as-a-Service (IaaS) paradigm allows a great deal of scalability, but resources must be used efficiently. Virtual Machines (VMs) can be provisioned to meet a users' demand. Services such as Amazon's EC2 [3] allow the advance reservation of VMs, which can be utilized at a lower price than the alternative on-demand instances. This introduces the risk of oversubscription and undersubscription, as users may waste money on excessive reservation, or overspend on on-demand instances, so uncertain demand must be considered when provisioning VMs. Stochastic programming [4] takes into account the uncertainty of future demand and chooses an optimal reservation amount.

Increasingly, VMs placed in the cloud require a large amount of bandwidth. The popular Video-on-Demand (VoD) service, Netflix [5], uses Amazon cloud services, and requires significant bandwidth between itself and users as a fundamental resource of its service. Guaranteeing bandwidth availability is very important, but has received inadequate attention in research. A combined approach to provision both VMs and bandwidth is necessary to guarantee performance and minimize cost. Software Defined Networking (SDN) [6] has emerged as a practical solution for guaranteeing bandwidth both within data centers and across the Internet. SDN decouples the routing of data from network control, allowing for virtual networks with customized bandwidth flow, that exceeds the abilities of previous techniques. There is great potential for network providers to extend their service provision to offer bandwidth guarantees in an end-to-end fashion, as cloud providers such as Google are already doing for their own WANs [7].

In this type of cloud environment, clients use a broker to provision virtual machines from cloud providers. Cloud providers deploy VMs using a hypervisor to manage instances. For example, Amazon’s EC2 service uses a hypervisor based on Xen’s [8] paravirtualization approach. SDN, as illustrated in Fig. 1.1 can be used to provide a virtual data center network, or Virtual Private Cloud (VPC), to allow communication between instances. The OpenFlow protocol [9] is a popular ‘Southbound’ API for passing routing instructions to OpenFlow-enabled switches and routers. OpenFlow allows an SDN controller to provide software-based virtual networks that can guarantee Service Level Agreements (SLAs). Cloud provider-based VPCs can be joined across data centers to form an SDN-based WAN. Google’s Andromeda [7] service already exploits OpenFlow for this purpose. Clients can then connect to their VPCs via VPN and make them an extension of their own networks. However, network providers currently lag behind Over-The-Top (OTT) service providers in supporting SDN. I therefore envision an SDN-based VPN that offers bandwidth provisioning through a similar model to VM provisioning, extending the SDN-based WAN concept to include the client. In this way, network providers can offer customisable bandwidth provisioning, by allowing clients to provision bandwidth along a specified network path. This provisioning of both VMs and bandwidth allows the optimization of cost, whilst guaranteeing performance. I assume that there is no inter-VM bandwidth requirement, as placing VMs efficiently to work together is beyond the scope of this thesis.

1.1.1 Virtual Machine Provisioning

To provision virtual machines in a cloud environment, a consumer typically creates virtual machines packaged with the required software to perform a task. These VMs may be uploaded to a repository or run on the customer’s private cloud (if they possess one). A cloud broker then distributes the VMs to cloud providers. Cloud providers use hypervisor software to run VMs on data center servers and manage the placement of VMs on physical servers. Particular areas of interest in VM provisioning are:

- **Inter-data center vs Intra-data center:** the placement of VMs between data centers is handled by the consumer-side broker, but is not concerned with

Figure 1.1: Typical SDN Architecture. The Data Plane is unchanged, with the key point of implementation being the interface between the Control and Data planes. Implementations such as OpenFlow address this interface.

the particular physical machines that host VMs. Cloud providers must make decisions regarding the placement of VMs at the intra-data center level and is typically of interest in energy-aware placement that consolidates VMs to minimize power consumption.

- **Online vs Offline:** some applications require rapid response times, so live, online decision-making may be more desirable. However, in large-scale systems, it may not be practical to perform computation in real time, and so an offline approach may be more appropriate.
- **Reactive vs Predictive:** as decisions may be made online or offline, so this choice may be influenced by whether the decision-making is reactive or predictive. Reactive decision-making is well suited to a VM consolidation algorithm that responds to current load and is executed online, whilst a predictive approach may suit an offline implementation that provisions VMs to anticipate future demand.

- **Stochastic vs Deterministic:** it is simpler to solve a deterministic problem, and if it can be assumed that there is no uncertainty in the system, a precise optimal solution can be efficiently found. However, in reality uncertainty exists in the system, so it may be more realistic to use a stochastic approach. However, the stochastic approach may require use of a heuristic and therefore a suboptimal solution.

This thesis focuses on the problem of inter-data center allocation, as intra-data center allocation is already well-researched. Rather, the focus is on allowing for uncertainty in a less regulated environment, where a stochastic approach is necessary. The use of stochastic optimization means the solution is predictive and calculated in an off-line fashion. This allows the possibility of advance planning and greater flexibility regarding scalability.

1.1.2 Bandwidth provisioning

To provision bandwidth in a cloud context, a network owner must offer virtualized networking functions. However, this has traditionally received less attention than virtualized computation at the cloud level because networks can be huge and complex, are highly heterogeneous, and are complex to manage. Existing network virtualization technologies are unable to cope with the scale of the problem in cloud computing, but the introduction of Software Defined Networking has made large scale, customised virtual networking a realistic option. SDN decouples the problem of network control from the problem of data forwarding whilst using existing networking devices and protocols. Network owners manage data routing at the physical level, whilst offering tenants control of a virtual network that operates transparently over the network infrastructure. Thus a tenant can provision a specific amount of bandwidth for a network link that the network owner is responsible for ensuring without giving the tenant direct control of the network infrastructure. Therefore, SDN can be deployed to provide large scale cloud bandwidth provisioning. Issues in bandwidth provisioning in the cloud typically revolve around considering bandwidth individually, or as a requirement of VM placement. Key issues in bandwidth provisioning include:

- **Solution Scalability:** whilst VM provisioning may only include a handful of hosts and clients, allowing relatively scalable problems, a bandwidth provisioning solution must address all the links and nodes connecting hosts together, and the routing of traffic between them. This can quickly become intractable, so solutions may simplify the problem by assuming specific network topologies or employing heuristics to find a sub-optimal, but computationally feasible solution.
- **Inter-data center vs Intra-data center:** this is also an issue in bandwidth provisioning, but for different reasons to VM provisioning. Specifically, the network environment defines the scale of the problem to be solved, and defines the assumptions that can be made about network topology. A data center network is typically more regularly structured than the Internet, and may conform to a known design, such as a tree, that can be used to improve scalability.
- **Latency and Delay:** as VM provisioning may be stochastic or deterministic, so bandwidth provisioning may assume that there are no external factors influencing the performance of a network link. In a data center a network controller may be aware of, and able to predict, all traffic on the network, meaning that network capacities remain fixed. However, in an Internet context, there may be other traffic that is difficult to anticipate, that is not sourced within an ISP's own area of control, and thus the available bandwidth on a link may be more varied.
- **Bandwidth-aware VM Placement vs Full Bandwidth Provisioning:** a number of VM placement solutions consider bandwidth as a requirement of VM placement as a parameter, similar to CPU power and memory capacity [10], or to minimize inter-communication [11]. This approach focuses on the placement of VMs and typically assumes a tree-shaped network architecture with fixed bandwidth routing [12]. This may be overly simplistic, and a more complete approach considers bandwidth provisioning as a valid problem to consider in its own right.

- **Joint vs Separate Provisioning:** until now, the vast majority of research focuses on VM placement and bandwidth provisioning as separate problems, except for ‘bandwidth-aware’ VM placement as discussed above. This is typically a question of scalability, but also the limited virtualization options offered for bandwidth prior to the introduction of SDN. In future, research will need to consider the two problems as connected, as I do in this thesis.

Bandwidth-aware VM placement has been addressed in the literature, so this thesis addresses the problem of full bandwidth provisioning, on a joint basis with VM allocation, to demonstrate the interdependence between the resources. Latency and delay are not addressed here, as it is an issue that requires further work to adequately address, rather this work focuses on improving the scalability of the allocation. This is, therefore, the first problem area addressed in this thesis. However, it is premised on the assumption that the user is not concerned about battery life, and the number of users is built into the system configuration. The second problem addressed in this thesis therefore considers a new approach to handling a scenario with varying numbers of battery-conscious users, and is introduced in the following section.

1.2 Bring-Your-Own-Application

Mobile computing is the dominant modern computing paradigm, with even less ‘tech-savvy’ users embracing laptops, tablets, and smartphones. The increased power and potential of mobile devices has led to ‘consumerization’ in computing, as users increasingly expect to be able to work and play on their own devices, rather than corporate-issue machines. The Bring-Your-Own-Device (BYOD) [13] paradigm is an excellent example of this, as organisations support the ability of their employees to work on their own devices and on the go, with an expected corresponding increase in worker satisfaction. However, whilst mobile devices are vastly more powerful than even their recent predecessors, both processing power, battery life, and screen size are limited. Accompanying the rise of mobile computing, cloud computing has seen a return to the traditional terminal-mainframe style of computation, with the addition

of Internet-connected mobility. Paradigms such as Infrastructure-as-a-Service (IaaS), and Software-as-a-Service (SaaS) provide the basis for users to now migrate their applications to the cloud, offloading the most computation-intensive parts of their work, speeding up execution and saving energy.

Users may migrate a complete virtual machine (VM) environment, or segments of code for execution, but all of these options require bandwidth and are subject to latency problems. In this chapter, I consider a local migration environment, where users may offload their computation to local equipment such as a smart TV, monitor, or desktop computer. Through the use of application migration, users who desire to work ‘on the go’ at locations such as coffee shops or libraries can do so by using equipment the way one might access a WiFi hotspot. This migration can be done through app and desktop virtualization technology, for example, Citrix [14]. On the other hand, Microsoft’s Continuum [15] allows the conversion of a mobile device to a full desktop through connection to a screen and peripherals. Continuum carries out computation on the mobile device, but transforms the application in use from a mobile version to a desktop one. Figure 1.2 illustrates mobile application offloading technology, as it is implemented on the mobile device, and the possible scenarios of offloading to the cloud or to a local computer.

I use the term ‘Bring-Your-Own-Application’ (BYOA) for the idea of the user bringing their device and app to offload to available equipment, inspired by the concepts of BYOD and IaaS to provide a local migration service. I consider a scenario where multiple users arrive at different locations, with varying states of battery charge and app requirements and design a stochastic optimization formulation to find the optimal allocation of users to equipment whilst anticipating future user arrivals. Users may move between locations to find the best equipment allocation, and so I minimize the users lost. The Continuum concept is in keeping with Microsoft’s aim for a consistent ecosystem between desktop and mobile, an idea that is mirrored in Google’s Chromebook. Given the established trend towards seamless integration of desktop and mobile computing, and given the technology for application migration that is

currently available, the BYOA system model is a logical and perhaps inevitable development. The goal of this thesis is not to invent a problem for the sake of solving it, but rather Chapter 4 aims to demonstrate that the proposed system model is practical and beneficial to users, and is in keeping with the current progression of mobile cloud computing trends.

Figure 1.2: Typical mobile application migration scenario. Middleware provides a common environment allowing the offloading of a whole VM or code segment to either a cloud data center, or local equipment. Typically, computation is paused on the mobile device and resumed when remote computation is complete and the resulting VM or program state is returned.

1.2.1 Mobile Application Migration

Mobile application migration is typically performed in one of two ways. Either an entire VM is offloaded, or an application is partitioned and particular segments of code are offloaded. To migrate an application, the process running on the mobile device is paused, and the application (either VM or code segment) copied to the cloud. Once

the needed work has been performed, the application is returned to the mobile device and the process is unpaused. It is typically necessary for a middleware application to provide an interface, due to the heterogeneity of applications. To perform this offloading in a public location to available equipment, it is also important to ensure security, for example, through the use of a Mobile Device Management (MDM) system, that can provide authentication of registered mobile devices. However, this area still has a number of outstanding research areas:

- **Security:** To prevent the infection of the host system, devices must be successfully authenticated, and must either be able to prove that they are free of infection or be isolated in a sandbox by the service provider. When using local equipment, this isolation is difficult to guarantee, so I use an MDMS-based approach here. Likewise, the user must be confident that the host equipment is secure. With a large scale cloud provider such as Amazon, a user may assume a degree of trustworthiness, but for locally provided equipment there must be a way for safety to be demonstrated.
- **Local vs Cloud Offloading:** Much of current research focuses on the mechanism of offloading, rather than the destination. However, there is a choice to be made between offloading to local resources (such as nearby hardware, or a cloudlet), or to the cloud. Offloading to the cloud offers greater flexibility in computing power, but may suffer from latency. Local resources may offer a faster response time, and allow the user to improve the interface experience through use of a larger screen or keyboard and mouse, but there is less scalability for demanding applications.
- **Virtual Machine vs Application Partitioning:** To offload, it must be determined exactly what to offload. This can either be done by migrating a complete VM, as in the VM provisioning introduced above, or by partitioning an application. Application partitioning is more complex, as it requires the target host to be able to run the offloaded code (hence Java is a popular language

for applications that may involve offloading), and the most relevant parts to offload must be identified. This identification must be either manual, which is time-consuming, or automatic, which is an open problem. Offloading a VM is simpler, and is the approach I prefer for this thesis, but requires more data to be transmitted, and therefore needs more bandwidth and consumes more battery.

The security of mobile application offloading is not directly addressed in Chapter 4, rather the work in Chapter 5 can be applied to this area in future work. Likewise, this work focuses on the problem of local offloading, as the goal is to preserve battery life. The scenario presented here is a novel one, and future research should investigate cloud-based offloading that integrates with the resource provisioning in Chapter 3. For the sake of security and simplicity, it is assumed that offloading is performed at the virtual machine granularity, as application partitioning still requires a mobile device to be in use, and the model introduced is intended to allow a user to save battery life by not using their mobile device. Given these design choices, Chapter 4 introduces the system model and problem formulation, with the aim of demonstrating the practicality of the BYOA scenario. Thus, having addressed the problem of cloud resource provisioning to guarantee performance, and local mobile application migration to preserve battery life, there remains one key area of mobile cloud computing that has not been addressed by stochastic optimization, namely the ever-present problem of cyber security, which is introduced in the following section.

1.3 Joint Security and Cyber Insurance Provisioning

In a 2014 report by McAfee it was estimated that financial losses due to cyber risks were between USD 300 billion and USD 1 trillion a year [16]. The Identity Theft Resource Center’s 2016 data breach category summary found that as of November, there were 873 recorded breaches in the US with over 29 million records exposed [17]. With the number of cyber attacks growing, successful attacks are now a question of ‘when’, not ‘if’. Yet, the Monetary Authority of Singapore observed that cyber

insurance adoption for small and medium-sized enterprises (SMEs) is less than 10% in Asia, despite 42% of the world's Internet users living in the region [18]. 90% of all cyber insurance is purchased by US companies, whilst in the UK, only 2% of companies have specialist cyber insurance. The market, therefore, has considerable room for further growth, with PricewaterhouseCoopers (PwC) estimating that annual premiums could grow USD 5 billion by 2018 and exceed USD 7.5 billion by 2020 [19].

Outsourcing computation to the cloud is now common practice, and it is not surprising that the Security-as-a-Service (SECaaS) paradigm has arisen to counter the growing level of cyber threats. Thus, an application may guard against attacks by provisioning security services from providers such as McAfee [20] and Trend Micro [21]. These services may take various forms, such as secure data storage, identity and access management (IAM), and intrusion detection services to screen incoming traffic [22]. These services can be provisioned in a similar manner to other cloud services, either through advance subscription or dynamically through on-demand options. SECaaS allows customers to reduce their security overheads while maintaining a high level of protection.

Despite the variety of security options available, it is inevitable that they will eventually be circumvented. Cyber insurance is used to provide explicit cover in the event that malicious activity leads to financial loss. Insurance coverage may be first- or third-party with first-party insurance covering eventualities such as theft of money and digital assets, business interruption, and cyber extortion. Third-party insurance may cover problems such as privacy breaches, loss of third-party data (e.g. user account information), and public relations expenses [23]. Major insurers, such as Allianz [23] or QBE [24] offer cyber insurance policies that cover a range of first- and third-party risk. Cyber insurance is an important and growing field, but carries some unique features that makes it challenging. For example, in 2013, millions of credit card details were stolen from Target [25]. The extent of a cyber attack can be difficult to assess, and Target revised its estimate of stolen user records from 40 million to over 70 million. Additionally, the costs of the attack were not measured in the literal value of the data stolen, but must be calculated from a combination of factors, such

as damage to business (sales dropped 3-4% over the previous year), settlement costs with credit card companies, management of public relations damage, and legal fees. By 2016, costs directly attributable to the breach neared USD 300 million, but the damages were offset by USD 90 million in cyber insurance payouts [26]. Therefore, it is clear that for the majority of companies, whose data breaches will be of lesser magnitude, cyber insurance is a crucial addition to their security plans.

As the Target case shows, damages are costly, indemnity payouts can be expensive, and it is in an insurance company's best interests to encourage customers to also provision adequate security, rather than only buying cyber insurance. [27] introduced a combined security and cyber insurance provisioning scheme that balances the need of a customer to invest in security to reduce the likelihood of a successful cyber attack, whilst also acknowledging the inevitability of cyber damage. The paper employed stochastic optimization to account for the uncertainty of attacks to avoid both overprovisioning and underprovisioning of services. I significantly extend the work in that paper to consider multiple time periods and uncertain costs of security and insurance premiums. I also improve the scalability of the solution through a partial Lagrange multiplier method, perform an analytical sensitivity analysis, and base my result on parameters derived from real data.

1.3.1 Cyber Insurance

Cyber insurance is provided to offer risk transfer, and encourage risk management. Risk is the possibility of suffering harm or loss, which, in the cyber case, is due to the exploitation of a vulnerability by an attacker. Risk transfer moves the damages from a successful attack from the customer to the insurer, whilst risk management aims to control or minimize risk, for example, through improved security. Damages caused by cyber attacks may be 'first-party', i.e. when the compensation must be paid to the customer, or they may be 'third-party' where costs are incurred by others. An insurance policy may cover a mixture of first- and third-party costs. To obtain insurance cover, a client must pay an insurance premium, which provides a particular level of coverage for a specified period of time. The cost of the policy is based on risk

assessments carried out by the insurer and the anticipated cost and probability of a data breach. In the event of a successful cyber attack, damages are determined, and an indemnity payment is made to the affected parties according to the terms of the insurance policy. Cyber insurance has some unique features that make it particularly challenging, and some key issues are as follows:

- **First- vs Third-Party Coverage:** In a cyber attack scenario, there may be a wide range of stakeholders. An attacker is often not after data belonging to the attack subject, but rather the stored data of their users, such as credit card information. Thus the potential damages from an attack may be to both the policy holder and their own customers. An insurer must determine the extent to which they will cover damages in this case, as there may be millions of users to consider.
- **Primary vs Secondary Losses:** Primary losses are direct consequences of an attack. Primary losses are easier to establish and attach cost values to. Secondary losses may be equally costly, but are indirect consequences of an attack, such as, lost custom through to a decrease in user confidence, or a lowered stock value. These losses are harder to evaluate and may not appear immediately, yet for the customer they may be just as expensive as primary losses. Thus an insurer must decide the extent to which they will cover secondary losses.
- **Establishing Risk Levels and Expenses:** There is a lack of widely available data on cyber attacks, unlike other fields of insurance. Typically insurers set premiums according to historic data, but in the case of cyber insurance, companies are typically reluctant to share details on their security and vulnerability states. This information asymmetry, combined with interdependent systems, evolving attacks, and software homogeneity, makes establishing risk levels challenging. In this thesis I use attack data from Internet honeypots to counter this

problem. Similarly, the intangibility of cyber assets, and the existence of primary and secondary losses makes it challenging for insurers to accurately assign damages.

- **Incentive to Increase Security:** The key question surrounding the cyber insurance market is whether it is an incentive to increase cyber security provisions. If insurance does not lead to increased security provisions, there is a danger insurers will not enter the market due to the high level of risk to them, and the market will disappear. The joint allocation of security and cyber insurance in this thesis is designed to investigate whether a causal relationship can be established, and therefore if there is a future for the cyber insurance market.

Due to the current difficulty of accurately assessing cyber damages and those who are affected by a cyber attack, I assume that the cyber insurance in this thesis provides first-party coverage of primary losses, as these are the most common types of coverage, and experimental parameters may be most reasonably determined for the insurance. Risk levels in this thesis are established through an analysis of real data from Internet honeypots, to ensure that the data is impartial and realistic. As one of the key questions surrounding cyber insurance is whether it incentivizes improved security, Chapter 5 employs a joint optimization of the two resources to addresses this question.

1.4 Stochastic Optimization

This thesis aims to demonstrate the usefulness of optimization in mobile cloud computing. In this section I introduce stochastic optimization in detail for the benefit of readers unfamiliar with the technique. Mathematical optimization techniques focus on the maximization or minimization of a function subject to a set of constraints. This has many useful applications in computer science, particularly in the case of resource allocation problems, where the maximization of profit, or the minimization of cost, may be important. Traditional optimization problems are deterministic, assuming that all data is known when the solution is calculated. A typical linear optimization problem may take the form:

$$\min_{\vec{x}} \{ \vec{c}^\top \vec{x} : \mathbf{A} \vec{x} \geq \vec{b}, \vec{x} \geq 0 \}, \quad (1.1)$$

when written in vector notation. Where $\vec{x}$ denotes the vector of decision variables and $\vec{c}$ their attached cost values. In the case of (1.1), the goal is to minimize the total cost of the decision variables. This minimization is performed in relation to a set of constraints, denoted by the matrix $\mathbf{A}$ and the vector $\vec{b}$. When these values are known and fixed, it is straightforward to efficiently solve the optimization problem through a technique such as the simplex algorithm or the interior point method. However, the real world contains a great deal of uncertainty, and it is not reasonable to imagine that in a system as large and complex as the Cloud, that all information may be determined in advance. Therefore, in this thesis I employ stochastic optimization to solve three different problems in cloud computing, allowing for various types of uncertainty. Stochastic optimization may be formulated in a similar manner to traditional optimization, but instead of minimizing (or maximizing) a particular cost value, the expected cost over a range of possible scenarios is calculated. Thus the formulation in (1.1) may be rewritten:

$$\min_{\vec{x}} \{ \mathbb{E}_{\Omega} [\vec{c}^\top \vec{x}] : \mathbf{A} \vec{x} \geq \vec{b}, \vec{x} \geq 0 \}, \quad (1.2)$$

where Ω denotes the set of scenarios that may occur. Solution of a stochastic optimization problem in general is difficult, with heuristics necessary, such as stochastic approximation. However, if it is possible to find an accurate probability distribution to give probability values to the set of scenarios, and the problem has finite support (i.e. the system elements can be treated as discrete), then it is possible to find an exact optimal solution through the use of a deterministic equivalent formulation. The deterministic equivalent formulation is a linear program (LP) with each decision variable duplicated so that there is a version of the variables for each scenario in Ω . The cost vector $\vec{c}$ is also modified by multiplying each cost by the probability of that scenario occurring. The final result is therefore found by calculating the optimum cost in each scenario, multiplied by the probability of that scenario, for every scenario in

Ω . Thus the problem is still a deterministic problem and can be solved using any technique that is appropriate for a traditional optimization problem.

The above example is the simple, single case of stochastic optimization. Stochastic optimization is used in this thesis to solve multi-stage problems across multiple time periods. Multi-stage stochastic optimization allows decisions such as how much of a resource to reserve in advance, that fits the possible resource demand in multiple succeeding time periods. In a multi-stage stochastic optimization problem, the total cost is therefore the cost of provisioning variables in the present, added to the expected cost in the future. Solving this optimization allows a decision to be taken in the present that does not waste money by overprovisioning unneeded resources, nor by underprovisioning leading to inadequate resource availability. If I denote the present reservation variables by $\vec{y}$, and their costs by $\vec{c}'$, and the corresponding constraint matrix and limits by $\mathbf{A}'$ and $\vec{b}'$, respectively, I may write the following two-stage stochastic optimization problem:

$$\min_{\vec{x}, \vec{y}} \{ \vec{c}'^\top \vec{y} + \mathbb{E}_\Omega[\vec{c}^\top \vec{x}] : \mathbf{A}\vec{x} \geq \vec{b}, \mathbf{A}'\vec{y} \geq \vec{b}', \vec{x} \geq 0, \vec{y} \geq 0 \}. \quad (1.3)$$

This two-stage formulation may be easily extended to multiple stages through a nested formulation with each subsequent stage being calculated as its own value added to the expected cost of the following stage. However, many of the problems in this paper feature integer variables, which are NP-hard, and therefore solution algorithms have high computational complexity. This is a significant problem when operating at the large scale of cloud computing, and heuristics are often used to improve scalability (such as the scenario tree reduction employed in Chapter 3). The following two sections introduce an alternative method called the partial Lagrange multiplier method, that relies on the property of total unimodularity, to provide both optimality and scalability.

1.5 Total Unimodularity

To counter the problem of integer or mixed integer problems being NP-hard, I take advantage of a property called total unimodularity (TU) [28]. TU is a property of

the constraint matrix (previously denoted by $\mathbf{A}$) that allows the integer restriction on decision variables to be relaxed to linear ones, while still giving integer decision variable values as the solution. Total unimodularity is defined as follows:

- A submatrix is defined as a square matrix that can be found by deleting some rows and columns from the original matrix.
- The original matrix is defined as totally unimodular if (and only if) the determinants of all its submatrices are equal to either -1 , 0 , or 1 .

To calculate the determinant of every submatrix is time-consuming and therefore counter-productive. However, there are alternative methods to determine total unimodularity. For example, a matrix satisfies total unimodularity if each column contains no more than two non-zero values, which must be 1 or -1 , and the sum of all columns with two non-zero values is 0 . [29] provides a number of criteria for proving the unimodularity of a matrix, which can be shown to be ‘total’ if it holds true for every square non-singular submatrix. This can be more efficient than calculating the determinant as many of the criteria can be established through simple inspection of the matrix rather than through calculation. In Chapter 4 I use Theorem 5 from [29] to prove that total unimodularity holds for part of the constraint matrix, likewise in Chapter 5 I use Theorem 3. Proving that total unimodularity holds for part of the matrix is essential for the partial Lagrange multiplier method, which I briefly introduce in the next section.

1.6 Partial Lagrange Multiplier Method

Solving a mixed integer problem involves using a method such as the branch-and-bound algorithm, which has exponential computational complexity [30]. However, if a problem satisfies the total unimodularity property and the integer variables can be relaxed to linear ones, which allows the problem to be solved in polynomial time, using a method such as the Interior Point Method [30]. In reality, there are only a limited number of problems that will fit this property, so [31] introduces the partial Lagrange

multiplier method as an alternative algorithm that can offer polynomial complexity where the constraint matrix partially satisfies TU.

1.6.1 Partial Total Unimodularity of Constraint Matrix

Consider the constraint matrix $\mathbf{A}$ and vector $\vec{\mathbf{b}}$, as in standard LP notation. If $\vec{\mathbf{b}}$ is all-integer, and $\mathbf{A}$ satisfies TU, I can solve as an LP. However, if $\mathbf{A}$ does not satisfy total unimodularity, it may be possible to partition the matrix into two matrices, one that satisfies total unimodularity, and one that does not. The TU-satisfying matrix, denoted by $\bar{\mathbf{A}}$, is kept as the constraint matrix, while the other, denoted by $\hat{\mathbf{A}}$, is moved into the objective function with a Lagrange multiplier, as in the Lagrange multiplier method.

1.6.2 Partial Lagrange Multiplier

The full Lagrange multiplier method is an established technique for solving convex optimization problems using linear constraints. In the vector notation of standard form, the problems are written as follows:

$$\min_{\vec{\mathbf{x}}} f_0(\vec{\mathbf{x}}) \quad \left| \begin{array}{l} f_i(\vec{\mathbf{x}}) \leq 0, \quad i = 1, \dots, m; \\ h_i(\vec{\mathbf{x}}) = 0, \quad i = 1, \dots, p, \end{array} \right. \quad (1.4)$$

where $\vec{\mathbf{x}} \in \mathbb{R}^n$. The constraints in (5.36) can be moved into the objective function weighted by Lagrange multipliers, as in the following equation:

$$L(\vec{\mathbf{x}}, \vec{\boldsymbol{\lambda}}, \vec{\mathbf{v}}) = f_0(\vec{\mathbf{x}}) + \sum_{i=1}^m \lambda_i f_i(\vec{\mathbf{x}}) + \sum_{i=1}^p v_i h_i(\vec{\mathbf{x}}), \quad (1.5)$$

where $\vec{\boldsymbol{\lambda}} \in \mathbb{R}^m$ and $\vec{\mathbf{v}} \in \mathbb{R}^p$ are the Lagrange multipliers. The dual function, expressed in $\vec{\boldsymbol{\lambda}}$ and $\vec{\mathbf{v}}$ is therefore:

$$g(\vec{\boldsymbol{\lambda}}, \vec{\mathbf{v}}) = \min_{\vec{\mathbf{x}} \in \mathbb{R}^n} L(\vec{\mathbf{x}}, \vec{\boldsymbol{\lambda}}, \vec{\mathbf{v}}), \quad (1.6)$$

and the corresponding dual problem can be written:

$$\max_{\vec{\boldsymbol{\lambda}}, \vec{\mathbf{v}}} g(\vec{\boldsymbol{\lambda}}, \vec{\mathbf{v}}) \quad \left| \quad \vec{\boldsymbol{\lambda}} \geq 0. \quad (1.7)$$

When used for convex optimization problems, the dual problem can be solved for $\vec{\lambda}$ and $\vec{v}$ via the subgradient method. The corresponding values found by the dual problem for $\vec{x}$ are also the optimal values for the primal problem, due to the convexity of the primal problem. The partial Lagrange multiplier method for linear programs adopts a similar strategy, with the constraints that do not satisfy TU being moved into the objective function with the set of $\vec{\lambda}$ Lagrange multipliers. The remaining, TU-satisfying, constraints remain as constraints in the problem, allowing it to be solved as an LP. Therefore the dual problem LP to be solved in a single iteration of the partial Lagrange multiplier algorithm is written as follows:

$$g(\vec{\lambda}) = \min_{\vec{x} \in \mathbb{R}^n} \{ \vec{c}^\top \vec{x} + \vec{\lambda}^\top (\hat{A}\vec{x} - \hat{\vec{b}}) : \bar{A}\vec{x} \leq \bar{\vec{b}}, \vec{x} \geq 0 \} \quad (1.8)$$

1.6.3 Solution to the Partial Lagrange Multiplier

To solve my optimization problem using the partial Lagrange multiplier method, I iteratively minimize the dual problem in (1.8) in terms of $\vec{\lambda}$.

1.6.3.1 Dual Problem Solution

In each iteration of the algorithm, I solve the dual problem, and use the new values of the primal variables to determine the value of the original primal objective function. This value is then used to calculate the Lagrange multipliers for the next iteration of the algorithm.

1.6.3.2 Updating Lagrange Multipliers

The behaviour of the algorithm is governed by the updated values of the Lagrange multipliers. This calculation is governed by the stepsize α_k for a given step k . The Lagrange multipliers are updated by adding the old value to the gradient of the dual function weighted by the step size, or 0, whichever is larger:

$$\vec{\lambda}^{(k+1)} = \vec{\lambda}^{(k)} + \alpha_k \cdot \nabla g(\vec{\lambda}^{(k)}). \quad (1.9)$$

The stepsize, α_k can be calculated in a variety of ways, for example, as a constant value, $1/k$, or $1/\sqrt{k}$. The ideal value will depend on the problem being solved, with a calculation dependent on k giving a decreasing stepsize as k increases, providing greater precision. A constant value may converge more rapidly, but may also result in oscillation between two fixed values, of which one is the optimum. The function $\nabla g(\vec{\lambda}^{(k)})$ is the derivative of the dual function with respect to $\vec{\lambda}$.

1.6.3.3 Stopping Criterion

The partial Lagrange multiplier method will always converge to the optimal solution, as I later prove for each of the problem formulations that use it. To determine when the solution has been reached, I use the stopping criterion. If the best value for the primal objective does not improve by more than the marginal value ϵ and Φ iterations have passed since the best value was found, then the algorithm terminates. If ϵ is sufficiently small and Φ is sufficiently large, the best value on termination would be guaranteed to be the optimum. However, the user may choose values for ϵ and Φ that suit their desired tradeoff between accuracy and runtime. The pseudocode for the partial Lagrange multiplier algorithm outlined in this section, with the described stopping criterion, is given in Algorithm 1.

Algorithm 1 Partial Lagrange Multiplier Method

Input : Dual problem $g(\vec{\lambda})$, Φ
Output: Optimal solution for $\vec{x}$
Initialize $g_{best}^{(0)} = -\infty$;
Set the iteration index: $k = 1, k_x = 0$;
Initialize $\vec{\lambda}$ values ≥ 0 ;
do
 Find optimal solution for $g(\vec{\lambda})$, given in Eq. (1.8) ;
 Compute solution of primal problem ;
 if $g_{best}^{(k)} - g_{best}^{(k-1)} \geq \epsilon$ **then** $k_x = k$;
 $k = k + 1$;
 Compute step size α_k ;
 Update Lagrange multipliers using Eq. (1.9);
while $k - k_x \leq \Phi$;

1.7 Key Contributions

This thesis aims to investigate the usefulness of stochastic optimization by applying it to three problem areas that each address key issues in mobile cloud computing. The first problem area optimizes cloud resource provisioning, to guarantee performance, the second optimizes mobile application migration to preserve battery life, and the third optimizes cyber security and insurance to improve security. Performance guarantees, battery life, and security are three of the most important issues in mobile cloud computing, and stochastic optimization can contribute to solving all of these issues. Figure 1.3 lays out the techniques applied in each chapter, as each problem area is formulated as a stochastic optimization problem, with scalability and sensitivity analysis methods applied. In Chapter 3 the scalability improvements are provided by a scenario tree reduction algorithm, which provides a good heuristic. This method is not used in Chapter 4 because the number of individual users in the system is the primary influence on problem size, rather than the size of the scenario tree. Therefore, the partial Lagrange multiplier method was found to replace it. Due to its effectiveness, I reused the method for Chapter 5, as the presence of binary variables makes it particularly appropriate. For reference and clarity, the key contributions of this thesis to each issue are outlined here according to chapter.

1.7.1 Chapter 3

In this chapter I take advantage of SDN and VM technology to present a joint optimization of VM and bandwidth allocation in a cloud computing environment that handles both demand and price uncertainty. My contributions can be summarized as follows:

- I devise a stochastic optimization to jointly reserve VMs and bandwidth across multiple time stages in a multi-user, multi-provider cloud environment, with both VM and bandwidth demand and price uncertainty. I demonstrate its optimality by formulating a deterministic equivalent problem and testing it on

real historical demand data. My formulation is able to obtain the optimal solution even when the probability distribution changes.

- The stochastic optimization is made more scalable by the use of scenario tree reduction techniques. This reduces the size of the problem space significantly whilst maintaining a close semantic match to the full problem, and yields solutions that are close to the optimum.
- I perform a sensitivity analysis, analytically determining the tolerance of the optimal solution to parameter changes. I reveal that it is important to consider routing of bandwidth, as changes in cost can result in rerouted traffic. I find sensitivity analysis is useful not only for users analyzing their decisions, but also for providers in setting optimal prices.

1.7.2 Chapter 4

In this chapter I build on existing migration technology and make the following contributions:

- I devise a stochastic binary optimization to optimally allocate users to equipment across multiple time periods considering uncertainty factors such as user numbers, battery levels, app requirements, and usage lengths. I demonstrate the optimality of this formulation through a series of numerical results tested on real user trace data.
- I exploit the property of total unimodularity to introduce a novel partial Lagrange multiplier algorithm to solve the optimization in polynomial time, countering the problem of exponential complexity of integer programming problems. I prove that the algorithm converges to the optimal solution and does so in polynomial time.
- I derive an analytical approach to sensitivity analysis by formulating two linear programs that can be solved to find exact values for variable sensitivity to parameter changes. This gives the ability to examine which parameters have

the largest impact on user allocation, for example how high can costs be raised without user allocation changing?

1.7.3 Chapter 5

In this chapter I make the following contributions to combined security and cyber insurance provisioning in the context of cloud computing:

- I devise a stochastic optimization for a customer to jointly provision security services and buy cyber insurance premiums across multiple time periods. I account for uncertainty in traffic quantities and attack frequency, as well as future uncertainty of security service prices and insurance premiums.
- I again use the partial Lagrange multiplier algorithm to find the optimal solution to the integer problem in, at worst, polynomial time. I provide proofs of convergence and scalability.
- I perform a sensitivity analysis, which provides precise values for solution tolerance to parameter change. I then demonstrate the effectiveness of my methods through evaluation of an example scenario, based on analysis of real attack data to provide realistic parameter settings.

Figure 1.3: Overview of the techniques used in each chapter. Each chapter addresses a resource allocation problem using stochastic optimization, followed by a runtime improvement technique and a sensitivity analysis.

Chapter 2

Literature Review

The work in this thesis stands on the shoulders of extensive research in the field of mobile cloud computing. In this chapter, I review the state of research in the relevant fields of mobile cloud computing, identifying areas that require further work, and demonstrating that the contributions of this thesis are a reasonable step forward, premised on ideas and technology that have a legitimate technical and intellectual basis.

2.1 Virtual machine and bandwidth allocation in cloud computing environments

Resource provisioning in cloud computing can be broadly divided into two categories: virtual machine allocation, and bandwidth allocation. There has been extensive work in this area, each of which approaches the problem from a different angle. A summary of the works outlined here may be found in Table 2.1.

2.1.1 Virtual Machine Allocation

VM allocation approaches typically focus on either offline allocation or online migration. [32] and [33] decouple the mapping of VMs to physical machines from the provisioning of computation by a cloud application. Each cloud application environment may run across multiple virtual machines, with VMs positioned across physical servers. It is assumed that the number of physical servers is fixed, but that VMs can be migrated live between them. Applications request resources at VM-level granularity to maintain their SLAs, while a manager allocates those VMs to physical servers,

running a constraint optimization to optimize SLA-compliance and running costs by maximizing a global utility function. [34] takes an alternative approach to the problem of meeting SLAs, by aiming to minimize violations through server consolidation. In this case, the objective is to minimize running costs through the live migration of VMs to maximize the utilization of physical servers. Under-utilization and overloading are both penalized using penalty terms. The algorithm uses historic workload data to identify usage patterns and place VMs accordingly, but also adapts dynamically by identifying servers that would benefit from consolidation through workload signatures. In [35], EnaCloud is an approach that encapsulates applications within a VM to simplify the placement of applications without the need for an intermediary managing multiple VMs. A heuristic algorithm is used to avoid the high complexity of calculating an optimal solution so that migration can be dynamic. The allocation is treated as a bin-packing problem that includes energy-awareness, it tolerates a degree of overprovisioning as it is deemed preferable to an increased frequency of migration and resizing.

These approaches take a live and dynamic approach to VM placement. However, these papers do not consider the bandwidth requirements of migration. [36] performs live migration but removes the simplifying assumption that VM migration can be seamless without accounting for the bandwidth consumption of the migration. The size of a VM may be large, possibly encapsulating an entire OS. VM migration ideally takes place in three phases, the first ‘push’ phase copies the VM to the new location, the second ‘stop-and-copy’ phase stops the service at the old location, activates at the new location, and copies any dirty memory pages remaining. The final ‘pull’ phase copies any missing request pages from the old location. In this work, the authors adopt a ‘pre-copy’ approach that uses a combination of the first two phases. This approach iteratively copies memory pages to minimize dirty pages while managing the tradeoff between speed of copying and maintaining QoS by avoiding excessive bandwidth usage. This tradeoff is managed by an M/M/1 queue model to account for varying numbers of service requests, and assumes the customer only receives service from one server at a time. [37] also considers the cost of migration but includes it in a

wider scope optimization problem that looks at the problem of VM consolidation as discussed above. The optimization maximizes profit while minimizing cost from power consumption and SLA violations. The number of physical and virtual machines are allocated to cloud services, and the CPU share allocated to each VM is also optimized. A limited lookahead control (LLC) is used to avoid excessive switching in the face of varying workloads, similar to the idea used in [35]. [38] also considers the bandwidth requirements of VMs by considering their data rate requests, to minimize traffic delay. Bandwidth is considered simplistically, as single links between computing centers and their connecting WAN, with bandwidth satisfaction done by considering the total bandwidth requirements of VMs placed at a computing centre, and the load it places on the communication link. Bandwidth routing within the WAN and within the computing centers are not considered in the paper.

However, whilst these solutions can dynamically manage VM allocation, they do not address the bandwidth requirements that are increasingly part of VM applications. [39] addresses the problem of placing VMs with high bandwidth requirements. Bandwidth allocation may be much more complex than VM placement, as communication links may be shared by multiple physical machines with multiple source and destination points. This work simplifies the complexity of bandwidth allocation by analyzing bandwidth usage in relation to a single terminus called the ‘root’. The optimization problem considers bandwidth as a constraint to the VM placement optimization, maximizing the performance of VM placement, with the ability to route traffic to the root used as a cost value. Since the formulated problem is NP-hard, the paper adopts a polynomial-time constant approximation algorithm, that can be made more efficient based on the assumption that the network is a tree structure. [12] also aims to sidestep the challenge of network routing by placing VMs with a high degree of inter-communication in close proximity, limiting congestion and improving scalability. Thus, rather than requiring data centers to modify their network or routing protocols, and rather than handle the complexity of bandwidth allocation, the paper focuses on VM placement. Bandwidth management is therefore left to the existing network management protocols, under the assumption that the VM placement

has ensured that the communication requirements can be met. Allocation is done by solving an optimization problem by minimizing the bandwidth cost of allocation, for example, by minimizing the number of total network switches used, as defined by the established routing paths between two hosts. This consolidation can also be employed at the inter-data center level.

[11] aims to optimize VM placement at both intra- and inter-data center levels. Bandwidth is accounted for by placing VMs first at data centers to limit the communication cost and latency between data centers, whilst intra-data center placement is similar to [12]. It is assumed that users provide exact VM requirements, and the placement of VMs with data centers in the distributed cloud is determined by minimizing the maximum distance or latency between the data centers. This approach does not directly consider the routing between data centers, but uses latency as the bandwidth allocation metric. Having allocated VMs at the inter-data center level, the exact placement of a VM on a physical machine is optimized, by choosing the exact rack and server on which to run the VM. A tree architecture is assumed so as to allow for a more efficient solution, but means that a change in architecture may not offer the same efficiency. However, this is not unreasonable, as data center networks are more regularly structured than the Internet, which allows for greater efficiencies when allocating bandwidth. However, the technology used to guarantee data center bandwidth can also be employed in WANs. [40] uses a mixed integer optimization problem to allocate VMs on an inter-data center basis, factoring in the bandwidth requirements of VMs and aiming to maximize bandwidth availability and minimize latency. Bandwidth routing is not considered but the bandwidth requirements and latency is considered for links between data centers. This method is similar to the already established techniques but is applied to the inter-data center level, which is not as commonly addressed. Additionally, resource requirements are considered according to CPU and memory, rather than a simplistic ‘number of VMs’. [41] also considers VM placement in a bandwidth-aware manner, but extends to consider live migration and inter-VM communication. VMs are not considered individually, but

are provisioned in groups, clustered together and then placed with physical machines according to a bucket-code learning algorithm.

The works outlined here show that virtual machine allocation is well-established as an area of study. The key ingredient missing, however, is an adequate consideration of the bandwidth needs of modern cloud applications. Whilst some research attempts to simplistically address bandwidth allocation within a data center context, there is limited work on external bandwidth allocation. Given the bandwidth-intensive nature of many cloud applications, this is a significant omission that should be addressed. This thesis attempts to show the importance of combined virtual machine allocation and bandwidth routing. To demonstrate the viability of the system model that will be proposed in Chapter 3, I now provide an overview of the current state of research on cloud bandwidth allocation, showing that there is a firm foundation on which to base this work.

2.1.2 Bandwidth Allocation

Bandwidth guarantees throughout the network are increasingly important in cloud applications. In SecondNet [42], the authors introduce a virtual data center (VDC), or virtual cluster, that can provide bandwidth guarantees combined with VM allocation. Instead of provisioning resources as VMs, VDCs are the unit of resource allocation, which encapsulates VMs and bandwidth. To handle the resources allocated to a VDC, the authors propose SecondNet, a network virtualization architecture. SecondNet virtualizes network bandwidth and computation together, and is not dependent on the architecture of the network. [43] extends this idea of abstracting resource allocation for both bandwidth and VMs to heterogeneous bandwidth demands. Thus, a tenant in a multi-tenant data center may specify a range of bandwidth requirements for a set of VMs. A heuristic approach is used to evaluate the acceptability of a VM allocation as the problem is proved to be NP-complete. However, despite allowing for heterogeneous bandwidth demand, this problem still requires a known set of bandwidth requests provided by tenants. [44] therefore aims to satisfy bandwidth requirements between multiple users on a data center network within a certain probability, allowing

for a stochastic approach where bandwidth requirements are not known in advance. Taking advantage of the notion of virtualized clusters the paper uses ‘stochastic virtual clusters’, which allow the sharing of network bandwidth between multiple tenants, when this demand is variable over time. The use of a virtual cluster allows virtualized resources to be redistributed in a way that is transparent to the user. Allocation of VMs to physical machines is established by allocating bandwidth to links, and giving a probabilistic assessment of the acceptability of a VM placement if the network links can satisfy the required bandwidth. [45] formulates an MILP to jointly place VMs and route bandwidth, with power efficiency as the main priority. This routing of bandwidth is not extensively covered in the works outlined thus far, and is usually left to the network’s normal protocols. However, the MILP in the paper addresses bandwidth routing directly and is a form of multi-commodity network flow problem. Whilst bandwidth demand uncertainty is not considered, and servers are assumed to be homogeneous, this approach is time-aware, using a heuristic to more quickly solve the NP-complete optimization problem, allowing quicker reaction to varying demand. [46] aims to take the joint approach to a large scale, performing joint request mapping and response routing for a geo-distributed cloud service. The varying costs of allocation in different regions are considered, with a distributed algorithm based on the alternating direction method of multipliers (ADMM) used to solve the allocation problem. The algorithm achieves near-optimal performance using this approach, which is different to other works, which tend to adopt a centralized approach.

These works largely focus on the application of bandwidth allocation within data centers, where the network layout may be carefully designed and is typically more conventionally structured, with more predictable demand, than a WAN. The current research shows that a multi-user scheme of bandwidth allocation is possible through the use of network virtualization. I rely on this idea to provide the technical foundation of the work in this thesis, allowing multiple clients to request specific bandwidth guarantees on the same network. However, to achieve the scalability and flexibility required by bandwidth allocation at the Internet level, I require a newer technology to go beyond the intra-data center limitations of the current work.

2.1.3 Software Defined Networking

Bandwidth allocation and reservation is important, but unlike virtualized computation, traditional virtualized networking technologies (such as virtual networks) are not adequate for the scale of bandwidth allocation required by cloud computing. To address this problem, Software Defined Networks (SDN) [6] have been introduced to offer flexible, custom virtualized networks that do not require extensive hardware replacement and reconfiguration. SDN decouples network control from data routing, and allows network providers to guarantee custom bandwidth provisions. Data routing is handled by the network owner, who can offer virtual networks that individual tenants can control, without revealing, or giving access to, the underlying network structure. To do this, SDN follows a three-layer structure, the infrastructure layer, control layer, and application layer. SDN is a network architecture with various implementations that incorporate its features, with OpenFlow being the most prominent [9].

2.1.3.1 Infrastructure Layer

The infrastructure layer consists of the physical devices that makes up the network and is managed by the network owner. Research in this area focuses on the performance of physical devices and the correctness of traffic forwarding. For example, [47] addresses the implementation of the OpenFlow SDN design on the NetFPGA platform, showing that OpenFlow can be implemented to run full line-rate switching. The purpose of Openflow is to allow the implementation of SDN on existing networking devices, such as an Ethernet switch or IPv4 router implementation. Since most OpenFlow complexity is centralized in a controller, which manages the forwarding tables of OpenFlow routers, there is little modification required at the infrastructure layer. To address performance bottlenecks, [48] connects a more powerful CPU to commodity switches to increase the processing of packets, with larger forwarding tables and packet buffers, with the additional aim of achieving more programmable switches.

2.1.3.2 Control Layer

The SDN controller is the key to networks running SDN and is responsible for the performance, scalability, and reliability of the network. In a large network, either large

scale LANs or WANs such as the Internet, the control is partitioned into multiple areas, with each area, or domain, managed by a separate SDN controller. This control may be horizontally or vertically partitioned, with either ‘flat’ or ‘hierarchical’ SDN control. In the horizontal case, the controllers each manage a subset of the network switches, where the switches are topologically (if not geographically) local, and each subset may be divided due to different ownership of physical networks. SDN permits this flexibility, as different network owners manage the infrastructure layer, and communicate at the control layer, which is transparent to the end user, who could end up running their network applications over networks managed by multiple owners. This approach is particularly likely to occur if implementing SDN across a WAN, such as the Internet, where local Internet connections are owned by local ISPs. Such a scenario might arise if a VoD service wished to guarantee end-to-end bandwidth for users. In contrast, the vertical approach organises the controllers hierarchically with different tasks and domains assigned to specific controllers, with a root controller overseeing the whole network. This is more practical where the entire network is managed by a single controller, such as in a data center [49].

2.1.3.3 Application Layer

Whilst the control layer provides the interface that enables SDN to function, it is the application layer that contains the greatest interest for the client and its usefulness as applied in this thesis. Through the northbound interface of the control layer, applications may access a global view of the network, and implement a variety of applications, which I give some examples of here as applicable to the areas covered in this thesis. SDN enables adaptive routing, such as load balancing and cross-layer design. In [50] traffic load is balanced across the network with differentiation according to traffic type. Controllers slice the network into sections and handle the balancing of individual traffic types to optimize performance. [51] uses SDN to enable cross-layer sharing of bandwidth availability data in a video conferencing context, allowing minimum bandwidth guarantees to be reserved by users, along with path control and imposing rate-limits. In addition to adaptive routing, SDN also allows for boundless

roaming across networks. This is particularly relevant for the type of user considered in Chapter 4, who need to maintain a consistent Internet connection while offloading mobile computation to the cloud but who may transition between WiFi and cellular networks. In [52] the OpenRoads platform is developed to support migration between WiFi and WiMAX networks. Using SDN to control both network types avoids the static nature of an IP-based network by providing a common interface to the application. SDN also permits improved network security. Security services in the cloud, such as those considered in Chapter 5, can be implemented because SDN allows greater control of security policies and services despite the heterogeneity of network hardware. SDN also provides greater potential for collecting data on attacks and routing data through an Intrusion Prevention System (IPS), which can be offered as a cloud service. For example, in [53] dynamic flow routing can be used to redirect traffic to a detection system. As discussed, the chief benefit of SDN for the work in Chapter 3 is the increased power for network virtualization. Traditional methods like VLAN require configuration of all network devices, whilst a technology like FlowVisor [54] sits between guest controllers and switching devices to provide users with access to a virtual network they can control, sharing the physical network with other users whilst remaining isolated. The drawback with SDN in the context of this thesis, is that it typically requires the network in question to be controlled by a single manager, and thus VM migration and bandwidth provisioning can be implemented easily at the intra-data center level, but becomes more difficult across the Internet as there may be multiple ISPs and network owners involved. To apply SDN to cloud computing explicitly, [55] implements CrossRoads with SDN controllers in each data center to migrate VMs between data centers, allowing for maintenance of IP addresses of VPNs, firewall access, and maintaining communication with external clients.

An architecture like SDN is alluring, as it offers extensive virtualization possibilities without requiring significant investment in new hardware. This is important, as an ISP would be reluctant to consider retooling their entire network to support bandwidth reservation. SDN allows bandwidth provisioning to be easily implemented and customised, so that users may allocate VMs and obtain end-to-end bandwidth

performance guarantees. Considering joint allocation in detail at the inter-cloud level is novel, but there remains one significant consideration that must first be addressed.

2.1.4 Uncertainty in Allocation

The works outlined thus far consider problems with predictable demand, but in reality, demand is uncertain. [56] co-allocates VMs and bandwidth for data streaming applications, as it is established that overprovisioning of CPU and bandwidth does not guarantee optimal performance. To cope with uncertainty, allocation is dynamic, using fuzzy logic to allocate VMs with bandwidth allocated accordingly by iteratively observing processing and storage status. Thus uncertainty is handled by remaining dynamically flexible to changing demand. In contrast, [57] takes a two-phase planning approach to anticipate demand uncertainty through resource reservation. Cloud-based VM services can be provisioned using either long-term reservation or short-term, on-demand, subscriptions. In the first phase, past usage data is used to estimate future reservation, by calculating the upper and lower bounds of resources to be obtained under long-term subscriptions. The second phase provisions VMs in the short term through utilization of reserved resources and on-demand allocation in a dynamic fashion using linear programming and the Kalman filter. [10] extends this two-stage optimization across multiple time periods, allowing variable length reservation contracts to be used. Instead of a two-stage optimization, the paper allows for multiple time periods each containing two provisioning stages, with reservations made in one time period being available for utilization in future time periods. Further, this allocation is performed across multiple cloud providers, considering the costs of geographic locations, although inter-communication and bandwidth requirements are not considered beyond a simple capacity constraint. The paper employs stochastic optimization to find a precise expected value based on a probability distribution determined from test data, that considers both price and demand uncertainty. [58] addresses the problem of consolidation of VMs on physical devices at the intra-data center level. Since VMs do not use a constant amount of bandwidth, it is modeled as a random variable, with probabilities attached to the level of bandwidth consumption

required by a VM at a given point in time. The problem of allocating VMs is then modeled as a stochastic bin-packing problem, which is an extension of previous works on consolidation that also used a bin-packing approach, but without the element of uncertainty. The bandwidth capacity of the rack and its top-of-rack switch capacity is the primary constraint on placement of VMs. [59] approaches uncertainty in bandwidth allocation for data centers that supports users requesting connections that guarantee both bandwidth and time to transfer data of a specific volume. Pricing is variable, offering users discounts for flexibility in SLA requirements. The proposed solution is two-stage that dynamically allocates minimal bandwidth in the first stage to arriving user requests, with further bandwidth allocation performed in the second stage to maximize revenue. This approach improves on deterministic methods by satisfying more requests and generating higher revenue, demonstrating the necessity of an approach that considers uncertainty in a real-world setting. The majority of work involving bandwidth allocation focuses on intra-data center allocation, however with the popularity of video-on-demand services, this problem needs addressing at the Internet level as well. [60] proposes an approach using a bandwidth broker that allows service providers to reserve bandwidth to guarantee a minimum level of performance for media streaming. In addition to optimizing bandwidth allocation, the authors also propose a bandwidth pricing scheme which is used to force the broker to maximize profits and optimize workload consolidation such that these two goals are identical in a scenario where both users and the bandwidth broker are selfish.

The existing work addressed in this section shows that there is a sound technical basis for VM and bandwidth allocation in the cloud, but there is not yet research that satisfactorily ties these elements together in a realistic way. Thus, in this thesis I apply stochastic optimization to provision these resources with optimal VM and bandwidth placement while considering uncertainty. Given the high bandwidth nature of cloud-based services, and the variable demand from users with limited patience, it is important to guarantee end-to-end performance, and the system model proposed here aims to provide this.

Table 2.1: Summary of works addressing problems in virtual machine and bandwidth allocation

P	Article	Objective	Solution
Virtual Machine Allocation	[32] [33]	Separate problem of VM placement from computation resource requests	Run applications across multiple VMs, support live migration
	[34]	Guarantee SLAs through VM consolidation	Live migration of VMs using historic workload data for initial assignment
	[35]	Simplify VM placement and remove need for intermediary	Encapsulate applications within VM, use heuristic method
	[36] [37]	Account for performance overhead of live migration	Multi-phase migration with M/M/1 queue model
	[38]	Account for performance overhead of live migration	Consider bandwidth requirements of VMs for communications and network overhead of migration
	[39]	Account for bandwidth as a resource requirement in VM placement	Consider bandwidth in relation to a single 'root' terminus
	[12]	Account for bandwidth as a resource requirement in VM placement	Place VMs with high degree of inter-communication in clusters in close proximity
	[11] [40]	Place VMs at the inter-cloud level	Distribute VMs to minimize distance and latency between data centers
	[41]	Place VMs at the inter-cloud level	Consider VMs as clusters of close inter-communication, not as individual VMs
Bandwidth Allocation	[42]	Provide bandwidth guarantees	Implement Virtual Data Center to encapsulate VM and bandwidth provisioning
	[43]	Provide bandwidth guarantees in multi-tenant data centers	Allow tenants to specify bandwidth requirements
	[44]	Provide bandwidth guarantees in multi-tenant data centers	Satisfy requirements within a certain probability, not pre-specified demand
	[45]	Jointly place VMs and route bandwidth	Optimize power consumption, assume bandwidth demand is known in advance
	[46]	Jointly place VMs and route bandwidth	Use distributed algorithm similar to ADMM

Software Defined Networking	[6] [9]	Provide working definition and implementation of SDN	Introduce OpenFlow to implement SDN without significant hardware replacement
	[47]	Implement SDN at the infrastructure layer	Implement OpenFlow on the NetFPGA platform
	[48]	Remove bottlenecks at the infrastructure layer	Attach powerful CPUs to commodity switches to process packets faster
	[49]	Implement SDN at the control layer	Partition large SDN networks into controller domains so each controller manages a limited space
	[50]	Implement load balancing at the application layer	Differentiate traffic by type and assign separate SDN controllers to each
	[51]	Implement cross-layer design at the application layer	Allow communication of bandwidth capacities to guarantee VOD performance
	[52]	Implement boundless roaming at the application layer	Create OpenRoads platform to investigate migration between WiFi and WiMAX networks
	[53]	Implement improved security at the application layer	Use SDN to redirect traffic through Intrusion Prevention System
Uncertainty in Allocation	[56]	Co-allocate VMs and bandwidth under uncertainty	Dynamic allocation using fuzzy logic
	[57] [10]	Co-allocate VMs and bandwidth under uncertainty	Use multi-phase allocation scheme with on-demand allocation to respond to uncertainty
	[58]	Consolidate VMs under bandwidth uncertainty	Model bandwidth requirements as random variable. Use stochastic bin-packing
	[59]	Guarantee bandwidth and transfer time in data centers	Offer discounts for SLA flexibility. Two-stage bandwidth allocation
	[60]	Guarantee bandwidth allocation at the inter-data center level	Use broker to guarantee performance and consolidate bandwidth

	[54]	Implement virtualization at the application layer	Design FlowVisor to sit between guest controllers and switching devices to allow multi-tenant networks
	[55]	Implement SDN at the inter-data center level	Implement CrossRoads, a design with SDN controllers in each data center to allow VM migration between the two seamlessly

2.2 Bring-Your-Own-Device and Mobile Device Migration

The problem of virtual machine and bandwidth allocation is applicable to general cloud computing, but may not always be practical for mobile users. If a user is constrained by their battery level, offloading computation to the cloud may not solve the problem, as they still need to interface through their mobile device. The Bring-Your-Own-Application model proposed in Chapter 4 relies on the extensive work done in the field of mobile application migration to enable the offloading of applications to local equipment at a variety of public locations. It is inspired by the model of Bring-Your-Own-Device computing, which has encouraged work in mobile device management and security which provide a viable technical basis for the novel system model proposed in this thesis. These contributions, along with the current state of the art in the field of mobile application offloading, are covered in this section, and a summary of the works reviewed can be found in Table 2.2.

2.2.1 Mobile Application Migration

Mobile cloud computing is increasingly popular, but the limitations of processing power, screen size, and battery life are widely addressed in the literature. Thus the notion of mobile application offloading has been researched, in various forms, for several years, as application migration requires portable applications and well-designed infrastructure. In [61], the authors take inspiration from the abstract concept of objects in object oriented programming to design an abstract syntax model to

specify languages for cloud applications. By designing the application at a sufficient degree of abstraction, an application may be migrated more easily between local and cloud-based computation without significant modification, as it is the implementation of the language on each platform that differs.

Employing a device-independent middleware platform like J2EE allows the seamless migration of an application between two mobile devices. In [62], the authors imagine a scenario where mobile application use can be migrated seamlessly between two mobile devices through a simple swiping action, where the application state is paused on the first device, and transferred via middleware on each device, and resumed on the second device. This is similar to the methods described above for migrating VMs in VM consolidation, except at the application level rather than a complete virtual machine. Using a similar technique, much of the recent work focuses on the idea of utilizing the cloud to relieve the demands on mobile devices. Due to the heterogeneity of hardware, migrating at the application level may be desirable, and applications need to be adapted to fit different environments whilst preserving the functionality and application state as far as possible. [63] introduces an application migration framework that partitions applications into input, output, and logic components, adapting each according to the new devices that are available.

Traditional mobile application migration techniques may incur significant overheads and there is a limitation on the parts of a program's code that can be migrated. [64] partitions Java applications to minimize overhead, as the use of a platform-independent language like Java means migration can take place at almost any point in the code. Rather than migrate complete processes or threads, a 'stack-on-demand' approach is used, which only migrates the topmost part of the Java stack, allowing computation to be offloaded whilst only using limited bandwidth to perform the migration. [65] also uses J2EE to perform live migration of applications, but uses a monitor on each device and central controller to choose the optimal node to migrate to and migrates at the Java Bean level to avoid service interruption. [66] focuses on minimizing the amount of data required to be transferred by caching data at the cloud side using a variant of the Least Recently Used caching strategy to identify the most commonly

migrated application components. Many of these applications rely on Java to offer seamless transitions via middleware for application migration. However, in addition to Java-based partitioning, many modern applications use web technologies, and HTML 5's web workers allow efficient and selective offloading [67]. Web workers are a new addition to the HTML 5 technology that execute distinct segments of Javascript code that can be run in a separate thread, allowing multi-threaded Javascript programs where tasks can be performed in the background of a web application in a way that was difficult to achieve before. This ability to partition a web application into threads allows offloading to the cloud, where a call to a worker may be executed locally or remotely. However, one of the dangers of application migration is that the user is no longer in control of all parts of the application and therefore lacks security relative to migration of an entire VM, which can be run in an isolated sandbox. [68] introduces a security framework to protect information security during application partitioning, by setting up virtual environments in the cloud that emulate an Android phone, that can then be used to run a migrated application. This approach uses an agent program to commission a delegate system to create virtual environments run on a VM, with one VM potentially hosting multiple virtual environments. This approach takes the encapsulation of the VM and applies it to the application level, thus offering a similar level of isolation. One of the key works on partial application migration is MAUI [69], a method-level migration service that requires the .NET runtime, a limitation of the technique. MAUI optimizes power consumption through fine-grained partitioning of applications by identifying 'remoteable' code segments. MAUI relies on using a managed code environment, such as .NET or Java, and uses an optimization approach to identify at runtime which methods should be executed remotely. The objective is to reduce battery usage whilst minimizing the workload for the programmer. The technique is tested for a face recognition application, an arcade game application, and a voice-based language translation application. The partitioning and offloading technique reduces battery consumption in all three cases. Clonecloud [70] performs a similar function, using device clones in the cloud to seamlessly offload mobile applications that are run in an application-level virtual machine. This method migrates

a chosen thread from the mobile device to the cloud-based clone, and re-integrates the thread at a point in the code chosen by the CloneCloud system. Threads are migrated rather than the complete application as mobile applications have certain features that must remain on the device. The automatic identification of remoteable code is similar to MAUI, but does not require any annotations in the source code by the programmer. Thus it reduces the workload further and is completely automated. The cost of a segment is identified, and an optimization run to choose the segments to offload so as to maximize the energy-aware performance, without input from the programmer. ThinkAir [71] addresses the scalability limitations of MAUI by taking advantage of parallelization through the creation of multiple VMs in the cloud in an online manner to execute code fragments, dynamically creating and destroying VMs as required. Thus the resource allocation is performed in an on-demand manner, with the goal being to optimize resource usage, as only VMs that are currently needed are actually instantiated. Once a VM has finished execution it is destroyed, rather than waiting for usage. These VMs are clones of the smartphone environment, similar to those used in CloneCloud, but ThinkAir considers a multi-user commercial system model to achieve scalability and efficiency for cloud providers, rather than the focus on ease-of-use for programmers of the previous work. This wider view is important, as application migration does not take place in a vacuum, and cloud providers do not only service a single client.

Although application partitioning is useful, it is not the only form of mobile application migration. For example, a complete VM may also be migrated from a device to run in the cloud. [72] introduces a decision support system to match users to cloud providers. The premise of the paper is similar to the works already discussed on VM provisioning, but adopts the use of a knowledge base rather than an optimization formulation. An offerings matcher receives a request to offload a VM and traverses the knowledge base of provider information to find a candidate that satisfies the cost and performance requirements of the user. Having searched the knowledge base, a migration can be performed. Migration to the cloud allows applications to access powerful processing options, but can suffer from latency and bandwidth limitations

that undermine the performance gains from migration. The case is then made for the introduction of local computation resources. [73] employs a two-tier architecture which implements ‘CloudAP’, a cloud base station that offers local cloud functionality, with a conventional cloud data center at the second tier. If a user is in an area covered by CloudAP, they can offload VMs to the base station with improved time to offload and latency. If the user’s movement takes them out of a covered area, the VM may be migrated seamlessly to the second tier cloud center. The combined approach offers better SLA satisfaction than a standard single-tier approach. This idea is formalized in the concept of ‘cloudlets’. Cloudlets operate as a resource-rich computer, or cluster of computers, with connections to the cloud. The cloudlet allows the user to offload most common applications locally, without having to resort to the distant cloud [74]. Cloudlets are decentralized, situated at locations such as coffee shops, managed not by cloud providers, but by local third parties. The resulting latency and bandwidth is therefore that which is afforded by a LAN, which is typically significantly greater than a WAN. [75] distinguishes what can be executed by the cloudlet and what should go to the full cloud, taking advantage of the fact that the connection between the cloudlet and the cloud will be higher bandwidth and lower latency than the connection between mobile device and cloud. Service requests are considered on the required computation and memory requirements and potential offloading cost as well as the device profile and location. A dynamic offloading decision maker chooses an execution plan for each service request and allocates requests to providers. [76] uses auction techniques to optimize resource utilization by appropriately valuing cloudlets and charging users. For example, a user with high computation requirements will give high value to a powerful cloudlet, whilst one requiring a rapid response time will give high value to one with a low-latency connection. To optimally match users (buyers) to cloudlets (sellers), a third-party auctioneer is employed, charging users for services used, as an alternative to optimization.

The works described here may be broadly divided into two categories; those that partition applications for offloading, and those that offload complete VMs. The latter

approach is more useful for the system model proposed in this thesis, as a key motivation is solving the problem of inadequate battery life. However, whilst these works show that application offloading is eminently viable from a technical point of view, the focus of the current body of work is entirely on remote cloud services, or, at the nearest, a local cloudlet. This leaves a key area unaddressed, namely the potential to offload to local computing equipment. It is already common for public locations, such as libraries and Internet cafés, to have large numbers of computing devices. This equipment lacks the customization that modern users require, but the device management and security infrastructure introduced by the Bring-Your-Own-Device paradigm shows that these computers could be utilized in a more powerful and personal way.

2.2.2 Bring-Your-Own-Device

The increased power of mobile devices has resulted in the ‘consumerization’ of productivity, and the rise of the Bring-Your-Own-Device (BYOD) paradigm [13]. BYOD means employees use their personal devices to do their work, with the aim being to create a greater sense of comfort and satisfaction, and a concomitant rise in productivity. For an organisation, BYOD is an opportunity to achieve lower equipment costs (as devices are effectively supplied by the employee), and improved productivity, but it also comes with new risks to manage. Traditionally, a company may safeguard security by requiring staff to use company-issued devices (such as laptops and smartphones), with encryption and restricted privileges (such as software installation), or even preventing all but the most trusted employees from taking work home. As a result, most BYOD work focuses on information security, which I briefly survey here, although it is not the focus of the work in this thesis. [77] outlines a number of security issues that arise from BYOD. For example, data confidentiality is at risk due to potentially sensitive corporate data being stored on personal devices. Any corporate data should therefore be encrypted. Isolation is also threatened as the security requirements for personal and private applications are significantly different. Thus, user-installed apps may provide a backdoor vulnerability, that circumvents the work-related app security. It is necessary, then, for organisations to enforce policies governing what a user may

install on their device to ensure only trusted apps are used. Finally, it is challenging to ensure compliance with security policies as BYOD solutions for this are limited, and the large number of BYOD devices makes manual audit impractical. In the context of this thesis, I consider the implications of a user migrating computation from a personal device to a publicly available piece of equipment, such as a desktop computer. In this case, similar security issues arise, and the user must be assured that the security provisions of the target equipment are sufficient to protect their personal data, and, in return, that a user that is not sufficiently security-conscious will not jeopardize the public equipment.

One solution for improving BYOD security is the use of Mobile Device Management systems, which have been proposed to support the discovery of devices and authenticate a device's access to data. These systems operate centrally, and use an agent installed on an approved mobile device to connect to the MDM system and verify the identity of the device. However, portable MDM has also been proposed, managing a smartphone's ability to control its peripheral devices in a secure context [78]. A portable MDM determines the context in which a device is being used and may limit the use of peripherals, for example, shutting off the phone's camera in an area where photography is not allowed. Likewise, the ability to access certain applications or services in an insecure environment, for example, when connected to an unencrypted network. This idea links to the concept of Mobile Application Management (MAM), which controls application administration. Application administration restricts certain applications and enforces updates. This is important for undiscerning users who can be tricked into installing malware, or when software has security vulnerabilities. Enforcing updates is also important for BYOD, as software (such as Java) is frequently patched as security flaws are discovered and fixed. A third model of BYOD management is Mobile Information Management (MIM), which improves security by controlling access to sensitive information. This approach limits sensitive data access to trusted applications [79], and keeps data stored in a central location, rather than on user devices, to ensure that when not in use it remains secure.

A significant area of vulnerability is in communication, as users may communicate via untrusted public networks. Virtual Private Networks (VPNs) are useful for securing communications when a device is communicating over public WiFi networks, such as those in a university [80]. Another problem on public networks is the risk of mobile malware, which has significantly increased in recent years [81]. For example, as a user migrates their application to another device that has been used by others and is accessible from the Internet, they expose their device to malware access. White-listing of equipment by providers can offer safety guarantees to users, but mobile devices should also have anti-malware measures installed, as user-owned devices cannot be monitored [82].

My problem formulation in Chapter 4 is a stochastic optimization problem, with binary decision variables. Binary optimizations problems (BPs), may be solved by the branch-and-bound method, but problems with integer variables are NP-hard [1]. The consequence is an exponential complexity for general solution methods [2]. In a stochastic scenario where there are many scenarios to consider, this can easily make the problem intractable. If a property known as total unimodularity [28] holds, however, solving the problem as an LP yields integer results in polynomial time [1]. If it cannot be guaranteed that this property holds for all constraints, [31] introduces an algorithm using a partial Lagrange multiplier to solve a sequence of LPs, reducing the problem to polynomial complexity.

The works covered in this section provide a sound technical basis for mobile application offloading in a cloud context, with BYOD research showing that a framework exists to provide device administration and security that can apply this offloading at the local level to fully preserve battery life. This thesis aims to address this ‘gap in the market’ - offloading to local equipment. Having shown that it is a viable concept given current technology, I will propose a system model and optimization problem in Chapter 4, providing mobile users with an alternative to the virtual machine and bandwidth allocation introduced in Chapter 3. However, whilst BYOD research takes the problem of security seriously, there is always more to be done, and thus I address the problem of cloud security in the following section.

Table 2.2: Summary of works addressing problems in mobile application offloading and Bring-Your-Own-Device

P	Article	Objective	Solution
Mobile Application Migration	[61]	Specify languages for cloud applications	Design abstract syntax model as higher abstraction allows easier migration
	[62]	Migrate applications between two mobile devices	Use middleware like J2EE to migrate applications
	[63] [64] [65]	Migrate applications between heterogeneous hardware	Partition applications and partially migrate to optimize performance
	[66]	Minimize the amount of data to be transferred when migrating to the cloud	Cache data at the cloud side
	[67]	Implement migration without depending on Java	Use HTML 5 web workers to enable partitioning of Javascript web applications
	[68]	Protect information security during offloading	Provision virtual environments in the cloud to emulate an Android phone
	[69] [70] [71]	Automate the partitioning of applications to reduce workload for the programmer	Identify ‘remoteable’ code using managed code environments and device clones in the cloud.
	[72]	Match migrated VMs to cloud providers	Combine existing VM allocation methods with knowledge base to find best provider
	[73] [75]	Improve SLA satisfaction and reduce latency	Use two-tier architecture with local cloudlets and cloud providers available
	[76]	Optimize utilization of cloudlet resources	Use third-party and auction mechanism to set cloudlet prices and allocate users

Bring Your Own Device	[13] [77]	Improve security of Bring Your Own Device operations	Provide comprehensive survey of current security issues
	[78]	Improve security of Bring Your Own Device operations	Implement a portable Mobile Device Management system to control smartphone behaviour
	[79]	Improve security of Bring Your Own Device operations	Implement a Mobile Information Management system to limit access to sensitive data to trusted applications
	[80]	Improve security of network communications	Deploy VPNs over public or untrusted networks
	[81] [82]	Avoid malware infection	Survey currently malware, install anti-virus software on mobile devices, and implement whitelisting of trusted offloading equipment

2.3 Cyber Insurance

The first contribution of this thesis is optimizing virtual machine and bandwidth allocation. The second is to provide a mobile application offloading alternative for battery-conscious mobile users. Once any computation leaves a user's device and is executed on a third-party-controlled computer, either locally or in the cloud, security becomes a major concern. The third contribution of this thesis is to provide an approach to cloud security that may be applied to either of these contexts. There are two aspects to the cloud security system model which I propose in this thesis. The first is the problem of Security-as-a-Service allocation, and the second is cyber insurance provisioning. Cyber insurance is a developing area of research, premised on the reality that security is imperfect and a breach is inevitable. Research in this area primarily addresses the problem of security allocation, the setting of cyber insurance parameters, and whether there is a symbiotic relationship between Internet security and cyber insurance. This section identifies the key works in these areas, and identifies areas for expansion. A summary of the works outlined is given in Table 2.3.

2.3.1 Security-as-a-Service

The notion of Security-as-a-Service was introduced in [83], where it was proposed as a way of securing cloud-based data through encryption and distribution of data. Encryption was performed using a cryptographic coprocessor located in the cloud that allows for good quality encryption to be executed quickly, meaning that data stored in the cloud is kept in a more secure fashion, as the encryption serves as an additional line of defence behind the standard system-level security. Not all data is encrypted, but sensitive data is identified in chunks that can be encrypted separately, increasing the speed of data operations. These chunks are then stored in a distributed manner, with access to the master database recording the location of each chunk allowed at only the highest level of security. [84] addresses the problem of selecting cloud service providers (CSP) with security considerations as a priority. The authors propose a framework to manage risk through a combination of technology, processes, and people. There are four components to the framework. Firstly, the security functionality offered by a provider is identified, along with whether the provider satisfies the client's requirements for that function. Secondly the auditability of a provider is assessed, to determine how transparent a provider is about their security processes, so that the client can accurately establish whether the provider maintains high standards. Thirdly, governance is assessed, as the client has to suffer the consequences of security breaches, they must be allowed sufficient control over how their data is handled. Finally, SECaaS providers must offer a service that is compatible with the services offered by other providers. In the cloud, users often employ services from multiple sources, so it is important for them to function together. [85] considers allocation of resources in a parallel computing context with the security overhead considered for both heterogeneous and homogeneous systems. Security overheads are calculated based on the performance of the available authentication methods, hash functions, and encryption algorithms. Security allocation is optimized based on a tradeoff between the strength of a security method and the required computation to perform it. For heterogeneous clusters a real-time approach is used to maximize the probability

that all jobs are executed in a timely fashion. [86] similarly divides security services by priority to optimize processing requirements in a mobile cloud context. However, instead of distinguishing between secure and non-secure data, services are defined as ‘critical’ or ‘normal’. The critical security services offer stronger security but cost the customer more. A Semi-Markov Decision Process is used to admit services, modeling the system reward for the cloud provider. Reward is calculated as the expected income for the provider offset by the expected running costs. Real-time systems may have critical security requirements, such as railway signaling or medical electronics. It is essential for rapid response times, but if a malicious outsider can gain access to the systems, it may lead to catastrophic results, including loss of life. Therefore [87] provisions security services in real-time systems to optimize performance, where the scheduling of jobs is combined with the allocation of security services. Security services are divided into groups according to type of service, with services provisioned from each group according to an individual service’s performance. The full security provision is then built by combining the chosen services from each group. Scheduling of jobs is done using a security-aware version of the earliest deadline first real-time scheduling algorithm. [88] addresses a shortcoming in the above scheduling methods, which is that they focus on CPU usage as the measure of resource requirements. In reality, cloud services also require memory and storage, which may alter the computation time and financial cost. The paper takes an approach that is both security-aware and budget-aware for scheduling tasks among multiple cloud service providers. [89] introduces the idea of firewall-style SECaaS providers. Traditional cloud security typically involves repurposing of existing security techniques that may lack the ability to scale up sufficiently to the cloud level. Therefore, a new architecture revolving around dedicated security services functioning as a firewall protecting cloud services. Similarly, [90] considers the problem of security for microservice architectures, and designs a network monitoring API called FlowTap to handle security. Microservices can be distributed and independently scaled, making them useful in cloud applications, but more complex to secure, particularly as each microservice assumes that the

others are trustworthy. FlowTap provides security policy enforcement and a monitoring infrastructure for network traffic, securing the network connection, rather than individually securing each application. This is shown to be important as, whilst users can install security software in a virtual environment, they have no control over the network traffic in the cloud. This a good solution when systems are interdependent and a weakness in one component can compromise the whole. I consider this approach for the SECaaS providers in this thesis, focusing on network traffic analysis as the primary means of providing security services.

The current state of the art in the field of cloud security provides a sound basis for the Security-as-a-Service model used in this paper. This approach outsources security to a service provider with more power and expertise than the average user, just as other cloud paradigms do. However, these works can also be accused of having too narrow a focus. Security provisioning is important and valid, but must now only be considered half the story. With the high profile and expensive nature of security breaches, there is concern not only over the security of a system, but also the consequences of a breach. With the rise of cyber insurance, a holistic approach must also consider the insurance costs and benefits when provisioning security.

2.3.2 Cyber Insurance

Cyber insurance is still developing as an area of interest, and there are a number of problem areas that will be addressed by researchers in the coming years. In general, insurance involves the transfer of risk from the user to the insurer, and cyber insurance provides explicit coverage of damages from cyber attacks. In this section I provide an overview of the key issues and current work in cyber insurance. In addition to risk transfer, risk management is also an essential part of the insurance process, and in this case, incorporates cyber security.

2.3.2.1 Risk Transfer

The field of cyber insurance contains a number of unique problems compared to other fields of insurance. The European Network and Information Security Agency

(ENISA) found that information asymmetry is a particular problem, where companies are reluctant to share full details of their security provisioning with insurers [91]. There is a lack of definition regarding what constitutes ‘good’ cyber security, and therefore companies design their own measures without reference to an international standard. However, this is changing, and companies may show that they conform to certain established ‘best practices’, such as the ISO2700x security management standards. Conforming to standards can reduce information asymmetry without compromising a security provider’s proprietary secrecy. Further, the SANS Institute found that the frequency of breaches is difficult to predict [92], as historic data on cyber security breaches is not available, unlike more traditional branches of insurance, such as car, or fire insurance. As a result the exact degree of risk in a system is difficult to quantify, and there is a lot of uncertainty in setting premiums and deductibles, as the actual cost and frequency of an attack may prove to be of much larger magnitude than anticipated. As the cyber insurance market matures, there will be an incentive for insurers and security providers to cooperate more closely, and increase the quantity of available data. The problem of non-transparent security provisioning is exacerbated by the interdependence of computer systems. With many cloud applications designed in a distributed manner with close intercommunication between them it is difficult to assess the overall vulnerability of a system, particularly where components are hosted by different cloud service providers, or protected by different security providers [93]. This correlation extends not only to different parts of a single system, but also to different insured entities, where the vulnerability of one client may impact another, serving as an access point to the whole cloud. This is made worse by the homogeneity of modern computer systems and lack of geographical variation. Thus, a vulnerability in one client’s software package installation may be mirrored in many others, and may allow the infection of another client through the first’s system. The scarcity of data available further compounds the problem, making it challenging to accurately assess the risk level and determine the correctness of that assessment [94]. Unlike other fields of insurance, it is even challenging to determine when an attack has actually taken place, as many are lengthy and leave liability unclear [95]. This differs from

traditional insurance where, for example, it is obvious if your car has crashed, or your house burned down. However, a skillful attacker may steal user data and not make the fact known until months, or even years, later. Even when an attack is discovered, the quantification of damage is difficult to define, due to the intangibility of losses. Since data has no intrinsic physical value, losses are not simply a calculation of the quantity of data stolen, but must be collated from other expenses. A company may invest in public relations to mitigate damage to their image, may increase security in the wake of an attack, which incurs expense, or be required to reach settlements with affected customers, which will incur legal fees on top of the reparations to be made. Damages may also extend to loss of business, as a significant attack may damage perceptions of company's competence and reliability. For the sake of this thesis, I make assumptions that the probability distribution of risk is known, and damages incurred can be correctly assessed. This area of cyber insurance requires further work to develop that is beyond the scope of this research.

2.3.2.2 Risk Management

Cyber insurance, as already discussed, is primarily concerned with the problem of risk transfer - where the cost from risk is transferred to the insurer. However, risk management and assessment is also a notable part of the literature, attempting to address the issues described above. Risk management is achieved through the establishment of methodologies and best practices that are adapted from normal business practices, with consideration for IT-specific measures. The Microsoft Security Risk Management Guide [96] recommends a cyclical four-phase approach to risk management. The first phase is assessing risk, identifying potential security flaws and the scope for damage from exploitation. The second phase is conducting decision support. This involves identifying risk management solutions and establishing their value according to cost-benefit analysis. The third phase is the implementation phase where the evaluated solutions are put into practice in the business. Finally, the effectiveness of these measures in practice should be measured. This requires establishing that the implementation is of a high enough standard, and that the methods are providing

the expected level of protection against risk. ISO/IEC 27001, along with the entire ISO2700x suite is an example of an international standard for IT security risk management [97], showing that solutions are primarily based on business practices, rather than technology. As already discussed, the introduction of standards allows the effective measurement of risk management and allows companies to follow a template to improve their provisions. In this thesis I assume that the security service providers in my system model are in compliance with international standards on risk management, and therefore the level of risk can be reasonably quantified.

2.3.2.3 Relationship Between Cyber Security and Insurance

The most important aspect of risk transfer in cyber insurance is the specification of the premium. The research in this area is primarily divided between independent security and interdependent security. In this thesis I focus on the independent case, where clients are independent of one another. The interdependent case is more complex and is beyond the scope of this thesis. In addition to the factors already discussed, such as risk assessment, predictability of attacks, and damage quantification, it is important to consider the potential impact on security provisioning when setting the insurance premium. [98] presents a mathematical model and experimental results to show that cyber insurance encourages good security practices, one of the key aspects of cyber insurance. Thus, ‘self-protection’ and ‘self-insurance’ are not mutually exclusive, but complementary. Intuitively this is a logical conclusion, as door locks and a burglar alarm do not prevent people from purchasing insurance for their possessions. However, in many cases, there are both security-related and non-security-related IT risks, and these are not always easy to separate. [99] proposes a model of cyber insurance called Aegis where users choose to take a proportion of liability on themselves. When insurance is compulsory, this is shown to be preferable. Thus, rather than choosing traditional cyber insurance contracts, that only cover damages incurred by cyber attacks, customers choose a contract that requires the payment of a deductible but offers coverage of non-security cyber damages. The presence of a deductible also functions as an incentive to customers to increase their security provisions, as they

are taking more personal responsibility. [100] looks at non-life cyber insurance, that is, insurance that covers repeated events rather than insuring against a single event, such as a major cyber attack. Using a Markov chain to model attack scenarios where software in a system is considered as a series of states with or without damage, the probability of damages/survival can be established by considering the amount of time a piece of software spends in each state and the probability of transitions. Insurance premiums are then set accordingly. [27] considers the problem of allocating non-life insurance alongside network monitoring SECaaS services, with a stochastic optimization model finding the optimum allocation of security and insurance as a joint decision to minimize the customer's cost. The use of stochastic optimization allows provisioning decisions to be made that match varying future demands as closely as possible. This relies on the ability to model incoming data and attack probabilities accurately. I use this system model as the basis for the cyber insurance work in this thesis, focusing on non-life insurance in an independent-security context. I must assume that the liabilities can be determined, and that the probability distribution for attacks is known. This assumption requires good test data, and so my numerical results are based on parameters derived from an analysis of cyber attack data. In this thesis I aim to demonstrate a connection between cyber insurance and SECaaS provisioning, as properly designed insurance contracts can incentivize security investment. [101] shows that cyber insurance does not intrinsically encourage security investment unless contract discrimination is allowed. Without this discrimination, the market either reaches an equilibrium that either does not result in security improvements, or, in the monopolistic case, results in the cyber insurer making zero profits. In the latter case there need to be incentives imposed on the market for insurers to participate. In this thesis I add the stipulation that if packets are not assessed by security services, they are not liable for insurance coverage, acting as a form of simple contract discrimination. I can demonstrate that in this case, cyber insurance encourages investment in security.

This thesis attempts to address the problem of joint provisioning of cloud security and cyber insurance. I have shown in this review that the security model considered

in Chapter 5 is realistic. The field of cyber insurance requires further development, although it is already gaining traction with insurance companies. To justify further research into improving the accuracy of cyber insurance models and parameter settings, it is important to show that it has a mutually beneficial relationship with cyber security. This thesis examines if this relationship can be established, and uses real attack data to provide authentic parameter settings across a multi-stage stochastic optimization. This is an area that has not been adequately addressed by the literature, and could be applied as security for the system models introduced in Chapters 3 and 4.

Table 2.3: Summary of works addressing problems in Security-as-a-Service and cyber insurance

P	Article	Objective	Solution
Security-as-a-Service	[83]	Improve security of cloud-based data	Split data into chunks, encrypt sensitive chunks, and distribute storage in the cloud
	[84]	Choose cloud providers to optimize security	Assess compatibility, auditability and governance of potential providers
	[85]	Allocate resources for security in heterogeneous systems	Provision security according to tradeoff between quality of security and computation required to perform it
	[86]	Allocate resources for security services	Prioritize security services by ‘critical’ and ‘normal’ using semi-Markov decision process
	[87]	Provision security in real-time	Provision security and schedule jobs simultaneously with security services grouped to reduce workload
	[88]	Allocate security according to complete set of resource requirements	Consider network and storage requirements and introduce budget constraints
	[89]	Design dedicated architecture for cloud security	Implement firewall-style services that scan incoming network traffic
	[90]	Implement security for microservices architecture	Implement security on incoming traffic not explicit application protection

Cyber Insurance	[91]	Survey issues in cyber insurance	Highlight problem of information asymmetry and encourage definition of ‘good’ cyber security
	[92]	Survey issues in cyber insurance	Highlight problem of predicting frequency, and therefore likelihood of attack
	[93]	Survey issues in cyber insurance	Highlight problem of system interdependence, as related systems share weaknesses
	[94] [95]	Survey issues in cyber insurance	Highlight problem of identifying risk levels due to lack of data widely available
	[96]	Improve cyber risk management	Enact four-phase approach to risk management, assessing and improving security flaws
	[97]	Improve cyber risk management	ISO 27001 on IT security risk management to define effective business practices
	[98]	Investigate whether insurance can encourage security investment	Provide mathematical and experimental models to show that the two go together
	[99]	Design insurance market to encourage security investment	Design market where users take partial liability to invite insurers to take part
	[100]	Set cyber insurance premiums effectively	Use Markov chain to model attack scenarios and set premiums accordingly
	[27]	Jointly allocate security services and cyber insurance	Use stochastic optimization model to optimize advance allocation of each resource
	[101]	Investigate whether insurance can encourage security investment	Establish that there must be contract discrimination and security must be adequate

Chapter 3

Joint Optimization of Resource Provisioning in Cloud Computing

The first application of stochastic optimization to mobile cloud computing in this thesis is to address the issue of guaranteeing the performance of cloud applications. In this chapter, I do this by addressing the problem of joint VM and bandwidth provisioning. This chapter is organised as follows. Section 3.1 introduces the cloud environment and system model. Section 3.2 outlines the detailed stochastic problem formulation. Section 3.3 lays out the scenario tree reduction algorithms. Section 3.4 presents the sensitivity analysis formulation and method. Section 3.5 contains an example scenario with corresponding numerical results to compare with the previously introduced techniques.

3.1 System Model

The cloud computing environment under consideration, with an example allocation scenario of a user streaming video, is illustrated in Fig. 3.1. Cloud providers host data centers, providing cloud services under the IaaS paradigm. Clients may provision VMs from providers, choosing an appropriate VM specification according to their needs. Providers may locate their data centers in different geographic locations, requiring communications between client and provider to go via Internet links. This traffic is transferred through routers provided by network service providers. By implementing SDN-enabled networking, bandwidth can be provisioned according to each client's demand. Clients provide VM demand requirements to an SDN controller, which places

Figure 3.1: Layout of network with regions, cloud providers, and clients (users) labeled.

VMs with providers and provisions router bandwidth accordingly. Network routers are connected in clusters according to geographic location, with clusters connected by higher capacity gateway routers. The SDN controller must provision VMs and bandwidth in combination to optimize costs, by trading off VM savings against bandwidth costs incurred by geographic distance.

In order to save costs in the long term, both VM and network providers offer reservation options. Resources can be reserved in advance, and then utilized at a reduced rate. VM demand, and costs, may be uncertain in advance, as well as each VM's bandwidth requirements, and bandwidth costs of each router, during use. Well chosen reservation quantities and placement can provide significant cost savings over time. However, as well as meeting demand, the provisioning amounts of each resource must remain within the capacity constraints of each resource provider.

I now provide elaboration on key features of the system model. Major mathematical notations are listed in Table 5.1.

Table 3.1: List of Key Notations for Chapter 3.

Symbol	Definition
$\mathcal{U}$	Set of all users, while $u \in \mathcal{U}$ denotes the user index
$\mathcal{P}$	Set of all cloud providers, while $p \in \mathcal{P}$ denotes the provider index
$\mathcal{V}$	Set of all VM classes, while $V_i \in \{V_1, V_2, \dots, V_{last}\}$ denotes the VM class index
$\mathcal{R}$	Set of all network routers, while $\mathcal{R} = \{1, \dots, R\}$ denotes the router index
$\mathcal{T}$	Set of all provisioning stages, while $t \in \mathcal{T}$ denotes the time stage index
$\mathcal{L}$	Set of reservation contracts for bandwidth, while $L_1 \in \{L_1, L_2, \dots, L_{last}\}$ denotes the contract index
$\mathcal{K}$	Set of reservation contracts for VMs, while $K_1 \in \{K_1, K_2, \dots, K_{last}\}$ denotes the contract index
$t_p^{(h)}, t_p^{(s)}, t_p^{(n)}$	Capacity of cloud providers for processing, storage, and internal network bandwidth
t_r	Bandwidth capacity of routers
$d_i^{(h)}, d_i^{(s)}, d_i^{(n)}$	Resource demands of VM classes for processing, storage, internal network bandwidth
$d_i^{(b)}$	Resource demands of VM classes for Internet bandwidth
$v_{iut}(\omega)$	Number of VMs required under scenario ω
$c_{ipk}^{(R)}, c_{rl}^{(R)}$	Fixed first stage costs for VMs and network bandwidth
$c_{ipkt}^{(re)}(\omega), c_{ipkt}^{(u)}(\omega),$ $c_{ipkt}^{(o)}(\omega)$	VM costs for reservation, utilization, and on-demand under all stages and scenarios
$c_{rlt}^{(re)}(\omega), c_{rlt}^{(u)}(\omega),$ $c_{rlt}^{(o)}(\omega)$	Bandwidth costs for reservation, utilization, and on-demand under all stages and scenarios
$c_{ipkt}^{(s)}(\omega), c_{ipkt}^{(b)}(\omega)$	Storage and outbound bandwidth additional costs for VM provisioning
$X_{rul}^{(R)}, Y_{ipk}^{(R)}$	Deterministic first stage decision variables for bandwidth and VM reservation
$X_{rult}^{(re)}(\omega), X_{rult}^{(u)}(\omega),$ $X_{rult}^{(o)}(\omega)$	Decision variables for bandwidth reservation, utilization and on-demand allocation
$Y_{uipkt}^{(re)}(\omega), Y_{uipkt}^{(u)}(\omega),$ $Y_{uipkt}^{(o)}(\omega)$	Decision variables for VM reservation, utilization and on-demand allocation
Ω	Set of all scenarios, while $\omega \in \Omega$ denotes the scenario index

3.1.1 Provisioning Scheme

Resources are provisioned from both cloud providers and network providers under one of two provisioning plans. The first is reservation, allowing clients to guarantee resource availability in advance for an extended period (e.g. 3 months, 1 year). At

the time of use, resources may be utilized under the reservation plan, or provisioned under the alternative, on-demand, plan at a costlier rate. Since the duration of a reservation plan is relatively long, I divide time into a set of provisioning stages of shorter duration (e.g. 1 hour, 1 day). User demand and on-demand costs are uncertain ahead of time, meaning that each provisioning stage involves a combination of utilizing reserved resources, and provisioning on-demand resources. A provisioning stage is therefore divided into three phases. The first phase allows the initiation of additional reservation plans if necessary - for example, if a previous plan needs renewing. The second phase takes the realized demand and utilizes as much of the available reserved resources as required. In the event that this is insufficient, the third, on-demand, phase provisions resources under the on-demand plan to fulfill the shortfall.

3.1.2 Reservation Contracts

Reservation plans are offered to clients in the form of reservation contracts. For example, a cloud provider may offer a 3-month or 6-month contract, where resources are reserved for either 3 or 6 months, respectively. A contract may be started at any point where sufficient time stages remain to complete the duration of the contract. VM contracts are denoted by $k \in \mathcal{K}$, whilst bandwidth contracts are denoted by $l \in \mathcal{L}$; each contract may be of differing length and price. In this section I define the behaviour of VM contracts, but bandwidth contracts are defined similarly, and can be found by simply replacing k with l in each expression. The set of stages at which a contract can be provisioned is therefore denoted by $\mathcal{T}_k$, defined as follows:

$$\mathcal{T}_k = \{1, \dots, |\mathcal{T}| - |k| + 1\}, \quad (3.1)$$

for cloud providers, and $\mathcal{T}_l$ for network providers. Resources reserved under a contract may be utilized at any of the subsequent time stages within the duration of the contract. The resources available for utilization in time stage t , are therefore all resources reserved in previous time stages whose contracts have not yet expired. $\mathcal{M}_{kt}$, defined as follows:

$$\mathcal{M}_{kt} = \{\max(1, t - |k| + 1), \dots, \min(t, |\mathcal{T}| - |k| + 1)\}, \quad (3.2)$$

is given as the set of time stages prior to t in which contract k could have been reserved, such that it is available for utilization in time stage t . $\mathcal{M}_{lt}$ denotes the equivalent term for bandwidth contract. Similarly, I give $\mathcal{N}_{kt}$, defined as follows:

$$\mathcal{N}_{kt} = \{t, \dots, \min(|\mathcal{T}|, t + |k| - 1)\}. \quad (3.3)$$

as the set of time stages covered by contract k , if it was reserved in time stage t . I define $\mathcal{N}_{lt}$ equivalently for bandwidth contracts.

3.1.3 Uncertainty of Parameters

If all parameters are known exactly in advance, the solution is a simple deterministic optimization, with no need for an on-demand phase. In this system model, I consider the uncertainty of three parameters. The number of VMs of each class required by each client is unknown in advance. Each VM class has a combination of VM type and bandwidth requirement, and to provide per-VM bandwidth uncertainty, I allow for VM classes with identical resource requirements, but with differing external bandwidth requirements. Thus the bandwidth requirements from each provider is dependent on the number and bandwidth requirements of VMs assigned to a cloud provider, making the placement of bandwidth demands unknown in advance. Additionally, and independently, the costs of both VMs and bandwidth are also uncertain in advance, with an exception given for reservation costs in the first time stage, which are assumed to be known. The demand level of VMs and bandwidth should be treated as uncertain as they are the primary resource to be provisioned, and assuming that they are deterministic is unrealistic and an oversimplification. Uncertainty of VM demand is an established issue in the literature, and with the bandwidth requirements of applications varying significantly (e.g. Netflix offering variable download rates), it is important to consider bandwidth uncertainty. Internet bandwidth allocation has a number of uncertainties, notably delay and latency caused by competing traffic. However, accurately modeling all these would be a legitimate topic for research in its own right, so I assume that the network capacity remains consistent, guaranteed by the ISP. It would be possible to extend the model to consider capacity uncertainty,

so I examine the influence of capacity changes on allocation in the numerical results. Price uncertainty is considered as the model may be implemented over a long time period (i.e. several months), and prices may well be changed by the service provider within that time. It is assumed that the possible values and probability distributions of the uncertain parameters are known. To solve the problem with these uncertain factors, I formulate a stochastic integer programming problem, which considers a set of scenarios, which encompass the parameters listed. The set of all scenarios in every provisioning stage is denoted by Ω ; with Ω_t denoting the set of all scenarios in provisioning stage t . Thus Ω is given by

$$\Omega = \prod_{t \in \mathcal{T}} \Omega_t = \Omega_1 \times \Omega_2 \times \cdots \times \Omega_{|\mathcal{T}|}. \quad (3.4)$$

Once a scenario is known, it is called a realization. Each realization is a composite of the possible time stage demands and is defined as $\omega = (\omega_1, \dots, \omega_{|\mathcal{T}|}) \in \Omega$. Each realization can be considered as a tuple, containing the set of VM demands for each client and the cost realization.

3.1.4 Provisioning Costs

Reserving VMs and bandwidth incurs three different cost types - one for each of the three provisioning phases. The reservation, utilization and on-demand costs may all be varied by the cloud providers, with only reservation in the first provisioning stage remaining fixed. The purpose of the optimization is to minimize the client's costs despite the uncertain demand and prices. First stage reservation costs are fixed, as they are assumed to be known. All subsequent pricing may vary with uncertainty. In addition to VM and bandwidth provisioning costs, cloud providers, such as Amazon [3], also charge for outbound bandwidth and storage, for certain VM types. I model this with additional cost weighting for VMs provisioned in the utilization and on-demand phases.

3.2 Problem Formulation

The goal of the optimization problem presented in this section is to find the optimal reservation of virtual machines and bandwidth. This reservation consists of both the

quantity of the resource reserved, the duration of the reservation, and the location of the reservation (i.e., the cloud provider or network path hosting the resource). This decision is made in an offline fashion, with the reservation variable values determined by the expected future values of the utilization and on demand variables. The quality of the solution is therefore dependent on the quality of the parameter settings, which must be anticipated from past usage and pricing data patterns. In this case, it is helpful to consider the application that is being optimized, for example, a company using the cloud for big data processing (such as MapReduce). The number of VMs required will vary according to the amount of nodes required by the processing, and whether there is processing currently being conducted. If there are a number of jobs being run, the bandwidth requirement between the company and the cloud may be high to transfer data, requiring high bandwidth end-to-end to prevent delays in business. The problem formulation must therefore aim to find the best tradeoff of reservation considering the varying workload, provisioning sufficient resources to handle the computation, without exceeding the capacity of the cloud provider or the network provider, and ensuring that the bandwidth provisioning is consistent along the whole route.

3.2.1 Stochastic Optimization Formulation

To solve the problem, I present a stochastic programming optimization with multi-stage recourse. The recourse action provisions utilization and on-demand resources in the multiple provisioning stages. The problem formulation is as follows:

$$\min_{X_{rl}^{(R)}, Y_{ipk}^{(R)}} \sum_{r \in \mathcal{R}} \sum_{p \in \mathcal{P}} \sum_{u \in \mathcal{U}} \sum_{V_i \in \mathcal{V}} \sum_{k \in \mathcal{K}} \sum_{l \in \mathcal{L}} \left(c_{rl}^{(R)} X_{rul}^{(R)} + c_{ipk}^{(R)} Y_{uipk}^{(R)} + \mathbb{E}_{\Omega}[\mathcal{Q}(X_{rul}^{(R)}, Y_{uipk}^{(R)}, \omega)] \right), \quad (3.5)$$

$$Y_{uipk}^{(R)} \in \mathbb{N}_0, \forall u \in \mathcal{U}, \forall V_i \in \mathcal{V}, \forall p \in \mathcal{P}, \forall k \in \mathcal{K}, \quad (3.6)$$

$$X_{rul}^{(R)} \geq 0, \forall r \in \mathcal{R}, \forall u \in \mathcal{U}, \forall l \in \mathcal{L}. \quad (3.7)$$

The objective function given in (3.5) minimizes the total cost of resource provisioning by minimizing the cost of reservation in the first stage, and the expected cost

$$\begin{aligned}
 \mathcal{Q}(X_{rul}^{(R)}, Y_{uipk}^{(R)}, \omega) = & \min \sum_{r \in \mathcal{R}} \sum_{u \in \mathcal{U}} \sum_{l \in \mathcal{L}} \sum_{t \in \mathcal{T}_l} c_{rlt}^{(re)}(\omega) X_{rult}^{(re)}(\omega) + \sum_{u \in \mathcal{U}} \sum_{V_i \in \mathcal{V}} \sum_{p \in \mathcal{P}} \sum_{k \in \mathcal{K}} \sum_{t \in \mathcal{T}_k} c_{ipkt}^{(re)}(\omega) Y_{uipkt}^{(re)}(\omega) + \\
 & \sum_{r \in \mathcal{R}} \sum_{u \in \mathcal{U}} \sum_{l \in \mathcal{L}} \sum_{t \in \mathcal{T}} \left(c_{rlt}^{(u)}(\omega) X_{rult}^{(u)}(\omega) + c_{rlt}^{(o)}(\omega) X_{rult}^{(o)}(\omega) \right) + \\
 & \sum_{u \in \mathcal{U}} \sum_{V_i \in \mathcal{V}} \sum_{p \in \mathcal{P}} \sum_{k \in \mathcal{K}} \sum_{t \in \mathcal{T}} \left((c_{ipkt}^{(u)}(\omega) + c_{ipkt}^{(s)}(\omega) d_i^{(s)} + c_{ipkt}^{(b)}(\omega) d_i^{(b)}) Y_{uipkt}^{(u)}(\omega) \right. \\
 & \quad \left. + (c_{ipkt}^{(o)}(\omega) + c_{ipkt}^{(s)}(\omega) d_i^{(s)} + c_{ipkt}^{(b)}(\omega) d_i^{(b)}) Y_{uipkt}^{(o)}(\omega) \right). \tag{3.8}
 \end{aligned}$$

from provisioning in the remaining stages. (3.6) ensures the VM reservation is an integer, whilst (3.7) ensures the bandwidth reservation is non-negative. The expected cost from provisioning under the uncertainty set Ω is given by $\mathbb{E}_\Omega[\mathcal{Q}(X_{rul}^{(R)}, Y_{uipk}^{(R)}, \omega)]$, where $\mathcal{Q}(X_{rul}^{(R)}, Y_{uipk}^{(R)}, \omega)$ minimizes the provisioning cost given scenario ω .

The cost function (3.8) minimizes the cost under uncertainty scenario ω using six decision variables. These variables represent the bandwidth and VM allocations for each provisioning phase. The total provisioning cost is summed across all routers, providers, contracts, clients, and time stages - weighted by the corresponding provisioning costs. This cost function, together with (3.5), form the objective for our stochastic formulation. This formulation is subject to a set of constraints.

$$Y_{uipkt}^{(u)}(\omega) \leq \sum_{\hat{t} \in \mathcal{M}_{kt}} Y_{uipk\hat{t}}^{(re)}(\omega), \forall u \in \mathcal{U}, \forall V_i \in \mathcal{V}, \forall p \in \mathcal{P}, \forall k \in \mathcal{K}, \forall t \in \mathcal{T}, \tag{3.9}$$

$$X_{rult}^{(u)}(\omega) \leq \sum_{\hat{t} \in \mathcal{M}_{lt}} X_{rul\hat{t}}^{(re)}(\omega), \forall u \in \mathcal{U}, \forall r \in \mathcal{R}, \forall l \in \mathcal{L}, \forall t \in \mathcal{T}, \tag{3.10}$$

(3.9) and (3.10) ensure that the VMs and bandwidth provisioned in the utilization phase do not exceed the available reserved resources, as only pre-reserved resources are entitled to the cheaper utilization rate. If a reserved contract's duration does not cover a given time stage, no resources from that contract can be utilized - a new reservation is required.

$$X_{rul}^{(R)} = X_{rul\bar{t}}^{(re)}(\omega), \bar{t} = 1, \forall u \in \mathcal{U}, \forall r \in \mathcal{R}, \forall l \in \mathcal{L}, \quad (3.11)$$

$$Y_{uipk}^{(R)} = Y_{uipk\bar{t}}^{(re)}(\omega), \bar{t} = 1, \forall u \in \mathcal{U}, \forall V_i \in \mathcal{V}, \forall p \in \mathcal{P}, \forall k \in \mathcal{K}, \quad (3.12)$$

(3.11) and (3.12) ensure that any reservations made in the first stage are exempt from price uncertainty, as the reservation decision is identical across all scenarios in the first time stage.

$$\sum_{u \in \mathcal{U}} \sum_{l \in \mathcal{L}} \left(X_{rult}^{(u)}(\omega) + X_{rult}^{(o)}(\omega) \right) \leq t_r, \forall r \in \mathcal{R}, \forall t \in \mathcal{T}, \quad (3.13)$$

$$\sum_{V_i \in \mathcal{V}} d_i^{(h)} \left(\sum_{u \in \mathcal{U}} \sum_{k \in \mathcal{K}} (Y_{uipkt}^{(u)}(\omega) + Y_{uipkt}^{(o)}(\omega)) \right) \leq t_p^{(h)}, \forall p \in \mathcal{P}, \forall t \in \mathcal{T}, \quad (3.14)$$

$$\sum_{V_i \in \mathcal{V}} d_i^{(s)} \left(\sum_{u \in \mathcal{U}} \sum_{k \in \mathcal{K}} (Y_{uipkt}^{(u)}(\omega) + Y_{uipkt}^{(o)}(\omega)) \right) \leq t_p^{(s)}, \forall p \in \mathcal{P}, \forall t \in \mathcal{T}, \quad (3.15)$$

$$\sum_{V_i \in \mathcal{V}} d_i^{(n)} \left(\sum_{u \in \mathcal{U}} \sum_{k \in \mathcal{K}} (Y_{uipkt}^{(u)}(\omega) + Y_{uipkt}^{(o)}(\omega)) \right) \leq t_p^{(n)}, \forall p \in \mathcal{P}, \forall t \in \mathcal{T}, \quad (3.16)$$

(3.13)-(3.16) ensure that the VMs and bandwidth that are used by the client do not exceed the resource capacities of the cloud providers and routers from which they are provisioned. The total quantities of each resource used should not exceed the capacity of each provider for that resource.

$$\sum_{p \in \mathcal{P}} \sum_{k \in \mathcal{K}} \left(Y_{uipkt}^{(u)}(\omega) + Y_{uipkt}^{(o)}(\omega) \right) \geq v_{iut}(\omega), \forall V_i \in \mathcal{V}, \forall u \in \mathcal{U}, \forall t \in \mathcal{T}, \quad (3.17)$$

(3.17) requires that the utilization and on-demand provisioning of VMs from each class across all cloud providers is sufficient to meet the realized demand for that class of VM, for each client, in each time stage. This demand is summed across all providers,

allowing the placing of VMs with cloud providers such that both VM and bandwidth costs are optimized jointly.

$$Y_{uipkt}^{(re)}(\omega) \in \mathbb{N}_0, \forall u \in \mathcal{U}, \forall V_i \in \mathcal{V}, \forall p \in \mathcal{P}, \forall k \in \mathcal{K}, \forall t \in \mathcal{T}_k, \quad (3.18)$$

$$Y_{uipkt}^{(u)}(\omega), Y_{uipkt}^{(o)}(\omega) \in \mathbb{N}_0, \forall u \in \mathcal{U}, \forall V_i \in \mathcal{V}, \forall p \in \mathcal{P}, \forall k \in \mathcal{K}, \forall t \in \mathcal{T}, \quad (3.19)$$

$$X_{rult}^{(re)}(\omega) \geq 0, \forall r \in \mathcal{R}, \forall u \in \mathcal{U}, \forall l \in \mathcal{L}, \forall t \in \mathcal{T}_l, \quad (3.20)$$

$$X_{rult}^{(u)}(\omega), X_{rult}^{(o)}(\omega) \geq 0, \forall r \in \mathcal{R}, \forall u \in \mathcal{U}, \forall l \in \mathcal{L}, \forall t \in \mathcal{T}. \quad (3.21)$$

(3.18) and (3.19) ensure that the number of VMs provisioned in each phase is a non-negative integer. Similarly, (3.20) and (3.21) require that all router bandwidth provisioning is non-negative. Treating VMs as atomic means that our problem is mixed integer, and therefore more complex to solve. It is possible to consider VM provisioning as a percentage of time used rather than as an atomic unit, thus I could reasonably convert a mixed integer problem into a linear program. As long as VM demand is an integer, I find that the result in this case remains the same. This relaxation becomes important when performing sensitivity analysis, covered in Section 3.4.

As well as meeting VM demand, it is essential that bandwidth demand is also met for a joint solution to work correctly. This means that as well as sufficient bandwidth being allocated to the routers adjacent to the cloud providers' data centers, the allocation must be preserved consistently across the network back to the client. I consider the networks as a graph, $G = (\mathcal{R}, \mathcal{E})$, where each vertex is a router in $\mathcal{R}$, and each edge in $\mathcal{E}$ is a link. I assume that provisioned upload and download bandwidth is symmetric. In general, traffic is considered to flow 'out' from cloud providers and 'in' to clients. All routers have both 'in' and 'out' links. This network labelling is illustrated in Fig. 3.1, where example nodes and edges are labelled accordingly. The flow constraints are defined as follows:

$$\sum_{e \in \mathcal{E}_r^{out}} \sum_{l \in \mathcal{L}} f_{eult}(\omega) - \sum_{e \in \mathcal{E}_r^{in}} \sum_{l \in \mathcal{L}} f_{eult}(\omega) = 0, \forall r \in \mathcal{R}, \forall u \in \mathcal{U}, \forall t \in \mathcal{T}, \quad (3.22)$$

$$\sum_{e \in \mathcal{E}_p^{out}} \sum_{l \in \mathcal{L}} f_{eult}(\omega) - \sum_{l \in \mathcal{L}} \left(X_{rult}^{(u)}(\omega) + X_{rult}^{(o)}(\omega) \right) = 0, \forall r \in \mathcal{R}, \forall u \in \mathcal{U}, \forall t \in \mathcal{T}, \quad (3.23)$$

$$\sum_{e \in \mathcal{E}_p^{out}} \sum_{l \in \mathcal{L}} f_{eult}(\omega) \geq \sum_{V_i \in \mathcal{V}} \sum_{k \in \mathcal{K}} d_i^{(b)} \left(Y_{uipkt}^{(u)}(\omega) + Y_{uipkt}^{(o)}(\omega) \right), \forall p \in \mathcal{P}, \forall u \in \mathcal{U}, \forall t \in \mathcal{T}, \quad (3.24)$$

$$\sum_{e \in \mathcal{E}_u^{in}} \sum_{l \in \mathcal{L}} f_{eult}(\omega) \geq \sum_{p \in \mathcal{P}} \left(\sum_{e \in \mathcal{E}_p^{out}} \sum_{l \in \mathcal{L}} f_{eult}(\omega) \right), \forall u \in \mathcal{U}, \forall t \in \mathcal{T}. \quad (3.25)$$

- (3.22) is a flow conservation constraint. The variable $f_{eult}(\omega)$ denotes the amount of network flow provisioned on edge e , for client u , under contract l , at time t . The set $\mathcal{E}_r^{out}$ contains the edges that connect directly to router r in the outward direction. Similarly, $\mathcal{E}_r^{in}$ denotes the set of edges that connect directly to router r in the inward direction. Thus I ensure that the sum of all traffic flowing into a router is equal to the sum of all traffic leaving the router - no bandwidth use is gained or lost between cloud provider and client throughout the network.
- (3.23) relates the flow variables defined on edges, to the bandwidth allocated to routers. Bandwidth is obtained from routers in the network. Consequently, this constraint ensures that the traffic flowing through the router is equal to the total bandwidth allocated to the router in a given time stage.
- (3.24) ensures that the bandwidth requirements of all VMs on all cloud providers are satisfied by the edges that connect directly to those cloud providers (denoted by the set $\mathcal{E}_p^{out}$). The total bandwidth demand on each cloud provider in scenario ω is the sum of the individual bandwidth requirements of each of the VMs provisioned from that cloud provider.

- (3.25) serves a similar purpose to (3.23), guaranteeing that the bandwidth requirements of all VMs associated with client u are satisfied by the edges that connect directly to the client, denoted by the set $\mathcal{E}_u^{in}$. Together with (3.22), this ensures that sufficient bandwidth is provisioned throughout the network, in a routing solution that is as cost effective as possible. This constraint offers a guarantee that the same amount of traffic that leaves a cloud provider reaches the client that it was intended for. Thus, demand is met and routing is optimized.

3.2.2 Deterministic Equivalent Formulation

If a stochastic program has finite support, the set of different uncertainty scenarios can be enumerated. By formulating the scenarios as another set in the domain, and their probabilities as weights for the cost coefficients in the objective function, I can formulate an equivalent deterministic optimization problem that can be solved using established methods. This is useful as the program can be solved as a mixed integer or linear program. However, even for a small number of demand scenarios, the size of the problem space can grow rapidly. Thus, the tractability of this method needs to be enhanced. Therefore, the virtue of Scenario Tree Reduction becomes apparent, in making a reasonable tradeoff between optimality and complexity. In the following section, I introduce this technique in detail.

3.3 Scenario Tree Reduction

The uncertainty introduced in Section 3.1.3 can be modeled using a tree of scenarios. Assuming that there is a finite set of possible scenarios, a tree can be drawn like the one shown in Fig. 3.2a. Each level of the tree represents a time stage in which provisioning decisions are made with each possible sequence of events being mapped as a path from the root node to the leaf nodes. Each node in the tree represents a scenario in Ω , with each node label corresponding to a scenario subscript, such as ω_2 . To solve the stochastic optimization formulated in (3.5)-(3.25), I can enumerate the set of possible scenarios and solve the problem as a deterministic mixed integer

Figure 3.2: Possible scenario tree layouts.

linear program, by assigning a probability coefficient, such as π_{w_2} , to each scenario. To enumerate each scenario, I can reformulate the tree as a fan, as shown in Fig. 3.2b. In the fan structure, each node in tree stage $t - 1$ is replicated to match the number of scenarios in time stage t . With the exception of the root node, therefore, each node has a maximum of one ancestor and one descendant, with the number of tree branches equal to the number of leaf nodes. The scope of a scenario realization is a single time stage, so a given scenario may occur in consecutive time stages. With the exception of the root node, the probability of each tree node is equal to its ancestor and descendant, found by multiplying together the probabilities of all scenarios in its branch. For simple enumeration of scenarios, I can relabel the tree as shown in Fig. 3.2c, with each node given a unique ID.

Solving a stochastic program is straightforward when the number of scenarios is small. However, the number of nodes in the tree can grow very large as the numbers of scenarios and time stages increase. In this case, the problem space can become enormous, leading to tractability problems. Thus, to improve the performance of the optimization, it is desirable to reduce the size of the scenario tree and reconstruct it as a reduced tree, similar to Fig. 3.2d.

3.3.1 Tree Reduction

In this section, I introduce an algorithm to reduce the size of the scenario tree heuristically whilst retaining a good approximation of the full tree's performance. The reduction algorithm creates two sets - Ω_t denoting the scenarios in time stage t that remain in the tree, and Φ_t containing the set of deleted scenarios from time stage t . Scenarios are deleted such that the *probability distance* measured between Ω_t and Φ_t is minimized. Different metrics can be used to measure the probability distance. In this case, I employ the Kantorovich distance [102], denoted by $\mathcal{L}[\Omega_t, \Phi_t]$, and is given as follows:

$$\mathcal{L}[\Omega_t, \Phi_t] = \sum_{\omega \in \Omega_t} \pi_{\omega} \min_{\omega' \in \Phi_t} \ell_t[\omega, \omega'], \quad (3.26)$$

where $\ell_t[\omega, \omega']$ is the probability distance between scenarios ω and ω' , and is as follows:

$$\ell_t[\omega, \omega'] = \sum_{\bar{t} \in t_0, \dots, t} \|\xi(\omega(\bar{t})) - \xi(\omega'(\bar{t}))\|^r, \quad (3.27)$$

where $\xi(\omega(\bar{t}))$ is the random vector of scenario $\omega(\bar{t})$ in time stage $\bar{t}$. Minimizing the Kantorovich distance gives an optimization problem, which provides the maximum cardinality of the removed scenario set. However, solving this can be NP-hard, which would be contrary to the objective of scenario reduction. Therefore, a heuristic algorithm can be formulated to obtain a set of removed scenarios. This algorithm offers a tradeoff between accuracy and execution time. The reduction heuristic algorithm is as follows:

- The Backward Reduction Algorithm [103], shown in Algorithm 2, is used to reduce the scenario tree. The algorithm functions as follows: given the desired number of scenarios, N_t , to be removed from time stage t , the algorithm iteratively selects a scenario for deletion. A scenario is chosen for deletion that minimizes the probability distance between the deleted scenario and the nearest remaining scenario in Ω_t . The algorithm terminates once N_t scenarios have been deleted from Ω_t and added to Φ_t .

- A second algorithm, given in Algorithm 3, is then run to redistribute the probabilities among the remaining scenarios in Ω_t . Algorithm 3 finds the closest scenario in Ω_t for each scenario in Φ_t . The probability of the scenario in Φ_t is added to the scenario in Ω_t . This is repeated N_t times.
- (3.26) is then calculated and compared to a threshold value ϵ_t . If the result is lower than the threshold, the scenario set is retained. Otherwise, N_t is reduced and the process is repeated.

Algorithm 2 Backward Reduction Algorithm

Input : Scenario set and probabilities, number of scenarios to remove, Ω_t, π_ω, N

Output: Output set of N removed scenarios, $\Phi^{[N]}$

For $v = 1$: Select an initial scenario for deletion.

forall $\hat{\omega} \in \Omega_t$ **do**

$c_{\hat{\omega}}^{[1]} \leftarrow \min_{\omega' \in \Omega_t \setminus \{\hat{\omega}\}} \ell_t[\hat{\omega}, \omega']$.

$d_{\hat{\omega}}^{[1]} \leftarrow \pi_{\hat{\omega}} c_{\hat{\omega}}^{[1]}$.

if $d_{\hat{\omega}}^{[1]} \leq d_{\omega'}^{[1]}$ **then**

$d_{\omega'}^{[1]} \leftarrow d_{\hat{\omega}}^{[1]}$.

$\omega^{[1]} \leftarrow \hat{\omega}^{[1]}$.

end

end

$\Phi^{[1]} \leftarrow \{\omega^{[1]}\}$.

for $v \leftarrow 2, N$ **do** Iteratively choose the remaining $N - 1$ scenarios to remove.

forall $\hat{\omega} \in \Omega_t \setminus \Phi^{[v-1]}$ **do**

$c_{\hat{\omega}}^{[v]} \leftarrow \min_{\omega' \notin \Phi^{[v-1]} \cup \{\hat{\omega}\}} \ell_t[\hat{\omega}, \omega']$

$d_{\hat{\omega}}^{[v]} \leftarrow \pi_{\hat{\omega}} c_{\hat{\omega}}^{[v]}$.

if $d_{\hat{\omega}}^{[v]} \leq d_{\omega'}^{[v]}$ **then**

$d_{\omega'}^{[v]} \leftarrow d_{\hat{\omega}}^{[v]}$.

$\omega^{[v]} \leftarrow \hat{\omega}^{[v]}$.

end

end

$\Phi^{[v]} \leftarrow \Phi^{[v-1]} \cup \{\omega^{[v]}\}$.

end

Algorithm 3 Probability Redistribution Algorithm

Input : Input set of deleted scenarios with their original probabilities and original scenario set, $\Phi_t, \pi_\omega, \Omega_t$

Output: Output reduced scenario set with redistributed probabilities, $\pi_{\omega'}^*, \forall \omega' \in \Omega'_t$
 $\Omega'_t \leftarrow \Omega_t \setminus \Phi_t$.

forall $\omega \in \Phi_t$ **do**

if $\omega' \in \arg \min_{\omega' \in \Omega'_t} \ell_t[\omega, \omega']$ **then**
 $\pi_{\omega'}^* \leftarrow \pi_{\omega'} + \pi_\omega$.
end

end

3.3.2 Tree Construction

The algorithm outlined above is sufficient if there is only a single time stage below the root node. With more scenarios, the tree must be reconstructed as each branch of the scenario tree must be connected to the root node, with a uniform number of levels. For leaf nodes that are removed, its ancestors in the tree are also removed. For higher levels of the tree, its predecessors are removed, as for the leaf nodes, and its descendants are assigned as descendants of the nearest remaining scenario. Probabilities are recalculated as before, by adding the probability of the removed scenario to the nearest remaining scenario's probability. The algorithm described is given in Algorithm 4. The result is a reduced scenario tree, which is quicker to traverse yet maintains a good semantic approximation of the full scenario tree.

Algorithm 4 Full Tree Construction Algorithm

Input : Input scenario tree as relabeled fan, with T time stage, as set Ω

Output: Output reconstructed reduced scenario tree as set Ω'

For $t = T$:

Perform **BackReduce**(Ω_t, N) to obtain Φ_t such that:

$$\sum_{\omega \in \phi_t} \pi_\omega \min_{\omega' \in \Omega_t / \Phi_t} \ell_t[\omega, \omega'] \leq \epsilon_t.$$

Apply **ProbRedist**(Φ_T, Ω_T) to obtain Ω'_T .

forall $\omega' \in \Phi_T$ **do** Reconstruct the tree

 | $\Omega' \leftarrow \Omega' \setminus \mathcal{F}^{[pre]}(\omega')$. where $\mathcal{F}^{[pre]}(\omega')$ is the set of predecessors of ω' .

end

for $t = T - 1, 1$ **do**

 Perform **BackReduce**(Ω_t, N) to obtain Φ_t such that:

$$\sum_{\omega \in \phi_t} \pi_\omega \min_{\omega' \in \Omega_t / \Phi_t} \ell_t[\omega, \omega'] \leq \epsilon_t.$$

forall $\omega' \in \Phi_t$ **do** Reconstruct scenario tree to reconnect predecessors and descendants

 | $\hat{\omega} \leftarrow \arg \min_{\hat{\omega} \in \Omega'_T} \ell[\omega', \hat{\omega}]$.

forall $\omega \in \mathcal{F}^{[des]}(\omega')$ **do** $\mathcal{F}^{[des]}(\omega')$ is the set of descendants of ω'

 | $\mathcal{F}'^{[pre]}(\omega) \leftarrow \mathcal{F}^{[pre]}(\hat{\omega}) \cup \{\hat{\omega}\}$.

 | $\mathcal{F}'^{[des]}(\hat{\omega}) \leftarrow \mathcal{F}^{[des]}(\hat{\omega}) \cup \{\omega\}$.

end

 | $\pi_{\hat{\omega}} \leftarrow \pi_{\hat{\omega}} + \pi_{\omega'}$.

 | $\Omega' \leftarrow \Omega' \setminus \mathcal{F}^{[pre]}(\omega')$.

end

end

3.4 Sensitivity Analysis

The purpose of stochastic programming is to find an optimal solution given uncertainty in parameters. In this thesis, I allow for uncertainty of pricing and demand. Sensitivity analysis allows the examination of a solution to discover its sensitivity to parameter changes. In [104], an analytical sensitivity analysis approach is derived for linear programming, which I employ here. I assume that VM and bandwidth demand has finite support, and can therefore treat the stochastic program outlined previously as a deterministic equivalent formulation with scenario probabilities becoming cost coefficients in the objective function. I relax the integer restriction on VM provisioning, making the primal formulation a linear program, convert any $\geq$ inequalities to be $\leq$ and replace all equalities with a pair of inequalities. The primal problem (given in (3.5)-(3.25)) is therefore of the standard form:

$$\min_x \{c^T x : Ax \geq b, x \geq 0\}, \quad (3.28)$$

and accompanying dual problem is of the form:

$$\max_{y,s} \{b^T y : A^T y + s \leq c, s \geq 0\}. \quad (3.29)$$

The dual formulation that accompanies the primal problem is given in detail in (3.30)-(3.41) as follows:

$$\begin{aligned} \max & - \sum_{r \in \mathcal{R}} \sum_{t \in \mathcal{T}} \sum_{\omega \in \Omega} W_{rt\omega}^{(3.13)} t_r - \sum_{p \in \mathcal{P}} \sum_{t \in \mathcal{T}} \sum_{\omega \in \Omega} (W_{pt\omega}^{(3.14)} t_p^{(h)} + W_{pt\omega}^{(3.15)} t_p^{(s)} + W_{pt\omega}^{(3.16)} t_p^{(n)}) \\ & + \sum_{V_i \in \mathcal{V}} \sum_{u \in \mathcal{U}} \sum_{t \in \mathcal{T}} \sum_{\omega \in \Omega} W_{iut\omega}^{(3.17)} v_{iut\omega}, \end{aligned} \quad (3.30)$$

$$\sum_{\omega \in \Omega_1} (W_{rult\omega}^{(3.12LT)} - W_{rult\omega}^{(3.12GT)}) \leq c_{rl}^{(R)}, \bar{t} = 1, \forall r \in \mathcal{R}, \forall u \in \mathcal{U}, \forall l \in \mathcal{L}, \quad (3.31)$$

$$\sum_{\omega \in \Omega_1} (W_{piukt\omega}^{(3.11LT)} - W_{piukt\omega}^{(3.11GT)}) \leq c_{ipk}^{(R)}, \bar{t} = 1, \forall p \in \mathcal{P}, \forall V_i \in \mathcal{V}, \forall u \in \mathcal{U}, \forall k \in \mathcal{K}, \quad (3.32)$$

$$\begin{aligned} -W_{rult\omega}^{(3.12LT)} + W_{rult\omega}^{(3.12GT)} + W_{rult\omega}^{(3.10)} & \leq c_{rlt\omega}^{(re)} p(\omega), \\ \bar{t} & = 1, \forall r \in \mathcal{R}, \forall u \in \mathcal{U}, \forall l \in \mathcal{L}, \forall \omega \in \Omega, \end{aligned} \quad (3.33)$$

$$\sum_{\hat{t} \in \mathcal{N}_{lt}} W_{rult\omega}^{(3.10)} \leq c_{rlt\omega}^{(re)} p(\omega), \forall r \in \mathcal{R}, \forall u \in \mathcal{U}, \forall l \in \mathcal{L}, \forall t \in \mathcal{T}_l / \{t_1\}, \forall \omega \in \Omega, \quad (3.34)$$

$$\begin{aligned} -W_{piukt\omega}^{(3.11LT)} + W_{piukt\omega}^{(3.11GT)} + W_{piukt\omega}^{(3.9)} & \leq c_{ipkt\omega}^{(re)} p(\omega), \\ \bar{t} & = 1, \forall p \in \mathcal{P}, \forall V_i \in \mathcal{V}, \forall u \in \mathcal{U}, \forall k \in \mathcal{K}, \forall \omega \in \Omega, \end{aligned} \quad (3.35)$$

$$\sum_{\hat{t} \in \mathcal{N}_{kt}} W_{piukt\omega}^{(3.9)} \leq c_{ipkt\omega}^{(re)} p(\omega), \forall p \in \mathcal{P}, \forall V_i \in \mathcal{V}, \forall u \in \mathcal{U}, \forall k \in \mathcal{K}, \forall t \in \mathcal{T}_k / \{t_1\}, \forall \omega \in \Omega, \quad (3.36)$$

$$\begin{aligned}
 -W_{rult\omega}^{(3.10)} - W_{rt\omega}^{(3.13)} - W_{rut\omega}^{(3.23LT)} + W_{rut\omega}^{(3.23GT)} &\leq c_{rlt\omega}^{(u)} p(\omega), \\
 \forall r \in \mathcal{R}, \forall u \in \mathcal{U}, \forall l \in \mathcal{L}, \forall t \in \mathcal{T}, \forall \omega \in \Omega,
 \end{aligned} \tag{3.37}$$

$$\begin{aligned}
 -W_{rt\omega}^{(3.13)} - W_{rut\omega}^{(3.23LT)} + W_{rut\omega}^{(3.23GT)} &\leq c_{rlt\omega}^{(o)} p(\omega), \\
 \forall r \in \mathcal{R}, \forall u \in \mathcal{U}, \forall l \in \mathcal{L}, \forall t \in \mathcal{T}, \forall \omega \in \Omega,
 \end{aligned} \tag{3.38}$$

$$\begin{aligned}
 -W_{piukt\omega}^{(3.9)} - d_i^{(h)} W_{pt\omega}^{(3.14)} - d_i^{(s)} W_{pt\omega}^{(3.15)} - d_i^{(n)} W_{pt\omega}^{(3.16)} + W_{uit\omega}^{(3.17)} - d_i^{(b)} W_{put\omega}^{(3.24)} \\
 \leq (c_{ipkt\omega}^{(u)} + c_{ipkt}^{(s)}(\omega) d_i^{(s)} + c_{ipkt}^{(b)}(\omega) d_i^{(b)}) p(\omega), \\
 \forall p \in \mathcal{P}, \forall V_i \in \mathcal{V}, \forall u \in \mathcal{U}, \forall k \in \mathcal{K}, \forall t \in \mathcal{T}, \forall \omega \in \Omega,
 \end{aligned} \tag{3.39}$$

$$\begin{aligned}
 -d_i^{(h)} W_{pt\omega}^{(3.14)} - d_i^{(s)} W_{pt\omega}^{(3.15)} - d_i^{(n)} W_{pt\omega}^{(3.16)} + W_{uit\omega}^{(3.17)} - d_i^{(b)} W_{put\omega}^{(3.24)} \\
 \leq (c_{ipkt\omega}^{(o)} + c_{ipkt}^{(s)}(\omega) d_i^{(s)} + c_{ipkt}^{(b)}(\omega) d_i^{(b)}) p(\omega), \\
 \forall p \in \mathcal{P}, \forall V_i \in \mathcal{V}, \forall u \in \mathcal{U}, \forall k \in \mathcal{K}, \forall t \in \mathcal{T}, \forall \omega \in \Omega,
 \end{aligned} \tag{3.40}$$

$$\begin{aligned}
 \sum_{r \in \mathcal{R}_e^{out}} \left(W_{rut\omega}^{(3.22LT)} - W_{rut\omega}^{(3.22GT)} + W_{rut\omega}^{(3.23LT)} - W_{rut\omega}^{(3.23GT)} \right) + \\
 \sum_{r \in \mathcal{R}_e^{in}} \left(-W_{rut\omega}^{(3.22LT)} + W_{rut\omega}^{(3.22GT)} \right) + \\
 \sum_{p \in \mathcal{P}_e^{out}} \left(W_{put\omega}^{(3.24)} - W_{ut\omega}^{(3.25)} \right) + \sum_{u \in \mathcal{U}_e^{in}} W_{ut\omega}^{(3.25)} \leq 0, \\
 \forall e \in \mathcal{E}, \forall u \in \mathcal{U}, \forall l \in \mathcal{L}, \forall t \in \mathcal{T}, \forall \omega \in \Omega.
 \end{aligned} \tag{3.41}$$

(3.30) is the dual objective function. The dual variables are given by W , where the superscript indicates the equation number from the primal problem that the dual variable corresponds to. In the case of the equality constraints, (3.11), (3.12), (3.22), and (3.23), in the original formulation, a suffix (either LT or GT) is added to the superscript to represent the two inequality constraints ($\leq$ and $\geq$, respectively) that replace the equalities. (3.31) corresponds to the primal variables $X^{(R)}$, (3.32) to the primal variables $Y^{(R)}$. (3.33) and (3.34) correspond to the variables $X^{(re)}$. (3.33) addresses the special case of the first time stage, whilst (3.34) corresponds to all

subsequent time stages. (3.35) and (3.36) correspond similarly to the primal variables $Y^{(re)}$. (3.37)-(3.40) correspond to the primal variables X^u , X^o , Y^u , and Y^o , respectively. Finally, (3.41) is the dual equation for the primal flow variables, f . The parameter $p(\omega)$ is the probability of ω .

Given this dual formulation, from [104] I have the following rules for the shape of the optimal value function graph as the cost coefficient changes:

- The optimal value function, $f(\gamma)$, is defined as the change in value of the optimal solution as the cost coefficient of one of the variables, for example $c_{rl}^{(R)}$, varies by the offset value γ .
- The optimal value function is concave and piecewise linear.
- $\mathcal{P}_\gamma^*$ denotes the set of optimal solutions for the primal problem with the offset value γ , for example $c_{rl}^{(R)} + \gamma$. $\mathcal{P}_\gamma^*$ is constant for (γ_1, γ_2) , if $f(\gamma)$ is linear across the interval $[\gamma_1, \gamma_2]$.
- If I have γ_1 and γ_2 such that $\mathcal{P}_{\gamma_1}^* = \mathcal{P}_{\gamma_2}^*$, I can define $\bar{\mathcal{P}}^* = \mathcal{P}_\gamma^*$ for all $\gamma \in [\gamma_1, \gamma_2]$. This is a linear interval of $f(\gamma)$.

The behaviour of the optimal value function within a linear interval is now defined. To obtain sensitivity analysis results, I must simply find the breakpoints dividing the intervals from each other. I denote the optimal value by z^* , found by solving the primal or dual problem. This allows the finding of the sensitivity results for any cost coefficient. Two linear programming problems can be solved to give γ_{min} and γ_{max} , which are the lower and upper bounds, respectively, for the perturbation variable γ , such that the optimal values of the primal variables X^* and Y^* , and the dual variables, W^* remain correct. I give the standard form of the two optimization problems in (5.65) and (5.66), where $e_j = 1$ if j is the index of the c_j under analysis, else $e_j = 0$.

$$\gamma_{min} = \min\{\gamma : A^T y + s = c + \gamma e_j, b^T y = c^T x^* + \gamma x_j^*, s \geq 0\}, \quad (3.42)$$

$$\gamma_{max} = \max\{\gamma : A^T y + s = c + \gamma e_j, b^T y = c^T x^* + \gamma x_j^*, s \geq 0\}. \quad (3.43)$$

$$\begin{aligned}
 \gamma_{min} = & \min \left\{ \gamma : - \sum_{r \in \mathcal{R}} \sum_{t \in \mathcal{T}} \sum_{\omega \in \Omega} W_{rtw}^{(3.13)} t_r - \sum_{p \in \mathcal{P}} \sum_{t \in \mathcal{T}} \sum_{\omega \in \Omega} (W_{ptw}^{(3.14)} t_p^{(h)} + W_{ptw}^{(3.15)} t_p^{(s)} \right. \\
 & + W_{ptw}^{(3.16)} t_p^{(n)}) + \sum_{V_i \in \mathcal{V}} \sum_{u \in \mathcal{U}} \sum_{t \in \mathcal{T}} \sum_{\omega \in \Omega} W_{iutw}^{(3.17)} v_{iutw} \\
 = & \sum_{r \in \mathcal{R}} \sum_{u \in \mathcal{U}} \sum_{l \in \mathcal{L}} \left[c_{rl}^{(R)} X_{rul}^{(R)*} + \sum_{\omega \in \Omega} p(\omega) \left(\sum_{t \in \mathcal{T}_l} c_{rlt\omega}^{(re)} X_{rult\omega}^{(re)*} + \sum_{t \in \mathcal{T}} \left(c_{rlt\omega}^{(u)} X_{rult\omega}^{(u)*} \right. \right. \right. \\
 & \left. \left. + c_{rlt\omega}^{(o)} X_{rult\omega}^{(o)*} \right) \right] + \sum_{u \in \mathcal{U}} \sum_{V_i \in \mathcal{V}} \sum_{p \in \mathcal{P}} \sum_{k \in \mathcal{K}} \left[c_{ipk}^{(R)} Y_{uipk}^{(R)*} + \sum_{\omega \in \Omega} p(\omega) \left(\sum_{t \in \mathcal{T}_k} c_{ipkt\omega}^{(re)} Y_{uipkt\omega}^{(re)*} + \right. \right. \\
 & \left. \left. \sum_{t \in \mathcal{T}} \left(c_{ipkt}^{(s)}(\omega) d_i^{(s)} + c_{ipkt}^{(b)}(\omega) d_i^{(b)} \right) \left(c_{ipkt\omega}^{(u)} Y_{uipkt\omega}^{(u)*} + c_{ipkt\omega}^{(o)} Y_{uipkt\omega}^{(o)*} \right) \right] + \gamma X_j \right\}.
 \end{aligned} \tag{3.44}$$

In (3.44), I give an example of the objective function for finding the lower bound on the cost coefficient of one of the decision variables, X_j , which can be any of the decision variables, for example, one of the $X_{rul}^{(R)}$ variables.

(3.45) gives an example of the modified dual problem constraint in (3.31) corresponding to the coefficient under analysis, in this case for one of the coefficients of $X_{rul}^{(R)}$. In practice, when implementing the sensitivity analysis, the other dual constraints can be left in their original form with no modification required. The constraint is as follows:

$$\sum_{\omega \in \Omega_1} W_{rul\omega}^{(3.12)} \leq c_{rl}^{(R)} + \gamma e_j, \bar{t} = 1, \forall r \in \mathcal{R}, \forall u \in \mathcal{U}, \forall l \in \mathcal{L}, \tag{3.45}$$

where, as in (5.65) and (5.66), $e_j = 1$ if j denotes the variable whose coefficient is under analysis, else $e_j = 0$.

These optimization problems can be encoded for a solver such as CPLEX [105], and solved as a linear program. A constraint of this method is that it is limited to modifying a single variable at a time. When studying the cost coefficient of a decision variable in the deterministic linear programming equivalent formulation of our stochastic program, it is also necessary to divide by the probability weight of the variable, to find the true cost sensitivity range, after the linear programs have been solved. Nonetheless, despite these restrictions, this remains a useful technique for

examining the tolerance of the stochastic program solution, as well as the interaction between decision variables.

3.5 Performance Evaluation

3.5.1 Parameter Settings

3.5.1.1 VMs and Cloud Providers

In our test environment I consider three distinct VM applications, representing three classes. The computing requirements for $V1$, $V2$, and $V3$ are 24, 24, and 48 CPU-hours per day, respectively. The daily storage requirements for storage are 160, 410, and 840 GBs for $V1$, $V2$, and $V3$, respectively. Daily internal network requirements are 5 Mbps, 10 Mbps, and 100 Mbps for $V1$, $V2$, and $V3$, respectively. These values are selected to be reasonable choices in keeping with present computer specifications, reflecting the different instance sizes offered by cloud providers. Our simulated environment is divided into three distinct geographic locations in the UK: North, South, and West. I include four cloud providers, all four are large scale commercial providers, and thus can be considered exempt from capacity constraints. Cloud providers offer two contract lengths: 1 hour, and 2 hours. These time periods are shorter than a real application, which would offer reservation periods of several months or even a year, but I use these time periods as it allows the effect of various parameter changes to be seen more clearly. Even at this short reservation period, it is possible to see the benefits of stochastic optimization over alternative methods, an effect that should become more pronounced at longer intervals.

3.5.1.2 Network Provider Routers

our test environment contains 9 routers, arranged according to Fig. 3.1. Routers belong to one of two classes, with gateway routers connecting regions having higher capacity than those within regions. The capacities of the router classes are 50 Mbps for regional routers, and 100 Mbps for gateway routers. The price of provisioning bandwidth from each geographic region is different, with bandwidth also obtainable under two contracts, as with VMs.

3.5.1.3 Clients

I allow for two clients, one based in the North, the other in the West. Each client has their own VM demands. For simplicity of illustration, I make the number of required VMs from each class the same for each client and cloud provider in a given demand realization.

3.5.1.4 Uncertainty Parameters

I base the uncertainty of VM demand on test data obtained from Google Cluster Trace files [106], as shown in Fig. 3.3. Bandwidth demand for V_1 and V_2 is reasonably synthesized based on IP traces for web traffic [107], with two demand levels for each class. Bandwidth demand for V_3 is based on usage estimates for Netflix Video on Demand [108], representing a different class of network traffic. Demand for class V_3 varies from 1 to 5, and 100 to 500 (in units of 100) for classes V_1 and V_2 (representing more popular but less bandwidth-intensive activities), for each client on a cloud provider. For simplicity, for a given realization, VM demand level is taken to be equivalent for each VM class. For example, if the demand for V_3 is 1, the demand for V_1 and V_2 will be 100. Prices for VMs are given in Table 3.2, listing upfront costs for reservation and hourly costs for utilization and on-demand. Bandwidth prices are given in Table 3.3. VM pricing varies according to the cloud provider location, with prices derived from Amazon EC2 costs. On-demand prices are identical regardless of contract length so are only stated once. Router prices in each region are listed per Mbps, and are based on average prices according to NetIndex [109]. All prices are listed in USD.

Table 3.3: Cost per Mbps of bandwidth provisioning through network provider routers

Router Region	Contract, Provisioning Phase				
	1 – hour			2 – hour	
	R	U	O	R	U
$R_1(West)$	0.110	0.038	0.625	0.213	0.029
$R_2(North)$	0.120	0.025	0.500	0.233	0.021
$R_3(South)$	0.154	0.013	0.375	0.296	0.010

Figure 3.3: Probability distribution of VM demand test data. Each curve represents a different VM demand scenario.

3.5.2 Numerical Results

I encode the deterministic equivalent model in GAMS with the listed parameters and solve it with CPLEX [105].

3.5.2.1 Effects of Reservation

The purpose of using stochastic optimization is to achieve resource reservation that minimizes both oversubscription and undersubscription. I first vary the total quantity of reserved VMs in the system and observe the effects on total cost, shown in Fig 3.4. As expected, the first stage cost increases and the second stage cost decreases, as the increased reservation requires less on-demand provisioning. The lowest point on the Total Cost curve shows the ideal tradeoff point to allow for multiple demand scenarios. It should be noted that there is still some on-demand usage at this point, as indicated by the second stage graph, which is still decreasing. This is the consequence of accounting for differing amounts of anticipated demand. The second stage cost becomes constant as the reservation amount passes the point where all possible demand scenarios are accounted for, and the only increase in cost is due to gratuitous reservation. I observe a similar pattern for the bandwidth results, which can be found in Fig. 3.5. The difference is less dramatic due to the difference in cost and demand

values compared to VMs, but the effect is the same. I combine the two and observe a joint effect in Fig. 3.6. This figure shows the importance of optimizing both VM and bandwidth reservation, as well as showing that erring on the side of overprovisioning is preferable to underprovisioning, as the cost penalties are more severe in the latter case.

Figure 3.4: Variation of cost with VM reservation.

3.5.2.2 Importance of joint approach

The problem formulation in this thesis is based on the premise that a joint approach to VM and bandwidth allocation is superior to attempting to provision these resources separately. To test this hypothesis, I conduct two tests to compare the performance of our joint solution with a separate approach. The first test case, given in Fig 3.7, limits the CPU capacity on cloud providers P3 and P4. These two providers offer the same pricing plan and are geographically close. The joint solution is calculated and compared to an alternative solution that provisions VM and bandwidth separately. The separate solution uses expected demand values for VM demand (with bandwidth

Figure 3.5: Variation of cost with bandwidth reservation.

derived from the expected VM demand) as the basis for VM and bandwidth reservation decisions. To compare the solutions, I apply the reservation decisions of each method to the same problem. The resulting cost margin demonstrates the superior solution to the chosen problem. In this case, despite the relatively low costs of bandwidth used in our parameter settings, I find that the joint solution is clearly superior at all levels of CPU capacity. This shows that the reservation choice made in the joint solution allows the repositioning of bandwidth to match the limitations on available CPU. In contrast, the separate solution reserves bandwidth so as to minimize the expected bandwidth costs, which involves reserving bandwidth in the cheaper South region, where providers P3 and P4 are located. This reserved bandwidth becomes essentially wasted, as VM allocation switches to the other providers. The joint solution is flexible enough to reroute bandwidth reservation to avoid the redundant South region routers, and thus can minimize the increase in cost. Increasing the relative cost of bandwidth would highlight this further.

In the second case, I modify the bandwidth capacity of the gateway routers for accessing the southern region, which contains P3 and P4. I choose these routers as P3

Figure 3.6: Variation of cost with VM and bandwidth reservation.

and P4 are a popular choice for VM allocation, and limiting bandwidth access highlights the importance of a joint approach. These results are shown in Fig. 3.8. There is an initial significant drop in cost when the bandwidth capacity exceeds zero, as P3 and P4 become useable. However, particularly at low capacities, the cost difference between the joint and separate solutions are significant. In the separate solution, VMs are reserved from providers without knowledge of the bandwidth available to communicate with those providers. With this lack of information, the cheaper providers P3 and P4 are preferred in advance, but at the time of use, the reserved resources are inaccessible, leading to the difference in price. The additional knowledge of the bandwidth routing allows the circumvention of the bandwidth bottlenecks whilst minimizing the increase in cost. Contrastingly, the joint solution is less affected by the variation in demand, as it is able to rearrange VM reservation to compensate for the bandwidth bottlenecks, which the separate allocation cannot match. An increase in bandwidth demand per VM would highlight this even further.

Figure 3.7: Comparison of joint and separate provisioning solutions when provider CPU capacity varies.

Figure 3.8: Comparison of joint and separate provisioning solutions when South gateway router bandwidth capacity varies.

3.5.2.3 Scenario Tree Reduction Performance

To test the performance of the Scenario Tree Reduction algorithm outlined earlier, I format the resulting data as a new set of probability distributions and use them as the demand input to generate the reservation decisions in our stochastic programs. The algorithm can be run to different probability distance thresholds, in this case: 0.1, 0.01, and 0.001. The smaller threshold values yield probabilities that are closer to the original distribution, but even so, the differences are subtle. In Fig. 3.9 I highlight the clearest example of difference between the distributions. The subtlety of

Figure 3.9: Magnified view of changed probability distributions to highlight shape changes.

the differences demonstrates the ability of the algorithm to exclude scenarios that do not contribute significantly to the nature of the scenario tree. The statistical details of the scenario reductions are given in Table 3.4, along with figures showing the runtime and memory improvements as the tree becomes smaller. The cost rises as the tree is reduced, and thus the user must choose an acceptable tradeoff of cost and speed. However, the algorithm manages to offer scenario reduction that differs only slightly from the optimum, even at a probability distance of 0.1.

Table 3.4: Impact of scenario reduction on optimization performance

Threshold	T1 Scenarios Removed	T2 Scenarios Removed	Cost	Iterations	Memory (MB)
0.001	72	33	205.0588	29170	66
0.01	108	29	205.0667	25260	60
0.1	144	25	205.2798	21437	53

3.5.2.4 Study of bandwidth allocation under parameter changes

An interesting factor in bandwidth optimization is studying how the distribution of bandwidth varies under different scenarios and parameter settings. Similarly, it is also interesting to observe what influences bandwidth reservation decisions. In Fig. 3.10 I observe the realized bandwidth usage across the network under different VM demand

scenarios. I highlight the ‘gateway’ routers that connect geographic regions together. Allocation is particularly high for routers connecting to region 3, which contains 2 cloud providers, offering the best prices, and therefore is the most influenced by increases in VM demand. In Fig. 3.11, I alter the pricing on routers $R3$ and $R8$, the gateway between regions 1 and 3, and observe the redistribution of traffic across the other gateway routers to compensate for the increased cost. If the implementation was a separate one, it would be logical to expect more traffic through the North region. However, the joint approach means that much of each user’s demand is relocated to their own local providers, as evidenced by the rise in $R1$, which is adjacent to the West cloud provider. This responsiveness to price change is important, and shows that well-placed reservations can compensate for increased costs. It also highlights the importance of a joint approach for finding the optimal cost.

Figure 3.10: Effect of varying VM demands on network router bandwidth allocation across the network.

Figure 3.11: Effect of price change on bandwidth allocation to gateway router between network regions 3 and 1.

3.5.2.5 Performance of alternative methods under different probability distribution shapes

Stochastic optimization is a more complex method than some alternatives. Thus, to demonstrate its usefulness, I compare it against alternative methods. The first alternative is an on-demand only method, with no reservation, representing an extreme case of undersubscription. The second alternative is a simple Expected Value Function (EVF) - using the mean of the probability distribution as the reservation amount. This is an important comparison, as stochastic optimization would be needlessly complex if it could not improve on EVF. Finally I take the 0.1 threshold scenario tree reduction solution to compare the reduced tree's performance. I expect stochastic optimization to be most effective. However, to prove that it is the best method, I demonstrate that it works even when the input parameters change. In Fig. 3.12, I change the variance of the input scenario distribution and compare the performance of the four methods. The patterns are the same with some interesting caveats. The exclusively on-demand approach is largely immune to the change in distribution, as it is purely

reactive, and is thus not influenced by probability. EVF performs markedly better with a narrower variance, as the expected value is closer to the possible outcomes. Surprisingly, however, it is noticeably worse than the on-demand only option. This is a consequence of the probability distribution, with the mean allocation resulting in oversubscription. This proves the necessity of a more sophisticated approach. Given the scenario tree reduction results already highlighted, it is not surprising that the reservation based on a reduced tree performs very close to the level of the full size stochastic optimization.

Figure 3.12: Total cost of different solution methods under different demand probability distributions.

3.5.2.6 Benefits of sensitivity analysis

I test our sensitivity analysis method by examining price sensitivity of VMs. To examine the effects of changing cost on reservation, I define a VM demand for each user on each provider. By using the sensitivity analysis as outlined in Section 3.4, I can find the precise cost thresholds that trigger reservation changes. This result is given in Fig. 3.13. Sensitivity analysis ensures I find exact price points without having

to make unnecessary measurements. This graph can be reversed to find the upper and lower cost bounds for the whole provisioning range of a variable.

I also test the relevance of sensitivity analysis for bandwidth allocation. Close examination of the bandwidth pricing in Table 3.3 reveals that the longer contract, *L2*, is undesirable. This contract should be more expensive to reserve, but reduce long term costs, thus rewarding forward planning. However, it is apparent that the prices are currently too high, making contract *L1* more desirable. Examination of the optimization results supports this observation. I can employ sensitivity analysis to find the necessary reduction in cost for *L2*, such that it becomes financially viable again. In Fig. 3.14, I examine the router prices from each location and contrast the current, poorly chosen, reservation cost for *L2*, with the new, optimal reservation cost found through sensitivity analysis. Thus I can see that sensitivity analysis has useful potential applications not only for the client, but also for the cloud provider, who can ensure that their profits are optimized.

Figure 3.13: Precise cost coefficient thresholds triggering VM reservation changes.

3.6 Conclusion

In this chapter, I have presented a stochastic programming formulation to optimally reserve virtual machines and bandwidth in a cloud computing environment. The formulation makes optimal decisions to reserve resources across multiple time stages

Figure 3.14: Full price of contract $L2$ in each network region, with the price needed for the contract to become useful.

despite uncertain demand. The joint VM and bandwidth provisioning is shown to be necessary, as each is interdependent. In a real world application, the problem space can quickly become very large, so I have applied a scenario tree reduction algorithm to find a reasonable heuristic that can be solved efficiently. I have found that whilst reducing the scenario tree increases total cost, the heuristic retains enough accuracy to give a desirable cost-performance tradeoff. Finally I have performed a sensitivity analysis of the stochastic programming problem to examine the solution's tolerance to parameter change. The optimization has sizeable intervals, and I have shown that sensitivity analysis is useful for providers in setting system parameters, as well as for clients. This chapter has shown that stochastic optimization may be effectively applied to guaranteeing the performance of cloud applications, in the following chapter, I will address how it may be employed to save mobile users' battery life through application migration.

Table 3.2: VM costs for each cloud provider and provisioning phase.

Provider	VM, Contract, and Phase											
	V_1			V_2			V_3					
	$1 - hour$			$1 - hour$			$1 - hour$			$1 - hour$		
	R	U	O	R	U	O	R	U	O	R	U	O
P_1	0.013	0.077	0.154	0.025	0.154	0.308	0.031	0.126	0.616	0.050	0.308	0.063
P_2	0.014	0.087	0.158	0.028	0.175	0.315	0.036	0.139	0.630	0.056	0.349	0.071
P_3	0.013	0.064	0.113	0.025	0.127	0.225	0.031	0.101	0.450	0.050	0.254	0.063
P_4	0.013	0.064	0.113	0.025	0.127	0.225	0.031	0.101	0.450	0.050	0.254	0.063

Chapter 4

Bring-Your-Own-Application (BYOA): A Scalable Stochastic Mobile Application Migration Service

The previous chapter guaranteed cloud performance through optimizing the offloading of computation in the form of VMs provisioned jointly with bandwidth. However, this scenario is primarily of benefit to static users with no battery concerns. For mobile users keen to preserve their battery life, I now consider a scenario where the mobile user can offload to local equipment, as an alternative to the remote cloud. The novel ‘Bring-Your-Own-Application’ model and the application of stochastic optimization to the scenario is introduced in this chapter as a new solution to this problem. I consider this model to be the next step in mobile cloud technology, as established in the previous chapters. There is sufficient existing technology to enable mobile application migration, and companies such as Microsoft and Google are keen to provide a seamless computing experience, allowing users to offload to local equipment while on the move is a logical way to do this. The chapter is organised as follows. Section 4.1 elaborates on the scenario design and lays out the key elements of the proposed environment. Section 4.2 provides the formulation of the stochastic optimization problem, accounting for all relevant factors. Section 4.3 introduces the partial Lagrange multiplier method, laying out the mathematical principles of total unimodularity and giving the algorithm to find the optimal solution. Section 4.4 contains a theoretical performance analysis, proving the convergence of the algorithm, and showing mathematically that

the method offers polynomial execution time. In Section 4.5, I present an analytical sensitivity analysis approach, finding two linear programs that give the sensitivity range of my formulation to parameter changes. Finally, Section 4.6 sets out a plausible example scenario based on real trace data to explore the behaviour of the scenario and demonstrate the effectiveness of the partial Lagrange multiplier method.

4.1 System Model

I consider an environment, with an example illustrated in Fig. 4.1 in which users are able to travel to locations (such as a library or coffee shop) whilst working on their mobile devices. At these locations, the user may choose to rent a nearby piece of equipment that is mains-powered (such as a desktop or smart-television), and offload their computation to it by submitting a request to the service broker. In so doing they preserve the battery life and data usage of their own mobile device, or enjoy the greater usability of a larger screen and more powerful computer. Offloading a virtual machine provides an isolated environment [110], protecting the user from malware on the target OS. Authentication is done using MDM through an app provided by the service broker. The user’s device communicates with an MDM server, such as VMWare’s Airwatch [111], via a VPN, such as Pulse Secure Connect [112], which authenticates the user, permitting migration. In addition to the location the user arrives at, other locations are available, and the broker serves as an intermediary between the individual providers at each location, and the user. The broker seeks to maximize the profits of the service providers, by maximizing equipment utilization and minimizing user rejection across all locations. The MDM server also connects locations, allowing a user to authenticate easily at new locations. BYOD security is beyond the scope of this thesis, but provides a viable basis for my system model.

The broker must consider a number of factors when allocating users. Allowance must be made for future demand, as the allocation takes place over multiple time stages. Users arrive with varying app requirements, needing compatible equipment, different battery charge levels, and usage duration, as well as start location and total

Figure 4.1: Illustration of multi-location system model and intra-location equipment allocation.

user numbers. If a user cannot be offered equipment at their starting location, they may be migrated physically to an alternative location, incurring a cost and battery penalty, but improving overall equipment utilization. In the event that there is no available equipment, or a user's battery will not last until they can reach an alternative location, they must be turned away and are considered 'lost'. In addition to the migration of users by the broker, users may choose to migrate themselves, as their needs change over time (for example, moving to a coffee shop for a 'working lunch'). This migration is treated as effectively instantaneous, as equipment is reserved for the user at the new location, and they can continue working on the go, however the migration does incur a battery charge penalty.

4.1.1 Uncertainty Parameters

In a real environment, there are many uncertainties regarding future requirements, which makes decision-making that considers future situations essential. In my environment, I consider a number of uncertain factors. In each time stage, the number of users arriving at each location, the duration of usage of each user, the battery charge of a user's mobile device, their app requirements (and consequently network bandwidth, equipment type, and computation power requirements), and whether they choose to move location of their own free will, are all uncertain factors. The uncertainties for an individual user are defined as a tuple, (i, p, l, b, k) , where $i \in \mathcal{I}$ is the user index, $p \in \mathcal{P}$ is the starting location, $l \in \mathcal{L}$ is the length of usage, $b \in \mathcal{B}$ the initial battery charge, and $k \in \mathcal{K}$ the application requirement (thus indicating the type of equipment required). The scenario, ω , is then defined as a set of user tuples, with $|\omega|$ defined by the number of users in that scenario. The set of all scenarios, Ω , is the set of all scenario sets ω . The maximum size of Ω is therefore the combination of all possible user parameters, for all possible users, given by the Cartesian product: $\mathcal{I} \times \mathcal{P} \times \mathcal{L} \times \mathcal{B} \times \mathcal{K}$. In reality this size will be reduced to allow for a smaller number of distinct user demand numbers. Fig. 4.2 illustrates user arrival and duration across time stages at a single location with distinct user demand numbers. Given all these uncertainties, I propose a stochastic optimization formulation, which considers future uncertainties when choosing which equipment to allocate a user to or whether to reject them. The optimization may choose to allocate the first arriving user, but by considering later demand, it may rather choose to reject the user considering the probability that a user who intends to stay for a longer time may come after, or who has more specific equipment requirements, and may be rejected if their equipment is unavailable.

The notations used for all these elements in the system model are given in Table 4.1.

Figure 4.2: Illustration of user allocation at time stages and uncertainty scenarios for an individual location.

4.2 Problem Formulation

To maximize the profit from user allocation whilst considering the multiple uncertainties, I formulate a multi-stage stochastic optimization problem. The goal of this optimization is to find an optimal matching between users as they arrive at a location and a compatible piece of computing equipment. A user is treated as unique and is modeled individually, allocated to a single piece of equipment. When a user arrives at a location, they are either allocated to a piece of equipment, migrated to another location, or rejected. Users pay a fee to use the equipment, and the objective of this formulation is to maximize the profit from users, and therefore maximize the number of users who find equipment. In each time period, the number of users allocated to devices is decided with consideration to the expected number of future users. The optimality of the solution is therefore dependent on realistically defining user arrival probabilities and their properties (equipment type, battery level, duration of stay), for example, in this thesis I use movement traces of students around a university campus to determine arrival probabilities. To minimize the number of rejected and migrated

Table 4.1: List of Key Notations for Chapter 4.

Symbol	Definition
$\mathcal{I}$	Set of all users, while $i \in \mathcal{I}$ denotes the user index.
$\mathcal{J}$	Set of all equipment classes that can be offloaded to. $j \in \mathcal{J}$ denotes the equipment class index.
$\mathcal{T}$	Set of all time periods, while $t \in \mathcal{T}$ denotes the time period index.
$\mathcal{P}$	Set of all physical locations, while $p \in \mathcal{P}$ denotes the location index.
$\mathcal{L}$	Set of all possible user durations, while $l \in \mathcal{L}$ denotes the duration index.
$\mathcal{K}$	Set of all possible application types, while $k \in \mathcal{K}$ denotes the application index.
$\mathcal{B}$	Set of all initial user battery charge levels, while $b \in \mathcal{B}$ denotes the battery charge index.
$F_{ij}(\omega)$	Binary compatibility switch, indicating compatibility between user's application and offloading equipment.
$D_i^{(n)}(\omega), D_i^{(b)}(\omega), D_i^{(m)}(\omega)$	Demand parameters, for network and battery for a user's application, and the battery migration cost for user i .
$D_{it}(\omega), D_{ipt}(\omega)$	Binary demand value for user i and the equivalent value by provider for Case 2.
$Q^{(n)}, Q_i^{(b)}(\omega)$	Capacity of network, user's mobile device battery under scenario ω .
Q_j, Q_{jp}	The amount of available equipment of class j , and the equivalent capacity for each provider for Case 2.
B_j	Binary parameter indicating whether equipment class j uses mobile battery.
$C_{jp}, C_i, C_i^{(m)}$	Equipment revenue, and cost of losing users and migrating users, respectively.
$x_{ijpt}(\omega)$	Decision variables for equipment allocation.
$z_{it}(\omega), m_{it}(\omega)$	Penalty variables for lost users and user migration.
$\vec{\lambda}, \lambda_{ptw}^1, \lambda_{ijtw}^2, \lambda_{itw}^3, \lambda_{iptw}^4$	Denotes vector notation and corresponding equation representations of Lagrange multipliers.
$W_{jptw}^{(4.4)}$	Example dual variable used for sensitivity analysis. Superscript indicates the corresponding equation number.

users, I include cost penalties in the objective function (4.1). These are not real financial costs, but rather represent the undesirable consequences of not serving users. These costs should be chosen appropriately, with migration incurring a lower penalty, as it is still preferable to rejection. I assume that location migration takes place within a time stage, with all locations equidistant from one another, thus the user may continue working while roaming, so a user does not experience downtime while

working, however it does incur a battery penalty. The objective function maximizes total profit and minimizes penalty costs, as shown in (4.1). The other constraints in this problem formulation are designed to enforce that each user is accounted for by a decision variable (assigned, rejected, or migrated), and users are only allocated to available, compatible equipment.

$$\max \mathbb{E}_{\Omega} \left[\sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} \sum_{p \in \mathcal{P}} \sum_{t \in \mathcal{T}} (C_{jp} x_{ijpt}(\omega) - C_i z_{it}(\omega) - C_i^{(m)} m_{it}(\omega)) \right]. \quad (4.1)$$

The objective function has three decision variable types. $x_{ijpt}(\omega)$, represents the binary allocation of a user, i , to an equipment class, j , at a particular location, p , and time, t . $z_{it}(\omega)$, a binary variable that alternatively indicates if a user is ‘lost’, and $m_{it}(\omega)$ for users that are physically migrated, and allocated to equipment at an alternative location to the one the user arrives at. Each physical location has a wireless network, which is used by devices for migration and their cloud activity. The limit on this bandwidth is modeled in (4.2), where the total network usage by a user-equipment allocation, with $D_i^{(n)}(\omega)$ as the bandwidth needed by a user, must total less than the location’s bandwidth capacity, $Q_p^{(n)}$.

$$\sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} D_i^{(n)}(\omega) x_{ijpt}(\omega) \leq Q_p^{(n)}, \quad \forall p \in \mathcal{P}, \forall t \in \mathcal{T}. \quad (4.2)$$

The battery levels of all users must remain non-zero regardless of where they are in the environment, as enforced in (4.3). Some equipment still requires the user to power their mobile device, and so incur battery cost, $D_i^{(b)}(\omega)$, whilst others do not, and therefore conserve energy, determined by the binary parameter, B_j . The battery cost of any migration, $D_i^{(m)}(\omega)$, must also be factored in, when considering the user’s charge level. If the user battery cannot last for the duration of their stay, $\mathcal{T}_{i\omega}$, the user is lost.

$$\sum_{t \in \mathcal{T}_{i\omega}} \left(\sum_{p \in \mathcal{P}} (B_j D_i^{(b)}(\omega) x_{ijpt}(\omega)) + D_i^{(m)}(\omega) m_{it}(\omega) \right) \leq Q_i^{(b)}(\omega), \quad \forall i \in \mathcal{I}, \forall j \in \mathcal{J}. \quad (4.3)$$

Each physical location has a limited number of each equipment type (equipment class). At each location, the total number of users allocated to each class cannot exceed the supply of that equipment, Q_{jp} ; this is enforced by (4.4).

$$\sum_{i \in \mathcal{I}} x_{ijpt}(\omega) \leq Q_{jp}, \quad \forall j \in \mathcal{J}, \forall p \in \mathcal{P}, \forall t \in \mathcal{T}. \quad (4.4)$$

Meeting demand as far as possible is essential, but cannot exceed the number of users in the system. Thus, (4.5) prevents the allocated user numbers exceeding the demand, $D_{ipt}(\omega)$, (i.e. allocating fictional users to increase profit), as this should be impossible. Conversely, (4.6) ensures that all user demand is either met by equipment allocation at one of the physical locations, or is accounted for as a lost user.

$$\sum_{j \in \mathcal{J}} \sum_{p \in \mathcal{P}} x_{ijpt}(\omega) \leq \sum_{p \in \mathcal{P}} D_{ipt}(\omega), \quad \forall i \in \mathcal{I}, \forall t \in \mathcal{T}, \quad (4.5)$$

$$\left(z_{it}(\omega) + \sum_{j \in \mathcal{J}} \sum_{p \in \mathcal{P}} x_{ijpt}(\omega) \right) \geq \sum_{p \in \mathcal{P}} D_{ipt}(\omega), \quad \forall i \in \mathcal{I}, \forall t \in \mathcal{T}. \quad (4.6)$$

If a user cannot be allocated at the location they arrive at, they must be either rejected or migrated to another location. This is enforced by (4.7), which considers the demand at individual locations. This works in conjunction with (4.6), which ensures that users that are migrated are not lost between locations.

$$m_{it}(\omega) + z_{it}(\omega) + \sum_{j \in \mathcal{J}} x_{ijpt}(\omega) \geq D_{ipt}(\omega), \quad \forall i \in \mathcal{I}, \forall p \in \mathcal{P}, \forall t \in \mathcal{T}. \quad (4.7)$$

Different users have different app requirements, and since not all equipment is compatible with all apps or is powerful enough to run all software, I must check that each user-equipment allocation is legitimate, controlled by the user-equipment binary parameter $F_{ij}(\omega)$. This is done in (4.8). This avoids the scenario of low-computation apps being run on high-powered equipment at the expense of high-computation apps while low-powered machines are idle.

$$x_{ijpt}(\omega) \leq F_{ij}(\omega), \quad \forall i \in \mathcal{I}, \forall j \in \mathcal{J}, \forall p \in \mathcal{P}, \forall t \in \mathcal{T}. \quad (4.8)$$

Each user is modeled individually, with each decision variable corresponding to an individual. Thus, user allocation is binary, as given in (4.9), making my formulation a binary optimization problem.

$$x_{ijpt}(\omega), z_{it}(\omega), m_{it}(\omega) \in \{0, 1\}. \quad (4.9)$$

4.2.1 Deterministic Equivalent Formulation

To solve a stochastic optimization using conventional optimization methods, I assume that the problem has finite support, a reasonable assumption in this case as all parameters are naturally discrete or can be reasonably discretized. The formulation can then be converted to a deterministic formulation, with the optimization being calculated for every scenario, thus $x_{ijpt}(\omega)$ becomes $x_{ijpt\omega}$, for example, and a probability weighting is attached to the revenue/cost coefficients, denoted by $\pi(\omega)$. It must be assumed that the probability distribution is known in advance. Then the solution can be calculated by solving as an integer program by a technique such as the branch-and-bound algorithm. It is therefore vital for $\pi(\omega)$ to be accurately determined, for the result to be a good reflection of the real world. For example, in this thesis, I use past user behavioural data to create a custom distribution for my experimental results. The deterministic equivalent formulation is given as follows:

$$\max \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} \sum_{p \in \mathcal{P}} \sum_{t \in \mathcal{T}} \sum_{\omega \in \Omega} \pi(\omega) (C_{jp} x_{ijpt\omega} - C_i z_{it\omega} - C_i^{(m)} m_{it\omega}). \quad (4.10)$$

The constraints given below are the equivalents of the constraints in the stochastic formulation, but are now evaluated across all scenarios, $\omega \in \Omega$.

$$\sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} D_{i\omega}^{(n)} x_{ijpt\omega} \leq Q_p^{(n)}, \quad \forall p \in \mathcal{P}, \forall t \in \mathcal{T}, \forall \omega \in \Omega. \quad (4.11)$$

$$\sum_{t \in \mathcal{T}_{i\omega}} \left(\sum_{p \in \mathcal{P}} (B_j D_{i\omega}^{(b)} x_{ijpt\omega}) + D_{i\omega}^{(m)} m_{it\omega} \right) \leq Q_{i\omega}^{(b)}, \quad \forall i \in \mathcal{I}, \forall j \in \mathcal{J}, \forall \omega \in \Omega. \quad (4.12)$$

$$\sum_{i \in \mathcal{I}} x_{ijpt\omega} \leq Q_{jp}, \quad \forall j \in \mathcal{J}, \forall p \in \mathcal{P}, \forall t \in \mathcal{T}, \forall \omega \in \Omega. \quad (4.13)$$

$$\sum_{j \in \mathcal{J}} \sum_{p \in \mathcal{P}} x_{ijpt\omega} \leq \sum_{p \in \mathcal{P}} D_{ipt\omega}, \quad \forall i \in \mathcal{I}, \forall t \in \mathcal{T}, \forall \omega \in \Omega, \quad (4.14)$$

$$z_{it\omega} + \sum_{j \in \mathcal{J}} \sum_{p \in \mathcal{P}} x_{ijpt\omega} \geq \sum_{p \in \mathcal{P}} D_{ipt\omega}, \quad \forall i \in \mathcal{I}, \forall t \in \mathcal{T}, \forall \omega \in \Omega. \quad (4.15)$$

$$m_{it\omega} + z_{it\omega} + \sum_{j \in \mathcal{J}} x_{ijpt\omega} \geq D_{ipt\omega}, \quad \forall i \in \mathcal{I}, \forall p \in \mathcal{P}, \forall t \in \mathcal{T}, \forall \omega \in \Omega. \quad (4.16)$$

$$x_{ijpt}(\omega) \leq F_{ij}(\omega), \quad \forall i \in \mathcal{I}, \forall j \in \mathcal{J}, \forall p \in \mathcal{P}, \forall t \in \mathcal{T}, \forall \omega \in \Omega. \quad (4.17)$$

$$x_{ijpt\omega}, z_{it\omega}, m_{it\omega} \in \{0, 1\}. \quad (4.18)$$

4.3 Partial Lagrange Multiplier Method

The deterministic equivalent formulation of my stochastic optimization is a binary optimization problem. This can be solved by binary optimization solution methods, such as branch-and-bound, which has exponential time complexity except in certain special cases. Consequently, it is important to find a way to improve performance for large scale problems (such as large numbers of users or scenarios). In this section I introduce a partial Lagrange multiplier method that uses the subgradient method to solve an LP in each iteration with a worst-case polynomial limit on the number of iterations.

4.3.1 Total Unimodularity of Constraint Matrix

The constraints in (4.11)-(4.17) can be expressed as a matrix. If it satisfies the total unimodularity property, I can relax the optimization to a linear problem and obtain the optimal solution, with the decision variables returning integer values. However,

in my case, the whole matrix does not satisfy this property, but I can establish that if constraints (4.11), (4.12), (4.15), and (4.16) are removed I have a constraint matrix where total unimodularity holds. I can demonstrate this, for example, by partitioning the rows of the constraint matrix into those corresponding to (4.13) and (4.17) and those corresponding to (4.14). By Theorem 5 in [29], one can see easily that the constraint matrix is unimodular, and that this property will hold for every square, non-singular submatrix. Thus total unimodularity holds, and this result is essential for the method outlined here.

4.3.2 Partial Lagrange Multiplier

The full Lagrange multiplier method is commonly employed to solve convex optimization problems, expressed in vector notation as follows:

$$L(\vec{x}, \vec{\lambda}, \vec{v}) = f_0(\vec{x}) + \sum_{i=1}^m \lambda_i f_i(\vec{x}) + \sum_{i=1}^p v_i h_i(\vec{x}), \quad (4.19)$$

where λ and v are the Lagrange multipliers. The dual function of the Lagrange is then given by:

$$g(\vec{\lambda}, \vec{v}) = \min_{\vec{x} \in \mathbb{R}^n} L(\vec{x}, \vec{\lambda}, \vec{v}). \quad (4.20)$$

The corresponding dual problem is therefore:

$$\max_{\vec{\lambda}, \vec{v}} g(\vec{\lambda}, \vec{v}) \mid \lambda \geq 0. \quad (4.21)$$

The full Lagrange multiplier method is commonly used for convex optimization problems, and the dual problem is therefore concave. The two problems will typically have the same optimal solution, with the dual problem efficiently solvable by the sub-gradient method. The partial total unimodularity of my problem formulation enables us to exploit this, by moving the constraints that do not satisfy total unimodularity into the objective function using the Lagrange multiplier. To match the standard form given above, I convert the maximization problem in (4.10)-(4.18) to a minimization problem. The dual function corresponding to (4.20) then becomes:

$$\begin{aligned}
g(\vec{\lambda}) = \min & - \sum_{i,j,p,t,\omega} \pi(\omega) (C_{jp} x_{ijpt\omega} - C_i z_{it\omega} - C_i^{(m)} m_{it\omega}) \\
& + \sum_{p,t,\omega} \lambda_{pt\omega}^1 \left(\sum_{i,j} D_{i\omega}^{(n)} x_{ijpt\omega} - Q_p^{(n)} \right) \\
& + \sum_{i,j,t_{i\omega},\omega} \lambda_{ijt\omega}^2 \left(\sum_{p \in \mathcal{P}} (B_j D_{i\omega}^{(b)} x_{ijpt\omega}) + D_{i\omega}^{(m)} m_{it\omega} - Q_{i\omega}^{(b)} \right) \\
& - \sum_{i,t,\omega} \lambda_{it\omega}^3 \left(z_{it\omega} + \sum_{j,p} x_{ijpt\omega} - \sum_p D_{ipt}(\omega) \right) \\
& - \sum_{i,p,t,\omega} \lambda_{ipt\omega}^4 \left(m_{it\omega} + z_{it\omega} + \sum_j x_{ijpt\omega} - D_{ipt\omega} \right). \tag{4.22}
\end{aligned}$$

For the sake of space and readability, a shorthand notation is used for the summations, with, for example, $\sum_{i,p}$ representing $\sum_{i \in \mathcal{I}} \sum_{p \in \mathcal{P}}$. This notation allows the shortening of already lengthy equations whilst remaining clear.

4.3.3 Solution to the Partial Lagrange Multiplier

To find the optimum value of my original problem formulation, expressed in (4.10)-(4.18) I must first solve the dual problem, as in (4.21), by iteratively maximizing the dual function in (4.22), updating each λ value to reach the optimal solution:

$$\max g(\vec{\lambda}). \tag{4.23}$$

Subject to (4.13), (4.14), and (4.17), with the additional restriction that $\vec{\lambda} \geq 0$, and relaxed range constraints on all decision variables:

$$0 \leq x_{ijpt}(\omega), z_{it}(\omega), m_{it}(\omega) \leq 1. \tag{4.24}$$

This method is shown in Algorithm 1, solving the dual problem, computing the corresponding solution to the primal problem, and updating the Lagrange multipliers.

4.3.3.1 Dual Problem Solution

The optimization problem in (4.22) can be solved as an LP subject to the constraints (4.13), (4.14), (4.17), and (4.24). These constraints completely satisfy total unimodularity, and therefore the solution to the LP is guaranteed to give integer

decision variable values. As a result, it is solvable in polynomial time, using the Interior Point Method (IPM) [1]. Each iteration of the partial Lagrange multiplier method solves the dual problem using the updated Lagrange multipliers. The decision variables from this result can then be used to find the value of the original primal objective (4.10), which is then used to update the Lagrange multiplier values.

4.3.3.2 Updating Lagrange multipliers

The iterative updates of the solution are governed by the calculation of the Lagrange multipliers for the following iteration. For any given step, k , I calculate the following step's multipliers with the following equation (written in vector notation):

$$\vec{\lambda}^{(k+1)} = \vec{\lambda}^{(k)} + \alpha_k \cdot \nabla g(\vec{\lambda}^{(k)}). \quad (4.25)$$

The step size α_k can be calculated in various ways, with a diminishing stepsize being ideal for guaranteeing convergence to the optimum. If the dual function is differentiable (as ours is), a constant stepsize will also converge to the optimum, and may do so more quickly, assuming the constant is small enough [113]. I perform a practical comparison of step size choices in Section 4.6.2.10. The function $\nabla g(\vec{\lambda}^{(k)})$ can be derived from equation (4.22) and is defined as:

$$\begin{aligned} \nabla g(\vec{\lambda}) = & \sum_{p,t,\omega} \left(\sum_{i,j} D_{i\omega}^{(n)} x_{ijpt\omega}^{(k)} - Q_p^{(n)} \right) + \sum_{i,j,t_{i\omega},\omega} \left(\sum_{p \in \mathcal{P}} (B_j D_{i\omega}^{(b)} x_{ijpt\omega}^{(k)} + D_{i\omega}^{(m)} m_{it\omega}^{(k)} - Q_{i\omega}^{(b)}) \right. \\ & \left. - \sum_{i,t,\omega} \left(z_{it\omega}^{(k)} + \sum_{j,p} x_{ijpt\omega}^{(k)} - \sum_p D_{ipt\omega} \right) - \sum_{i,p,t,\omega} \left(m_{it\omega}^{(k)} + z_{it\omega}^{(k)} + \sum_j x_{ijpt\omega}^{(k)} - D_{ipt\omega} \right) \right). \end{aligned} \quad (4.26)$$

Finally, I require that each instance of λ must be non-negative, and is defined as the larger of the calculated update and 0.

4.3.3.3 Stopping Criterion

In the following section, I prove the convergence of the partial Lagrange multiplier method. In order to stop the algorithm, I keep a record of the best valid solution. If there is not a significant improvement in the best value after Φ iterations, I may

terminate the algorithm and take the current best solution. The value of Φ and the definition of ‘significant’ should be chosen to trade off running time and solution quality.

Algorithm 1 Partial Lagrange Multiplier Method

Input : Dual problem $g(\vec{\lambda})$, Φ

Output: Optimal solution values for x , m , z

Initialize $g_{best}^{(0)} = -\infty$;

Set the iteration index: $k = 1, k_x = 0$;

Initialize $\vec{\lambda}$ values e.g. $= 0.5$;

do

 Find optimal solution for $g(\vec{\lambda})$, given in Eq. (4.22) ;

 Compute solution of primal problem ;

if $g_{best}^{(k)} - g_{best}^{(k-1)} \geq \epsilon$ **then** $k_x = k$;

$k = k + 1$;

 Compute step size α_k ;

 Update Lagrange multipliers using Eq. (4.25) and (4.26);

while $k - k_x \geq \Phi$;

4.4 Theoretical Performance Analysis

In this section I analyse the convergence performance of the partial Langrange multiplier algorithm and its scalability.

4.4.1 Convergence Analysis

I first show that the minimization problem’s dual function, $g(\vec{\lambda})$, is concave. Given this concavity, I can prove that it converges within a finite number of iterations. I then show by strong duality that the solution at the point of convergence is optimal.

4.4.1.1 Concavity Proof

The dual formulation of the minimization problem is given in (4.22). The domain of x , z and m are determined by the remaining constraints in (4.13), (4.14), (4.17), and (4.24). I demonstrate the concavity of (4.22) with the parameter θ as follows, with θ defined in the range $0 \leq \theta \leq 1$. First, I give the equations for $\theta \cdot g(\vec{\lambda})$ and $(1 - \theta) \cdot g(\vec{\lambda})$. I use the notation $\vec{\lambda}^1$ and $\vec{\lambda}^2$ to indicate two distinct versions of the Lagrange multiplier vectors. The corresponding notation in the equations, such as

$\lambda_{pt\omega}^{2.1}$ indicates a multiplier from set λ^2 that corresponds to the equivalent $\lambda_{pt\omega}^1$ used in (4.22).

$$\begin{aligned}
\theta \cdot g(\vec{\lambda}^1) = \min - & \sum_{i,j,p,t,\omega} \theta \pi(\omega) (C_{jp} x_{ijpt\omega} - C_i z_{it\omega} - C_i^{(m)} m_{it\omega}) \\
& + \theta \left(\sum_{p,t,\omega} \lambda_{pt\omega}^{1.1} \left(\sum_{i,j} D_{i\omega}^{(n)} x_{ijpt\omega} - Q_p^{(n)} \right) \right. \\
& + \sum_{i,j,t_{i\omega},\omega} \lambda_{ijt\omega}^{1.2} \left(\sum_p (B_j D_{i\omega}^{(b)} x_{ijpt\omega}) + D_{i\omega}^{(m)} m_{it\omega} - Q_{i\omega}^{(b)} \right) \\
& - \sum_{i,t,\omega} \lambda_{it\omega}^{1.3} \left(z_{it\omega} + \sum_{j,p} x_{ijpt\omega} - \sum_p D_{ipt\omega}(\omega) \right) \\
& \left. - \sum_{i,p,t,\omega} \lambda_{ipt\omega}^{1.4} \left(m_{it\omega} + z_{it\omega} + \sum_j x_{ijpt\omega} - D_{ipt\omega} \right) \right), \quad (4.27)
\end{aligned}$$

$$\begin{aligned}
(1 - \theta) \cdot g(\vec{\lambda}^2) = \min - & \sum_{i,j,p,t,\omega} (1 - \theta) \pi(\omega) (C_{jp} x_{ijpt\omega} - C_i z_{it\omega} - C_i^{(m)} m_{it\omega}) \\
& + (1 - \theta) \left(\sum_{p,t,\omega} \lambda_{pt\omega}^{2.1} \left(\sum_{i,j} D_{i\omega}^{(n)} x_{ijpt\omega} - Q_p^{(n)} \right) \right. \\
& + \sum_{i,j,t_{i\omega},\omega} \lambda_{ijt\omega}^{2.2} \left(\sum_p (B_j D_{i\omega}^{(b)} x_{ijpt\omega}) + D_{i\omega}^{(m)} m_{it\omega} - Q_{i\omega}^{(b)} \right) \\
& - \sum_{i,t,\omega} \lambda_{it\omega}^{2.3} \left(z_{it\omega} + \sum_{j,p} x_{ijpt\omega} - \sum_p D_{ipt\omega} \right) \\
& \left. - \sum_{i,p,t,\omega} \lambda_{ipt\omega}^{2.4} \left(m_{it\omega} + z_{it\omega} + \sum_j x_{ijpt\omega} - D_{ipt\omega} \right) \right). \quad (4.28)
\end{aligned}$$

Using these two equations, I can combine them to obtain $g(\theta\lambda^1 + (1 - \theta)\lambda^2)$ and perform the following derivation:

$$\begin{aligned}
& g(\theta\vec{\lambda}^1 + (1 - \theta)\vec{\lambda}^2) \\
& = \min - \sum_{i,j,p,t,\omega} \pi(\omega) (C_{jp} x_{ijpt\omega} - C_i z_{it\omega} - C_i^{(m)} m_{it\omega}) \\
& \quad + \left(\sum_{p,t,\omega} \left(\theta \lambda_{pt\omega}^{1.1} + (1 - \theta) \lambda_{pt\omega}^{2.1} \right) \left(\sum_{i,j} D_{i\omega}^{(n)} x_{ijpt\omega} - Q_p^{(n)} \right) \right. \\
& \quad + \sum_{i,j,t_{i\omega},\omega} \left(\theta \lambda_{ijt\omega}^{1.2} + (1 - \theta) \lambda_{ijt\omega}^{2.2} \right) \left(\sum_p (B_j D_{i\omega}^{(b)} x_{ijpt\omega}) + D_{i\omega}^{(m)} m_{it\omega} - Q_{i\omega}^{(b)} \right) \\
& \quad \left. - \sum_{i,t,\omega} \left(\theta \lambda_{it\omega}^{1.3} + (1 - \theta) \lambda_{it\omega}^{2.3} \right) \left(z_{it\omega} + \sum_{j,p} x_{ijpt\omega} - \sum_p D_{ipt\omega} \right) \right)
\end{aligned}$$

$$\begin{aligned}
 & - \sum_{i,p,t,\omega} \left(\theta \lambda_{ipt\omega}^{1.4} + (1-\theta) \lambda_{ipt\omega}^{2.4} \right) \left(m_{it\omega} + z_{it\omega} + \sum_j x_{ijpt\omega} - D_{ipt\omega} \right) \\
 = \min & - \sum_{i,j,p,t,\omega} (\theta + (1-\theta)) \left(\pi(\omega) (C_{jp} x_{ijpt\omega} - C_i z_{it\omega} - C_i^{(m)} m_{it\omega}) \right) \\
 & + \left(\sum_{p,t,\omega} \left(\theta \lambda_{pt\omega}^{1.1} + (1-\theta) \lambda_{pt\omega}^{2.1} \right) \left(\sum_{i,j} D_{i\omega}^{(n)} x_{ijpt\omega} - Q_p^{(n)} \right) \right. \\
 & + \sum_{i,j,t_{i\omega},\omega} \left(\theta \lambda_{ijt\omega}^{1.2} + (1-\theta) \lambda_{ijt\omega}^{2.2} \right) \left(\sum_p (B_j D_{i\omega}^{(b)} x_{ijpt\omega}) + D_{i\omega}^{(m)} m_{it\omega} - Q_{i\omega}^{(b)} \right) \\
 & - \sum_{i,t,\omega} \left(\theta \lambda_{it\omega}^{1.3} + (1-\theta) \lambda_{it\omega}^{2.3} \right) \left(z_{it\omega} + \sum_{j,p} x_{ijpt\omega} - \sum_p D_{ipt\omega} \right) \\
 & - \sum_{i,p,t,\omega} \left(\theta \lambda_{ipt\omega}^{1.4} + (1-\theta) \lambda_{ipt\omega}^{2.4} \right) \left(m_{it\omega} + z_{it\omega} + \sum_j x_{ijpt\omega} - D_{ipt\omega} \right) \\
 = \min & \theta \left(- \sum_{i,j,p,t,\omega} (\pi(\omega) (C_{jp} x_{ijpt\omega} - C_i z_{it\omega} - C_i^{(m)} m_{it\omega})) \right. \\
 & + \sum_{p,t,\omega} \lambda_{pt\omega}^{1.1} \left(\sum_{i,j} D_{i\omega}^{(n)} x_{ijpt\omega} - Q_p^{(n)} \right) \\
 & + \sum_{i,j,t_{i\omega},\omega} \lambda_{ijt\omega}^{1.2} \left(\sum_p (B_j D_{i\omega}^{(b)} x_{ijpt\omega}) + D_{i\omega}^{(m)} m_{it\omega} - Q_{i\omega}^{(b)} \right) \\
 & - \sum_{i,t,\omega} \lambda_{it\omega}^{1.3} \left(z_{it\omega} + \sum_{j,p} x_{ijpt\omega} - \sum_p D_{ipt\omega}(\omega) \right) \\
 & - \sum_{i,p,t,\omega} \lambda_{ipt\omega}^{1.4} \left(m_{it\omega} + z_{it\omega} + \sum_j x_{ijpt\omega} - D_{ipt\omega} \right) \\
 & + (1-\theta) \left(- \sum_{i,j,p,t,\omega} (\pi(\omega) (C_{jp} x_{ijpt\omega} - C_i z_{it\omega} - C_i^{(m)} m_{it\omega})) \right. \\
 & + \sum_{p,t,\omega} \lambda_{pt\omega}^{2.1} \left(\sum_{i,j} D_{i\omega}^{(n)} x_{ijpt\omega} - Q_p^{(n)} \right) \\
 & + \sum_{i,j,t_{i\omega},\omega} \lambda_{ijt\omega}^{2.2} \left(\sum_p (B_j D_{i\omega}^{(b)} x_{ijpt\omega}) + D_{i\omega}^{(m)} m_{it\omega} - Q_{i\omega}^{(b)} \right) \\
 & - \sum_{i,t,\omega} \lambda_{it\omega}^{2.3} \left(z_{it\omega} + \sum_{j,p} x_{ijpt\omega} - \sum_p D_{ipt\omega} \right) \\
 & \left. - \sum_{i,p,t,\omega} \lambda_{ipt\omega}^{2.4} \left(m_{it\omega} + z_{it\omega} + \sum_j x_{ijpt\omega} - D_{ipt\omega} \right) \right). \tag{4.29}
 \end{aligned}$$

Since $\min_{p_i \in \vec{p}} \{p_i\} + \min_{q_i \in \vec{q}} \{q_i\} \leq \min_{c_i \in \vec{p} + \vec{q}} \{c_i\}$, then from (4.27)-(4.29) I can say that $\theta \cdot g(\vec{\lambda}^1) + (1-\theta) \cdot g(\vec{\lambda}^2) \leq g(\theta \vec{\lambda}^1 + (1-\theta) \vec{\lambda}^2)$, and therefore $g(\vec{\lambda})$ is concave.

4.4.1.2 Convergence

Given the concavity of $g(\vec{\lambda})$, I can show that the algorithm can converge to the optimum through iterative updates of the Lagrange multipliers, $\vec{\lambda}$. The gradient function, given in (4.26), can be assumed to follow the constraint $\nabla g(\vec{\lambda}) \leq G$, which

I give detail of in Section 4.4.2. Each multiplier, λ_i , is also bounded below by 0 and above. For example, from Eq. (4.22), I derive the following upper bound for $\lambda_{ipt\omega}^4$:

$$\lambda_{ipt\omega}^4 \leq \pi(\omega)C_i^{(z)}it\omega, \quad \forall i \in \mathcal{I}, \forall p \in \mathcal{P}, \forall t \in \mathcal{T}, \forall \omega \in \Omega. \quad (4.30)$$

I can derive similar scalar bounds for λ^1 , λ^2 , and λ^3 , thus I know that $\|\vec{\lambda}^{(1)} - \vec{\lambda}^*\|_2$, where $\vec{\lambda}^{(1)}$ is the set of multipliers at the first iteration, is bounded. This bound can be expressed as $\|\vec{\lambda}^{(1)} - \vec{\lambda}^*\|_2 \leq R$, which I expand on in Section 4.4.2. Given this, I can utilize the proof from [31] which I reproduce here:

$$\begin{aligned} \|\vec{\lambda}^{(k+1)} - \vec{\lambda}^*\|_2^2 &= \|\vec{\lambda}^{(k)} + \alpha_k \nabla g(\vec{\lambda}^{(k)}) - \vec{\lambda}^*\|_2^2 \\ &= \|\vec{\lambda}^{(k)} - \vec{\lambda}^*\|_2^2 + 2\alpha_k \nabla g(\vec{\lambda}^{(k)}) (\vec{\lambda}^{(k)} - \vec{\lambda}^*) + \alpha_k^2 \|\nabla g(\vec{\lambda}^{(k)})\|_2^2 \\ &\leq \|\vec{\lambda}^{(k)} - \vec{\lambda}^*\|_2^2 + 2\alpha_k (g(\vec{\lambda}^{(k)}) - g(\vec{\lambda}^*)) + \alpha_k^2 \|\nabla g(\vec{\lambda}^{(k)})\|_2^2. \end{aligned} \quad (4.31)$$

The inequality above is based on the fact that $g(\vec{\lambda}^*) \leq g(\vec{\lambda}^{(k)}) + \nabla g(\vec{\lambda}^{(k)}) (\vec{\lambda}^* - \vec{\lambda}^{(k)})$ for a concave function. This can be applied recursively to give equation (4.32).

$$\begin{aligned} \|\vec{\lambda}^{(k+1)} - \vec{\lambda}^*\|_2^2 &\leq \|\vec{\lambda}^{(1)} - \vec{\lambda}^*\|_2^2 + 2 \sum_{i=1}^k \alpha_i (g(\vec{\lambda}^{(i)}) - g(\vec{\lambda}^*)) + \sum_{i=1}^k \alpha_i^2 \|\nabla g(\vec{\lambda}^{(i)})\|_2^2 \\ &\leq R^2 + 2 \sum_{i=1}^k \alpha_i (g(\vec{\lambda}^{(i)}) - g(\vec{\lambda}^*)) + G \sum_{i=1}^k \alpha_i^2. \end{aligned} \quad (4.32)$$

Keeping the best result, and given the proof of concavity, we have:

$$\sum_{i=1}^k \alpha_i (g(\vec{\lambda}^{(i)}) - g(\vec{\lambda}^*)) \leq (g_{best}^{(k)} - g(\vec{\lambda}^*)) \left(\sum_{i=1}^k \alpha_i \right). \quad (4.33)$$

Given the upper bounds on the Lagrange multipliers, we can get:

$$g_{best}^{(k)} - g(\vec{\lambda}^*) \leq \frac{R^2 + G^2 \sum_{i=1}^k \alpha_i^2}{2 \sum_{i=1}^k \alpha_i}. \quad (4.34)$$

Given the optimal version of the stepsize calculation rule $\alpha_k = (R/G)/\sqrt{k}$, $g_{best}^{(k)}$ will converge to $g(\vec{\lambda}^*)$ as k , the step count, increases, giving the upper bound on the number of required iterations as $K \leq (RG/\epsilon)^2$, where ϵ is the acceptable error margin

within which the best solution only changes fractionally (see Section 4.3.3.3 on the stopping criterion).

4.4.1.3 Strong Duality

Having demonstrated that the partial Lagrange algorithm will converge to an optimal solution of the dual problem, I must also show that the corresponding variable values give the optimal solution to the original primal problem. Total unimodularity guarantees that the solution will be integer, and according to Slater's constraint qualification, if the optimization problem is convex, and there is at least one strictly feasible solution, then the solution to the Lagrange dual problem is equivalent to the primal solution [114]. Since the LP relaxation of my model is convex, and it is easy to find at least one strictly feasible solution, I can conclude that if $\vec{\lambda}^*$ is the optimal solution to the dual problem, $g(\vec{\lambda})$, then the corresponding $\vec{x}^*$, $\vec{m}^*$, and $\vec{z}^*$ are the optimal integer solutions to the primal problem, and therefore the original binary optimization problem.

4.4.2 Scalability Analysis

In the previous section, I proved that the partial Lagrange multiplier method will converge within $K \leq (RG/\epsilon)^2$ iterations. Given that for each Lagrange multiplier (I can denote an individual multiplier by λ_i), considering my earlier result that $\|\vec{\lambda}^{(1)} - \vec{\lambda}^*\|_2 \leq R$, I can assume that R is equal to, at most, the largest upper bound of the Lagrange multipliers, multiplied by $\sqrt{M}$, where M is the total number of Lagrange multipliers. Thus R is linearly proportional to the square root of the number of Lagrange multipliers, which I denote by $R = v\sqrt{M}$. I have previously stated that $\|\nabla g(\vec{\lambda})\|_2 \leq G$, with $\nabla g(\vec{\lambda})$ defined in (4.26). If $(\sum_{i,j} D_{iw}^{(n)} x_{ijptw}^k - Q_p^{(n)}) \geq 0$, $(\sum_{p \in \mathcal{P}} (B_j D_{iw}^{(b)} x_{ijptw}^k) + D_{iw}^{(m)} m_{itw}^k - Q_{iw}^{(b)}) \geq 0$, $-((z_{itw}^k + \sum_{j,p} x_{ijptw}^k) - \sum_p D_{iptw}) \leq 0$, and $-(m_{itw}^k + z_{itw}^k + \sum_{j \in \mathcal{J}} x_{ijptw}^k - D_{iptw}) \leq 0$, then $\|\nabla g(\vec{\lambda})\|_2 \leq \|\mathbf{A}\vec{y}\|_2 \leq \|\mathbf{A}\mathbf{I}_{|\vec{y}| \times 1}\|_2$, where $\vec{y}$ and $\mathbf{A}$ are the vector and matrix representations of the variables and their coefficients, respectively, in $\nabla g(\vec{\lambda})$. Thus, I can assume that $G = Q_1 \sqrt{|\vec{y}|}$, where $|\vec{y}|$ is the total number of decision variables, and Q_1 is a constant defined by the larger of

the network or battery demands of an application. Conversely, if I reverse the bounds on the above expressions I have the scalars as the upper bound of $\|\nabla g(\vec{\lambda})\|_2$, given by: $(\sum_{i,j,p,t,\omega} (Q_p^{(n)})^2 + \sum_{i,j,t_{i\omega},\omega} (Q_{i\omega}^{(b)})^2 + \sum_{i,p,t,\omega} (D_{ipt\omega})^2 + \sum_{i,p,t,\omega} (D_{ipt\omega})^2)^{\frac{1}{2}}$. Since the value here is simply proportional to the problem size, I can again reasonably assume that $G = Q_2\sqrt{|\vec{y}|}$, where Q_2 is another constant defined by the largest parameter value. Thus I can confidently define $G = Q\sqrt{|\vec{y}|}$, where $Q = \max\{Q_1, Q_2\}$.

Taking these derivations and applying them to $K \leq (RG/\epsilon)^2$, I have an upper bound on the number of iterations required to converge given by $K \leq Q^2 \times v^2 M \times |y|/\epsilon^2$. Given that the coefficients of the Lagrange multipliers in the dual function are taken from the constraints that do not satisfy total unimodularity, the number of iterations is bounded by a linear function of the number of these constraints and the total number of variables in the original problem formulation. Therefore, having each iteration of the partial Lagrange multiplier algorithm solvable in polynomial time (assuming the Interior Point Method is used), and the number of iterations bounded by a linear function, I can say the total computation complexity of the partial Lagrange multiplier method is at worst polynomial, and therefore guarantees improvement over the traditional branch-and-bound [1] method for integer programming.

4.5 Sensitivity Analysis

The purpose of using stochastic programming is to find an optimal allocation of resources at the present time which accounts for the uncertainty of future events, for example user demand, equipment revenue, battery levels. Sensitivity analysis examines how flexible a solution is, and determines its tolerance to changes in parameter values. In [104], an analytical method for sensitivity analysis of a linear program is derived. Whilst my original problem is a binary program, the partial Lagrange multiplier algorithm provides the solution of a linear programming problem, which due to total unimodularity, means I can apply this sensitivity analysis approach to my problem without loss of accuracy. To do this, I take the equation in (4.22), using the final Lagrange values found by the algorithm as constant coefficients, as

my primal objective function. This may be rearranged to give (4.35), removing the scalar values as they only influence the final profit, rather than the individual variable values. This objective function is subject to the constraints satisfying total unimodularity: (4.13), (4.14), (4.17), and variables constraint (4.24), which I divide into three separate constraints for clarity, listed here:

$$\begin{aligned} \min \sum_{i,j,p,t,\omega} & \left((-p(\omega)C_{jp} + \lambda_{pt\omega}^{1*}D_{i\omega}^{(n)} + \lambda_{ijt\omega}^{2*}B_jD_{i\omega}^{(b)} - \lambda_{it\omega}^{3*} - \lambda_{ipt\omega}^{4*})x_{ijpt\omega} \right. \\ & \left. + (p(\omega)C_i - \lambda_{it\omega}^{3*} - \lambda_{ipt\omega}^{4*})z_{it\omega} + (p(\omega)C_i^{(m)} + \lambda_{ijt\omega}^{2*}D_{i\omega}^{(m)} - \lambda_{ipt\omega}^{4*})m_{it\omega} \right), \end{aligned} \quad (4.35)$$

$$0 \leq x_{ijpt\omega} \leq 1, \quad \forall i \in \mathcal{I}, \forall j \in \mathcal{J}, \forall p \in \mathcal{P}, \forall t \in \mathcal{T}, \forall \omega \in \Omega, \quad (4.36)$$

$$0 \leq z_{it\omega} \leq 1, \quad \forall i \in \mathcal{I}, \forall t \in \mathcal{T}, \forall \omega \in \Omega, \quad (4.37)$$

$$0 \leq m_{it\omega} \leq 1, \quad \forall i \in \mathcal{I}, \forall t \in \mathcal{T}, \forall \omega \in \Omega. \quad (4.38)$$

This primal problem is therefore of the standard LP form, which expressed in vector notation is:

$$\min_x \{ \vec{c}^\top \vec{x} : \mathbf{A}\vec{x} \leq \vec{b}, \vec{x} \geq 0 \}, \quad (4.39)$$

whilst the accompanying dual problem is of the form:

$$\max_{y,s} \{ \vec{b}^\top \vec{y} : \mathbf{A}\vec{y} + \vec{s} \geq \vec{c}, \vec{s} \geq 0 \}. \quad (4.40)$$

I may obtain exact sensitivity results for the coefficients in the objective function by formulating an LP of the dual problem. The dual problem accompanying this formulation is therefore given in detail as follows:

$$\begin{aligned} \max \sum_{j,p,t,\omega} & Q_{jp}W_{jpt\omega}^{(4.13)} + \sum_{i,p,t,\omega} D_{ipt\omega}W_{it\omega}^{(4.14)} \\ & + \sum_{i,j,p,t,\omega} (F_{ij\omega}W_{ijpt\omega}^{(4.17)} + W_{ijpt\omega}^{(4.36)}) + \sum_{i,t,\omega} (W_{it\omega}^{(4.37)} + W_{it\omega}^{(4.38)}), \end{aligned} \quad (4.41)$$

$$\begin{aligned}
 W_{jpt\omega}^{(4.13)} + W_{it\omega}^{(4.14)} + W_{jpt\omega}^{(4.17)} + W_{ijpt\omega}^{(4.36)} \geq \\
 -p(\omega)C_{jp} + \lambda_{pt\omega}^{1*}D_{i\omega}^{(n)} + \lambda_{ijpt\omega}^{2*}B_jD_{i\omega}^{(b)} - \lambda_{it\omega}^{3*} - \lambda_{ipt\omega}^{4*}, \quad (4.42) \\
 \forall i \in \mathcal{I}, \forall j \in \mathcal{J}, \forall p \in \mathcal{P}, \forall t \in \mathcal{T}, \forall \omega \in \Omega,
 \end{aligned}$$

$$W_{it\omega}^{(4.37)} \geq p(\omega)C_i - \lambda_{it\omega}^{3*} - \sum_p \lambda_{ipt\omega}^{4*} \quad \forall i \in \mathcal{I}, \forall t \in \mathcal{T}, \forall \omega \in \Omega, \quad (4.43)$$

$$W_{it\omega}^{(4.38)} \geq p(\omega)C_i^{(m)} + \sum_j \lambda_{ijpt\omega}^{2*}D_{i\omega}^{(m)} - \sum_p \lambda_{ipt\omega}^{4*} \quad \forall i \in \mathcal{I}, \forall t \in \mathcal{T}, \forall \omega \in \Omega. \quad (4.44)$$

(4.41) is the objective function, with dual variables, W , corresponding to each of the constraint equations in the original formulation. For clarity, the superscript is the equation number that the variable corresponds to. Equations (4.42), (4.43), and (4.44) match the variables x , z , and m , respectively. Since this is an LP, strong duality holds, and the solution of the dual problem is equivalent to the result of the primal problem.

Given this dual formulation, from [104], I have the following rules for the shape of the optimal value function graph as a variable's cost coefficient changes:

- The optimal value function, $f(\gamma)$, is defined as the change in value of the optimal solution as the cost coefficient of one of the variables, for example $x_{ijpt\omega}$, varies by the offset value γ .
- The optimal value function is convex and piecewise linear.
- $\mathcal{F}_\gamma^*$ denotes the set of optimal solutions for the primal problem with the offset value γ , for example $C_{jp} + \gamma$. $\mathcal{F}_\gamma^*$ is constant for (γ_1, γ_2) , if $f(\gamma)$ is linear across the interval $[\gamma_1, \gamma_2]$.
- If I have γ_1 and γ_2 such that $\mathcal{F}_{\gamma_1}^* = \mathcal{F}_{\gamma_2}^*$, I can define $\bar{\mathcal{F}}^* = \mathcal{F}_\gamma^*$ for all $\gamma \in [\gamma_1, \gamma_2]$. This is a linear interval of $f(\gamma)$.

Having thus defined the behaviour of the optimal value function, the aim of my sensitivity analysis is to find the breakpoints where the linear intervals are defined. I can denote the optimal value, found by solving either the primal or dual problem by z^* . I can find the sensitivity analysis result from modifying any cost coefficient by solving two linear equations to give γ_{min} and γ_{max} , which are the lower and upper bounds of the perturbation variable γ such that the optimal variable choices for the primal variables, denoted by x^* , z^* , and m^* , and the dual variables, W^* , remain constant. I give the general vector form of the two LPs in (4.45) and (4.46), where $e_q = 1$ if q is the vector index of the c_q under analysis, else $e_q = 0$.

$$\gamma_{min} = \min\{\gamma : \mathbf{A}^\top \vec{\mathbf{y}} + \vec{\mathbf{s}} \geq \vec{\mathbf{c}} + \gamma e_j, \vec{\mathbf{b}}^\top \vec{\mathbf{y}} = \vec{\mathbf{c}}^\top \vec{\mathbf{x}}^* + \gamma x_j^*, \vec{\mathbf{s}} \geq 0\}, \quad (4.45)$$

$$\gamma_{max} = \max\{\gamma : \mathbf{A}^\top \vec{\mathbf{y}} + \vec{\mathbf{s}} \geq \vec{\mathbf{c}} + \gamma e_j, \vec{\mathbf{b}}^\top \vec{\mathbf{y}} = \vec{\mathbf{c}}^\top \vec{\mathbf{x}}^* + \gamma x_j^*, \vec{\mathbf{s}} \geq 0\}. \quad (4.46)$$

In (4.47) I give an example of the objective function to find γ_{min} where one of the x variable's revenue coefficient is being examined. The optimal values for the primal variables are included as constants, with the main decision variable being γ . The decision variable under examination is denoted by x_q . It is also necessary to change the corresponding dual constraint, which I give in (4.48). The other dual constraints, (4.44) and (4.43), remain unchanged.

$$\begin{aligned} \gamma_{min} = \min \{ \gamma : & \sum_{j,p,t,\omega} Q_{jp} W_{jpt\omega}^{(4.13)} + \sum_{i,p,t,\omega} D_{ipt\omega} W_{it\omega}^{(4.14)} \\ & + \sum_{i,j,p,t,\omega} (F_{ij\omega} W_{ijpt\omega}^{(4.17)} + W_{ijpt\omega}^{(4.36)}) + \sum_{i,t,\omega} (W_{it\omega}^{(4.37)} + W_{it\omega}^{(4.37)}) \\ = & \sum_{i,j,p,t,\omega} \left((-p(\omega)C_{jp} + \lambda_{pt\omega}^{1*} D_{i\omega}^{(n)} + \lambda_{ijt\omega}^{2*} B_j D_{i\omega}^{(b)} - \lambda_{it\omega}^{3*} - \lambda_{ipt\omega}^{4*}) x_{ijpt\omega}^* \right. \\ & \left. + (p(\omega)C_i - \lambda_{it\omega}^{3*} - \lambda_{ipt\omega}^{4*}) z_{it\omega}^* + (p(\omega)C_i^{(m)} + \lambda_{ijt\omega}^{2*} D_{i\omega}^{(m)} - \lambda_{ipt\omega}^{4*}) m_{it\omega}^* \right) + \gamma x_q \}, \end{aligned} \quad (4.47)$$

$$\begin{aligned}
W_{jpt\omega}^{(4.13)} + W_{it\omega}^{(4.14)} + W_{jpt\omega}^{(4.17)} + W_{ijpt\omega}^{(4.36)} \geq \\
(-p(\omega)C_{jp} + \lambda_{pt\omega}^{1*}D_{i\omega}^{(n)} + \lambda_{ijt\omega}^{2*}B_jD_{i\omega}^{(b)} - \lambda_{it\omega}^{3*} - \lambda_{ipt\omega}^{4*}) + \gamma e_q, \\
\forall i \in \mathcal{I}, \forall j \in \mathcal{J}, \forall p \in \mathcal{P}, \forall t \in \mathcal{T}, \forall \omega \in \Omega.
\end{aligned} \tag{4.48}$$

Both the minimization of γ and the equivalent maximization problem can be solved as linear programs. The formulation given here examines the behaviour of the cost coefficients, however it would also be simple to modify the method (by using the optimal dual variables as constants and the primal constraints) to examine the effects of changing the constraint limits. As a result, this technique is extremely useful as it provides exact sensitivity results, as opposed to an experimental approach, which may yield imprecise results with more measurements taken than is necessary. The main limitation of this approach is that each variable must be studied separately, the LPs described cannot be applied to a range of variables at once. It is also important to note that for problems where there are multiple factors contributing to the variable coefficients (as there are here), the LP result needs to be adjusted for these factors. Nonetheless these are practical considerations that do not alter the usefulness of this method in analyzing the tolerance of the stochastic programming solution to changes in parameter values, and the way decision variables interact.

4.6 Performance Evaluation

The value of my model, and the partial Lagrange multiplier algorithm are dependent on their real world performance. To evaluate the performance of my methods, I design a scenario based on real usage trace data and present a number of numerical results to support my claims.

4.6.1 Parameter Settings

I draw data from a survey of Internet cafe users [115] and usage trace of wireless network users on a university campus [116] to design my scenario. In this case, I consider students moving between three locations, $P1$, $P2$, and $P3$, situated on a

university campus. The three locations represent three different location types: a library, a computer lab, and a coffee shop, respectively, with equipment demand lowest at the coffee shop. The scenario runs across 4 time periods of 1 hour each, with each of the 400 unique users (each with their own battery level and application requirements) staying for up to 2 hours at a time. Those who stay for longer have a 20% chance of migrating location for the second time period. A user only appears once in the system, with either 15, 30, or 40 new users arriving at a single location in a time period (demand at $P3$ is 0.6 of the other locations by default). A user may offload to equipment for one of three different applications - either productivity, video streaming, or online communication. The wireless LAN at each location has a capacity of 300, 600, or 450Mbps, for $P1$, $P2$, and $P3$ respectively, using multiple access points to maintain the bandwidth. Each instance of the productivity application requires 256kbps, the video application requiring 1500kbps (based on Netflix recommended bandwidth [108]), and online communication requiring 500kbps (derived from Skype bandwidth estimates [117]). Equipment comes in 4 varieties, $J1$, $J2$, $J3$, and $J4$, and is charged at prevailing Internet cafe rates in Singapore [118], with $J1$ giving a \$2 profit, whilst the other three return \$1 per period. I choose reasonable penalty cost values of \$0.50 and \$0.20 for lost users and migration, respectively. Battery levels are defined as a decimal, where 1.00 is a full battery. For equipment that still requires device battery, applications $K1$, $K2$, and $K3$ drain 0.10, 0.20, and 0.30 battery per hour, respectively. A user may arrive with battery level, 0.25, 0.50, 0.75, or 1.00, with physical migration incurring a 0.10 battery penalty. The probability of each combination of user numbers, user battery level, application demand, duration, and migration is calculated based on reasonable conclusions drawn from the real data.

4.6.2 Numerical Results

Based on my chosen parameter values, the stochastic optimization has finite support, and thus I can implement a deterministic equivalent formulation, and solve the model using GAMS scripts to solve both the default branch-and-bound binary program, and

iteratively, to solve the partial Lagrange multiplier algorithm. The results and my descriptions are presented here.

4.6.2.1 Effect of available equipment on lost users

In Fig. 4.3 the level of available equipment is set to 20 and varied for each class in turn. This is done simultaneously for each provider, and I measure the expected number of lost users. $J1$ varies greatly because it has the widest range of application compatibility, whilst $J3$ is compatible with the least popular application types. Thus I see that the primary cause of lost users when varying equipment quantities is compatibility - as this determines the number of users that need assigning to alternative equipment and the flexibility of moving them. This also explains the shape of the graph if I were to increase the equipment levels further; each would become constant, as the lost users would require equipment other than the increased one. The exception is $J1$, which has universal compatibility and further increase would bring the lost users to 0.

Figure 4.3: Impact of equipment availability on lost users.

4.6.2.2 Effect of changing battery level on lost users

In Fig. 4.4 I set all user battery levels to 0.50 and then vary scenarios $B1$ and $B2$ in turn. Since $B1$ and $B4$ are equally likely, their results are identical, as are $B2$ and $B3$. The impact of reduced battery levels on lost users drops as the levels increase, proportionately to the likelihood of that scenario, with the number reaching a constant, as the battery demands of the users do not exceed 0.60 in my test scenario.

Figure 4.4: Impact of battery levels on lost users.

4.6.2.3 Comparison with greedy approach

In Fig. 4.5 I compare the behaviour of the stochastic optimization with a greedy approach that assigns users on arrival without considering future demand. The percentage of users requiring two time periods is increased, and the resultant profit measured, with the stochastic optimization delivering a noticeably better result. This result has an unexpected aspect, as the gap actually closes between the two results, but this may simply be caused by the increased user duration having a generally beneficial effect.

4.6.2.4 Benefits of migration

In Fig. 4.6 I vary the number of devices available at location $P1$ and observe the change in total profit. I compare my migratory method with a similar stochastic optimization without migration and observe that at low equipment levels, the lost

Figure 4.5: Comparison of greedy approach with stochastic optimization with variation in user duration.

users result in a substantial loss of income, whilst migration allows us to dramatically increase the utilization of equipment, justifying the use of an approach that considers multiple locations. If I continue to increase the number of devices available at $P1$, the curves move closer together and become steady, but ultimately the non-migration option would never quite reach the migration curve, as other locations would still require migration, and money would be saved on lost users.

Figure 4.6: Benefits of migration on total profit vs non-migratory method.

4.6.2.5 Behaviour of migration

In Fig. 4.7 I again vary the number of devices at $P1$ and observe the variation in user allocation across the system. As the number of devices increases, the level of migration drops, and the allocation of users between $P1$ and $P2$ tends towards equality. $P3$ is a less popular location, as indicated by the demand data I use, but is significantly more utilized as a result of the low equipment levels at $P1$. This again shows the benefit of migration, as the migration curve shown here represents the profit difference between the migration and non-migration methods shown in the previous figure. As the $P1$ equipment continues to increase, $P1$ would continue to rise with equipment increases as it would then absorb unsatisfied demand from $P2$ and $P3$, which would become steady, as they reach maximum utilization. The migration curve would, however, start to increase again.

Figure 4.7: Behaviour of migration as equipment quantity varies.

4.6.2.6 Effects of migration probability

In Fig. 4.8 the probability of the number of users migrating is varied. The trace data indicates that users have a chance of migrating during their session, so I vary this

probability to observe the effects on the number of forced migrations and lost users. As the users increase their own movements, the number of lost users tends to increase, as demand becomes less predictable and the risk of a location being oversubscribed rises. If a user reaches an oversubscribed location, they may not have sufficient battery to migrate again, and so be lost. Interestingly, the number of migrated users goes down, perhaps again due to battery issues, as the optimization minimizes the number of total migrations to conserve each user's battery.

Figure 4.8: Effect of migration probability on user allocation.

4.6.2.7 Following user movement from trace data

In Fig. 4.9, I use real life trace data and track the expected movements of a user between locations on the Nanyang Technological University campus. The user arrives at location $P1$, a coffee shop, in time stage $T1$, and I indicate the probability of their movements across two time stages between $P2$, the Lee Wee Nam Library, and $P3$, an academic laboratory. In $T1$ they are most likely to be allocated to equipment at $P1$, but have a small chance of being relocated to $P2$, if their battery is good enough. In $T2$ they have a higher chance of moving away from $P1$ as there is a 20% chance that they will choose to migrate of their own accord. Whether they actually move, however, is dependent on the situation at $P1$, and the broker may consider it better for them to remain at $P1$, hence the smaller probability of moving. It is also possible that the user will move to $P3$, as they may have been migrated to $P2$ in the first

time stage, yet want to move again. This probability is, inevitably, smaller. There also remains a possibility that the user will be lost in each time stage. Probabilities shown for $T2$ do not consider users who only stay for one time period, which occurs at a probability of 0.6.

Figure 4.9: Probability of user movement based on trace data.

4.6.2.8 Influence of price variation on user allocation

In Fig. 4.10, I vary the revenue from device allocation at $P1$ and study the effect on user allocation. At low prices, it is still undesirable to reject a user, but as the price rises, I observe movement of users from other locations as the system migrates them to $P1$.

4.6.2.9 Runtime performance of partial Lagrange multiplier method

The purpose of introducing the partial Lagrange multiplier method is to take advantage of the tractability of linear programming. I use a simplified version of the problem (necessary to be tractable) to compare the performance benefits from the algorithm as I vary the number of users. In Fig. 4.11 I compare the execution time of the traditional branch-and-bound method for solving integer programming problems

Figure 4.10: Effect of price variation on user allocation.

with the partial Lagrange multiplier algorithm, and find that for the larger problem I gain a significant performance improvement that will only grow considerably with larger problem sizes. At the smaller size, the traditional method is only a little worse, but increases exponentially, the gap in execution time growing with each step up in problem size. By contrast, my algorithm increases at a linear rate and at a much more gentle slope. This is in keeping with the theoretical result proven in Section 4.4.2.

Figure 4.11: Comparison of execution time of B&B algorithm vs partial Lagrange multiplier method.

4.6.2.10 Effect of stepsize on convergence rate

The calculation of λ is determined by the stepsize, and is the main influence on the algorithm's behaviour. I have proved that the algorithm converges, and does so in polynomial time. However, the choice of stepsize can control the exact number of iterations to reach the optimum. In Fig. 4.12 I compare two methods of calculating stepsize - one constant and one using $\sqrt{1/k}$, where k is the iteration number. Both methods will reach the optimum, however the constant stepsize reaches more quickly, but oscillates on and below the optimum, as the stepsize never decreases. However, this is acceptable as the algorithm terminates if the solution does not improve. By contrast, the $\sqrt{1/k}$ curve decreases more slowly and less predictably with the stepsize but reaches the exact optimal value, although it slows down as it does so. The best step size calculation method will vary based on the individual problem, which in this case lends itself to a rapid constant descent.

Figure 4.12: Comparison of convergence rates with different stepsize calculation methods.

4.6.3 Sensitivity Analysis Result

In Fig. 4.13 I study the results from performing the Sensitivity Analysis introduced in Section 4.5. The first two curves show the breakpoint for the price and probability coefficients of a single user allocation variable. Due to the total unimodularity restriction, the variable is effectively binary, so there is a single breakpoint. Since the two

parameters are both coefficients, they behave in a similar manner. The breakpoint indicates the price and probability points at which the variable switches between 0 and 1. To contrast the behaviour of the variable coefficients, the third curve shows the change in the expected value function as the limit on one constraint varies, in this case one of the constraints limiting total allocation by the number of pieces of equipment. Unlike the cost coefficients, this parameter has multiple breakpoints, as with my parameter settings it is the main limiting factor on equipment allocation. The sensitivity analysis used in (4.47) can be modified, replacing the dual problem with the primal problem and finding the γ values for the dual variables. This result can give us the breakpoints for the dual variables, which correspondingly yields the breakpoints for the primal constraints, showing how they influence the primal variables. For example, I can apply this method to find that a single variable will change between 0 and 1 by adding 5 pieces of equipment for that particular scenario. Thus Sensitivity Analysis can be used to find the behaviour of variables and the precise relationship between parameter settings and variable values.

4.7 Conclusion

In this chapter I have presented a stochastic optimization formulation to solve a novel mobile application migration scenario. I have explained the importance of the scenario in the mobile cloud computing context and demonstrated that the formulation finds the optimal solution. In a real world application, a stochastic approach that allows for user migration is most effective. I have also presented a polynomial time algorithm that guarantees an optimal solution to the integer problem. I have proved the convergence and runtime of the algorithm and demonstrated that the practical results support the theoretical analysis. This algorithm is essential for a practical application, as even moderately large scenarios can become intractable for traditional integer programming methods and has potential for application in other fields. I believe this is an important work that considers the logical next step for cloud computing and I have presented an optimal, realistic, and scalable approach to solving

Figure 4.13: Comparison sensitivity analysis results for different parameters.

the allocation problem. This area merits further research and presents a significant opportunity, in addition to clearly demonstrating the power of stochastic optimization as applied to the problem of battery limitations in mobile cloud computing. Having now addressed both performance guarantees and battery life, in the next chapter I will apply stochastic optimization to the problem of cyber security.

Chapter 5

A Scalable Approach to Joint Cyber Insurance and Security-as-a-Service Provisioning in Cloud Computing

Chapters 3 and 4 have demonstrated that stochastic optimization is a powerful tool in both guaranteeing performance and preserving battery life in the context of mobile cloud computing. In this chapter I intend to demonstrate how it may be applied to the remaining major issue in mobile cloud computing - security. Whether offloading computation to the cloud, or to a local computer, users need to be assured that their data is secure. By considering whether cyber insurance can encourage the provisioning of cyber security services in a cloud context, I will show that stochastic optimization can be applied usefully to this problem. This chapter is organised as follows: Section 5.1 outlines the system model for my combined security and insurance environment, and the stochastic optimization formulation is provided in Section 5.2. To improve the runtime performance of my solution, I introduce the partial Lagrange multiplier algorithm in Section 5.3, and give proofs of its performance in Section 5.4. The technique used in my sensitivity analysis is presented in Section 5.5. I give comprehensive practical results in Section 5.6 demonstrating the effectiveness of my ideas.

5.1 System Model

I consider a SECaaS framework similar to the one used in [27], containing a customer who uses applications, which receive Internet traffic in the form of packets. These

packets are scanned by services from SECaaS providers, provisioned by a subscription management process (SMP). In the event that harmful packets elude security, cyber insurers, subscribed to by an insurance management process (IMP), provide compensation for damages incurred. In this section I outline the components of the system as illustrated in Fig. 5.1. The notations for the system model elements are given in Table 5.1.

Figure 5.1: Illustration of interaction between system components.

5.1.1 Security-as-a-Service Model

In my security model, a customer runs applications (denoted by set $\mathcal{A}$) that I assume to be Internet-accessible, either on a cloud service such as Amazon EC2, or in-house servers managed by the customer. Applications receive data packets in accordance with their operating purpose, e.g. email data or financial transactions. Legitimate packets are called safe packets, while packets used in cyber attacks are called unsafe packets. Unsafe packets are deemed handled if they are correctly detected by security services, or unhandled if they are not successfully processed (for example if they are undetected). These unhandled packets will cause damage, which incurs costs to the

customer. Security services are provisioned and charged on a per-packet basis. Security providers (denoted by set $\mathcal{S}$) can offer customers prepaid plans tailored to each application (denoted by set $\mathcal{P}_{as}$) which have a particular per-packet price, duration, and maximum number of packets. Prepaid services are provisioned in advance, with services utilized according to the incoming traffic at the present time. In the event that the prepaid services are not sufficient, further services can be provisioned under an on-demand pricing scheme. On-demand allocation can be done dynamically, at a per-packet usage price that is higher than that of the prepaid scheme.

5.1.2 Cyber Insurance Model

Customers can purchase cyber insurance products in the form of insurance policies (denoted by set $\mathcal{L}$) from insurance companies (denoted by set $\mathcal{I}$) that compensate them for incidents and disasters such as data breach, data corruption, and business interruption. The IMP purchases insurance policies, which includes the premium, types of risks covered, indemnity value, and policy duration. These insurance policies can be packaged with security services, to cover damages caused by undetected unsafe packets. The price for insurance purchased in advance is charged at a rate known as a ‘present premium’. However, the insurer may adjust charges for the same policy at a future time, which is obtained in an on-demand fashion, known as a ‘future premium’. Insurance covers an application and the coverage of an application can only be purchased from one insurer at a given time. Further, I assume that damages can be accurately determined. This may be in various forms, such as a ‘ransom’ paid [119], or the cost of legal fees and court settlements, as in the case of Target’s 2013 data breach [26]. I assume that both first- and third-party damages are covered by the insurance, and that each unhandled packet for an application has the same damage cost and insurance coverage. These assumptions simplify the model with removal of each assumption left for future work.

5.1.3 Decision Stages

My system model takes place across multiple time periods (e.g. one hour or one day), as illustrated in Fig. 5.2. The set of all time periods is defined as $\mathcal{T} =$

Figure 5.2: In each time period, there are three provisioning stages. This figure illustrates an example case with 3 time periods, however this can be freely expanded to n time periods.

$\{t_1, t_2, t_3, \dots, t_n\}$, where n denotes the number of time periods. Security service reservation and cyber insurance policies may last for multiple time periods. For example, a user may purchase prepaid security services for two time periods, allowing them to use those services in the following time period at no additional cost. Let t denote the current time period, then $\mathcal{F}_{pt}$ denotes the set of previous time periods (including the current time period) in which security plan p could have been purchased and still be active. Let $|p|$ denote the length of plan p , then $\mathcal{F}_{pt} = \{\max(1, t - |p| + 1), \dots, t\}$. $\mathcal{F}_{lt}$ denotes the equivalent set for insurance policies, where l denotes the policy. Each individual time period consists of three decision stages. The first stage is the ‘preparation stage’. In this stage, the SMP and IMP make initial provisioning choices to provision prepaid security services and buy insurance policies using the present pre-

mium. The second stage is the ‘packet arrival stage’, where the number of incoming packets is realized. The number of required prepaid services and any additional on-demand services are provisioned, and additional insurance can be purchased using the future premium. The third and final stage is the ‘claim stage’, where the number of unhandled packets is realized and the damage costs and per-packet indemnity are calculated. It is assumed that insurance claims for unhandled packets are approved and there are no attempts to defraud the insurer.

5.1.4 Uncertainty Parameters

My model contains a number of uncertain parameters, which are not known in advance. For example in the preparation stage, the number of arriving packets at an application, and the corresponding number of unhandled packets, are unknown, and are represented by random parameters. Likewise, insurance premium costs and SE-CaaS prices may be uncertain, as they can be adjusted by providers, so these are also treated as random parameters (since the pricing strategy is opaque to the customer). In the preparation stage, the prepaid security and present premium prices are known. In the packet arrival stage, the number of arriving packets and the future premiums are known. In the claim stage, the number of unhandled packets is known.

An unrealized circumstance is known as a scenario, which is called a realization when it occurs at a particular time. Let $\Omega^\dagger$ represent the set of scenarios in the preparation stage. Then, given realization $\omega^\dagger$, $\Omega^\ddagger(\omega^\dagger)$ represents the set of possible scenarios in the packet arrival stage. Likewise, given the pair of realizations $(\omega^\dagger, \omega^\ddagger)$, $\Omega^{\ddagger\ddagger}(\omega^\dagger, \omega^\ddagger)$ represents the set of possible scenarios in the claim stage. The terms $\pi_{\omega^\dagger}$, $\pi_{\omega^\ddagger}$, and $\pi_{\omega^{\ddagger\ddagger}}$ represent the corresponding probabilities of the scenarios, and scenario sets of each provisioning stage have finite support and have probabilities summing to 1. The scenario probabilities of a time period t are independent from those of other time periods, so the probabilities of scenarios in period t are not affected by events in period $t - 1$.

5.1.5 Decision Variables

The solution of a stochastic optimization formulation is known as a decision, and is represented by a set of values assigned to the decision variables.

5.1.5.1 Preparation Stage Variables

The following are the decision variables for the preparation stage. (5.1) defines the domain of prepaid security services, where $X_{aspt\omega^\dagger}^\dagger$ denotes prepaid services from provider s , covering application a under security plan p , and (5.2) defines the domain of $W_{asilt\omega^\dagger}^\dagger$, which denotes the binary decision to buy insurance policy l using the present premium from insurer i to cover application a as protected by security provider s . These are evaluated over time stage t , and therefore must be considered for each scenario $\omega^\dagger$.

$$X_{aspt\omega^\dagger}^\dagger \in \mathbb{N}_0, \forall a \in \mathcal{A}, s \in \mathcal{S}, p \in \mathcal{P}_{as}, t \in \mathcal{T}, \omega^\dagger \in \Omega^\dagger, \quad (5.1)$$

$$W_{asilt\omega^\dagger}^\dagger \in \{0, 1\}, \forall a \in \mathcal{A}, s \in \mathcal{S}, i \in \mathcal{I}, l \in \mathcal{L}, t \in \mathcal{T}, \omega^\dagger \in \Omega^\dagger. \quad (5.2)$$

5.1.5.2 Packet Arrival Stage Variables

The variables for the packet arrival stage are defined as follows:

$$W_{asilt\omega^\ddagger}^\ddagger \in \{0, 1\}, \forall a \in \mathcal{A}, s \in \mathcal{S}, i \in \mathcal{I}, l \in \mathcal{L}, t \in \mathcal{T}, \omega^\ddagger \in \Omega^\ddagger(\omega^\dagger), \quad (5.3)$$

$$Y_{aspt\omega^\ddagger} \in \mathbb{N}_0, \forall a \in \mathcal{A}, s \in \mathcal{S}, p \in \mathcal{P}_{as}, t \in \mathcal{T}, \omega^\ddagger \in \Omega^\ddagger(\omega^\dagger), \quad (5.4)$$

$$Z_{ast\omega^\ddagger} \in \mathbb{N}_0, \forall a \in \mathcal{A}, s \in \mathcal{S}, t \in \mathcal{T}, \omega^\ddagger \in \Omega^\ddagger(\omega^\dagger). \quad (5.5)$$

$W_{asilt\omega^\ddagger}^\ddagger$ in (5.3) represents the choice to purchase insurance using the future premium, as an alternative to $W_{asilt\omega^\dagger}^\dagger$, subject to scenario $\omega^\ddagger$ in time t . $Y_{aspt\omega^\ddagger}$ in (5.4) indicates the number of prepaid services to be executed and $Z_{ast\omega^\ddagger}$ in (5.5) is the number of on-demand services that additionally need to be provisioned to cover any shortfall.

5.1.5.3 Claim Variables

The variables for the claim stage are defined as follows:

$$C_{asit\omega^{\dagger\ddagger}}^{\omega^\ddagger} \in \mathbb{N}_0, \forall a \in \mathcal{A}, s \in \mathcal{S}, i \in \mathcal{I}, t \in \mathcal{T}, \omega^\dagger \in \Omega^\dagger, \omega^{\dagger\ddagger} \in \Omega^{\dagger\ddagger}(\omega^\dagger, \omega^\ddagger), \quad (5.6)$$

$$C_{asit\omega^{\dagger\dagger}}^{\dagger\omega^{\dagger}} \in \mathbb{N}_0, \forall a \in \mathcal{A}, s \in \mathcal{S}, i \in \mathcal{I}, t \in \mathcal{T}, \omega^{\dagger} \in \Omega^{\dagger}, \omega^{\dagger\dagger} \in \Omega^{\dagger\dagger}(\omega^{\dagger}, \omega^{\dagger}). \quad (5.7)$$

$C_{asit\omega^{\dagger\dagger}}^{\omega^{\dagger}}$, whose domain is defined in (5.6), is the number of unhandled packets claimed for that have insurance coverage paid for by a present premium whilst $C_{asit\omega^{\dagger\dagger}}^{\dagger\omega^{\dagger}}$, defined in (5.7), is the number of unhandled packets claimed for that have coverage paid for by a future premium. This is under the realization of unhandled packets in scenario $\omega^{\dagger\dagger} \in \Omega^{\dagger\dagger}(\omega^{\dagger}, \omega^{\dagger})$.

5.2 Problem Formulation

The stochastic optimization problem can be represented as a nested formulation with multi-stage recourse. The solutions for both the IMP and the SMP can be obtained using the following sequence of equations:

$$\begin{aligned} \min \mathbb{E}_{\Omega^{\dagger}} \left[\mathcal{Q}^{\dagger}(X^{\dagger}, W^{\dagger}, \omega^{\dagger}) + \mathbb{E}_{\Omega^{\dagger}} \left[\mathcal{Q}^{\dagger}(Y, Z, W^{\dagger}, \omega^{\dagger}, \omega^{\dagger}) \right. \right. \\ \left. \left. + \mathbb{E}_{\Omega^{\dagger\dagger}} [\mathcal{Q}^{\dagger\dagger}(C, C_{\dagger}^{\dagger}, \omega^{\dagger}, \omega^{\dagger}, \omega^{\dagger\dagger})] \right] \right]. \quad (5.8) \end{aligned}$$

The objective function in (5.8) minimizes the total cost of both security service and insurance allocation across all time periods in the set $\mathcal{T}$, and thus it is only necessary to solve the optimization once for all $t \in \mathcal{T}$. The problem formulation is nested, as the expected value of each decision stage is affected by the expected value of the following stages. For example, the expected value of the packet arrival stage is determined, not only by the packet arrival stage variables, but also by the expected value of the claim stage. Therefore, the decisions made in the preparation stage and packet arrival stage are made in consideration of the effect those decisions will have on the expected value of the subsequent stages. This allows decisions to be made in the present that will have the best outcome in the future. The function $\mathcal{Q}^{\dagger}(\cdot)$ is an optimization problem to minimize the cost of the preparation stage across all time periods, and is defined as follows:

$$\mathcal{Q}^{\dagger}(X^{\dagger}, W^{\dagger}, \omega^{\dagger}) = \min \left[\mathcal{R}^{\dagger}(\omega^{\dagger}) \right] \quad (5.9)$$

Table 5.1: List of Key Notations for Chapter 5.

Symbol	Definition
$\mathcal{A}$	Set of all user applications, while $a \in \mathcal{A}$ denotes the application index.
$\mathcal{S}$	Set of all SECaaS providers, while $s \in \mathcal{S}$ denotes the provider index.
$\mathcal{T}$	Set of all time periods, while $t \in \mathcal{T}$ denotes the time period index.
$\mathcal{P}_{as}$	Set of all security plans, while $p \in \mathcal{P}_{as}$ denotes the plan index.
$\mathcal{I}$	Set of all cyber insurers, while $i \in \mathcal{I}$ denotes the insurer index.
$\mathcal{L}$	Set of all insurer policies while, $l \in \mathcal{L}$ denotes the policy index.
$\Omega^\dagger, \Omega^\ddagger(\omega^\dagger),$ $\Omega^{\ddagger\ddagger}(\omega^\dagger, \omega^\ddagger)$	Set of all scenarios for preparation stage, packet arrival stage, and claim stage with each set dependent on the scenario choice from the previous stage, while $\omega^\dagger, \omega^\ddagger, \omega^{\ddagger\ddagger}$ are the corresponding scenario indices.
$k_{aspt\omega^\dagger}^\dagger, k_{ast\omega^\ddagger}^\ddagger$	The cost of prepaid security services and on-demand under scenarios $\omega^\dagger$, and $\omega^\ddagger$, respectively.
$f_{asilt\omega^\dagger}^\dagger, f_{asilt\omega^\ddagger}^\ddagger$	Present insurance premium and future premium under scenarios $\omega^\dagger$, and $\omega^\ddagger$, respectively.
e_{asp}	Limit of prepaid packets for application a set by provider s under plan p .
$n_{a\omega^\ddagger}$	The total number of packets arriving at application a in scenario $\omega^\ddagger$.
u_a	The damages cost caused per packet .
$m_{as\omega^{\ddagger\ddagger}}^{\omega^\ddagger}$	The number of packets unhandled by provider s arriving at application a .
h_{asi}	The indemnity payable for unhandled packets by insurer i .
g_{asi}	The number of packets covered by insurer i for security service s .
$X_{aspt\omega^\dagger}^\dagger$	Decision variable for prepaid security services in the preparation stage.
$W_{asilt\omega^\dagger}^\dagger, W_{asilt\omega^\ddagger}^\ddagger$	Decision variables for present and future insurance premiums, respectively.
$Y_{aspt\omega^\ddagger}, Z_{ast\omega^\ddagger}$	Decision variables for security services to be utilized under the prepaid and on-demand schemes, respectively.
$C_{asit\omega^{\ddagger\ddagger}}^{\omega^\ddagger}, C_{asit\omega^{\ddagger\ddagger}}^{\ddagger\omega^\ddagger}$	Decision variables for number of packets claimed for under present and future premiums, respectively.

$$\text{where: } \mathcal{R}^\dagger(\omega^\dagger) = \sum_{a \in \mathcal{A}} \sum_{s \in \mathcal{S}} \sum_{t \in \mathcal{T}} \left(\sum_{p \in \mathcal{P}_{as}} k_{aspt\omega^\dagger}^\dagger X_{aspt\omega^\dagger}^\dagger + \sum_{i \in \mathcal{I}} \sum_{l \in \mathcal{L}} f_{asilt\omega^\dagger}^\dagger W_{asilt\omega^\dagger}^\dagger \right), \quad (5.10)$$

s.t. (5.1), (5.2),

$$\sum_{p \in \mathcal{P}_{as}} X_{aspt\omega^\dagger}^\dagger \leq e_{asp}, \forall a \in \mathcal{A}, s \in \mathcal{S}, t \in \mathcal{T}, \quad (5.11)$$

$$\sum_{i \in \mathcal{I}} \sum_{l \in \mathcal{L}} W_{asilt\omega^\dagger}^\dagger \leq 1, \forall a \in \mathcal{A}, s \in \mathcal{S}, t \in \mathcal{T}. \quad (5.12)$$

The objective function in (5.9) is to minimize the value of the function $\mathcal{R}^\dagger(\cdot)$, which denotes the cost of the preparation stage in all time periods, where $k_{aspt\omega^\dagger}^\dagger$ gives the prepaid security service cost, and $f_{asilt\omega^\dagger}^\dagger$ is the present insurance premium price. The constraint in (5.11) ensures that the number of prepaid services does not exceed the packet limit of provider s , for plan p , denoted by e_{asp} . Similarly, the constraint in (5.12) ensures that each application is only covered by one insurer at a time. Given fixed values of the decision variables, $X_{aspt\omega^\dagger}^\dagger$ and $W_{asilt\omega^\dagger}^\dagger$, I optimize the cost of the packet arrival stage as follows:

$$\mathcal{Q}^\dagger(Y, Z, W^\dagger, \omega^\dagger, \omega^\dagger) = \min \left[\mathcal{R}^\dagger(\omega^\dagger, \omega^\dagger) \right] \quad (5.13)$$

$$\text{where: } \mathcal{R}^\dagger(\omega^\dagger, \omega^\dagger) = \sum_{a \in \mathcal{A}} \sum_{s \in \mathcal{S}} \sum_{t \in \mathcal{T}} \left(k_{ast\omega^\dagger}^\dagger Z_{ast\omega^\dagger} + \sum_{i \in \mathcal{I}} \sum_{l \in \mathcal{L}} f_{asilt\omega^\dagger}^\dagger W_{asilt\omega^\dagger}^\dagger \right), \quad (5.14)$$

s.t. (5.3) – (5.5),

$$Y_{aspt\omega^\dagger} \leq \sum_{\hat{t} \in \mathcal{F}_{pt}} X_{aspt\omega^\dagger}^\dagger, \forall a \in \mathcal{A}, s \in \mathcal{S}, p \in \mathcal{P}_{as}, t \in \mathcal{T}, \quad (5.15)$$

$$n_{a\omega^\dagger} \leq \sum_{s \in \mathcal{S}} \left(Z_{ast\omega^\dagger} + \sum_{p \in \mathcal{P}_{as}} Y_{aspt\omega^\dagger} \right), \forall a \in \mathcal{A}, t \in \mathcal{T}, \quad (5.16)$$

$$\sum_{l \in \mathcal{L}} (W_{asilt\omega^\dagger}^\dagger + W_{asilt\omega^\dagger}^\dagger) \leq 1, \forall a \in \mathcal{A}, s \in \mathcal{S}, i \in \mathcal{I}, t \in \mathcal{T}, \quad (5.17)$$

$$\sum_{i \in \mathcal{I}} \sum_{l \in \mathcal{L}} W_{asilt\omega^\dagger}^\dagger \leq 1, \forall a \in \mathcal{A}, s \in \mathcal{S}, t \in \mathcal{T}. \quad (5.18)$$

The optimization problem in (5.14) minimizes the cost of the packet arrival stage. $k_{ast\omega^\dagger}^\dagger$ gives the on-demand security cost and $f_{asilt\omega^\dagger}^\dagger$ is the future insurance premium cost. The constraint in (5.15) ensures that utilized security does not exceed the prepaid provisioning. The constraint in (5.16) ensures that the mixture of prepaid and on-demand security services are sufficient to cover incoming packets $n_{a\omega^\dagger}$. The constraints in (5.17) and (5.18) are similar to that in (5.12), ensuring that only one insurer is bought from under present or future premiums per application. Finally, I also minimize the cost of the claim stage. Assuming fixed values of the decision

variables $W_{asilt\omega^\dagger}^\dagger$, $Y_{aspt\omega^\dagger}$ and $Z_{ast\omega^\dagger}$, I can define the function $\mathcal{Q}^{\dagger\dagger}(\cdot)$ as a minimization problem in the following way:

$$\mathcal{Q}^{\dagger\dagger}(C, C^\dagger, \omega^\dagger, \omega^\dagger, \omega^{\dagger\dagger}) = \min \left[\mathcal{R}^{\dagger\dagger}(\omega^\dagger, \omega^\dagger, \omega^{\dagger\dagger}) \right] \quad (5.19)$$

$$\text{where: } \mathcal{R}^{\dagger\dagger}(\omega^\dagger, \omega^\dagger, \omega^{\dagger\dagger}) = \sum_{a \in \mathcal{A}} \sum_{s \in \mathcal{S}} \sum_{t \in \mathcal{T}} \left(u_a m_{as\omega^{\dagger\dagger}}^\dagger - \sum_{i \in \mathcal{I}} h_{asi} \left(C_{asit\omega^{\dagger\dagger}}^{\omega^\dagger} + C_{asit\omega^{\dagger\dagger}}^{\dagger\omega^\dagger} \right) \right), \quad (5.20)$$

s.t. (5.6) – (5.7),

$$C_{asit\omega^{\dagger\dagger}}^{\omega^\dagger} \leq \sum_{l \in \mathcal{L}} \sum_{\hat{t} \in \mathcal{F}_{lt}} g_{asi} W_{asilt\omega^\dagger}^\dagger, \forall a \in \mathcal{A}, s \in \mathcal{S}, i \in \mathcal{I}, t \in \mathcal{T}, \quad (5.21)$$

$$C_{asit\omega^{\dagger\dagger}}^{\dagger\omega^\dagger} \leq \sum_{l \in \mathcal{L}} \sum_{\hat{t} \in \mathcal{F}_{lt}} g_{asi} W_{asilt\omega^\dagger}^\dagger, \forall a \in \mathcal{A}, s \in \mathcal{S}, i \in \mathcal{I}, t \in \mathcal{T}, \quad (5.22)$$

$$C_{asit\omega^{\dagger\dagger}}^{\omega^\dagger} + C_{asit\omega^{\dagger\dagger}}^{\dagger\omega^\dagger} \leq Z_{ast\omega^\dagger} + \sum_{p \in \mathcal{P}_{as}} Y_{aspt\omega^\dagger}, \forall a \in \mathcal{A}, s \in \mathcal{S}, i \in \mathcal{I}, t \in \mathcal{T}, \quad (5.23)$$

$$C_{asit\omega^{\dagger\dagger}}^{\omega^\dagger} + C_{asit\omega^{\dagger\dagger}}^{\dagger\omega^\dagger} \leq m_{as\omega^{\dagger\dagger}}^\dagger, \forall a \in \mathcal{A}, s \in \mathcal{S}, i \in \mathcal{I}, t \in \mathcal{T}. \quad (5.24)$$

The damages per packet are denoted by u_a , with the number of penetrating packets given by $m_{as\omega^{\dagger\dagger}}^\dagger$. This cost is offset by h_{asi} the per-packet indemnity. The claims are governed by the constraints in (5.21) and (5.22), which ensure that the numbers of packets claimed do not exceed those covered by the insurer, who can insure number of packets, g_{asi} , per policy. The constraint in (5.23) ensures that the packets were processed by security (a requirement for the insurance claims). The constraint in (5.24) ensures that they are not greater than the number of actual malicious packets.

Having defined the problem, I must now solve it. When a stochastic optimization problem has finite support, it is possible to create a deterministic equivalent formulation, which can be solved as a mixed integer problem using an established technique, such as the branch-and-bound algorithm (B&B) [30]. In my case, it is reasonable to assume that this criterion holds, and so the following set of equations provide the deterministic equivalent objective formulation:

$$\min \sum_{\omega^\dagger \in \Omega^\dagger} \pi_{\omega^\dagger} \left(\mathcal{R}^\dagger(\omega^\dagger) + \sum_{\omega^{\dagger\dagger} \in \Omega^{\dagger\dagger}(\omega^\dagger)} \pi_{\omega^{\dagger\dagger}} \left(\mathcal{R}^{\dagger\dagger}(\omega^\dagger, \omega^{\dagger\dagger}) \right) \right)$$

$$+ \sum_{\omega^{\dagger\dagger} \in \Omega^{\dagger\dagger}(\omega^\dagger, \omega^\dagger)} \pi_{\omega^{\dagger\dagger}} \mathcal{R}^{\dagger\dagger}(\omega^\dagger, \omega^\dagger, \omega^{\dagger\dagger})) \Bigg). \quad (5.25)$$

s.t. (5.1) – (5.7) holds,

$$\sum_{p \in \mathcal{P}_{as}} X_{aspt\omega^\dagger}^\dagger \leq e_{as}, \forall a \in \mathcal{A}, s \in \mathcal{S}, t \in \mathcal{T}, \omega^\dagger \in \Omega^\dagger, \quad (5.26)$$

$$\sum_{i \in \mathcal{I}} \sum_{l \in \mathcal{L}} W_{asilt\omega^\dagger}^\dagger \leq 1, \forall a \in \mathcal{A}, s \in \mathcal{S}, t \in \mathcal{T}, \omega^\dagger \in \Omega^\dagger, \quad (5.27)$$

$$Y_{aspt\omega^\dagger} \leq \sum_{\hat{t} \in \mathcal{F}_{pt}} X_{aspt\hat{t}\omega^\dagger}^\dagger, \forall a \in \mathcal{A}, s \in \mathcal{S}, p \in \mathcal{P}_{as}, t \in \mathcal{T}, \omega^\dagger \in \Omega^\dagger(\omega^\dagger), \quad (5.28)$$

$$n_{a\omega^\dagger} \leq \sum_{s \in \mathcal{S}} \left(Z_{ast\omega^\dagger} + \sum_{p \in \mathcal{P}_{as}} Y_{aspt\omega^\dagger} \right), \forall a \in \mathcal{A}, t \in \mathcal{T}, \omega^\dagger \in \Omega^\dagger(\omega^\dagger), \quad (5.29)$$

$$\sum_{l \in \mathcal{L}} (W_{asilt\omega^\dagger}^\dagger + W_{asilt\omega^\dagger}^\dagger) \leq 1, \forall a \in \mathcal{A}, s \in \mathcal{S}, i \in \mathcal{I}, t \in \mathcal{T}, \omega^\dagger \in \Omega^\dagger(\omega^\dagger), \quad (5.30)$$

$$\sum_{i \in \mathcal{I}} \sum_{l \in \mathcal{L}} W_{asilt\omega^\dagger}^\dagger \leq 1, \forall a \in \mathcal{A}, s \in \mathcal{S}, t \in \mathcal{T}, \omega^\dagger \in \Omega^\dagger(\omega^\dagger), \quad (5.31)$$

$$C_{asit\omega^{\dagger\dagger}}^{\omega^\dagger} \leq \sum_{l \in \mathcal{L}} \sum_{\hat{t} \in \mathcal{F}_{lt}} g_{asi} W_{asilt\hat{t}\omega^\dagger}^\dagger, \forall a \in \mathcal{A}, s \in \mathcal{S}, i \in \mathcal{I}, t \in \mathcal{T}, \omega^{\dagger\dagger} \in \Omega^{\dagger\dagger}(\omega^\dagger, \omega^\dagger), \quad (5.32)$$

$$C_{asit\omega^{\dagger\dagger}}^{\dagger\omega^\dagger} \leq \sum_{l \in \mathcal{L}} \sum_{\hat{t} \in \mathcal{F}_{lt}} g_{asi} W_{asilt\hat{t}\omega^\dagger}^\dagger, \forall a \in \mathcal{A}, s \in \mathcal{S}, i \in \mathcal{I}, t \in \mathcal{T}, \omega^{\dagger\dagger} \in \Omega^{\dagger\dagger}(\omega^\dagger, \omega^\dagger), \quad (5.33)$$

$$C_{asit\omega^{\dagger\dagger}}^{\omega^\dagger} + C_{asit\omega^{\dagger\dagger}}^{\dagger\omega^\dagger} \leq Z_{ast\omega^\dagger} + \sum_{p \in \mathcal{P}_{as}} Y_{aspt\omega^\dagger}, \quad \forall a \in \mathcal{A}, s \in \mathcal{S}, i \in \mathcal{I}, t \in \mathcal{T}, \omega^{\dagger\dagger} \in \Omega^{\dagger\dagger}(\omega^\dagger, \omega^\dagger), \quad (5.34)$$

$$C_{asit\omega^{\dagger\dagger}}^{\omega^\dagger} + C_{asit\omega^{\dagger\dagger}}^{\dagger\omega^\dagger} \leq m_{as\omega^{\dagger\dagger}}^{\omega^\dagger}, \forall a \in \mathcal{A}, s \in \mathcal{S}, i \in \mathcal{I}, t \in \mathcal{T}, \omega^{\dagger\dagger} \in \Omega^{\dagger\dagger}(\omega^\dagger, \omega^\dagger). \quad (5.35)$$

5.3 Partial Lagrange Multiplier Method

The mixed integer programming deterministic equivalent formulation is an effective way of finding the optimum solution to the stochastic optimization, but suffers from tractability problems, as I would normally use a technique such as branch-and-bound, which has exponential computational complexity [30]. However, if I can convert my problem to a linear program (LP) without loss of precision, I can solve it in polynomial time. In this section I introduce the partial Lagrange multiplier method [31], which solves a sequence of LPs with at worst a polynomial number of steps.

5.3.1 Total Unimodularity of Constraint Matrix

To solve the tractability problem, I take advantage of the Total Unimodularity (TU) property [120]. If the set of constraints in (5.11)-(5.24) are considered as a matrix $\mathbf{M}$ and vector $\vec{\mathbf{b}}$, as in standard LP notation (I give an example of standard notation in (5.55)), then if $\vec{\mathbf{b}}$ is all-integer (which in my case it is), and $\mathbf{M}$ satisfies TU, I can relax all integer variables to linear ones and solve as an LP in polynomial time while guaranteeing integer variable solutions.

In my case, however, $\mathbf{M}$ only partially satisfies TU, which I can establish according to Theorem 3 in [29]. The matrix $\mathbf{M}$ is trimmed by removing the constraints that obviously violate TU, for example, those with coefficients not equal to -1 , 0 , or 1 . The resultant matrix, $\mathbf{M}'$ can then be divided into two disjoint subsets, one containing the rows corresponding to constraints (5.12), (5.15), (5.17), and (5.24), and the other containing (5.11), (5.16), and (5.18). It is clear from these constraints that the conditions for unimodularity given in [29] are satisfied for the matrix, $\mathbf{M}'$, and that it will also hold for every square, non-singular submatrix, which is the criterion for unimodularity to be ‘total’. Therefore, total unimodularity is satisfied for $\mathbf{M}'$. The remaining constraints, which do not satisfy TU, mean that it is necessary to still use an inefficient method, therefore I resort to the partial Lagrange multiplier method, which is described as follows.

5.3.2 Partial Lagrange Multiplier

The full Lagrange multiplier method is an established technique for solving convex optimization problems using linear constraints. In the vector notation of standard form, the problems are written as follows:

$$\min_{\vec{\mathbf{x}}} f_0(\vec{\mathbf{x}}) \quad \left| \begin{array}{l} f_i(\vec{\mathbf{x}}) \leq 0, \quad i = 1, \dots, m; \\ h_i(\vec{\mathbf{x}}) = 0, \quad i = 1, \dots, p, \end{array} \right. \quad (5.36)$$

where $\vec{\mathbf{x}} \in \mathbb{R}^n$. The constraints in (5.36) can be moved into the objective function weighted by Lagrange multipliers, as in the following equation:

$$L(\vec{x}, \vec{\lambda}, \vec{v}) = f_0(\vec{x}) + \sum_{i=1}^m \lambda_i f_i(\vec{x}) + \sum_{i=1}^p v_i h_i(\vec{x}), \quad (5.37)$$

where $\vec{\lambda} \in \mathbb{R}^m$ and $\vec{v} \in \mathbb{R}^p$ are the Lagrange multipliers. The dual function, expressed in $\vec{\lambda}$ and $\vec{v}$ is therefore:

$$g(\vec{\lambda}, \vec{v}) = \min_{\vec{x} \in \mathbb{R}^n} L(\vec{x}, \vec{\lambda}, \vec{v}), \quad (5.38)$$

and the corresponding dual problem can be written:

$$\max_{\vec{\lambda}, \vec{v}} g(\vec{\lambda}, \vec{v}) \mid \vec{\lambda} \geq 0. \quad (5.39)$$

When used for convex optimization problems, the dual problem can be solved for $\vec{\lambda}$ and $\vec{v}$ via the subgradient method. The corresponding values found by the dual problem for $\vec{x}$ are also the optimal values for the primal problem, due to the convexity of the primal problem. The partial Lagrange multiplier method for linear programs adopts a similar strategy, with the constraints that do not satisfy TU being moved into the objective function with the set of $\vec{\lambda}$ Lagrange multipliers. The remaining, TU-satisfying, constraints remain as constraints in the problem, allowing it to be solved as an LP. Therefore, applying this to the objective function in (5.25) along with the remaining constraints in (5.21)-(5.23), I get the following dual function in terms of $\vec{\lambda}$:

$$\begin{aligned} g(\vec{\lambda}) = & \sum_{\omega^\dagger} \pi_{\omega^\dagger} \left(\mathcal{R}^\dagger(\omega^\dagger) + \sum_{\omega^\ddagger} \pi_{\omega^\ddagger} \left(\mathcal{R}^\ddagger(\omega^\dagger, \omega^\ddagger) + \sum_{\omega^{\ddagger\ddagger}} \pi_{\omega^{\ddagger\ddagger}} \mathcal{R}^{\ddagger\ddagger}(\omega^\dagger, \omega^\ddagger, \omega^{\ddagger\ddagger}) \right) \right) \\ & + \sum_{a,s,i,t,\omega^\dagger,\omega^\ddagger,\omega^{\ddagger\ddagger}} \lambda_{asit\omega^\dagger\omega^\ddagger\omega^{\ddagger\ddagger}}^1 \left(C_{asit\omega^{\ddagger\ddagger}}^{\omega^\ddagger} - \sum_{l,t \in \mathcal{F}_{lt}} g_{asi} W_{asilt\omega^\dagger}^\dagger \right) \\ & + \sum_{a,s,i,t,\omega^\dagger,\omega^\ddagger,\omega^{\ddagger\ddagger}} \lambda_{asit\omega^\dagger\omega^\ddagger\omega^{\ddagger\ddagger}}^2 \left(C_{asit\omega^{\ddagger\ddagger}}^{\dagger\omega^\ddagger} - \sum_{l,t \in \mathcal{F}_{lt}} g_{asi} W_{asilt\omega^\dagger}^\dagger \right) \\ & + \sum_{a,s,i,t,\omega^\dagger,\omega^\ddagger,\omega^{\ddagger\ddagger}} \lambda_{asit\omega^\dagger\omega^\ddagger\omega^{\ddagger\ddagger}}^3 \left((C_{asit\omega^{\ddagger\ddagger}}^{\omega^\ddagger} + C_{asit\omega^{\ddagger\ddagger}}^{\dagger\omega^\ddagger}) - (Z_{ast\omega^\dagger} + \sum_{p \in \mathcal{P}_{as}} Y_{aspt\omega^\dagger}) \right). \end{aligned} \quad (5.40)$$

In this formulation, I use an abbreviated notation for the sake of brevity and readability, where $\sum_{a,s}$ is equivalent to writing $\sum_{a \in \mathcal{A}} \sum_{s \in \mathcal{S}}$.

5.3.3 Solution to the Partial Lagrange Multiplier

To solve my optimization problem using the partial Lagrange multiplier method, I iteratively minimize the dual problem in terms of $\vec{\lambda}$, given by the objective function in (5.40):

$$\min g(\vec{\lambda}), \quad (5.41)$$

subject to the TU-satisfying constraints in (5.11)-(5.12), (5.15)-(5.18), and (5.24). I also relax the variable range constraints to be linear intervals.

5.3.3.1 Dual Problem Solution

Since the outlined dual problem can be solved as an LP with variables guaranteed to give integer values, I can solve it in polynomial time using the Interior Point Method (IPM) [30]. In each iteration of the algorithm, I solve the dual problem, and use the new values of the primal variables to determine the value of the original primal objective function. This value is then used to calculate the Lagrange multipliers for the next iteration of the algorithm.

5.3.3.2 Updating Lagrange Multipliers

The behaviour of the algorithm is governed by the updated values of the Lagrange multipliers. This calculation is governed by the stepsize α_k for a given step k . The Lagrange multipliers are updated by adding the old value to the gradient of the dual function weighted by the step size, or 0, whichever is larger:

$$\vec{\lambda}^{(k+1)} = \vec{\lambda}^{(k)} + \alpha_k \cdot \nabla g(\vec{\lambda}^{(k)}). \quad (5.42)$$

The stepsize, α_k can be calculated in a variety of ways, for example, as a constant value, $1/k$, or $1/\sqrt{k}$. The ideal value will depend on the problem being solved, with a calculation dependent on k giving a decreasing stepsize as k increases, providing greater precision. A constant value may converge more rapidly, but may also result in oscillation between two fixed values, of which one is the optimum. The function

$\nabla g(\vec{\lambda}^{(k)})$ is the derivative of the dual function with respect to $\vec{\lambda}$ and, when derived from (5.40) is as follows:

$$\begin{aligned} \nabla g(\vec{\lambda}) = & \sum_{a,s,i,t,\omega^\dagger,\omega^\ddagger,\omega^{\ddagger\ddagger}} \left(C_{asit\omega^{\ddagger\ddagger}}^{\omega^\ddagger} - \sum_{l,\hat{t} \in \mathcal{F}_{lt}} g_{asi} W_{asil\hat{t}\omega^\dagger}^\dagger \right) \\ & + \sum_{a,s,i,t,\omega^\dagger,\omega^\ddagger,\omega^{\ddagger\ddagger}} \left(C_{asit\omega^{\ddagger\ddagger}}^{\dagger\omega^\ddagger} - \sum_{l,\hat{t} \in \mathcal{F}_{lt}} g_{asi} W_{asil\hat{t}\omega^\ddagger}^\ddagger \right) \\ & + \sum_{a,s,i,t,\omega^\dagger,\omega^\ddagger,\omega^{\ddagger\ddagger}} \left((C_{asit\omega^{\ddagger\ddagger}}^{\omega^\ddagger} + C_{asit\omega^{\ddagger\ddagger}}^{\dagger\omega^\ddagger}) - (Z_{ast\omega^\ddagger} + \sum_{p \in \mathcal{P}_{as}} Y_{aspt\omega^\ddagger}) \right). \end{aligned} \quad (5.43)$$

5.3.3.3 Stopping Criterion

The partial Lagrange multiplier method will always converge to the optimal solution, as I prove in Section 5.4. To determine when the solution has been reached, I use the stopping criterion. If the best value for the primal objective does not improve by more than the marginal value ϵ and Φ iterations have passed since the best value was found, then the algorithm terminates. If ϵ is sufficiently small and Φ is sufficiently large, the best value on termination would be guaranteed to be the optimum. However, the user may choose values for ϵ and Φ that suit their desired tradeoff between accuracy and runtime. The pseudocode for the partial Lagrange multiplier algorithm outlined in this section, with the described stopping criterion, is given in Algorithm 1.

Algorithm 1 Partial Lagrange Multiplier Method

Input : Dual problem $g(\vec{\lambda})$, Φ

Output: Optimal solution for $X^\dagger$, $W^\dagger$, $W^\ddagger$, Y , Z , C , $C^\ddagger$

Initialize $g_{best}^{(0)} = -\infty$;

Set the iteration index: $k = 1, k_x = 0$;

Initialize $\vec{\lambda}$ values ≥ 0 , upper bounds given in (5.48) ;

do

 Find optimal solution for $g(\vec{\lambda})$, given in Eq. (5.40) ;

 Compute solution of primal problem ;

if $g_{best}^{(k)} - g_{best}^{(k-1)} \geq \epsilon$ **then** $k_x = k$;

$k = k + 1$;

 Compute step size α_k ;

 Update Lagrange multipliers using Eq. (5.42) and (5.43);

while $k - k_x \leq \Phi$;

5.4 Theoretical Performance Analysis

In this section I demonstrate analytically the convergence and scalability of the partial Lagrange multiplier method.

5.4.1 Convergence Analysis

To prove the convergence of the algorithm, I need to prove three elements. Firstly, I demonstrate that the dual function is concave. Secondly, since the function is concave, I can demonstrate that the algorithm will converge to the optimum by iteratively updating $\vec{\lambda}$. Finally I show that I can guarantee strong duality, thus the solution to the dual problem found by the algorithm, is also the optimal solution to the corresponding primal problem.

5.4.1.1 Concavity Proof

The dual function is given in (5.40), subject to the constraints in (5.11)-(5.12), (5.15)-(5.18), and (5.24). To show concavity, I introduce the parameter θ ($0 \leq \theta \leq 1$) and can write the equations $\theta \cdot g(\vec{\lambda})$ and $(1 - \theta) \cdot g(\vec{\lambda})$. I introduce notations $\vec{\lambda}^1$ and $\vec{\lambda}^2$, which indicate two versions of the vector of Lagrange multipliers. Combined with the notation from (5.40) I then have, for example, $\lambda_{asit\omega^\dagger\omega^\ddagger\omega^{\ddagger\ddagger}}^{2.1}$ to indicate the Lagrange multiplier $\lambda_{asit\omega^\dagger\omega^\ddagger\omega^{\ddagger\ddagger}}^1$ from the set of multipliers, $\vec{\lambda}^2$. The equations are therefore written as follows:

$$\begin{aligned} \theta \cdot g(\vec{\lambda}^1) = \min \theta & \left[\sum_{\omega^\dagger} \pi_{\omega^\dagger} \left(\mathcal{R}^\dagger(\omega^\dagger) + \sum_{\omega^\ddagger} \pi_{\omega^\ddagger} \left(\mathcal{R}^\ddagger(\omega^\dagger, \omega^\ddagger) \right. \right. \right. \\ & \left. \left. \left. + \sum_{\omega^{\ddagger\ddagger}} \pi_{\omega^{\ddagger\ddagger}} \mathcal{R}^{\ddagger\ddagger}(\omega^\dagger, \omega^\ddagger, \omega^{\ddagger\ddagger}) \right) \right) \right] \\ & + \theta \left[\sum_{a,s,i,t,\omega^\dagger,\omega^\ddagger,\omega^{\ddagger\ddagger}} \lambda_{asit\omega^\dagger\omega^\ddagger\omega^{\ddagger\ddagger}}^{1.1} \left(C_{asit\omega^{\ddagger\ddagger}}^{\omega^\ddagger} - \sum_{l,\hat{t} \in \mathcal{F}_{lt}} g_{asi} W_{asil\hat{t}\omega^\dagger}^\dagger \right) \right. \\ & + \sum_{a,s,i,t,\omega^\dagger,\omega^\ddagger,\omega^{\ddagger\ddagger}} \lambda_{asit\omega^\dagger\omega^\ddagger\omega^{\ddagger\ddagger}}^{1.2} \left(C_{asit\omega^{\ddagger\ddagger}}^{\ddagger\omega^\ddagger} - \sum_{l,\hat{t} \in \mathcal{F}_{lt}} g_{asi} W_{asil\hat{t}\omega^\ddagger}^\ddagger \right) \\ & \left. + \sum_{a,s,i,t,\omega^\dagger,\omega^\ddagger,\omega^{\ddagger\ddagger}} \lambda_{asit\omega^\dagger\omega^\ddagger\omega^{\ddagger\ddagger}}^{1.3} \left((C_{asit\omega^{\ddagger\ddagger}}^{\omega^\ddagger} + C_{asit\omega^{\ddagger\ddagger}}^{\ddagger\omega^\ddagger}) - (Z_{ast\omega^\ddagger} + \sum_{p \in \mathcal{P}_{as}} Y_{aspt\omega^\ddagger}) \right) \right], \quad (5.44) \end{aligned}$$

$$\begin{aligned}
(1 - \theta) \cdot g(\vec{\lambda}^2) = & \min(1 - \theta) \left[\sum_{\omega^\dagger} \pi_{\omega^\dagger} \left(\mathcal{R}^\dagger(\omega^\dagger) + \sum_{\omega^\ddagger} \pi_{\omega^\ddagger} \left(\mathcal{R}^\ddagger(\omega^\dagger, \omega^\ddagger) \right. \right. \right. \\
& \left. \left. \left. + \sum_{\omega^{\ddagger\ddagger}} \pi_{\omega^{\ddagger\ddagger}} \mathcal{R}^{\ddagger\ddagger}(\omega^\dagger, \omega^\ddagger, \omega^{\ddagger\ddagger}) \right) \right) \right] \\
& + (1 - \theta) \left[\sum_{a,s,i,t,\omega^\dagger,\omega^\ddagger,\omega^{\ddagger\ddagger}} \lambda_{asit\omega^\dagger\omega^\ddagger\omega^{\ddagger\ddagger}}^{2.1} \left(C_{asit\omega^{\ddagger\ddagger}}^{\omega^\dagger} - \sum_{l,\hat{t} \in \mathcal{F}_{lt}} g_{asi} W_{asilt\omega^\dagger}^\dagger \right) \right. \\
& + \sum_{a,s,i,t,\omega^\dagger,\omega^\ddagger,\omega^{\ddagger\ddagger}} \lambda_{asit\omega^\dagger\omega^\ddagger\omega^{\ddagger\ddagger}}^{2.2} \left(C_{asit\omega^{\ddagger\ddagger}}^{\dagger\omega^\ddagger} - \sum_{l,\hat{t} \in \mathcal{F}_{lt}} g_{asi} W_{asilt\omega^\ddagger}^\dagger \right) \\
& \left. + \sum_{a,s,i,t,\omega^\dagger,\omega^\ddagger,\omega^{\ddagger\ddagger}} \lambda_{asit\omega^\dagger\omega^\ddagger\omega^{\ddagger\ddagger}}^{2.3} \left((C_{asit\omega^{\ddagger\ddagger}}^{\omega^\dagger} + C_{asit\omega^{\ddagger\ddagger}}^{\dagger\omega^\ddagger}) - (Z_{ast\omega^\ddagger} + \sum_{p \in \mathcal{P}_{as}} Y_{aspt\omega^\ddagger}) \right) \right]. \quad (5.45)
\end{aligned}$$

These two equations can then be combined into a single expression, $g(\theta\vec{\lambda}^1 + (1 - \theta)\vec{\lambda}^2)$, which allows us to write the following derivation:

$$\begin{aligned}
g(\theta\vec{\lambda}^1 + (1 - \theta)\vec{\lambda}^2) = & \min \sum_{\omega^\dagger} \pi_{\omega^\dagger} \left(\mathcal{R}^\dagger(\omega^\dagger) + \sum_{\omega^\ddagger} \pi_{\omega^\ddagger} \left(\mathcal{R}^\ddagger(\omega^\dagger, \omega^\ddagger) + \sum_{\omega^{\ddagger\ddagger}} \pi_{\omega^{\ddagger\ddagger}} \mathcal{R}^{\ddagger\ddagger}(\omega^\dagger, \omega^\ddagger, \omega^{\ddagger\ddagger}) \right) \right) \\
& + \sum_{a,s,i,t,\omega^\dagger,\omega^\ddagger,\omega^{\ddagger\ddagger}} \left(\theta \lambda_{asit\omega^\dagger\omega^\ddagger\omega^{\ddagger\ddagger}}^{1.1} + (1 - \theta) \lambda_{asit\omega^\dagger\omega^\ddagger\omega^{\ddagger\ddagger}}^{2.1} \right) \left(C_{asit\omega^{\ddagger\ddagger}}^{\omega^\dagger} - \sum_{l,\hat{t} \in \mathcal{F}_{lt}} g_{asi} W_{asilt\omega^\dagger}^\dagger \right) \\
& + \sum_{a,s,i,t,\omega^\dagger,\omega^\ddagger,\omega^{\ddagger\ddagger}} \left(\theta \lambda_{asit\omega^\dagger\omega^\ddagger\omega^{\ddagger\ddagger}}^{1.2} + (1 - \theta) \lambda_{asit\omega^\dagger\omega^\ddagger\omega^{\ddagger\ddagger}}^{2.2} \right) \left(C_{asit\omega^{\ddagger\ddagger}}^{\dagger\omega^\ddagger} - \sum_{l,\hat{t} \in \mathcal{F}_{lt}} g_{asi} W_{asilt\omega^\ddagger}^\dagger \right) \\
& + \sum_{a,s,i,t,\omega^\dagger,\omega^\ddagger,\omega^{\ddagger\ddagger}} \left(\theta \lambda_{asit\omega^\dagger\omega^\ddagger\omega^{\ddagger\ddagger}}^{1.3} + (1 - \theta) \lambda_{asit\omega^\dagger\omega^\ddagger\omega^{\ddagger\ddagger}}^{2.3} \right) \left((C_{asit\omega^{\ddagger\ddagger}}^{\omega^\dagger} + C_{asit\omega^{\ddagger\ddagger}}^{\dagger\omega^\ddagger}) \right. \\
& \left. - (Z_{ast\omega^\ddagger} + \sum_{p \in \mathcal{P}_{as}} Y_{aspt\omega^\ddagger}) \right) \\
= & \min(\theta + (1 - \theta)) \left[\sum_{\omega^\dagger} \pi_{\omega^\dagger} \left(\mathcal{R}^\dagger(\omega^\dagger) + \sum_{\omega^\ddagger} \pi_{\omega^\ddagger} \left(\mathcal{R}^\ddagger(\omega^\dagger, \omega^\ddagger) \right. \right. \right. \\
& \left. \left. \left. + \sum_{\omega^{\ddagger\ddagger}} \pi_{\omega^{\ddagger\ddagger}} \mathcal{R}^{\ddagger\ddagger}(\omega^\dagger, \omega^\ddagger, \omega^{\ddagger\ddagger}) \right) \right) \right] \\
& + \sum_{a,s,i,t,\omega^\dagger,\omega^\ddagger,\omega^{\ddagger\ddagger}} \left(\theta \lambda_{asit\omega^\dagger\omega^\ddagger\omega^{\ddagger\ddagger}}^{1.1} + (1 - \theta) \lambda_{asit\omega^\dagger\omega^\ddagger\omega^{\ddagger\ddagger}}^{2.1} \right) \left(C_{asit\omega^{\ddagger\ddagger}}^{\omega^\dagger} - \sum_{l,\hat{t} \in \mathcal{F}_{lt}} g_{asi} W_{asilt\omega^\dagger}^\dagger \right) \\
& + \sum_{a,s,i,t,\omega^\dagger,\omega^\ddagger,\omega^{\ddagger\ddagger}} \left(\theta \lambda_{asit\omega^\dagger\omega^\ddagger\omega^{\ddagger\ddagger}}^{1.2} + (1 - \theta) \lambda_{asit\omega^\dagger\omega^\ddagger\omega^{\ddagger\ddagger}}^{2.2} \right) \left(C_{asit\omega^{\ddagger\ddagger}}^{\dagger\omega^\ddagger} - \sum_{l,\hat{t} \in \mathcal{F}_{lt}} g_{asi} W_{asilt\omega^\ddagger}^\dagger \right) \\
& + \sum_{a,s,i,t,\omega^\dagger,\omega^\ddagger,\omega^{\ddagger\ddagger}} \left(\theta \lambda_{asit\omega^\dagger\omega^\ddagger\omega^{\ddagger\ddagger}}^{1.3} + (1 - \theta) \lambda_{asit\omega^\dagger\omega^\ddagger\omega^{\ddagger\ddagger}}^{2.3} \right) \left((C_{asit\omega^{\ddagger\ddagger}}^{\omega^\dagger} + C_{asit\omega^{\ddagger\ddagger}}^{\dagger\omega^\ddagger}) \right)
\end{aligned}$$

$$\begin{aligned}
& - (Z_{ast\omega^\dagger} + \sum_{p \in \mathcal{P}_{as}} Y_{aspt\omega^\dagger}) \\
= & \min \theta \left[\sum_{\omega^\dagger} \pi_{\omega^\dagger} \left(\mathcal{R}^\dagger(\omega^\dagger) + \sum_{\omega^\ddagger} \pi_{\omega^\ddagger} \left(\mathcal{R}^\ddagger(\omega^\dagger, \omega^\ddagger) + \sum_{\omega^{\ddagger\ddagger}} \pi_{\omega^{\ddagger\ddagger}} \mathcal{R}^{\ddagger\ddagger}(\omega^\dagger, \omega^\ddagger, \omega^{\ddagger\ddagger}) \right) \right) \right. \\
& + \sum_{a,s,i,t,\omega^\dagger,\omega^\ddagger,\omega^{\ddagger\ddagger}} \lambda_{asit\omega^\dagger\omega^\ddagger\omega^{\ddagger\ddagger}}^{1.1} \left(C_{asit\omega^{\ddagger\ddagger}}^{\omega^\dagger} - \sum_{l,\hat{l} \in \mathcal{F}_{lt}} g_{asi} W_{asilt\omega^\dagger}^\dagger \right) \\
& + \sum_{a,s,i,t,\omega^\dagger,\omega^\ddagger,\omega^{\ddagger\ddagger}} \lambda_{asit\omega^\dagger\omega^\ddagger\omega^{\ddagger\ddagger}}^{1.2} \left(C_{asit\omega^{\ddagger\ddagger}}^{\dagger\omega^\dagger} - \sum_{l,\hat{l} \in \mathcal{F}_{lt}} g_{asi} W_{asilt\omega^\dagger}^\dagger \right) \\
& + \left. \sum_{a,s,i,t,\omega^\dagger,\omega^\ddagger,\omega^{\ddagger\ddagger}} \lambda_{asit\omega^\dagger\omega^\ddagger\omega^{\ddagger\ddagger}}^{1.3} \left((C_{asit\omega^{\ddagger\ddagger}}^{\omega^\dagger} + C_{asit\omega^{\ddagger\ddagger}}^{\dagger\omega^\dagger}) - (Z_{ast\omega^\dagger} + \sum_{p \in \mathcal{P}_{as}} Y_{aspt\omega^\dagger}) \right) \right] \\
& + (1 - \theta) \left[\sum_{\omega^\dagger} \pi_{\omega^\dagger} \left(\mathcal{R}^\dagger(\omega^\dagger) + \sum_{\omega^\ddagger} \pi_{\omega^\ddagger} \left(\mathcal{R}^\ddagger(\omega^\dagger, \omega^\ddagger) + \sum_{\omega^{\ddagger\ddagger}} \pi_{\omega^{\ddagger\ddagger}} \mathcal{R}^{\ddagger\ddagger}(\omega^\dagger, \omega^\ddagger, \omega^{\ddagger\ddagger}) \right) \right) \right. \\
& + \sum_{a,s,i,t,\omega^\dagger,\omega^\ddagger,\omega^{\ddagger\ddagger}} \lambda_{asit\omega^\dagger\omega^\ddagger\omega^{\ddagger\ddagger}}^{2.1} \left(C_{asit\omega^{\ddagger\ddagger}}^{\omega^\dagger} - \sum_{l,\hat{l} \in \mathcal{F}_{lt}} g_{asi} W_{asilt\omega^\dagger}^\dagger \right) \\
& + \sum_{a,s,i,t,\omega^\dagger,\omega^\ddagger,\omega^{\ddagger\ddagger}} \lambda_{asit\omega^\dagger\omega^\ddagger\omega^{\ddagger\ddagger}}^{2.2} \left(C_{asit\omega^{\ddagger\ddagger}}^{\dagger\omega^\dagger} - \sum_{l,\hat{l} \in \mathcal{F}_{lt}} g_{asi} W_{asilt\omega^\dagger}^\dagger \right) \\
& + \left. \sum_{a,s,i,t,\omega^\dagger,\omega^\ddagger,\omega^{\ddagger\ddagger}} \lambda_{asit\omega^\dagger\omega^\ddagger\omega^{\ddagger\ddagger}}^{2.3} \left((C_{asit\omega^{\ddagger\ddagger}}^{\omega^\dagger} + C_{asit\omega^{\ddagger\ddagger}}^{\dagger\omega^\dagger}) - (Z_{ast\omega^\dagger} + \sum_{p \in \mathcal{P}_{as}} Y_{aspt\omega^\dagger}) \right) \right]. \quad (5.46)
\end{aligned}$$

Given that from [114]:

$$\min_{p_i \in \vec{p}} \{p_i\} + \min_{q_i \in \vec{q}} \{q_i\} \leq \min_{e_i \in \vec{e} = \vec{p} + \vec{q}} \{e_i\}, \quad (5.47)$$

then taking equations (5.44)-(5.46) I can say that $\theta \cdot g(\vec{\lambda}^1) + (1 - \theta)g(\vec{\lambda}^2) \leq g(\theta\vec{\lambda}^1 + (1 - \theta)\vec{\lambda}^2)$. Therefore I have shown that $g(\vec{\lambda})$ is concave.

5.4.1.2 Proof of Convergence

Since I have proved that $g(\vec{\lambda})$ is concave, I can assume that $\nabla g(\vec{\lambda}) \leq G$, where G is a value that I define in (5.53) and (5.54). Further, each individual Lagrange multiplier value, denoted by λ_i , is bounded above, as well as by the defined bound of $\lambda_i \geq 0$. I can derive the upper bounds of $\vec{\lambda}$ from Eq. (5.40), giving the following limits:

$$\begin{aligned}
\lambda_{asit\omega^\dagger\omega^\ddagger\omega^{\ddagger\ddagger}}^1 & \leq \frac{\pi_{\omega^\dagger} f_{asilt\omega^\dagger}^\dagger}{g_{asi}} \\
\lambda_{asit\omega^\dagger\omega^\ddagger\omega^{\ddagger\ddagger}}^2 & \leq \frac{\pi_{\omega^\ddagger} f_{asilt\omega^\ddagger}^\dagger}{g_{asi}} \\
\lambda_{asit\omega^\dagger\omega^\ddagger\omega^{\ddagger\ddagger}}^3 & \leq \pi_{\omega^\ddagger} k_{as\omega^\ddagger}^\dagger.
\end{aligned} \quad (5.48)$$

Given these upper bounds, I can say $\|\vec{\lambda}^{(1)} - \vec{\lambda}^*\|_2 \leq R$, where $\vec{\lambda}^{(1)}$ denotes the Lagrange multipliers in the first step, $\vec{\lambda}^*$ denotes the optimal values for the Lagrange multipliers, and R is a value that I will define in Section 5.4.2. With this established, I can write the convergence proof from [31], which I include here as follows. For a concave function, $g(\vec{\lambda}^*) \leq g(\vec{\lambda}^{(k)}) + \nabla g(\vec{\lambda}^{(k)})(\vec{\lambda}^* - \vec{\lambda}^{(k)})$ is true, we can write the following derivation:

$$\begin{aligned} \|\vec{\lambda}^{(k+1)} - \vec{\lambda}^*\|_2^2 &= \|\vec{\lambda}^{(k)} + \alpha_k \nabla g(\vec{\lambda}^{(k)}) - \vec{\lambda}^*\|_2^2 \\ &= \|\vec{\lambda}^{(k)} - \vec{\lambda}^*\|_2^2 + 2\alpha_k \nabla g(\vec{\lambda}^{(k)})(\vec{\lambda}^{(k)} - \vec{\lambda}^*) + \alpha_k^2 \|\nabla g(\vec{\lambda}^{(k)})\|_2^2 \\ &\leq \|\vec{\lambda}^{(k)} - \vec{\lambda}^*\|_2^2 + 2\alpha_k (g(\vec{\lambda}^{(k)}) - g(\vec{\lambda}^*)) + \alpha_k^2 \|\nabla g(\vec{\lambda}^{(k)})\|_2^2, \end{aligned} \quad (5.49)$$

$$\begin{aligned} \|\vec{\lambda}^{(k+1)} - \vec{\lambda}^*\|_2^2 &\leq \|\vec{\lambda}^{(1)} - \vec{\lambda}^*\|_2^2 + 2 \sum_{i=1}^k \alpha_i (g(\vec{\lambda}^{(i)}) - g(\vec{\lambda}^*)) + \sum_{i=1}^k \alpha_i^2 \|\nabla g(\vec{\lambda}^{(i)})\|_2^2 \\ &\leq R^2 + 2 \sum_{i=1}^k \alpha_i (g(\vec{\lambda}^{(i)}) - g(\vec{\lambda}^*)) + G \sum_{i=1}^k \alpha_i^2. \end{aligned} \quad (5.50)$$

The partial Lagrange algorithm keeps track of the best result obtained, which means that after k iterations, we have the result:

$$\sum_{i=1}^k \alpha_i (g(\vec{\lambda}^{(i)}) - g(\vec{\lambda}^*)) \leq (g_{best}^{(k)} - g(\vec{\lambda}^*)) \left(\sum_{i=1}^k \alpha_i \right), \quad (5.51)$$

which when combined with the bounds given above, also gives the result:

$$g_{best}^{(k)} - g(\vec{\lambda}^*) \leq \frac{R^2 + G^2 \sum_{i=1}^k \alpha_i^2}{2 \sum_{i=1}^k \alpha_i}. \quad (5.52)$$

This proof is a known result from convex optimization, and thus I know that $g_{best}^{(k)}$, which denotes the best solution value for $g(\vec{\lambda})$ at step k , will converge to $g(\vec{\lambda}^*)$, the optimal solution of the dual problem. For the optimal version of the stepsize calculation $\alpha_k = (R/G)/\sqrt{k}$, the number of iterations required to do so will be bounded above by $K \leq (RG/\epsilon)^2$, where ϵ is the error margin covered under Section 5.3.3.3 on the stopping criterion.

5.4.1.3 Strong Duality

I have shown that the partial Lagrange multiplier algorithm will converge to an optimal solution of the dual problem. However, I must also show that this solution is optimal for the original primal problem. Since I have satisfied the property of total unimodularity, I am guaranteed an integer solution. By the Slater's constraint qualification, if the optimization problem is convex, and there is at least one solution that is strictly feasible (which it is not difficult to find), then the solution to the Lagrange dual problem is equivalent to the primal solution [114]. Therefore if $g(\vec{\lambda}^*)$ is the optimal solution to the dual problem, the corresponding values of the decision variables are the optimal solution to the original mixed integer optimization problem.

5.4.2 Scalability Analysis

I have proved that the partial Lagrange multiplier method will converge within a maximum of $K \leq (RG/\epsilon)^2$ iterations. I have already established that, using vector notation, $\|\vec{\lambda}^{(1)} - \vec{\lambda}^*\|_2 \leq R$. Given the upper bounds on the Lagrange multipliers in (5.48), I can reasonably assume that, given the probability multipliers and larger denominators giving upper bounds ≤ 1 , $R = |\vec{\lambda}|$, where $|\vec{\lambda}|$ denotes the total number of Lagrange multipliers. Additionally, I have also given the bound $\|\nabla g(\vec{\lambda})\|_2 \leq G$, with $g(\vec{\lambda})$ given in (5.43). If the expressions $(C_{asit\omega}^{\omega\dagger} - \sum_{l,\hat{t} \in \mathcal{F}_{lt}} g_{asi} W_{asil\hat{t}\omega}^{\dagger}) \geq 0$, $(C_{asit\omega}^{\dagger\omega} - \sum_{l,\hat{t} \in \mathcal{F}_{lt}} g_{asi} W_{asil\hat{t}\omega}^{\dagger}) \geq 0$, and $(C_{asit\omega}^{\omega\dagger} + C_{asit\omega}^{\dagger\omega} - (Z_{ast\omega}^{\dagger} + \sum_{p \in \mathcal{P}_{as}} Y_{aspt\omega}^{\dagger})) \geq 0$, then:

$$\|\nabla g(\vec{\lambda})\|_2 \leq \|\vec{c}\|_2 = G, \quad (5.53)$$

where $\vec{c}$ is the vector representation of the claim variables. Conversely, if the expressions given above from (5.43) are less than 0, then I have:

$$\|\nabla g(\vec{\lambda})\|_2 \leq \|\mathbf{W}\vec{u}\|_2 = G, \quad (5.54)$$

where $\vec{u}$ is the vector representation of the security utilization and insurance provisioning variables, while $\mathbf{W}$ is the matrix representation of their coefficients in the above equations. Substituting these values for R and G into the inequality $K \leq (RG/\epsilon)^2$, I

have a function that is bounded by a linear function of the problem size. The solution to the dual problem is the main point of complexity in the partial Lagrange multiplier algorithm, which can be solved in polynomial time using the IPM [30]. Therefore I have a worst case of polynomial execution time for the partial Lagrange multiplier algorithm.

The proofs establishing convergence and scalability in this chapter follow the same reasoning as applied to the algorithm in Chapter 4, but the exact values taken by the parameters in $K \leq (RG/\epsilon)^2$ and the upper limits on the values of the Lagrange multipliers are different as they are determined by the parameters of the problem.

5.5 Sensitivity Analysis

Stochastic optimization finds the optimal solution to a programming problem given uncertain parameters. In [104], an analytical sensitivity analysis was introduced to find the precise breakpoints for parameters at which the corresponding variable allocations change. This technique is specifically for linear programs. Therefore, I use the best result from the partial Lagrange multiplier algorithm, fixing the values of the Lagrange multipliers at their optimal values. The resulting LP fits the standard form:

$$\min_{\vec{x}} \{ \vec{c}^\top \vec{x} : \mathbf{A} \vec{x} \geq \vec{b}, \vec{x} \geq 0 \}, \quad (5.55)$$

and accompanying dual problem is of the form:

$$\max_{\vec{y}, \vec{s}} \{ \vec{b}^\top \vec{y} : \mathbf{A}^\top \vec{y} + \vec{s} \leq \vec{c}, \vec{s} \geq 0 \}. \quad (5.56)$$

The resulting dual objective is as follows, with each dual variable denoted by D , where the superscript gives the equation number corresponding to the relevant constraint:

$$\max \sum_{a,s,t,\omega^\dagger} D_{ast\omega^\dagger}^{(5.12)} - \sum_{a,t,\omega^\ddagger} n_{a\omega^\ddagger} D_{a,t,\omega^\ddagger}^{(5.16)} + \sum_{a,s,i,t,\omega^\ddagger} D_{asit\omega^\ddagger}^{(5.17)} + \sum_{a,s,t,\omega^\ddagger} D_{ast\omega^\ddagger}^{(5.18)}$$

$$+ \sum_{a,s,\omega^{\dagger\dagger}} m_{as\omega^{\dagger\dagger}}^{\omega^{\dagger}} D_{asit\omega^{\dagger\dagger}}^{(5.24)} + \sum_{a,s,t,\omega^{\dagger}} e_{as} D_{ast\omega^{\dagger}}^{(5.11)}. \quad (5.57)$$

The dual constraints are then as follows:

$$D_{ast\omega^{\dagger}}^{(5.11)} - D_{aspt\omega^{\dagger}}^{(5.15)} \geq \pi_{\omega^{\dagger}} k_{aspt\omega^{\dagger}}^{\dagger}, \forall a \in \mathcal{A}, s \in \mathcal{S}, p \in \mathcal{P}_{as}, t \in \mathcal{T}, \omega^{\dagger} \in \Omega^{\dagger}, \quad (5.58)$$

$$D_{ast\omega^{\dagger}}^{(5.12)} + D_{asit\omega^{\dagger}}^{(5.17)} \geq \pi_{\omega^{\dagger}} f_{asilt\omega^{\dagger}}^{\dagger} - \sum_{\omega^{\dagger}, \omega^{\dagger\dagger}} \lambda_{asit\omega^{\dagger}\omega^{\dagger}\omega^{\dagger\dagger}}^1 g_{asi}, \\ \forall a \in \mathcal{A}, s \in \mathcal{S}, i \in \mathcal{I}, l \in \mathcal{L}, t \in \mathcal{T}, \omega^{\dagger} \in \Omega^{\dagger}, \quad (5.59)$$

$$D_{asit\omega^{\dagger}}^{(5.17)} + D_{ast\omega^{\dagger}}^{(5.18)} \geq \pi_{\omega^{\dagger}} \pi_{\omega^{\dagger}} f_{asilt\omega^{\dagger}}^{\dagger} - \sum_{\omega^{\dagger\dagger}} \lambda_{asit\omega^{\dagger}\omega^{\dagger}\omega^{\dagger\dagger}}^2 g_{asi}, \\ \forall a \in \mathcal{A}, s \in \mathcal{S}, i \in \mathcal{I}, l \in \mathcal{L}, t \in \mathcal{T}, \omega^{\dagger} \in \Omega^{\dagger}(\omega^{\dagger}), \quad (5.60)$$

$$D_{aspt\omega^{\dagger}}^{(5.15)} - D_{a,t,\omega^{\dagger}}^{(5.16)} \geq - \sum_{\omega^{\dagger\dagger}} \lambda_{asit\omega^{\dagger}\omega^{\dagger}\omega^{\dagger\dagger}}^3, \forall a \in \mathcal{A}, s \in \mathcal{S}, p \in \mathcal{P}_{as}, t \in \mathcal{T}, \omega^{\dagger} \in \Omega^{\dagger}(\omega^{\dagger}), \quad (5.61)$$

$$D_{a,t,\omega^{\dagger}}^{(5.16)} \geq \pi_{\omega^{\dagger}} \pi_{\omega^{\dagger}} k_{aspt\omega^{\dagger}}^{\dagger} - \sum_{\omega^{\dagger\dagger}} \lambda_{asit\omega^{\dagger}\omega^{\dagger}\omega^{\dagger\dagger}}^3, \forall a \in \mathcal{A}, s \in \mathcal{S}, t \in \mathcal{T}, \omega^{\dagger} \in \Omega^{\dagger}(\omega^{\dagger}), \quad (5.62)$$

$$D_{asit\omega^{\dagger\dagger}}^{(5.24)} \geq \pi_{\omega^{\dagger}} \pi_{\omega^{\dagger}} \pi_{\omega^{\dagger\dagger}} h_{asi} + \lambda_{asit\omega^{\dagger}\omega^{\dagger}\omega^{\dagger\dagger}}^1 + \lambda_{asit\omega^{\dagger}\omega^{\dagger}\omega^{\dagger\dagger}}^3, \\ \forall a \in \mathcal{A}, i \in \mathcal{I}, s \in \mathcal{S}, t \in \mathcal{T}, \omega^{\dagger\dagger} \in \Omega^{\dagger\dagger}(\omega^{\dagger}, \omega^{\dagger}), \quad (5.63)$$

$$D_{asit\omega^{\dagger\dagger}}^{(5.24)} \geq \pi_{\omega^{\dagger}} \pi_{\omega^{\dagger}} \pi_{\omega^{\dagger\dagger}} h_{asi} + \lambda_{asit\omega^{\dagger}\omega^{\dagger}\omega^{\dagger\dagger}}^2 + \lambda_{asit\omega^{\dagger}\omega^{\dagger}\omega^{\dagger\dagger}}^3, \\ \forall a \in \mathcal{A}, i \in \mathcal{I}, s \in \mathcal{S}, t \in \mathcal{T}, \omega^{\dagger\dagger} \in \Omega^{\dagger\dagger}(\omega^{\dagger}, \omega^{\dagger}). \quad (5.64)$$

To perform the sensitivity analysis using the dual formulation, I must establish rules for its behaviour. From [104], I have the following rules which define the behaviour of the optimal value function as a cost coefficient changes:

- The optimal value function, $f(\gamma)$, is defined as the change in value of the optimal solution as the cost coefficient of one of the variables, for example $k_{aspt\omega^\dagger}^\dagger$, varies by γ . γ is a decision variable defined as the offset value for the cost coefficient of a single primal decision variable.
- The optimal value function is concave and piecewise linear.
- $\mathcal{P}_\gamma^*$ is the set of optimal solutions for the primal problem with the offset value, γ , applied, for example $k_{aspt\omega^\dagger}^\dagger + \gamma$. $\mathcal{P}_\gamma^*$ is constant for (γ_1, γ_2) , if $f(\gamma)$ is linear across the interval $[\gamma_1, \gamma_2]$.
- If for γ_1 and γ_2 , $\mathcal{P}_{\gamma_1}^* = \mathcal{P}_{\gamma_2}^*$ holds, I can define $\bar{\mathcal{P}}^* = \mathcal{P}_\gamma^*$ for all $\gamma \in [\gamma_1, \gamma_2]$. This is a linear interval of $f(\gamma)$.

Given this definition of behaviour within a linear interval, the key is therefore to find the breakpoints dividing the linear intervals, as these are the points at which the optimal values for the primal decision variables change. Two linear programming problems can be solved to give γ_1 and γ_2 , which are the lower and upper bounds of γ , within a linear interval. I give the standard form of the two optimization problems in (5.65) and (5.66), where $e_j = 1$ if j is the index of the c_j under analysis, else $e_j = 0$.

$$\gamma_1 = \min\{\gamma : \mathbf{A}^\top \bar{\mathbf{y}} + \bar{\mathbf{s}} = \bar{\mathbf{c}} + \gamma e_j, \bar{\mathbf{b}}^\top \bar{\mathbf{y}} = \bar{\mathbf{c}}^\top \bar{\mathbf{x}}^* + \gamma x_j^*, \bar{\mathbf{s}} \geq 0\}, \quad (5.65)$$

$$\gamma_2 = \max\{\gamma : \mathbf{A}^\top \bar{\mathbf{y}} + \bar{\mathbf{s}} = \bar{\mathbf{c}} + \gamma e_j, \bar{\mathbf{b}}^\top \bar{\mathbf{y}} = \bar{\mathbf{c}}^\top \bar{\mathbf{x}}^* + \gamma x_j^*, \bar{\mathbf{s}} \geq 0\}. \quad (5.66)$$

As an example, I give the minimization case for analyzing the cost coefficient of an $X^\dagger$ variable. I provide the objective function and the relevant constraint, as the remaining constraints are unchanged.

$$\begin{aligned} \gamma_1 = \min \left\{ \gamma : \sum_{a,s,t,\omega^\dagger} D_{ast\omega^\dagger}^{(5.12)} - \sum_{a,t,\omega^\dagger} n_{a\omega^\dagger} D_{a,t,\omega^\dagger}^{(5.16)} + \sum_{a,s,i,t,\omega^\dagger} D_{asit\omega^\dagger}^{(5.17)} + \sum_{a,s,t,\omega^\dagger} D_{ast\omega^\dagger}^{(5.18)} \right. \\ \left. + \sum_{a,s,\omega^{\dagger\dagger}} m_{as\omega^{\dagger\dagger}}^\dagger D_{asit\omega^{\dagger\dagger}}^{(5.24)} + \sum_{a,s,t,\omega^\dagger} e_{as} D_{ast\omega^\dagger}^{(5.11)} = g(\bar{\mathbf{x}}^*) + \gamma X_j^\dagger \right\}, \quad (5.67) \end{aligned}$$

where $g(\vec{\lambda}^*)$ denotes the scalar cost value of the final, optimal, solution to the partial Lagrange multiplier algorithm, and $X_j^\dagger$ denotes the value of the specific instance of $X^\dagger$ under investigation. The dual constraint corresponding to $X^\dagger$ is therefore modified to:

$$D_{ast\omega^\dagger}^{(5.11)} - D_{aspt\omega^\dagger}^{(5.15)} \geq \pi_{\omega^\dagger} k_{aspt\omega^\dagger}^\dagger + \gamma e_j, \forall a \in \mathcal{A}, s \in \mathcal{S}, p \in \mathcal{P}_{as}, t \in \mathcal{T}, \omega^\dagger \in \Omega^\dagger, \quad (5.68)$$

where $e_j = 1$ for the specific instance of $X^\dagger$ to be analyzed, and 0 for all other cases. For this example for $X^\dagger$, there are two parameters that contribute to the cost coefficient, i.e., the scenario probability, $\pi_{\omega^\dagger}$, and the provisioning cost, $k_{aspt\omega^\dagger}^\dagger$. The value calculated for γ gives the combined tolerance of the parameters, but it is simple to find the individual tolerance by simple division. For example, if the aim is to find the tolerance for the provisioning cost, I divide the combined bound by the scenario probability to find the bound for cost only. The two optimizations outlined can be solved as two LPs to give precise values for upper and lower bounds of γ . If the cost coefficient is changed beyond the upper or lower bound, there is a new linear interval and therefore the values of γ_1 and γ_2 can be recalculated. This process can be repeated to find the full, precise curve for a cost coefficient.

5.6 Performance Evaluation

5.6.1 Parameter Settings

To test the performance of my optimization formulation, I consider a scenario across two time periods with three SECaaS providers (i.e., $s1$, $s2$, and $s3$) and two cyber insurers (i.e. $i1$ and $i2$). Each SECaaS provider offers two prepaid plans (i.e. $p1$ and $p2$), with one lasting for one time period, the other lasting for two. Provider $s1$ offers services in units of 5 packets for $p1$ and 8 packets for $p2$ with each unit costing \$0.10 and \$0.15, respectively. Provider $s2$ offers units of 15 and 20 at \$0.25 and \$0.30. The damages for a packet that is undetected by the security services is priced at \$1.5 per packet. To simulate the possibility of packets not being processed by security, I

include an additional SECaaS provider, *s3*, with no cost and no insurance coverage. The insurers offer two premiums (i.e. q_1 and q_2), with q_1 lasting for one time period, and q_2 lasting for two, and being priced just below twice the price of q_1 . Values for service prices are synthesized based on cloud and security providers, such as Amazon EC2 [3] and Trend Micro [21]. Packet damages and indemnity are synthesized from estimated damages caused by historic cyber attacks and example insurance premiums for small scale businesses, scaled according to my incoming packet data [121] [122]. Packet demand levels and detection probabilities are generated from real honeypot data provided by the University of Waikato's Cyber Security Lab [123]. Honeypots positioned in Singapore, Sao Polo, Brazil, and San Jose, USA, collected packet data over a number of days. Using this data, I employed the Snort IDS (Intrusion Detection System) [124] with two different rulesets. The more complete ruleset provided the benchmark for the number of malicious packets, with the other ruleset giving the proportion of missed packets and the corresponding missed-detection probabilities for each scenario. This analysis provides us with realistic values for cyber attacks and provides more informative numerical results.

5.6.2 Numerical Results

The deterministic equivalent of the problem formulation was encoded in a GAMS script and solved using the CPLEX solver. The tests were designed to examine the performance of the stochastic optimization formulation, and explore the relationship between security provisioning and insurance coverage.

5.6.2.1 Influence of prepaid security on cost

In Fig. 5.3 I set the number of prepaid SECaaS services provisioned from each security provider and measure the impact on the expected cost and provisioning quantities across all scenarios. The upper plot measures the total used services. As the provisioned prepaid services increase, the number of on-demand services required decreases. This is the expected result, but compares well with the lower plot, which measures the change in prepaid services against total cost. The total cost curve is convex and shows

that there is an ideal prepaid point where the number of prepaid services mitigates the damages of both overprovisioning and underprovisioning. Thus the optimal solution requires some on-demand services when demand is high, as the increased expense does not outweigh the wasted expense from overprovisioning at low demand that more prepaid services would incur. This shows the usefulness of stochastic optimization, as my formulation finds this optimal point.

Figure 5.3: Impact of prepaid SecaaS services on service utilization and total cost.

5.6.2.2 Impact of packet damages and indemnity on security provisioning

One of the key challenges in cyber insurance is the accurate estimation of damages caused by cyber attacks. In Fig. 5.4 I investigate the effect of changing the damage values assigned to individual packets on security provisioning. The model allows for some incoming packets to not be covered by the security services. I then measure the decrease in the number of ignored packets as the damage per malicious unhandled packet. The result is clear - the damages caused by inadequate security necessitate the

increasing of security provisioning, and cyber insurance is not a sufficient mitigation. This also shows the importance of insurers correctly assessing the expected damages caused by cyber attacks so as to optimally set their insurance policies.

Figure 5.4: Impact of packet damages on security provisioning measured in unprocessed packets. As damages increase in cost, it becomes more costly to leave packets unprocessed, and so security provisioning is increased to ensure more packets are processed.

I also examine the influence that the indemnity values (set by the insurers) have on security provisioning. As before, I measure the quantity of unprocessed packets as the indemnity changes. This is shown in Fig. 5.5. In this graph I consider three different packet damage values, with each showing that the number of unprocessed packets decreases as the indemnity rises. This is because packets unprocessed by the security services are not covered by the insurance policy. Thus, higher indemnity values create an incentive for users to employ comprehensive security coverage. The height of the curves shows that this is complementary to the per-packet damage costs, as the greater the damage the lower the curve, and the faster it descends. Therefore I conclude that cyber insurance can be used to incentivize security provisioning.

Figure 5.5: Impact of packet indemnity on security provisioning measured in unprocessed packets. Higher indemnity provides an incentive to increase security provisioning so that packets are not left unprocessed.

5.6.2.3 Comparison of solution methods with varying attack probability

To justify my use of the stochastic optimization approach, I compare the performance of the stochastic optimization to various alternative methods under varying attack probabilities. This will highlight the importance of accurate security provisioning and insurance provisioning. In Fig. 5.6 I provide a comparison between the methods of stochastic optimization, expected value function, and no-insurance, and also include a fourth result, with no security (and therefore no insurance), to demonstrate the high cost of ignoring cyber security. The expected value function method purchases prepaid security services sufficient to process the mean number of packet arrivals as specified by the probability distribution. The four methods are compared across three different probability distributions - normal, uniform, and the one derived from my real data. The expected value function gives a good result but is inferior to stochastic optimization as it does not consider future impact, with the order of the bar heights following the expected order.

Figure 5.6: Comparison of alternative solution methods with varying attack probability distributions.

5.6.2.4 Influence of budget on provisioning

In the real world, a user will have a restricted budget to spend on security services and insurance. To examine the priorities of the user, I introduced a budget limit which acts as a maximum value for the objective function solution. In Fig. 5.7 I provide a comparison between the amount of security provisioning and the number of insurance claims. As the budget rises, the security provisioning increases until it reaches a maximum to correspond with the maximum incoming traffic, showing that security is the top priority. The increase in packet claims, however, shows that not only is the insurance provision increased, but using the prediction from stochastic optimization, the user can increase their security knowing that the indemnity benefits from covered packets will offset the security costs. Thus I see a symbiotic relationship between security and insurance provisioning where the insurance coverage provides an incentive for the security provisioning, which in turn limits the costs from packet damages.

Figure 5.7: Exploration of the relationship between security and insurance provisioning under budget constraints.

5.6.2.5 Influence of security capacity on total cost

Each security plan from a provider has a maximum capacity of prepaid SECaaS packets that can be provisioned. In Fig. 5.8 I vary the prepaid capacity of the security providers, and measure the impact on the total cost. As expected, with more limited prepaid options, costs are higher, as the number of packets that must be processed by on-demand services increases. This problem is more severe initially, as the number of on-demand services must be increased. The impact lessens non-linearly, as the prepaid plans reach a capacity sufficient to handle most demand without on-demand services, and further increases have limited impact. The number of on-demand services required decreases and so security costs also decrease. To provide a comparison I add results using the mean number of incoming packets and a curve with no insurance provision. The stochastic optimization performs best, as expected, but the mean value result offers worse performance, because it is inflexible and results in underprovisioning, making it worse than the no-insurance option, due to the cost penalty from excessive on-demand security.

Figure 5.8: Exploration of the effect security capacity has on total cost.

5.6.2.6 Benefits of joint optimization

The premise of my formulation is that optimizing the security provisioning and cyber insurance jointly is preferable to handling the two separately, due to the influence that the two services have on each other. I have demonstrated this connection in my previous results. Nonetheless, to justify my premise I implemented an alternative method that first, independently, provisions security, before making cyber insurance decisions. Again, I vary the security capacity, to show how the solutions perform. The result, given in Fig. 5.9 shows that the joint solution gives clear financial benefits over the separate approach. Thus, the symbiotic nature of security and insurance is demonstrated in cost benefits.

5.6.2.7 Performance of partial Lagrange multiplier method

The key purpose of introducing the partial Lagrange multiplier method is to achieve improved scalability over the conventional branch-and-bound (B&B) method for solving mixed integer programming problems. In Fig. 5.10 I introduce a reduced size version of my test case and vary the number of time periods, allowing us to increase the problem size linearly. I compare the execution time for a standard B&B algorithm

Figure 5.9: Demonstration of the benefits of joint optimization.

with the execution time for the partial Lagrange multiplier method and find that the increase in execution time matches the expectation from my analytical results. The B&B curve increases at a rapid rate, showing that for a large problem, the memory and processing time requirements would rapidly become impractical. By contrast, using a stepsize of $\alpha = 1$, I find that the partial Lagrange multiplier method converges quickly and only increases at a very slow rate, and linearly, as the problem size grows. This result supports my claims that the partial Lagrange multiplier method offers much improved scalability over traditional mixed integer program solution methods.

5.6.2.8 Convergence Behaviour

In Fig. 5.11 I demonstrate the manner in which the partial Lagrange multiplier method converges to the solution, testing the performance on the same problem as the previous result. I observe that the convergence is extremely rapid, as the initial value is high, but the solution drops quickly to near the optimum, and finds the optimal value on the third iteration. I find that this performance remains the same independent of the initial value of the Lagrange multipliers, or the value of the stepsize - the optimal value is still found within three steps. After the optimal value is found, the solution

Figure 5.10: Comparison of execution time and memory usage of partial Lagrange multiplier and branch-and-bound (B&B) methods as problem size increases.

alternates between this value and a slightly higher one, until the algorithm terminates. This is within the expected behaviour of the algorithm, which specifies termination if the algorithm does not improve within a predetermined number of iterations. In this case, I terminate after ten iterations, with the best value being the solution to the original integer problem. The speed of convergence is therefore dependent on the structure of the problem, and the tightness of the constraints (i.e. those that satisfy total unimodularity). If the constraints are tight, within the parameters set, then the solution can be found faster than the polynomial worst case, as I find here.

5.6.2.9 Sensitivity Analysis of Prepaid SECaaS Provisioning

I use parameter settings that allow the customer to choose security plans of varying lengths, with a choice of service provider. To understand how pricing influences the user's provisioning decisions, I perform an analytical sensitivity analysis on the variable instance $X_{1,1,1,1,1}^\dagger$. The sensitivity analysis formulation, using the final step of the reduced-size partial Lagrange multiplier algorithm result, gives precise values for the change in cost at which the customer changes their prepaid security allocation.

Figure 5.11: Example case of speed of convergence of the partial Lagrange algorithm to the optimal value.

The point of change is called a ‘breakpoint’, and I show this in Fig. 5.12. As the price increases, the value of $X_{1,1,1,1,1}^\dagger$ decreases, as expected, with prepaid security provisioning redistributed to the alternative plan. At each point on the graph, a threshold is reached, causing the customer to change their purchasing choices. The breakpoints are primarily determined by the cost of the alternative. However, since $X^\dagger$ is a prepaid service variable, it must also consider the increased cost against the on-demand cost. Thus, an increased cost of this variable may not immediately result in a drop to 0, if the on-demand cost is still high, rather the customer prefers to change plan. The primary benefit of this approach over a trial-and-error approach, where the cost values must be manually changed and the solution recalculated, is that I can guarantee that I find the exact point at which the cost changes, and minimize the number of tests required to find those points. To demonstrate this, a second line is plotted, showing a set of reasonable values for cost variation, and the corresponding variable allocation. Whilst the various levels of provisioning are found, the exact breakpoints are not, and further measurements would be required to find them. Service providers may employ this sensitivity analysis method to set

competitive prices, as they can see exactly how to price their services such that the customer chooses them over the competition, without offering a lower price than necessary. To examine the choice between providers, I take the result of the partial Lagrange multiplier method for the full-size problem in Fig. 5.13, and vary the cost of plan p_2 from provider s_1 across two time stages. I plot the breakpoints and measure the allocation for plans p_1 and p_2 from providers s_1 and s_2 . As the cost increases, the customer prefers to switch provider, up to the capacity limit of that provider, rather than choose a short-term plan. As the price increases further, the customer switches to the short-term plan to cover the remaining demand. Therefore, it is better to choose a long-term plan from any provider, as they are more cost-efficient than short-term plans, with stochastic optimization allowing us to minimize the long-term costs.

Figure 5.12: Cost breakpoints of the variable $X_{1,1,1,1,1}^\dagger$, comparing analytical and experimental approaches to sensitivity analysis.

5.6.2.10 Results conclusions

From the above numerical results, I can see clearly that the stochastic optimization provides benefits as it allows for variation in future demand and attack probabilities, as well as the corresponding damages and indemnity values. It also offers clues as to

Figure 5.13: Cost breakpoints of the variable $X_{1,1,2,1,1}^{\dagger}$, for the full-size problem showing the choices made between plans and providers.

how those values should be optimally set, as I see the relationship between security provisioning and cyber insurance, as good insurance encourages good security, and vice versa. Further, I demonstrate the significantly better scalability of the partial Lagrange multiplier method over traditional branch-and-bound for MIP solving. Finally, I implement my analytical sensitivity analysis, and show that it is more efficient and precise than an experimental approach.

5.7 Conclusion

In this chapter I have presented a combined approach to security and cyber insurance provisioning in the cloud. Using a stochastic optimization, I have presented a method of optimally provisioning both services in the face of uncertainty regarding future pricing, incoming traffic and cyber attacks. Since my optimization involves solving an integer programming problem, I present the partial Lagrange multiplier method, which exploits the total unimodularity property to guarantee integer solutions, while relaxing the problem to a linear programming problem. This problem is solved iteratively using a subgradient method, which I prove converges to the optimal solution in

at worst polynomial time. Using the solution produced by the algorithm, I apply an analytical sensitivity analysis approach that gives precise sensitivity values for individual parameters. Finally I provide an experimental evaluation of my contributions using realistic traffic and attack data derived by running real traffic data through an Intrusion Detection System. The main challenge of cyber insurance is the number of assumptions that must be made, for example, the ability to detect cyber attacks, establish accurate damages, and successfully make insurance claims.

The objective of this chapter was to show that stochastic optimization can be used to improve cloud security. I have done this particularly by demonstrating the interdependence between insurance and security provisioning. This addresses the third of the key issues that this thesis intended to advance through the use of stochastic optimization, and can be applied to improve the user protection in the scenarios presented in the previous chapters on guaranteeing performance and improving battery life. Therefore the investigation of its applications to mobile cloud computing have shown clearly that it is a useful tool that should be applied to extensions of these problems and other problem areas. A full summary of the contributions of this thesis, and some suggestions for future extensions are provided in the following chapter.

Chapter 6

Conclusions and Further Work

In this chapter I recapitulate the contents of the preceding chapters and highlight the major contributions of this thesis. I also present a number of potential future research directions for extending the work shown here.

6.1 Conclusions

As stated in Chapter 1, the goal of this thesis is to investigate the usefulness of stochastic optimization to mobile cloud computing by applying it to three related problem areas that address three of the most significant issues in mobile cloud computing. The first area optimizes resource allocation in the cloud, guaranteeing performance; the second optimizes mobile device migration, preserving battery life; and the third optimizes cloud security and cyber insurance, improving security. Thus, I have shown that stochastic optimization can be applied to improve the state of the art in these important areas of performance, battery life, and security. The contents of this thesis can be summarized as follows:

- **Chapter 1:** I provide introductions to the problem areas addressed in the thesis, namely virtual machine and bandwidth provisioning, mobile application migration, and cyber insurance. I also introduce stochastic optimization and the partial Lagrange multiplier method.
- **Chapter 2:** I outline the current state of research in the fields of virtual machine and bandwidth provisioning, mobile application migration, and security-as-a-service and cyber insurance. I discuss the contributions of each work and identify

problem areas that the prior work does not satisfactorily address to provide context and justification for the contributions of this thesis.

- **Chapter 3:** I apply stochastic optimization to the problem of jointly provisioning virtual machines and bandwidth. Stochastic optimization allows both resources to be reserved in advance despite uncertain future demand and prices. Numerical results using demand probabilities generated from real workload trace data are presented to justify the joint approach and show that stochastic optimization is superior to alternative, more simplistic methods. To improve the scalability of the mixed integer problem I employ a scenario tree reduction algorithm that reduces the problem size to give a heuristic solution that is a good approximation of the optimum. I also apply an analytical sensitivity analysis method to find the tolerance of solutions to parameter changes. The method provides exact values to determine how far a cost parameter can change before the corresponding decision variable value changes. Further numerical results are given to show the performance improvements offered by the scenario tree reduction and illustrate how the sensitivity analysis may be used for providers to optimize price setting to maximize profit without losing business.
- **Chapter 4:** I devise a new system model that I call ‘Bring-Your-Own-Application’. Inspired by the Bring-Your-Own-Device paradigm, I take advantage of existing mobile application migration techniques to propose a model whereby customers may offload mobile applications to more powerful equipment at a location such as a library or coffee shop. To optimize the matching of users to equipment across multiple locations I again employ stochastic optimization to solve a problem with binary variables. With the use of binary variables, I find a number of parameters are also binary, which allows the use of the partial Lagrange multiplier method. The method solves the binary problem as a sequence of linear programs. I prove that the algorithm will converge to the optimal solution, and that it will do so in polynomial time, which is an improvement over solution methods for NP-hard binary problems. I also re-apply the analytical sensitivity

analysis method from the previous chapter to find precise values for optimizing prices. Numerical results demonstrate the effectiveness of the stochastic optimization, demonstrate the runtime of the partial Lagrange multiplier method, and show the value of the sensitivity analysis. Parameters for the numerical results are derived from real data on mobile user movements around a university campus and applied to the campus of Nanyang Technological University.

- **Chapter 5:** With the increasing popularity of cyber insurance, I find that, similarly to Chapter 3, stochastic optimization can be effectively applied to the joint provisioning of security-as-a-service and cyber insurance. The result is a mixed integer problem that can optimally provision both resources, and I investigate whether there is an interdependence between the two. Using cyber attack data derived from Internet honeypots I find that the availability of cyber insurance incentivizes good security provisioning if the existence of security investment is a prerequisite of coverage. Since the problem is an integer problem with some binary variables, I again employ the partial Lagrange multiplier method. I prove its scalability and optimality, and demonstrate that its experimental performance matches the theoretical analysis. I also utilize the sensitivity analysis from the previous two chapters, as this system model supports variable length contracts and reservation, so the sensitivity analysis provides important insight into the setting of contract prices such that customers will be encouraged to commit to a provider and security coverage level for a longer period of time.

The three areas of contribution may be considered as three parts of a larger mobile cloud computing scenario, which is illustrated in Fig. 6.1. In this context, a user may provision resources from the cloud, with a broker optimizing the placement of VMs and bandwidth, as discussed in Chapter 3. This user may be static, located at home or work, or mobile, but with no battery or data concerns. However, if the user chooses to continue their activity while mobile, they may find that their battery life is too limited. In this scenario, the user would find it beneficial to offload their computation entirely to a piece of local computing equipment at a nearby location,

allowing them to continue their work without sacrificing their device battery. This can be optimized as discussed in Chapter 4 with multiple users being allocated by a broker to locations in a manner that preserves the battery of as many people as possible. In both these offloading scenarios, users must be able to trust the host that handles their computation, and this may be guaranteed in the form of security-as-a-service resources being provisioned. Thus, in Chapter 5 security and cyber insurance are purchased on behalf of the user to improve protection of their computation, and protect them against damages caused by malicious activity. In each aspect of this scenario, stochastic optimization is shown to be a powerful tool in improving the cost and performance of the system, and therefore, I can confidently state that stochastic optimization has proven its usefulness in mobile cloud computing. Future research directions should therefore focus on extending and fully integrating the three scenarios addressed in this thesis. This thesis should also serve as evidence that stochastic optimization should be applied to other problem areas in mobile cloud computing.

6.2 Future Work

6.2.1 Integrated System

In this thesis I have presented the three problem areas as related. Whilst my intent has been to show that stochastic optimization may be useful in solving a range of problems in mobile cloud computing, it is also my intent to show that these problem areas can be connected into a single larger system of mobile cloud computing, with each component optimized stochastically. As illustrated in Fig. 6.1, each issue addressed focuses on an aspect of a larger scenario. The next step in this research, therefore, should be to combine the problem formulations into a larger system model, where a user has a two-tiered offloading choice between local equipment (if available) and the cloud (if battery life permits). This offloading should be optimized end-to-end considering the computational and bandwidth requirements of multiple mobile users. In addition to these requirements, both users and providers may wish to provision security services and cyber insurance, with the numerical results being extended to

Figure 6.1: Larger mobile cloud computing context, where the user chooses between cloud offloading, with optimized VM and bandwidth allocation, and offloading to local equipment while mobile. Security and insurance services are provisioned to cover their offloading in each case.

consider a long period of time, to fully demonstrate the benefits of reservation. This integrated model would make the work on scalability done in this thesis of greater importance, as the number of decision variables would be very high, although both scenario tree reduction and the partial Lagrange multiplier method would be useful in this regard.

6.2.2 Joint Virtual Machine and Bandwidth Allocation

Future work should improve the realism of the problem formulation, by considering factors such as random network delay and VM migration. Network delay is unpredictable and there is a lack of good available data to model Internet delay and congestion, yet it is not realistic to assume that all network links will maintain the same capacity constantly. VM migration (within the cloud) is also an important part

of VM allocation, as the work outlined in Chapter 2 typically considers either advance placement or VM consolidation. Future extensions could focus on an approach that combines both the initial placement and the migration of VMs. The approach to user migration used in Chapter 4 may be a good way to model this. The model could also be extended to consider inter-VM communication and service composition. Often a VM does not exist in isolation but must communicate with other VMs, or serves a specific purpose as part of a larger web service, and so clusters of VMs must be provisioned together. Thus the work here could be combined with existing work that allocates VMs in clusters, and attaches a desirability weighting to a provider that already hosts a VM that communicates a lot with the VM being placed. Further attempts to improve computational performance should also be considered, for example by applying a distributed approach. The sensitivity analysis results show promise, but can also be extended to further examine the interaction between system elements, such as the significance of constraints, and the priority of variables.

6.2.3 Bring-Your-Own-Application

Further work should consider a two-tier migration to the cloud as well as local equipment as already discussed. The physical migration modeling could be extended to consider migration delay, and non-equidistant locations. Thus migration to other locations would be influenced by the time taken to travel, which may vary between locations, and the user's battery life. This is a more realistic approach, as it is unlikely that real locations would be equidistant, but requires a more complex model. A user may also have changing application requirements during a session. It is common while working for the nature of the work to change, and the applications being used may also change. Thus a user may perform low-demand word processing for a while, but then need to carry out more demanding image processing, which may necessitate the use of more powerful equipment. The stochastic optimization may be extended to allow for the probability of a user increasing or decreasing their equipment needs during a session.

6.2.4 Joint Security-as-a-Service and Cyber Insurance Provisioning

The main challenge of cyber insurance is the number of assumptions that must be made, for example, the ability to detect cyber attacks, establish accurate damages, and successfully make insurance claims. As mentioned in Chapter 2, the focus of work in this area should be on establishing international standards and collecting data to provide universal guidelines so that there is consistency in setting costs. Future extensions that are applicable to this work could consider the interaction of applications, where the security performance of one part of the system can impact the security of other parts. In our system model, we assume that there is no interaction between components and the security of each application is independent of the others. Real systems typically have interdependence, and an extension to the model should allow for conditional probabilities that increase the vulnerability of applications that are closely related or situated in the same cloud provider network. I introduced real honeypot data, but future extensions could consider more extensive data to produce more accurate options. The data used in this thesis is collected over a short period of a few days, but a honeypot run for a longer period of time, such as a few months or years, could generate more reliable data about the behaviour of attackers over time. Accuracy of data could further be extended through the implementation of systems to update parameters on a daily or weekly basis, to improve future decisions. Thus rather than assuming that attack and attack success probabilities remain constant for the duration of the system, the values may be updated after a period of time and the reservation decisions adapted accordingly.

6.3 Summary

To briefly summarize, in this thesis I have applied the technique of stochastic optimization to three related problem areas in mobile cloud computing, to determine whether stochastic optimization can be useful applied to the key issues in mobile cloud computing. I considered the optimization of offloading at two different tiers of the

cloud computing model, and the allocation of security and insurance to provide protection to the user (and provider) when offloading. In each case I applied a scalability improvement method to the stochastic optimization formulation, resulting in significantly better runtime performance. I also employed analytical sensitivity analysis to consider precisely the behaviour of variables under parameter changes, with potential benefits for providers in price-setting and service specification. I have proposed a number of directions for further research, demonstrating the richness of this research area, and the importance of stochastic optimization as a tool in the advancement of the state of the art.

References

- [1] C. P. Gomes, “Structure, duality, and randomization: Common themes in ai and or.” in *AAAI/IAAI*, H. A. Kautz and B. W. Porter, Eds. AAAI Press / The MIT Press, 2000, pp. 1152–1158. [Online]. Available: <http://dblp.uni-trier.de/db/conf/aaai/aaai2000.htmlGomes00>
- [2] W. Zhang, “Branch-and-Bound Search Algorithms and Their Computational Complexity,” Information Sciences Institute/University of Southern California, Tech. Rep. ISI/RR-96-443, May 1996.
- [3] (2016) Amazon EC2 pricing. [Online]. Available: <http://aws.amazon.com/ec2/pricing/>
- [4] A. Shapiro, D. Dentcheva, and A. Ruszcynski, *Lectures on Stochastic Programming: Modeling and Theory*. Society for Industrial and Applied Mathematics Philadelphia, 2009. [Online]. Available: http://www2.isye.gatech.edu/people/faculty/Alex_Shapiro/SPbook.pdf
- [5] (2014) AWS Case Study: Netflix. [Online]. Available: <http://aws.amazon.com/solutions/case-studies/netflix/>
- [6] “Software-Defined Networking: The New Norm for Networks,” White Paper, April 2012. [Online]. Available: <https://www.opennetworking.org/images/stories/downloads/sdn-resources/white-papers/wp-sdn-newnorm.pdf>
- [7] (2014) Enter the Andromeda Zone - Google Cloud Platforms Latest Networking Stack. [Online]. Available: <http://googlecloudplatform.blogspot.sg/2014/04/enter-andromeda-zone-google-cloud-platforms-latest-networking-stack.html>
- [8] (2014) Mirage OS A Cloud Operating System. [Online]. Available: <http://www.xenproject.org/developers/teams/mirage-os.html>

- [9] (2015) OpenFlow. [Online]. Available: <https://www.opennetworking.org/ja/sdn-resources-ja/onf-specifications/openflow>
- [10] S. Chaisiri, B.-S. Lee, and D. Niyato, "Optimization of Resource Provisioning Cost in Cloud Computing," *Services Computing, IEEE Transactions on*, vol. 5, no. 2, pp. 164–177, April 2012.
- [11] M. Alicherry and T. Lakshman, "Network Aware Resource Allocation in Distributed Clouds," in *INFOCOM, 2012 Proceedings IEEE*, March 2012, pp. 963–971.
- [12] X. Meng, V. Pappas, and L. Zhang, "Improving the Scalability of Data Center Networks with Traffic-aware Virtual Machine Placement," in *INFOCOM, 2010 Proceedings IEEE*, March 2010, pp. 1–9.
- [13] A. Scarfo, "New security perspectives around byod," in *Broadband, Wireless Computing, Communication and Applications (BWCCA), 2012 Seventh International Conference on*, Nov 2012, pp. 446–451.
- [14] (2015) App and desktop virtualization. [Online]. Available: <http://www.citrix.com/solutions/desktop-virtualization/overview.html>
- [15] (2015) Continuum for phone. [Online]. Available: [https://msdn.microsoft.com/en-us/library/windows/hardware/mt608594\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/windows/hardware/mt608594(v=vs.85).aspx)
- [16] McAfee, "Net Losses: Estimating the Global Cost of Cybercrime," Center for Strategic and International Studies, Economic Impact of Cybercrime II, Jun. 2014.
- [17] (2016) Identity theft resource center data breach reports. [Online]. Available: <http://www.idtheftcenter.org/2016databreaches.html>
- [18] (2016) A bold approach to cyber risk management. [Online]. Available: <http://www.mas.gov.sg/News-and-Publications/Speeches-and-Monetary-Policy-Statements/Speeches/2016/A-Bold-Approach-to-Cyber-Risk-Management.aspx>

- [19] (2016) Insurance 2020 & beyond: Reaping the dividends of cyber resilience. [Online]. Available: <http://www.pwc.com/gx/en/industries/financial-services/insurance/publications/insurance-2020-cyber.html>
- [20] (2016) McAfee security-as-a-service solutions. [Online]. Available: <https://www.mcafeeasap.com/MarketingContent/Products/ProductsLanding.aspx>
- [21] (2016) Deep security as a service. [Online]. Available: <http://www.trendmicro.com/us/business/saas/deep-security-as-a-service/usage-based-pricing>
- [22] B. Delamore and R. K. L. Ko, “Chapter 9 - security as a service (secaas)an overview,” in *The Cloud Security Ecosystem*, R. K. L. Ko and K.-K. R. Choo, Eds. Boston: Syngress, 2015, pp. 187 – 203. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/B9780128015957000094>
- [23] (2016) Allianz cyber protect. [Online]. Available: <http://www.agcs.allianz.com/services/financial-lines/cyber-insurance/>
- [24] (2016) Cyber and data security. [Online]. Available: <http://www.qbeurope.com/professional-financial/cyber-liability.asp>
- [25] M. Clark. (2014) Timeline of target’s data breach and aftermath: How cybertheft snowballed for the giant retailer. [Online]. Available: <http://www.ibtimes.com/timeline-targets-data-breach-aftermath-how-cybertheft-snowballed-giant-retailer-1580056>
- [26] C. A. Newman. (2016) Targets cyber insurance: A \$100 million policy vs. \$300 million (so far) in costs. [Online]. Available: <http://datasecuritylaw.com/blog/targets-cyber-insurance-a-100-million-policy-vs-300-million-so-far-in-costs/>
- [27] S. Chaisiri, R. K. L. Ko, and D. Niyato, “A joint optimization approach to security-as-a-service allocation and cyber insurance management,” in *Trust-com/BigDataSE/ISPA, 2015 IEEE*, Aug 2015, pp. 426–433.
- [28] E. Santos, Jr. and E. S. Santos, “Polynomial solvability of cost-based abduction,” *Artif. Intell.*, vol. 86, no. 1, pp. 157–170, Sep. 1996. [Online]. Available: [http://dx.doi.org/10.1016/0004-3702\(96\)00016-1](http://dx.doi.org/10.1016/0004-3702(96)00016-1)

- [29] A. J. Hoffman and J. B. Kruskal, *Integral Boundary Points of Convex Polyhedra*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 49–76. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-68279-0_3
- [30] C. P. Gomes, “Structure, duality, and randomization: Common themes in AI and OR.” in *AAAI/IAAI*, H. A. Kautz and B. W. Porter, Eds. AAAI Press / The MIT Press, 2000, pp. 1152–1158. [Online]. Available: <http://dblp.uni-trier.de/db/conf/aaai/aaai2000.htmlGomes00>
- [31] Z. Cao, H. Guo, J. Zhang, D. Niyato, and U. Fastenrath, “Improving the efficiency of stochastic vehicle routing: A partial lagrange multiplier method,” *Vehicular Technology, IEEE Transactions on*, vol. PP, no. 99, pp. 1–1, 2015.
- [32] H. N. Van, F. Tran, and J.-M. Menaud, “Autonomic Virtual Resource Management for Service Hosting Platforms,” in *Software Engineering Challenges of Cloud Computing, 2009. CLOUD '09. ICSE Workshop on*, May 2009, pp. 1–8.
- [33] —, “SLA-Aware Virtual Resource Management for Cloud Infrastructures,” in *Computer and Information Technology, 2009. CIT '09. Ninth IEEE International Conference on*, vol. 1, Oct 2009, pp. 357–362.
- [34] N. Bobroff, A. Kochut, and K. Beaty, “Dynamic Placement of Virtual Machines for Managing SLA Violations,” in *Integrated Network Management, 2007. IM '07. 10th IFIP/IEEE International Symposium on*, May 2007, pp. 119–128.
- [35] B. Li, J. Li, J. Huai, T. Wo, Q. Li, and L. Zhong, “EnaCloud: An Energy-Saving Application Live Placement Approach for Cloud Computing Environments,” in *Cloud Computing, 2009. CLOUD '09. IEEE International Conference on*, Sept 2009, pp. 17–24.
- [36] X. Xu, K. Yao, S. Wang, and X. Zhou, “A VM Migration and Service Network Bandwidth Analysis Model in IaaS,” in *Consumer Electronics, Communications and Networks (CECNet), 2012 2nd International Conference on*, April 2012, pp. 123–125.
- [37] D. Kusic, J. Kephart, J. Hanson, N. Kandasamy, and G. Jiang, “Power and Performance Management of Virtualized Computing Environments Via Lookahead Control,” in *Autonomic Computing, 2008. ICAC '08. International Conference on*, June 2008, pp. 3–12.

- [38] K. Zamanifar, N. Nasri, and M. Nadimi-Shahraki, “Data-Aware Virtual Machine Placement and Rate Allocation in Cloud Environment,” in *Advanced Computing Communication Technologies (ACCT), 2012 Second International Conference on*, Jan 2012, pp. 357–360.
- [39] R. Cohen, L. Lewin-Eytan, J. Naor, and D. Raz, “Almost Optimal Virtual Machine Placement for Traffic Intense Data Centers,” in *INFOCOM, 2013 Proceedings IEEE*, April 2013, pp. 355–359.
- [40] V. Justafort and S. Pierre, “Performance-Aware Virtual Machine Allocation Approach in an Intercloud Environment,” in *Electrical Computer Engineering (CCECE), 2012 25th IEEE Canadian Conference on*, April 2012, pp. 1–4.
- [41] F. Tao, C. Li, T. Liao, and Y. Laili, “Bgm-bla: a new algorithm for dynamic migration of virtual machines in cloud computing,” *IEEE Transactions on Services Computing*, vol. PP, Mar. 2015.
- [42] C. Guo, G. Lu, H. J. Wang, S. Yang, C. Kong, P. Sun, W. Wu, and Y. Zhang, “SecondNet: A Data Center Network Virtualization Architecture with Bandwidth Guarantees,” in *Proceedings of the 6th International Conference*, ser. Co-NEXT ’10. New York, NY, USA: ACM, 2010, pp. 15:1–15:12. [Online]. Available: <http://doi.acm.org/10.1145/1921168.1921188>
- [43] J. Zhu, D. Li, J. Wu, H. Liu, Y. Zhang, and J. Zhang, “Towards Bandwidth Guarantee in Multi-Tenancy Cloud Computing Networks,” in *Network Protocols (ICNP), 2012 20th IEEE International Conference on*, Oct 2012, pp. 1–10.
- [44] L. Yu and H. Shen, “Bandwidth Guarantee Under Demand Uncertainty in Multi-Tenant Clouds,” in *Distributed Computing Systems (ICDCS), 2014 IEEE 34th International Conference on*, June 2014, pp. 258–267.
- [45] A. Dalvandi, M. Gurusamy, and K. C. Chua, “Time-Aware VM-Placement and Routing with Bandwidth Guarantees in Green Cloud Data Centers,” in *Cloud Computing Technology and Science (CloudCom), 2013 IEEE 5th International Conference on*, vol. 1, Dec 2013, pp. 212–217.
- [46] H. Xu and B. Li, “Joint Request Mapping and Response Routing for Geodistributed Cloud Services,” in *INFOCOM, 2013 Proceedings IEEE*, 2013, pp. 854–862.

- [47] J. Naous, D. Erickson, G. A. Covington, G. Appenzeller, and N. McKeown, “Implementing an openflow switch on the netfpga platform,” in *Proceedings of the 4th ACM/IEEE Symposium on Architectures for Networking and Communications Systems*, ser. ANCS '08. New York, NY, USA: ACM, 2008, pp. 1–9. [Online]. Available: <http://doi.acm.org/10.1145/1477942.1477944>
- [48] G. Lu, R. Miao, Y. Xiong, and C. Guo, “Using cpu as a traffic co-processing unit in commodity switches,” in *Proceedings of the First Workshop on Hot Topics in Software Defined Networks*, ser. HotSDN '12. New York, NY, USA: ACM, 2012, pp. 31–36. [Online]. Available: <http://doi.acm.org/10.1145/2342441.2342448>
- [49] S. Schmid and J. Suomela, “Exploiting locality in distributed sdn control,” in *Proceedings of the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking*, ser. HotSDN '13. New York, NY, USA: ACM, 2013, pp. 121–126. [Online]. Available: <http://doi.acm.org/10.1145/2491185.2491198>
- [50] M. Koerner and O. Kao, “Multiple service load-balancing with openflow,” in *2012 IEEE 13th International Conference on High Performance Switching and Routing*, June 2012, pp. 210–214.
- [51] A. D. Ferguson, A. Guha, C. Liang, R. Fonseca, and S. Krishnamurthi, “Participatory networking: An api for application control of sdns,” in *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM*, ser. SIGCOMM '13. New York, NY, USA: ACM, 2013, pp. 327–338. [Online]. Available: <http://doi.acm.org/10.1145/2486001.2486003>
- [52] K.-K. Yap, M. Kobayashi, R. Sherwood, T.-Y. Huang, M. Chan, N. Handigol, and N. McKeown, “Openroads: Empowering research in mobile networks,” *SIGCOMM Comput. Commun. Rev.*, vol. 40, no. 1, pp. 125–126, Jan. 2010. [Online]. Available: <http://doi.acm.org/10.1145/1672308.1672331>
- [53] G. Huang, C. N. Chuah, S. Raza, and S. Seetharaman, “Dynamic measurement-aware routing in practice,” *IEEE Network*, vol. 25, no. 3, pp. 29–34, May 2011.
- [54] R. Sherwood, G. Gibb, K. kiong Yap, M. Casado, N. Mckeown, and G. Parulkar, “Flowvisor: A network virtualization layer,” Tech. Rep., 2009.
- [55] V. Mann, A. Vishnoi, K. Kannan, and S. Kalyanaraman, “Crossroads: Seamless vm mobility across data centers through software defined networking,” in *2012 IEEE Network Operations and Management Symposium*, April 2012, pp. 88–96.

- [56] J. Cao, W. Zhang, and W. Tan, “Dynamic control of data streaming and processing in a virtualized environment,” *Automation Science and Engineering, IEEE Transactions on*, vol. 9, no. 2, pp. 365–376, April 2012.
- [57] R.-H. Hwang, C.-N. Lee, Y.-R. Chen, and D.-J. Zhang-Jian, “Cost Optimization of Elasticity Cloud Resource Subscription Policy,” *Services Computing, IEEE Transactions on*, vol. 7, no. 4, pp. 561–574, Oct 2014.
- [58] D. Breitgand and A. Epstein, “Improving Consolidation of Virtual Machines with Risk-Aware Bandwidth Oversubscription in Compute Clouds,” in *INFOCOM, 2012 Proceedings IEEE*, March 2012, pp. 2861–2865.
- [59] D. Divakaran and M. Gurusamy, “Towards Flexible Guarantees in Clouds: Adaptive Bandwidth Allocation and Pricing,” *Parallel and Distributed Systems, IEEE Transactions on*, vol. PP, no. 99, pp. 1–1, 2014.
- [60] D. Niu, C. Feng, and B. Li, “A Theory of Cloud Bandwidth Pricing for Video-on-Demand Providers,” in *INFOCOM, 2012 Proceedings IEEE*, March 2012, pp. 711–719.
- [61] A. Ranabahu, E. Maximilien, A. Sheth, and K. Thirunarayan, “Application portability in cloud computing: An abstraction-driven perspective,” *Services Computing, IEEE Transactions on*, vol. 8, no. 6, pp. 945–957, Nov 2015.
- [62] V. Schuchardt, “Moving mobile applications between mobile devices seamlessly,” in *Software Engineering (ICSE), 2012 34th International Conference on*, June 2012, pp. 1595–1598.
- [63] K. P. Thant and T. T. Naing, “A migration framework for ubiquitous computing applied in mobile applications,” in *Information and Telecommunication Technologies, 2005. APSITT 2005 Proceedings. 6th Asia-Pacific Symposium on*, Nov 2005, pp. 213–218.
- [64] R. Ma and C.-L. Wang, “Lightweight application-level task migration for mobile cloud computing,” in *Advanced Information Networking and Applications (AINA), 2012 IEEE 26th International Conference on*, March 2012, pp. 550–557.
- [65] F. Deng, Q. Wang, and J. Shao, “Toward middleware-based online application migration,” in *Computer Software and Applications Conference, 2009. COMP-SAC '09. 33rd Annual IEEE International*, vol. 1, July 2009, pp. 222–227.

- [66] Y. Jiang, J. He, Q. Li, and X. Xiao, “A dynamic execution offloading model for efficient mobile cloud computing,” in *Global Communications Conference (GLOBECOM), 2014 IEEE*, Dec 2014, pp. 2302–2307.
- [67] M. Zbierski and P. Makosiej, “Bring the cloud to your mobile: Transparent offloading of html5 web workers,” in *Cloud Computing Technology and Science (CloudCom), 2014 IEEE 6th International Conference on*, Dec 2014, pp. 198–203.
- [68] S.-H. Hung, C.-S. Shih, J.-P. Shieh, C.-P. Lee, and Y.-H. Huang, “An online migration environment for executing mobile applications on the cloud,” in *Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2011 Fifth International Conference on*, June 2011, pp. 20–27.
- [69] 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*, ser. MobiSys ’10. New York, NY, USA: ACM, 2010, pp. 49–62. [Online]. Available: <http://doi.acm.org.ezlibproxy1.ntu.edu.sg/10.1145/1814433.1814441>
- [70] 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*, ser. EuroSys ’11. New York, NY, USA: ACM, 2011, pp. 301–314. [Online]. Available: <http://doi.acm.org/10.1145/1966445.1966473>
- [71] S. Kosta, A. Aucinas, P. Hui, R. Mortier, and X. Zhang, “Thinkair: Dynamic resource allocation and parallel execution in the cloud for mobile code offloading,” in *INFOCOM, 2012 Proceedings IEEE*, March 2012, pp. 945–953.
- [72] V. Andrikopoulos, Z. Song, and F. Leymann, “Supporting the migration of applications to the cloud through a decision support system,” in *Cloud Computing (CLOUD), 2013 IEEE Sixth International Conference on*, June 2013, pp. 565–572.
- [73] Y. Zhang, R. Yang, T. Wo, C. Hu, J. Kang, and L. Cui, “Cloudap: Improving the qos of mobile applications with efficient vm migration,” in *High Performance Computing and Communications, Embedded and Ubiquitous Computing (HPCC_EUC), 2013 IEEE 10th International Conference on*, Nov 2013, pp. 1374–1381.

- [74] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The case for vm-based cloudlets in mobile computing," *Pervasive Computing, IEEE*, vol. 8, no. 4, pp. 14–23, Oct 2009.
- [75] K. Elgazzar, P. Martin, and H. Hassanein, "Cloud-assisted computation offloading to support mobile services," *Cloud Computing, IEEE Transactions on*, vol. PP, no. 99, pp. 1–1, 2014.
- [76] A. Jin, W. Song, P. Wang, D. Niyato, and P. Ju, "Auction mechanisms toward efficient resource sharing for cloudlets in mobile cloud computing," *Services Computing, IEEE Transactions on*, vol. PP, no. 99, pp. 1–1, 2015.
- [77] Y. Wang, J. Wei, and K. Vangury, "Bring your own device security issues and challenges," in *Consumer Communications and Networking Conference (CCNC), 2014 IEEE 11th*, Jan 2014, pp. 80–85.
- [78] M. Ji, S. Kim, Y. Park, and J. H. Yi, "Mobile device management system with portable devices," in *Consumer Electronics (ISCE), 2015 IEEE International Symposium on*, June 2015, pp. 1–2.
- [79] M. Eslahi, M. Naseri, H. Hashim, N. Tahir, and E. Saad, "Byod: Current state and security challenges," in *Computer Applications and Industrial Electronics (ISCAIE), 2014 IEEE Symposium on*, April 2014, pp. 189–192.
- [80] M. P. Souppaya and K. A. Scarfone, "Guidelines for managing the security of mobile devices in the enterprise," *Special Publication (NIST SP)-800-124 Rev 1*, June 2013.
- [81] (2011) Juniper mobile security report 2011 unprecedented mobile threat growth. [Online]. Available: <http://forums.juniper.net/t5/Security-Now/Juniper-Mobile-Security-Report-2011-Unprecedented-Mobile-Threat/ba-p/129529>
- [82] K. Johnson, "Sans mobility/byod security survey," in *SANS Institute White Paper*, March 2012.
- [83] C. P. Ram and G. Sreenivaasan, "Security as a service (sass): Securing user data by coprocessor and distributing the data," in *Trendz in Information Sciences Computing(TISC2010)*, Dec 2010, pp. 152–155.

- [84] C. Tang and J. Liu, “Selecting a trusted cloud service provider for your saas program,” *Computers Security*, vol. 50, pp. 60 – 73, 2015. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0167404815000139>
- [85] T. Xie and X. Qin, “Security-aware resource allocation for real-time parallel jobs on homogeneous and heterogeneous clusters,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 19, no. 5, pp. 682–697, May 2008.
- [86] H. Liang, D. Huang, L. X. Cai, X. Shen, and D. Peng, “Resource allocation for security services in mobile cloud computing,” in *Computer Communications Workshops (INFOCOM WKSHPS), 2011 IEEE Conference on*, April 2011, pp. 191–195.
- [87] M. Lin, L. Xu, L. T. Yang, X. Qin, N. Zheng, Z. Wu, and M. Qiu, “Static security optimization for real-time systems,” *IEEE Transactions on Industrial Informatics*, vol. 5, no. 1, pp. 22–37, Feb 2009.
- [88] L. Zeng, B. Veeravalli, and X. Li, “Saba: A security-aware and budget-aware workflow scheduling strategy in clouds,” *Journal of Parallel and Distributed Computing*, vol. 75, pp. 141 – 151, 2015. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0743731514001658>
- [89] F. A. Guenane, “Network security management using a novel firewall cloud-based service,” in *2015 International Conference on Protocol Engineering (ICPE) and International Conference on New Technologies of Distributed Systems (NTDS)*, July 2015, pp. 1–6.
- [90] Y. Sun, S. Nanda, and T. Jaeger, “Security-as-a-service for microservices-based cloud applications,” in *2015 IEEE 7th International Conference on Cloud Computing Technology and Science (CloudCom)*, Nov 2015, pp. 50–57.
- [91] ENISA, “Incentives and barriers of the cyber insurance market in Europe,” Tech. Rep., Jun. 2012. [Online]. Available: <https://www.enisa.europa.eu/publications/incentives-and-barriers-of-the-cyber-insurance-market-in-europe>
- [92] B. Filkins, “Quantifying Risk: Closing the Chasm Between Cybersecurity and Cyber Insurance,” SANS Institute InfoSec Reading Room, Tech. Rep., Mar. 2016. [Online]. Available: <https://www.sans.org/reading-room/whitepapers/analyst/quantifying-risk-closing-chasm-cybersecurity-cyber-insurance-36770>

- [93] R. Anderson, R. Böhme, R. Clayton, and T. Moore, “Security Economics and the Internal Market,” ENISA, Tech. Rep., Jan. 2008. [Online]. Available: <https://www.enisa.europa.eu/publications/archive/economics-sec/>
- [94] C. Biener, M. Eling, and J. H. Wirfs, “Insurability of Cyber Risk: An Empirical Analysis,” University of St. Gallen, School of Finance, Working Papers on Finance 1503, Jan. 2015. [Online]. Available: <https://ideas.repec.org/p/usg/sfwphi/201503.html>
- [95] T. Bandyopadhyay, “Organizational adoption of cyber insurance instruments in it security risk management: A modeling approach,” in *Southern Association for Information Systems (SAIS) 2012 Proceedings*, 2012. [Online]. Available: <http://aisel.aisnet.org/sais2012>
- [96] (2006) The security risk management guide. [Online]. Available: <http://trygstad.rice.iit.edu:8000/Books/TheSecurityRiskManagementGuide-Microsoft.pdf>
- [97] “Information technology – security techniques – information security management systems – requirements,” ISO/IEC 27001, Standard, Jan. 2013.
- [98] W. J. Y. J. P. Kesan, R. P. Majuca, “The Economic Case for Cyberinsurance,” University of Illinois College of Law, Law and Economics Working Papers Series LE04-004, Nov. 2009.
- [99] R. Pal, L. Golubchik, and K. Psounis, *Aegis A Novel Cyber-Insurance Model*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 131–150. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-25280-8_12
- [100] C. Barracchini and E. A. M., “Cyber risk and insurance coverage: An actuarial multistate approach,” *Review of Economics Finance*, vol. 4, pp. 57–69, 2014. [Online]. Available: <http://EconPapers.repec.org/RePEc:bap:journl:140105>
- [101] R. Pal, L. Golubchik, K. Psounis, and P. Hui, “Will cyber-insurance improve network security? a market analysis,” in *IEEE INFOCOM 2014 - IEEE Conference on Computer Communications*, April 2014, pp. 235–243.
- [102] C. Villani, “The Wasserstein Distances,” in *Optimal Transport*, ser. Grundlehren der mathematischen Wissenschaften. Springer Berlin Heidelberg, 2009, vol. 338, pp. 93–111. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-71050-9_6

- [103] H. Heitsch and W. Rmisch, “Scenario Reduction Algorithms in Stochastic Programming,” *Computational Optimization and Applications*, vol. 24, no. 2-3, pp. 187–206, 2003. [Online]. Available: <http://dx.doi.org/10.1023/A3A1021805924152>
- [104] B. Jansen, J. de Jong, C. Roos, and T. Terlaky, “Sensitivity Analysis in Linear Programming: Just Be Careful!” *European Journal of Operational Research*, vol. 101, no. 1, pp. 15 – 28, 1997. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0377221796001725>
- [105] (2015) GAMS Solvers. [Online]. Available: <http://www.gams.com/dd/docs/solvers/allsolvers.html>
- [106] J. Wilkes and C. Reiss. (2014) Google Cluster Data. [Online]. Available: https://code.google.com/p/googleclusterdata/wiki/ClusterData2011_2
- [107] (1993) LBL-CONN-7. [Online]. Available: <ftp://ita.ee.lbl.gov/html/contrib/LBL-CONN-7.html>
- [108] (2014) How Can I Control How Much Data Netflix Uses? [Online]. Available: <https://help.netflix.com/en/node/87>
- [109] (2014) Ookla Net Index. [Online]. Available: <http://www.netindex.com/>
- [110] C. Li, A. Raghunathan, and N. K. Jha, “A trusted virtual machine in an untrusted management environment,” *IEEE Transactions on Services Computing*, vol. 5, no. 4, pp. 472–483, Jan. 2012. [Online]. Available: <http://dx.doi.org/10.1109/TSC.2011.30>
- [111] (2016) Enterprise mobility management - airwatch. [Online]. Available: <https://www.vmware.com/products/enterprise-mobility-management/features.html>
- [112] (2016) Pulse secure connect secure. [Online]. Available: <https://www.pulsesecure.net/products/connect-secure/>
- [113] (2003) Subgradient methods. [Online]. Available: https://web.stanford.edu/class/ee392o/subgrad_method.pdf
- [114] S. Boyd and L. Vandenberghe, *Convex Optimization*. New York, NY, USA: Cambridge University Press, 2004.

- [115] S. Alam, Z. Abdullah, and N. Ahsan, “Cyber caf usage in malaysia: An exploratory study,” *Journal of Internet Banking and Commerce*, vol. 14, no. 1, 4 2009.
- [116] D. Kotz and K. Essien, “Analysis of a campus-wide wireless network,” *Wirel. Netw.*, vol. 11, no. 1-2, pp. 115–133, Jan. 2005. [Online]. Available: <http://dx.doi.org/10.1007/s11276-004-4750-0>
- [117] (2015) How much bandwidth does skype need? [Online]. Available: <https://support.skype.com/en/faq/fa1417/how-much-bandwidth-does-skype-need>
- [118] (2015) Internet cafes in singapore. [Online]. Available: <http://www.world66.com/asia/southeastasia/singapore/internetcafes>
- [119] R. Winton. (2016) Hollywood hospital pays \$17,000 in bitcoin to hackers; fbi investigating. [Online]. Available: <http://www.latimes.com/business/technology/la-me-ln-hollywood-hospital-bitcoin-20160217-story.html>
- [120] E. Santos, Jr. and E. S. Santos, “Polynomial solvability of cost-based abduction,” *Artif. Intell.*, vol. 86, no. 1, pp. 157–170, Sep. 1996. [Online]. Available: [http://dx.doi.org/10.1016/0004-3702\(96\)00016-1](http://dx.doi.org/10.1016/0004-3702(96)00016-1)
- [121] (2015) 2015 Ponemon Institute cost of cyber crime study. [Online]. Available: <http://www8.hp.com/us/en/software-solutions/ponemon-cyber-security-report/>
- [122] C. Marciano, “How much does cyber/data breach insurance cost?” Jun. 2016. [Online]. Available: <https://databreachinsurancequote.com/cyber-insurance/cyber-insurance-data-breach-insurance-premiums/>
- [123] (2016) Cybersecurity Researchers of Waikato. [Online]. Available: <https://crow.org.nz/>
- [124] (2016) Snort IDS. [Online]. Available: <https://www.snort.org/>

Author's Publications (Published/Accepted)

Journal

- [1] **J. Chase**, and D. Niyato “Joint Optimization of Resource Provisioning in Cloud Computing,” *IEEE Transactions on Services Computing*, accepted for final publication. 2015.
- [2] **J. Chase**, D. Niyato, P. Wang, S. Chaisiri, and R.K.L. Ko “A Scalable Approach to Joint Cyber Insurance and Security-as-a-Service Provisioning in Cloud Computing,” *IEEE Transactions on Dependable and Secure Computing*, accepted for final publication. 2017.

Conference

- [1] **J. Chase**, D. Niyato, and S. Chaisiri, “Bring-Your-Own-Application (BYOA): Optimal Stochastic Application Migration in Mobile Cloud Computing,” *IEEE Global Communications Conference*, pp. 1-6, San Diego, CA, USA, December. 2015.
- [2] **J. Chase**, D. Niyato, R. Kaewpuang, and Y. Wen, “Joint virtual machine and bandwidth allocation in software defined network (SDN) and cloud computing environments,” *IEEE International Conference on Communications*, pp. 2969 - 2974, Sydney, Australia, June. 2014.

Author's Publications (Submitted)

Journal

- [1] **J. Chase**, D. Niyato, Z. Cao, and H. Guo “Bring-Your-Own-Application (BYOA): A Scalable Stochastic Mobile Application Migration Service,” *IEEE Transactions on Mobile Computing*, submitted in September 2016, under review.