The Pennsylvania State University
The Graduate School

RESOURCE ALLOCATION FOR EDGE CLOUD AND SMART GRID

A Dissertation in
Computer Science and Engineering
by
Vajiheh Farhadi

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

May 2022

The dissertation of Vajiheh Farhadi was reviewed and approved by the following:

Thomas F. La Porta
Evan Pugh Professor, William E. Leonhard Professor
Dissertation Co-Advisor
Committee Co-Chair

Ting He
Associate Professor
Dissertation Co-Advisor
Committee Co-Chair

Mort Webster
Professor

Nilanjan Ray Chaudhuri
Associate Professor

Chitaranjan Das
Professor
Head of the Department

Abstract

This dissertation develops solutions for optimized resource allocation in various distributed systems, including data-intensive applications in edge clouds and SCADA-based control systems in power grid networks.

The abstract resource allocation problem concerns how to optimally use resources for different tasks. In the context of this dissertation, the resources are CPU cycles, wireless link bandwidth, and storage space in mobile edge computing networks or the budget for deploying communication links in the smart grid network.

The optimized resource allocation problem for data-intensive applications in edge clouds: Mobile edge computing provides a highly distributed computing environment that can be used to deploy applications and services as well as to store and process content in close proximity to mobile users. In this dissertation, the research on mobile edge computing begins by jointly considering service placement and request scheduling problems for data-intensive applications in edge clouds under communication, computation, and storage constraints. A budget constraint to control the operation cost due to service replication/migration is also imposed. To properly formulate this problem, we separate the time scales of the two decisions: service placement occurs at a larger scale (frames) to prevent system instability, and request scheduling occurs at a smaller scale (slots) to support real-time services. We fully characterize the complexity of our problem by analyzing the hardness of various cases. By casting our problem as a set function optimization, we develop a polynomial-time algorithm that achieves a constant-factor approximation under certain conditions. Furthermore, we develop a polynomial-time request scheduling algorithm by computing the maximum flow in a constructed auxiliary graph, which satisfies hard resource constraints and is provably optimal in the special case where requests have homogeneous resource demands.

Extensive synthetic and trace-driven simulations show that the proposed algorithm serves 90% of the requests that could be served from the edge under the optimal solution.

The optimized resource allocation problem for smart grid systems: In this dissertation, the research on the control network of the smart grid addresses the optimized allocation of budget for deploying the communication links in a system in which the power grid is coupled with a geographically co-located SCADA-based communication network. We consider the problem of Power Line Carrier Communication (PLCC) and non-PLCC (fiber, microwave, etc.) link allocation and study the impact of the coupling between the communication and the power networks as it affects a SCADA-based preventive control system in the occurrence of the cascading failures. Loss of a power transmission line disables PLCC links; hence failure in the power network leads to an outage in the communication network and consequently disrupts monitoring and control of the power system. Non-PLCC links are immune to failures in the power grid; however, they are costly.

In the first part of the research on optimized resource allocation in smart grids, we start with a baseline where all the communication links are PLCC. Next, we pose the problem of allocating non-PLCC communication links with a budget constraint for a given power system with a given control center location. The ultimate objective of the design of the communication network is to ensure that the re-dispatch-based preventive control can effectively restrict the propagation of cascade. We propose a heuristic that enhances the total demand served at the end of cascade, and show its effectiveness when tested in a 2383-bus Polish network.

In the second part, we focus on (a) relaxing the idealistic assumption that the topology of the power grid (all breaker statuses of the lines) are known, and (b) assuming that only the minimum number of lines are associated with communication links. Meanwhile, the observability of all nodes at the control center before cascade is provided. Such a realistic scenario poses new challenges that were not present in the previous case. In the absence of a fully known admittance matrix (which provides the topology of the power system), we use the estimated admittance matrix proposed in [1]. In this work, the authors divide the problem of the admittance matrix estimation into two steps, first detecting the islands formed within in power grid due to failure, and second estimating connectivity of unobservable lines within islands. The effectiveness of the redispatch-based control depends upon the

accuracy of these two steps. Here, we propose a novel scheme in designing the communication network comprised of both PLCC and non-PLCC links in preparation of possible failures under a budget constraint on the communication link deployment cost. First, we characterize the fundamental hardness of our problem. Next, we develop a solvable MILP-based algorithm, which attains a constant-factor approximation under certain conditions. Finally, we show via simulations on the IEEE 118-bus system that the proposed algorithm achieves superior performance in terms of enabling more accurate topology estimation and more served demand in the face of cascades.

Contents

List of Figures **x**

List of Tables **xiii**

Acknowledgements **xv**

1 Introduction **1**
1.1 Background and Motivation 1
1.1.1 Mobile Edge Computing 1
1.1.2 Smart Grid 3
1.2 Challenges & Our Contributions 4
1.2.1 Mobile Edge Computing 4
1.2.2 Power Grid 7
1.3 Organization 8

2 Related Work **9**

3 Service Placement and Request Scheduling for Data-intensive Applications in Edge Clouds **14**
3.1 Introduction 15
3.2 Background & Contribution 15
3.3 Problem Formulation 18
3.3.1 System Model 18
3.3.2 Underlying Optimization Problems 21
3.3.2.1 Service Placement with Shadow Scheduling . 21
3.3.2.2 Request Scheduling under Soft Resource Constraints 22

3.3.2.3 Request Scheduling under Hard Resource Constraints . . . 23
3.4 Complexity Analysis . . . 24
3.4.1 Complexity of Service Placement . . . 24
3.4.1.1 Having B-constraint Only . . . 25
3.4.1.2 Having R-constraint Only . . . 26
3.4.1.3 Removing R- and B-constraints . . . 27
3.4.1.4 Summary of All Cases . . . 28
3.4.2 Complexity of Request Scheduling . . . 28
3.4.2.1 General Case . . . 29
3.4.2.2 Homogeneous Special Case . . . 30
3.4.2.3 Summary of All Cases . . . 30
3.5 Algorithms . . . 30
3.5.1 Approximation Algorithm for Service Placement . . . 31
3.5.1.1 Conversion to Set Function Optimization . . . 31
3.5.1.2 Algorithm . . . 35
3.5.1.3 Complexity . . . 35
3.5.2 Optimal Request Scheduling under Soft Constraints . . . 35
3.5.3 Optimal and Heuristic Request Scheduling under Hard Constraints . . . 36
3.5.3.1 Optimal Algorithm for Homogeneous Requests 36
3.5.3.2 Heuristic Algorithm for Heterogeneous Requests . . . 38
3.6 Extension to Multi-frame Optimization . . . 39
3.7 Performance Evaluation . . . 40
3.7.1 Benchmarks . . . 41
3.7.2 Results on Service Placement . . . 41
3.7.2.1 Synthetic simulation . . . 41
3.7.2.2 Trace-driven simulation . . . 45
3.7.3 Results on Request Scheduling . . . 49
3.7.3.1 Soft Constraints . . . 49
3.7.3.2 Hard Constraints . . . 50
3.7.4 Results on Multi-frame Extension . . . 52
3.8 Conclusion . . . 53

4 Budget-Constrained Reinforcement of SCADA for Cascade Mitigation 55
4.1 Introduction . . . 56

4.1.1 Summary of Contributions . . . 57
4.2 Background and Motivation . . . 58
4.2.1 Modeling Coupled Cascading Failure . . . 58
4.2.2 Modeling Preventive Control . . . 59
4.2.3 Motivating Example . . . 61
4.3 Budget-Constrained Reinforcement of Communication Network . . . 62
4.3.1 Problem Formulation . . . 62
4.3.2 Complexity Analysis . . . 64
4.3.3 Algorithm Design . . . 65
4.3.3.1 Generic heuristic . . . 65
4.3.3.2 Domain-specific heuristic . . . 65
4.4 Performance Evaluation . . . 69
4.5 Conclusion . . . 75

5 Improvement of SCADA-based Preventive Control Under Budget Constraints 77
5.1 Introduction . . . 78
5.1.1 Background & Contribution . . . 78
5.1.2 Research Gap and Challenges . . . 81
5.2 System Model & Motivation . . . 82
5.2.1 System Model . . . 83
5.2.2 Preventive Control & Topology Estimation . . . 83
5.2.2.1 Example Application 1 - Preventive Control . 84
5.2.2.2 Example Application 2 - Topology Estimation 84
5.2.3 Motivating Experiment . . . 85
5.3 Problem Formulation . . . 87
5.3.1 Underlying Optimization Problem . . . 88
5.3.2 Complexity analysis . . . 90
5.4 Approximation Algorithm . . . 91
5.5 Performance Evaluation . . . 96
5.5.1 Benchmarks and metrics . . . 96
5.5.1.1 (α, β)-Approximation of CMST . . . 97
5.5.1.2 BC method . . . 98
5.5.2 Simulation Setup . . . 99
5.5.3 Results . . . 99
5.5.3.1 Setting Design Parameters . . . 101
5.5.3.2 Overall Comparison of All Algorithms . . . 106

5.6 Conclusion 108

6 Conclusion and Future Works **110**
6.1 Summary of Contributions 110
6.2 Conclusion & Future Works 112

Bibliography **114**

List of Figures

3.1 System model. 18
3.2 Time scales of service placement and request scheduling. . . . 19
3.3 Illustration of 2DSC. 27
3.4 Complexity of service placement (3.1). 28
3.5 Complexity of request scheduling under hard constraints (3.2). 30
3.6 Auxiliary graph $\mathcal{G}$ for request scheduling. 36
3.7 Performance evaluation for service placement under the synthetic simulation setup in Section 3.7.2. 43
3.8 Performance evaluation for service placement under highly popular data-intensive service input. 45
3.9 Varying the frame size. 47
3.10 Varying the slot duration. 48
3.11 The variability of the user and requests in a cell 48
3.12 Performance evaluation in trace-driven simulation under soft constraints. 49
3.13 Performance evaluation for request scheduling under hard constraints. 51
3.14 Performance evaluation in trace-driven simulation under hard constraints (scheduling done by LRRS). 52
3.15 Performance of extended GSP-SS for multi-frame optimization in trace-driven simulation. 53

4.1 Flow chart highlighting the modeling of the coupled cascading failure in power and communication networks. 58
4.2 The effect of preventive control on cascade propagation. (a) initial failures in the Polish grid (red links), (b) cascading failures (black links) without control (as a zoom-in of the rectangle in (a)), (c) cascading failures (black links) with control of the two red nodes (as a zoom-in of the rectangle in (b)). . . . 62

4.3 Subgraphs of the system for the domain-specific heuristic, $h = 10$. 68
4.4 Performance evaluation in terms of change in D_{tot}^{p} with respect to 100% PLCC case for different node weights (non-PLCC links are placed by the generic heuristic under $L = 200$, $N = 50$, and $B = 17$). 69
4.5 Performance evaluation for the generic heuristic under different design parameters: (a) $N = 50$ & varying L, (b) $L = 200$ & varying N ($B = 17$ in both cases). 70
4.6 Performance evaluation for the domain-specific heuristic under different design parameters ($B = 17$). 72
4.7 Performance evaluation of change in D_{tot}^{p} with respect to 100% PLCC case (non-PLCC links are placed by different methods under (a-d) $B = 174$, (e-h) $B = 17$ ($L = 200$ and $N = 50$ in the generic heuristic and $h = 10$ in the domain-specific heuristic). 73
4.8 Graphs showing the non-PLCC links selected by different heuristics. Dark edges: non-PLCC, light edges: PLCC, CC: control center. 76

5.1 Performance evaluation of scenarios (a) randomly selecting PLCC links under $\mathcal{B} = 40$ and (b) mixture of suitably placed PLCC and non-PLCC links under $\mathcal{B} = 35$, in terms of median and lower adjacency of D_{tot}^{p}. 85
5.2 Graphs showing (a) the power grid topology and (b-g) the communication network obtained from different design methods. Black edges: PLCC, red edges: non-PLCC, CC: control center 100
5.3 Performance evaluation of approximation algorithm in terms of median and lower adjacency of D_{tot}^{p} for different node weight definition under $\mathcal{B} = 40$, $N = 5$ and $L = 20$. 102
5.4 Performance evaluation of approximation algorithm in terms of median and lower adjacency of D_{tot}^{p} for different N under $\mathcal{B} = 40$ and $L = 20$. 103
5.5 Performance evaluation of approximation algorithm in terms of median and lower adjacency of D_{tot}^{p} for different L under $\mathcal{B} = 40$ and $N = 5$. 103

5.6 Performance evaluation of proposed approximation algorithm in terms of median and lower adjacency of D^p_{tot} for different $\mathcal{B}$ under $N = 5$ and $L = 20$. 104
5.7 Performance evaluation of all algorithms in terms of median and lower adjacency of D^p_{tot} under $\mathcal{B} = 40$, ($N = 5$ and $L = 20$ for approximation). 106
5.8 Performance evaluation of different methods in terms of percentage of misses and false alarms of line outage identification under $\mathcal{B} = 40$, ($N = 5$ and $L = 20$ for proposed approximation method). 108

List of Tables

3.1 Table of notations 20
3.2 Capacity Violations (%) 50

4.1 The effect of preventive control on cascade in (i) 100%-PLCC links and (ii) no communication network. 61
4.2 Statistical measures of variables in the domain-specific heuristic under $B = 17$ 72

5.1 Literature on cascading failure in power grid in presence/absence of control network 82
5.2 Percentage of (i) valuable and (ii) highly observable islands in different scenarios (a) randomly selecting PLCC links under $\mathcal{B} = 40$ and (b) mixture of suitably placed PLCC and non-PLCC links under $\mathcal{B} = 35$. 86
5.3 Percentage of estimation accuracy in different scenarios (a) randomly selecting PLCC links under $\mathcal{B} = 40$ and (b) mixture of suitably placed PLCC and non-PLCC links under $\mathcal{B} = 35$ for (i) valuable, (ii) highly observable and (iii) slightly observable islands. 86
5.4 Table of notations 88
5.5 The effect of different node weight definition on approximation algorithm on percentage of (i) valuable and (ii) highly observable islands under $\mathcal{B} = 40$, $N = 5$ and $L = 20$. 101
5.6 The effect of varying N on percentage of (i) valuable and (ii) highly observable islands under $\mathcal{B} = 40$ and $L = 20$. 102
5.7 The effect of varying L on percentage of (i) valuable and (ii) highly observable islands under $\mathcal{B} = 40$ and $N = 5$. 103

5.8 The effect of varying budget in proposed approximation algorithm on percentage of (i) valuable and (ii) highly observable islands under $N = 5$ and $L = 20$. 105
5.9 The effect of varying budget in proposed approximation algorithm on percentage of estimation accuracy for (i) valuable, (ii) highly observable and (iii) slightly observable islands (under $N = 5$ and $L = 20$). 105
5.10 Performance of different methods in terms of (i) percentage of valuable and (ii) percentage of highly observable islands under $\mathcal{B} = 40$, ($N = 5$ and $L = 20$ for approximation). 106
5.11 Performance evaluation of all algorithms in terms of percentage of estimation accuracy in (i) valuable, (ii) highly observable and (iii) slightly observable islands under $\mathcal{B} = 40$, ($N = 5$ and $L = 20$ for approximation method). 107

Acknowledgment

I would first like to express my sincere gratitude to my advisor, Dr. Thomas La Porta, for his invaluable advice, motivation, and mentorship throughout my Ph.D. work. I would like to thank my co-advisor, Dr. Ting He, for the continuous support of my Ph.D. research, for her patience, enthusiasm, and immense knowledge. It is thanks to Dr. La Porta's and Dr. He's mentorships that I have become the researcher that I am today.

My sincere thanks also go to Dr. Nilanjan Ray Chaudhuri for his collaboration in my work in the smart grid area. Collaborating with him has been very enriching and informative.

I would like to thank the rest of my thesis committee, Dr. Nilanjan Ray Chaudhuri and Dr. Mort Webster, for their encouragement, insightful comments, and hard questions.

I would like to thank my other collaborators, Dr. Gopal Vennelaganti, Dr. Fidan Mehmeti, Dr. Hana Khamfroush, Dr. Shiqiang Wang, and Dr. Kevin Chan, for their help and support.

The work presented in this dissertation was funded by the U.S. Army Research Laboratory and the U.K. Ministry of Defence under Agreement W911NF-16-3-0001 and National Science Foundation (NSF) Grant Award ECCS 1836827. The completion of my Ph.D. would not have been possible without their support.

Last but not least, I would like to thank my family for supporting me unconditionally throughout my life.

Chapter 1

Introduction

This chapter presents an introduction to the work presented in this dissertation. It begins with a description of the key concepts in optimally determining the allocation of resources in edge cloud computing applications. Next, it addresses optimal resource allocation problems in the smart grid, with a focus on allocating different types of communication links for the control network.

This is followed by a brief discussion on motivations and challenges pertaining to the problems we address, and our contributions.

1.1 Background and Motivation

Resource allocation is the process by which network elements try to satisfy the competing demands that applications have for network resources such as CPU cycles, wireless link bandwidth, and storage space in mobile edge computing networks, or the budget for deploying communication links in the smart grid networks. Note that some applications/users may be allocated fewer network resources than they request, where selecting the appropriate amount is part of the resource allocation problem.

1.1.1 Mobile Edge Computing

The first part of this work focuses on resource allocation in mobile edge computing technology.

Progress in performance, cost, and availability of Internet user devices,

distributed computing platforms, and network technologies have resulted in the introduction of many novel applications such as augmented reality, cloud robotics, automated vehicles, video surveillance, streaming, smart homes, and Internet of Things (IoT) in diverse areas including healthcare, security, entertainment, mining, and transportation. These applications intensify the importance of supervision in managing and handling data and latency in the network. The new distributed applications are often bandwidth-hungry (in applications such as video conferencing/monitoring) and latency-sensitive (in applications such as robotic surgery and automated vehicles). These characteristics introduce challenges in terms of reliability, operating cost, and performance of applications and services [2].

The distributed mobile edge clouds give new opportunities to manage computing resources and allocate those resources to applications to minimize the overall cost of deployment while satisfying user demands. From desktop computing, through grid computing, and now to cloud computing, the concept of distributed computational power being made available to a huge number of end-users in an efficient fashion has evolved through different stages and moved onto different platforms. Cloud computing, another avatar of distributed computing, brings flexibility, scalability, and cost-effectiveness into various sciences and commercial practices [3].

However, as the computing power in the mobile edge computing environment increases, the demands on these networks have grown tremendously [4]. Disparate wireless users are running resource-intensive and delay-sensitive applications from the edge of mobile networks. The environments that run these applications are referred to as edge clouds [5], cloudlets [6], fog [7], follow me cloud [8], or micro clouds [9]. Mobile applications are increasingly resource-demanding as they address use cases based on big data and machine learning problems. As users access these resource-hungry applications via bandwidth-limited wireless links, how to optimally allocate the limited resources at edge clouds to competing demands (e.g., to minimize the response time or the operation cost) poses a difficult but intriguing research question, which has attracted tremendous interest in the research community.

The use of a centrally placed cloud to address these demands is now proved inadequate. The concept of mobile edge computing is emerging as an answer to meet the need of resource-intensive and delay-sensitive applications that need data processing in real-time. Optimally allocating the limited resources at edge clouds to competing demands is the focus of current research. There are notable subjects in this field that are being targeted here: (i) allo-

cation of computing resources, (ii) allocation of network resources, (iii) where to execute applications, and (iv) when to execute applications.

1.1.2 Smart Grid

The concept of the smart grid with the promise of revolutionizing the production, delivery, and utilization of electricity has attracted significant research attention in recent years. Power grids are critical infrastructures and their proper functioning is vital for modern society. For example, large-scale blackouts in the power grid due to natural disasters or malicious attacks can potentially lead to huge damage and could cost billions of dollars [10].

In the electrical energy cyber-physical system (CPS), the physical system includes generators, transformers, loads, and transmission lines, whereas the legacy Supervisory Control and Data Acquisition (SCADA) system along with the more modern Wide-Area Monitoring, Protection, and Controls (WAMPAC) system constitute the cyber system [11]. Cascading failure propagation influences the physical components coupled with the cyber component in a complex fashion, such that the failure of one or a few components may cause the failure of the entire interconnected system. The efficient preventive process requires the system to be designed precisely, particularly in the face of massive failures in the power grid.

In the smart grid, a Control Center (CC) attempts to mitigate cascading failures via monitoring node/link states in the grid and taking preventive control actions through actuators deployed in selected nodes (e.g., generators/loads). The communication medium, especially whether a communication link is realized through Power Line Carrier Communication (PLCC) [12], has a significant impact on the interdependency between the power and communication networks. A PLCC link piggybacks on a power line, and hence the outage of the power line will lead to the failure of the link. In contrast, a non-PLCC link carries information on a dedicated communication medium, and outages in the power grid do not directly fail the link. However, deploying such links incurs nontrivial costs and thus must be designed carefully.

This dissertation examines the impact of coupling between the communication and the power networks as it affects a SCADA-based preventive control system. In this regard, we combine the cost efficiency of PLCC links and the reliability of non-PLCC links to improve the robustness of the power system against cascading failures under a budget constraint.

1.2 Challenges & Our Contributions

The main focus of this dissertation is allocating resources in distributed systems to best serve the applications of interest under resource capacity and operation budget constraints. However, there are multiple challenges that need to be fully addressed in order to achieve our goal.

1.2.1 Mobile Edge Computing

There are issues that must be considered to optimally allocate resources in edge cloud computing applications:

(i) Data and Service Placement: Certain applications require a large amount of data on the server, and demands may arise from multiple parts of the network. This creates the need to offer the service at multiple edge clouds.

Early experimental studies were conducted under the strategy of always serving each user request from the closest edge cloud [13]. However, for mobile users to be continuously connected to the nearest location, the issue of when and where [14, 15, 5, 16], and how [17] to migrate services while maintaining a balance between service quality and migration cost becomes a challenge. It has been observed that at times of heavy demands, the nearest edge cloud may not be the answer for the best service [18, 19, 20]. Meanwhile, there are initiatives to develop standards [21, 22, 23] for pooling available edge computing resources within the same geographical region that can be shared among contending user requests. Hence, for the best placement of the services, where and how many, have to be determined carefully.

(ii) Optimized Request Scheduling: Scheduling in large scale computing clusters is critical to job performance and resource utilization. As a cluster size grows to thousands of users and scheduling needs become complex and varied, request scheduling in cloud-scale clusters poses unique challenges, i.e., on which edge server, if any, to schedule each user request such that certain objectives (e.g., cost, completion time) can be optimized [24, 25]. Existing works typically assume that serving each request requires a dedicated share of resources (e.g., CPU cycles, memory space, network bandwidth), such that the total resource consumption at a server is the accumulation of resource requirements scheduled to it.

Scheduling approaches for data-intensive applications may couple user

movement to request submission. Initial efforts in scheduling centered on the reuse of cached requests, namely heuristics for scheduling independent tasks sharing common files, on a Grid made of interconnected clusters [26]. While this approach works for applications that do not require huge amounts of data on the server applications, data-intensive applications i.e, applications dealing with augmented reality, video analytics, distributed machine learning and so on, require both a dedicated share of resources and a nontrivial amount of data at the server (e.g., object database, trained machine learning models). Resources required for storing such data as trained machine learning models or object databases differ critically from standard resources in that they are amortized over all requests over the same copy of data. Further, many other data-intensive applications use enormous amounts of user-provided data (e.g., images captured by the user), even though resources for collecting/storing such requests are dedicated on a request-by-request basis.

Thus, in addition to the traditional resources of CPU cycles and memory, resource allocation algorithms for data-intensive applications must also consider storage resources for storing server data and network bandwidth for receiving requests that contain large amounts of user-provided data.

(iii) Joint Service Placement and Request Scheduling: Joint allocation of dedicated and amortized resources splits one big challenge into two smaller ones [27]: (i) service placement, which determines replication and placement of each service (including server code and data) within the storage capacity of each edge cloud, and (ii) request scheduling, which determines whether and where to place scheduling requests according to the communication and computation capabilities of edge clouds and other constraints, such as maximum delay. However, these issues, i.e., service placement and request scheduling, are not decoupled from each other as the edge cloud scheduled to process a request must have a replica of the requested service.

(iv) Budget Constraints: The ability to serve requests and the overall cost depends on the selection of the replica of the requested service of sources and their scheduling. Selection of the best computing resources regardless of the placement of data, as is done in existing scheduling algorithms, does not give time and cost-efficient schedules when time to fetch data is comparatively larger than the computation time of requests. Significant bandwidth is required to stage-in and stage-out these data prior to serving requests. Similarly, if the data is to be re-used, the scheduling policy must select closer (in terms of network distance) computing resources. This affects the time and cost of transferring output data, and hence the overall execution time

and cost. The transfer time is drastically reduced by picking a computing host closer to the data-host.

Thanks to the fog and edge computing paradigms, maintaining the requirements of highly demanding services is attainable because computation capabilities have been shifted towards the edge of the network. To ensure that the quality of such services is still met in the event of user mobility, it is essential to migrate services across computing hosts. Several studies have been conducted to address service migration in different edge-centric research areas, including fog computing, multi-access edge computing, cloudlets, and vehicular clouds [28].

However, in cases of limited coverage of the computing sources, when users move, user communication may need to pass through multiple hops, which can impact the quality of service (QoS). To mitigate the consequences of such QoS degradation, services must be dynamically migrated to a better source, closer to the new user location [28]. Such a need for migration at the edge can be observed in the advent of the "Follow Me" trend, which leads to emerging terms in studies, such as Follow Me Cloud [29], Follow Me Edge [30], Follow Me Edge Cloud [31], Follow Me Fog [32], Move With Me [33], and Companion Fog Computing [34]. These works, along with many others, reemphasize the need for no interruption of service during migration, along with the need to adjust migration costs. In our work, we force a budget constraint to control the operation cost due to service replication/migration. These changes facilitate a controllable trade-off between the performance of serving requests and the reconfiguration cost, which requires critical changes in the underlying optimization problem.

Our main contributions in allocating resources in edge clouds are:

First, we aim to provide solutions for jointly considering service placement and request scheduling for data-intensive applications. We separate the time scales of the two decisions: to prevent system instability, service placement occurs at a larger scale (frames); to limit scheduling delay, request scheduling occurs at a smaller scale (slots). Besides, to control the operation cost due to service replication/migration, we force a budget constraint for service placement. Then, we extend the problem formulation to accommodate two formulations for the request scheduling subproblem, one under soft resource constraints where the resource capacities are only enforced on the average, and the other under hard resource constraints where the resource capacities are strictly enforced. This extension significantly improves the applicability

of our solutions, while introducing new challenges in terms of harder scheduling subproblem.

1.2.2 Power Grid

To optimally allocate different types of communication links for the control network in the smart grid, the following challenges must be met:

(i) Lack of Information: Lack of information is a serious challenge in the recovery phase of a power grid, which can be produced by the failure of sensors due to the same event that prompted power grid failure (e.g., a flood can destroy both the substation and the sensor/actuator at the substation), or the CC losing observability of sensors (e.g., due to the failure of a PLCC link). In either case, the CC faces uncertainty in the states (e.g., failed/operational) of both power grid elements (lines and substations) and communication network elements (sensors, actuators, PDCs, and communication links) in parts of the system that it can no longer reach. Moreover, the loss of a power line also leads to the loss of the corresponding PLCC link in the control network. This, in turn, may result in lack of observability and controllability, which negatively impacts the preventive control.

(ii) Budget Constraints: In the design of a SCADA communication network, budget constraints play a crucial role. Since power lines are already deployed in power grid, the deployment costs of PLCC links are largely confined to connecting repeaters and modems to the existing electrical grids, which typically costs much less than installing dedicated physically-separate communication media. Hence, PLCC links provide a means of supporting the SCADA system at a lower cost. However, power lines are primarily designed to be a transmission medium for electrical energy; hence, they may not be as suitable as data network media in terms of reliability, controllability, and security for data communication applications. Non-PLCC links are immune to failures in the power grid; however, they are more expensive than PLCC links. For non-PLCC links, fiber, copper, and microwave have been considered. Among these non-PLCC technologies, microwave provides the best flexibility and cost savings [35]. In our work, we force a budget constraint to control the deployment cost of communication networks in the smart grid.

In this dissertation, a coupled cascading failure model of a power grid with a SCADA-based communication network is studied, in which all the commu-

nication links are initially based on PLCC. To impose preventive control which restores power services in the the power grid as quickly as possible, we pose the problem of situating a set of non-PLCC links, albeit with a budget constraint limiting the number of such links.

We also minimize the chance of forming unobservable islands after cascade failure while addressing the cost of deploying both PLCC and non-PLCC links. Furthermore, we assumed that we are not aware of the full topology of the network (status of breakers), so our proposed algorithms must perform well in identifying islands correctly as well as maximizing total demands after failure.

1.3 Organization

In Chapter 2, we describe related works to our problems.

Chapter 3 presents our resource allocation problem for data-intensive applications in edge clouds by proposing a two-time-scale framework that jointly optimizes service (data/code) placement and request scheduling, under storage, communication, computation, and budget constraints [36]. Furthermore, we extend the problem formulation to improve the applicability of our solutions.

In Chapter 4, we study the progressive failure recovery under uncertainty on DC quasi-steady-state (QSS) models of cascading failure in power grid networks by optimally allocating non-PLCC communication links.

In Chapter 5, we propose a novel scheme in optimal design of the communication network comprised of both PLCC and non-PLCC links in the face of uncertain knowledge of failure and system topology, under a budget constraint on both PLCC and non-PLCC link deployment cost.

Finally, we summarize and conclude this dissertation in Chapter 6.

Chapter 2

Related Work

In this chapter, we provide a high-level overview of related work in resource allocation. Next, we narrow it down to edge cloud and smart grid applications. We include relevant related work in each chapter as well.

There are well-known methods for solving resource allocation problems. The most widely used mathematical optimization technique is linear programming (LP) [37, 38]. However, the chances of being able to formulate the problem using LP are rare. Researchers often attempt to approximate the non-linear parts of their problem by linear functions and then solve the modified problem using LP techniques. Branch and Bound, which first relaxes some of the constraints of the problem, is a technique often used to solve Integer Linear Programming (ILP) and Mixed-Integer Linear Programming (MILP) problems [39]. Lagrangian duality is another technique used for solving resource allocation problems. This method is based on dualizing all the major constraints, i.e., except for the non-negativity constraints [40]. This method uses the classical KKT conditions, assuming the relaxed problem is a convex programming problem [41, 42]. Lagrangian relaxation uses dualizing some of the constraints of an ILP problem [43, 44]. Usually, some constraints are relaxed in this method to approximate a difficult problem by a simpler one. It sets an initial value for the dual variables for those constraints and modifies the values through an iterative process.

Heuristics methods are problem-specific solutions in general and have been seen as techniques that are based on logical ideas on how to improve solutions [45]. There are also metaheuristic methods such as [46] and [47], which are based on genetic algorithms, simulated annealing, tabu search, ant colony, particle swarm, etc.

While early works on mobile edge computing assumed that every user can only access its closest edge server, studies in [18, 19, 20] have shown that users can benefit from accessing services on edge servers that are multiple hops away. Allowing the use of non-local edge servers creates the problem of edge workload scheduling, which has been extensively studied in recent years.

Existing works have used various objectives (e.g., minimizing the cost [24] or the makespan [25]), workload models (e.g., fluid model [24], tasks [25], multi-component applications [48]), and edge cloud architectures (e.g., flat versus hierarchical [49]). These works typically assume that each workload requires its own resource for execution, i.e., the resources are dedicated. Here "dedicated" means that each unit of resource can only be used by one workload, e.g., two workloads can share a processor, but each CPU cycle is only used by one workload.

While this assumption usually holds for computation and communication resources (assuming unicasts), it can be too restrictive for storage resources. Note that although [25] allows each service replica to serve multiple jobs, it does not optimize the service placement. For example, in data analytics applications, multiple requests based on the same data can be served by one replica of the data, although processing each request and communicating the input/output still require dedicated CPU and bandwidth. In [50], authors provide an ILP model for virtual machine (VM) placement in fog computing. The use of future user positions is adopted to improve the VM placement and decrease the number of migrations.

Meanwhile, works on content placement in cache networks have considered storage resources that can be shared among requests of the same type. Various solutions have been developed to place contents under cache capacity constraints based on predicted content popularities [51, 52], or request history [53]. Variations of the problem have been studied, e.g., a cache can serve requests from other caches [54], or the content placement and the routing of requests can be jointly optimized [55]. However, the content placement problem only considers the storage resource (i.e., cache space), while the other types of resources (e.g., CPU, bandwidth) are ignored. Note that although [53] was motivated by "hosting services", the problem was actually about caching.

In the spirit of content placement, the goal of [56] is to replicate services (and also delete replicas) so that latency is minimized. However, there is

no thorough complexity analysis, and also the authors do not consider request scheduling. In [57], a stochastic optimization problem is formulated to minimize the long-term average latency given a long-term cost budget. The authors analyze the impact of frequent migrations in the mobile edge cloud as a performance-cost tradeoff.

Only a few works have considered multiple types of resources (e.g., storage, computation, communication). In [58, 59], MILPs were formulated for placing contents or service functions and activating storage, computation, and communication resources in a distributed cloud network. However, no formal complexity analysis or algorithm with performance guarantee was provided.

In [60], a dynamic service placement and workload scheduling framework was proposed to jointly allocate storage and computation resources, but there is no hard constraint on computation resources and no consideration of bandwidth constraints. In [61], an algorithm with a performance guarantee was developed for placing virtual network functions (VNFs) in distributed cloud networks and routing service flows among the placed VNFs under chaining constraints. However, each unit of resource (CPU, memory, bandwidth) is dedicated to a flow (i.e., not amortized), and there is no "storage capacity" constraint on the VNF placement. In [62], an optimal algorithm was developed for joint resource placement and assignment in distributed networks, where a "resource" means a service, and a "type of resources" means a type of services. The solution assumed that each placed service can only serve one request (i.e., dedicated).

In [63], the authors consider the problem of resource provisioning and replica placement in cloud CDNs, where the objective is to minimize the cost. In contrast, our goal is to maximize the expected number of served requests per time slot. Furthermore, all the requests in [63] are of equal size, and have an identical computation requirement, as opposed to our setup where these parameters are different for different requests.

This is critically different from our problem in Chapter 3, which considers both sharable and non-sharable resources, leading to very different conclusions: the problem in [62] is polynomial-time solvable, but our problem is NP-hard.

The work closest to our problems is [27], which considers joint service placement and request scheduling under hard constraints on both dedicated resources (communication, computation) and amortized resources (storage). However, it assumed full knowledge of the requests, which means that to ap-

ply its solution in an online setting, one must batch requests and make placement/scheduling decisions simultaneously. To reduce cost and improve stability, we separate the time scales of service placement and request scheduling and impose a budget constraint on the cost of each service placement. These changes induce critical changes in the underlying optimization problem.

In this work, we also focus on the interaction between the power grid and the control network during cascading failures. Any degradation that limits the ability of the control network to either monitor or control elements in the power grid will increase the risk of a larger cascade of failures. Ideally, the control network should be deployed independently of the power grid. However, this is an expensive solution. Therefore, it is of high importance to optimally allocate the budget for deploying communication links in the smart grid network.

In general, interdependence tends to happen because of cross-system functional dependencies and geographical topology similarities [64]. These two factors imply the possible influences each network can have on the other in the state of cascading failures. Study [65] notes that blackouts in the US cost billions of dollars, and the losses due to blackouts increase dramatically with blackout duration. This example shows how inter-connectivity can significantly increase the scope and intensify the damage in an interdependent system during a large-scale cascade.

It is important to be able to accurately model cascading failures in the power grid to develop and evaluate solutions for preventing them. There are three well-accepted cascading failure models of power systems: DC-quasi-steady-state (QSS) [66, 67, 68, 69, 70], AC-QSS [71, 72, 73, 74, 75, 76, 77, 78, 79], and dynamic [80, 81, 82]. AC-QSS models are based on AC power flow, which can capture voltage collapse in addition to line overloading. Usually, these models suffer from divergence issues due to voltage collapse [75]. The dynamic models [81] present the most precise mechanism of cascade propagation. However, they are computationally costly for large-scale networks.

DC-QSS models work based on DC power flow and are computationally economical and easily implementable [83]. These advantages offer the chance to build interdependent models of power and communication control networks and apply statistical analysis on a large-scale system. These models assume a uniform voltage profile and neglect the resistive loss; therefore, they cannot consider reactive power in the power system.

Although there is a significant body of literature in the area of cascading

failures in power grid, very few papers [70, 83] focus on the coupled cascading failure of power grid and the associated communication network. The authors in [70], [64] and [83] use the DC-QSS model for the power grid. Reference [70] compares a topological contagion model to a power grid model and a percolation model of internetwork cascading to three models of interdependent power-communication systems. The authors also propose a model of a smart power grid coupled to a communication network and show that increased power-communication coupling decreases vulnerability, in contrast to the percolation model. The overall results suggest that robustness can be enhanced by interconnecting networks with complementary capabilities if modes of internetwork failure propagation are constrained. Authors in [84] came to the same conclusion.

Reference [83] considers the real-world scenario, where the location of failures in a coupled power-communication network might be unknown or only partially known. A model is considered where functionality of the power grid and its failure assessment relies on the operation of a monitoring system and vice-versa. The paper addresses ongoing cascading failures with a twofold approach – cascade prevention by re-dispatching generation and shedding loads, and formulation of a recovery plan to maximize the total amount of load served during the recovery intervention.

Chapter 3

Service Placement and Request Scheduling for Data-intensive Applications in Edge Clouds

Mobile edge computing provides the opportunity for wireless users to exploit the power of cloud computing without a large communication delay. To serve data-intensive applications (e.g., video analytics, machine learning tasks) from the edge, we need, in addition to computation resources, storage resources for storing server code and data as well as network bandwidth for receiving user-provided data. Moreover, due to time-varying demands, the code and data placement needs to be adjusted over time, which raises concerns of system stability and operation cost.

In this work [36, 85], we address these issues by proposing a two-time-scale framework that jointly optimizes service (code and data) placement and request scheduling, while considering storage, communication, computation, and budget constraints. First, by analyzing the hardness of various cases, we completely characterize the complexity of our problem. Next, we develop a polynomial-time service placement algorithm by formulating our problem as a set function optimization, which attains a constant-factor approximation under certain conditions. Furthermore, we develop a polynomial-time request scheduling algorithm by computing the maximum flow in a carefully constructed auxiliary graph, which satisfies hard resource constraints and is provably optimal in the special case where requests have homogeneous resource demands. Extensive synthetic and trace-driven simulations show that the proposed algorithms achieve 90% of the optimal performance.

3.1 Introduction

The emerging technology of *mobile edge computing* [4] enables wireless users to run resource-intensive and delay-sensitive applications from the edge of mobile networks. As users access these resource-hungry applications via bandwidth-limited wireless links, how to optimally allocate the limited resources at edge clouds to competing demands (e.g., to minimize the response time or the operation cost) poses a difficult but intriguing research question, which has attracted tremendous interest in the research community.

The remainder of this chapter is organized as follows. Section 3.3 formulates our problem within a single frame, for which Section 3.4 analyzes the complexity, and Section 3.5 presents our algorithms and their performance analysis. Section 3.6 extends our solution to multiple frames. Section 4.4 evaluates the proposed solution against benchmarks. Finally, Section 3.8 concludes the chapter.

3.2 Background & Contribution

Intuitively, one should strive to serve every user request from the nearest edge cloud. While this intuition has been supported by empirical studies [13], maintaining service locality for mobile users poses a significant challenge, including how to migrate services [17] and when/where to migrate services [14, 15, 5, 16], in order to attain a desirable tradeoff between the quality of service and the migration cost. When some of the edge clouds are heavily loaded, it has been shown that users can benefit from getting served by non-nearest edge clouds in the same metropolitan area network [18, 19, 20]. Meanwhile, there have been standardization initiatives [21, 22, 23] to create an open edge computing environment, such that edge clouds within the same geographical region form a shared resource pool, which can then be distributed among contending user requests. The existence of a shared resource pool creates the need for *request scheduling*, i.e., on which edge server, if any, to schedule each user request such that a given objective can be optimized [24, 25]. Existing works typically assume that serving each request needs a *dedicated* share of resources including CPU cycles, memory space, and network bandwidth, and that the total resource consumption at a server is the summation of resource demands scheduled to it.

While the above assumption holds for applications that do not need no-

table amounts of data on the server, it fails to capture the requirements of *data-intensive applications*. In such applications (e.g., video analytics, distributed machine learning), serving a request needs both a dedicated amount of resources and a significant amount of data at the server (e.g., object database, trained machine learning models). The storage resource for storing such data fundamentally differs from the other types of resources in that it is *amortized* over all requests against the same copy of data. Note that many data-intensive applications also require a nontrivial amount of user-provided data (e.g., images captured by the user), although the resources for collecting/storing such data are typically dedicated to each request. Hence, in addition to the conventional resources of CPU cycles and memory, resource allocation algorithms for data-intensive applications should also consider the storage resources for storing server data and the network bandwidth for receiving user-provided data.

Jointly allocating dedicated and amortized resources induces a decomposition of the resource allocation problem into two subproblems [27]: (i) *service placement*, which decides how to replicate and place each service (including server code and data) within the storage capacity of each edge cloud, and (ii) *request scheduling*, which decides whether/where to schedule each request within the communication and the computation capacities of edge clouds, as well as other constraints (e.g., maximum delay). The two subproblems are coupled by the fact that the edge cloud scheduled to process a request must have a replica of the requested service. The existing solution [27] makes both decisions at the same time, and thus may adjust service placement as frequently as the scheduling of requests, risking a high operation cost and even system instability.

In this work, we jointly consider service placement and request scheduling for data-intensive applications. In contrast to [27], we separate the time scales of the two decisions: to prevent system instability, service placement happens at a larger scale (*frames*); to limit scheduling delay, request scheduling happens at a smaller scale (*slots*). Frame length is tuned based on dynamics of user mobility and user request patterns, while slot length is tuned based on the desired scheduling delay and expected execution time of jobs. Furthermore, to control the operation cost due to service replication/migration, we impose a budget constraint for service placement. These changes enable a controllable trade-off between the cost of reconfiguration and the performance of serving requests, while inducing critical changes in the underlying optimization problem. For the request scheduling subproblem, we separately

consider a version with *soft constraints* where the resource capacities are only enforced on the average, and a version with *hard constraints* where the resource capacities are strictly enforced.

The main contributions of this chapter are as follows:

- We propose a two-time-scale framework for joint service placement and request scheduling, and formulate the underlying optimization as a mixed integer linear program (MILP) that jointly considers dedicated and amortized resources.

- By examining the complexity of our problem in carefully selected special cases, we not only prove that both the service placement subproblem and the request scheduling subproblem (under hard constraints) are generally NP-hard, but also determine all the cases that are polynomial-time solvable and identify the root cause of the hardness.

- By reformulating the service placement subproblem as a set function optimization, we develop a greedy service placement algorithm based on shadow request scheduling computed by a linear program (LP). By proving that our objective function is monotone sub-modular under certain conditions and our constraints form a p-independence system, we derive a constant-factor approximation guarantee for the proposed algorithm.

- We show that in the special case where all the requests demand the same amount of communication/computation resources, the request scheduling subproblem under hard constraints can be converted to a maximum flow problem in a carefully constructed auxiliary graph, based on which we develop a polynomial-time algorithm that is provably optimal.

- We show that both our formulation and our service placement algorithm can be extended to exploit request prediction over multiple frames.

- We perform extensive performance evaluations via synthetic and trace-driven simulations. The evaluations show that: (i) the key performance differentiator is the service placement algorithm (i.e., a simplistic algorithm suffices for request scheduling), (ii) the proposed service placement algorithm consistently outperforms benchmarks and achieves over 90% of the optimal performance, even when the approximation guarantee does not

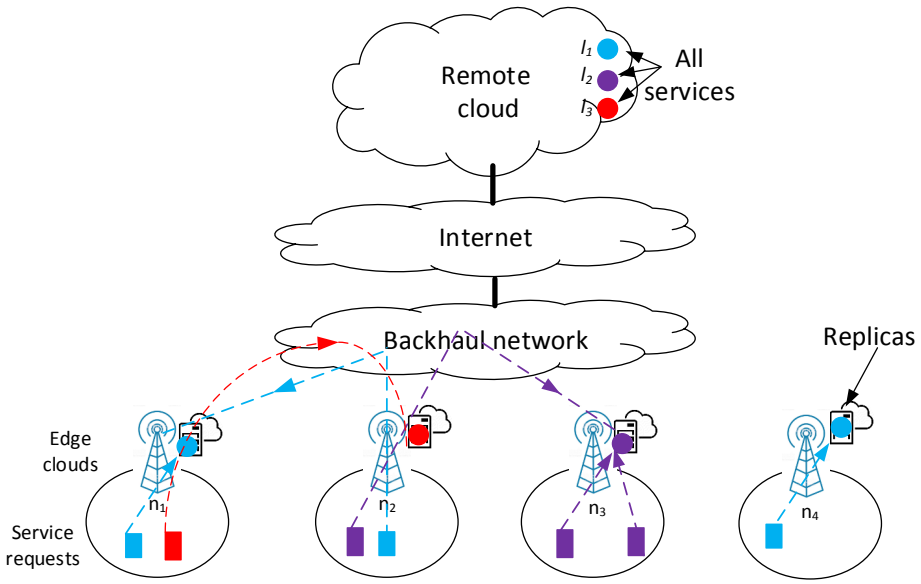


Figure 3.1: System model.

hold, and (iii) the performance can be notably improved by jointly planning service placements for multiple frames, while most of the improvement is already achieved by considering two frames at a time.

3.3 Problem Formulation

3.3.1 System Model

As illustrated in Fig. 3.1, we consider a wireless edge network consisting of a set N of edge clouds, each accessible via a wireless access point or base station covering a specified area. We assume that all the edge clouds are connected by back-haul links that can be used for inter-cloud communications. There is a set L of services, of which a subset can be hosted by each edge cloud at a given point in time, subject to storage capacity constraints.

Services may be migrated/replicated between edge clouds, and/or from a remote cloud to an edge cloud. To access a certain service, a user will first send a request for this service to its local edge cloud, which may then route the request to another edge cloud for processing. Serving a request for service l submitted to edge cloud n at edge cloud m (possibly $m \neq n$) consumes

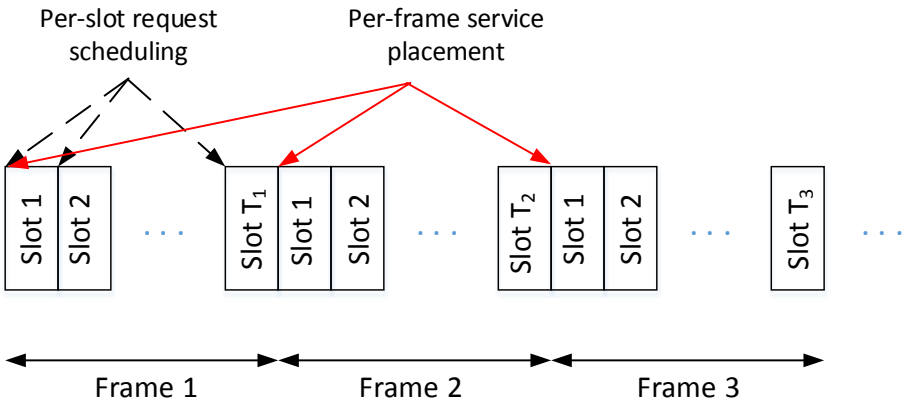


Figure 3.2: Time scales of service placement and request scheduling.

communication resources for transferring input/output between the user and edge cloud n, and computation resources at edge cloud m. Additionally, edge cloud m must have a replica of service l. If $m \neq n$, communication resources are also consumed for transferring input/output between edge cloud n and edge cloud m, but as back-haul links usually have much higher bandwidth than access links, we will focus on the communication resources consumed at the access link in edge cloud n.

To ensure system stability while providing timely services, we adopt a two-time-scale framework as illustrated in Fig. 3.2, where service placement is performed once per frame at the beginning of the frame, and request scheduling is performed once per slot at the beginning of the slot. We discuss how to tune the values of frame and slot length in Section 4.4. Furthermore, we impose a budget B to control the cost of migrating/replicating services in each frame. We refer to a request for service l that is submitted to edge cloud n as a "type-(l,n)" request. The average rate of type-(l,n) requests in frame f is denoted by λ_{ln}^{f} (unit: requests/slot), which is assumed to be predictable based on the request history [86], [87], [88]. The actual number of type-(l,n) requests in slot t is denoted by λ_{ln}^{t}, which is only known at the beginning of slot[1] t. We evaluate the impact of predicting the request rate in Section 4.4.

Each edge cloud has limited communication, computation, and storage capacities. The capacities of different edge clouds may be different. Like-

[1]This is feasible by considering all the requests received during slot $t-1$ as being "submitted" in slot t.

wise the size of each service replica and the communication/computation resources required by each request may be different. There may be other constraints (e.g., latency, security) on whether a given edge cloud m is permitted to serve type-(l, n) requests, and we model that by an indicator a_{lnm} ('0': not permitted; '1': permitted). The main notations used in this work are described in Table 5.4.

Table 3.1: Table of notations

Notation	**meaning**
N	set of edge clouds
$N_+ = N \cup \{n_0\}$	set of edge clouds plus the remote cloud n_0
L	set of all possible services
R_n	storage capacity of edge cloud n
W_n	processing capacity of edge cloud n (per slot)
K_n	communication capacity of edge cloud n (per slot)
r_l	size per replica of service l
κ_l	size of input/output data per request for service l
ω_l	computation requirement per request for service l
$a_{lnm} \in \{0, 1\}$	indicates whether edge cloud m is permitted to serve type-(l, n) requests
$\lambda^t_{ln}, \lambda^f_{ln}$	actual number of type-(l, n) requests in slot t and average number of type-(l, n) requests per slot in frame f
$c_{ln'n}$	cost of replicating or migrating service l from cloud n' to edge cloud n, where cloud n' can be either a remote cloud or an edge cloud
B	maximum cost for service placement in one frame
$x^f_{ln} \in \{0, 1\}$	placement variable for frame f, 1 if service l is placed on edge cloud n and 0 otherwise
$y^t_{lnm}, y^f_{lnm} \in [0, 1]$	scheduling variable representing the probability that a type-(l, n) request is scheduled to edge cloud m in slot t (under soft resource constraints) or frame f
z^t_{lnm}	scheduling variable representing the number of type-(l, n) requests that are scheduled to edge cloud m in slot t (under hard resource constraints)

3.3.2 Underlying Optimization Problems

Although at different time scales (Fig. 3.2), service placement and request scheduling are solving the same optimization problem with different decision variables as explained below. We now formulate the optimization problems for service placement and request scheduling formally. The service placement problem is solved once per frame; once the placement is set, the request scheduling problem is solved once per slot within the frame. Variables related to the frame are denoted with f and those related to a slot are denoted with t.

For the scheduling problem we consider both soft and hard constraints. For soft constraints we use probabilistic scheduling knowing that in some cases requests will not be accommodated within their slot but must be served in a subsequent slot. For hard constraints we guarantee that all scheduled requests are served within the next slot.

3.3.2.1 Service Placement with Shadow Scheduling

To evaluate the service placement cost, we assume that the services always exist on the remote cloud n_0, i.e., $x^f_{ln_0} \equiv 1$, and deleting a service replica from an edge cloud incurs no cost. Furthermore, we always replicate a service from the nearest location hosting the service. That is, the cost of placing service l at edge cloud n in frame f is $c^f_{ln} = \min_{n' \in N_+ : x^{f-1}_{ln'}=1} c_{ln'n}$, where $c_{lnn} \equiv 0$.

The optimization problem for service placement can be formulated as (3.1): Objective (3.1a) maximizes the expected number of requests served per slot. Constraint (3.1b) guarantees that the scheduling variables are valid. Constraint (3.1c) ensures that each edge cloud n does not store more than its storage capacity R_n. Constraint (3.1d) guarantees that the total communication demand on an edge cloud n stays within its communication capacity K_n on the average. Constraint (3.1e) ensures that the total computation demand scheduled to an edge cloud m is within its computation capacity W_m on the average. Constraint (3.1f) states that an edge cloud can only serve a request if it contains the requested service and is a candidate server. Constraint (3.1g) ensures that the total service placement cost is within the budget. Constraint (3.1h) specifies valid ranges of the decision variables.

$$\max \sum_{l\in L}\sum_{n\in N}\lambda_{ln}\sum_{m\in N} y_{lnm} \tag{3.1a}$$

$$\text{s.t.} \sum_{m\in N} y_{lnm} \leq 1, \qquad \forall l \in L, n \in N, \tag{3.1b}$$

$$\sum_{l\in L} x_{lm} r_l \leq R_m, \qquad \forall m \in N, \tag{3.1c}$$

$$\sum_{l\in L} \lambda_{ln}\kappa_l \sum_{m\in N} y_{lnm} \leq K_n, \qquad \forall n \in N, \tag{3.1d}$$

$$\sum_{l\in L} \omega_l \sum_{n\in N} \lambda_{ln} y_{lnm} \leq W_m, \qquad \forall m \in N, \tag{3.1e}$$

$$y_{lnm} \leq a_{lnm} x_{lm}, \qquad \forall l \in L, n \in N, m \in N, \tag{3.1f}$$

$$\sum_{l\in L}\sum_{n\in N} x_{ln} c_{ln} \leq B, \tag{3.1g}$$

$$x_{ln} \in \{0,1\}, y_{lnm} \geq 0, \qquad \forall l \in L, n \in N, m \in N. \tag{3.1h}$$

At the beginning of each frame f, we solve (3.1) with the predicted request rates[2] $\lambda_{ln} = \lambda^f_{ln}$ and the placement costs $c_{ln} = c^f_{ln}$ for the service placement x^f_{ln} and the corresponding request scheduling y^f_{lnm}. Then only x^f_{ln} will be used (to place services). Although the scheduling variable y^f_{lnm} will not be used for actual scheduling, it is needed to evaluate the served request rate (3.1a) under a given service placement. For this reason, we refer to y^f_{lnm} as the *shadow scheduling variable*.

3.3.2.2 Request Scheduling under Soft Resource Constraints

Depending on whether requests submitted in a slot can be postponed till a later slot, the optimization problem for request scheduling differs slightly. If the requests can be postponed, then the average resource constraints (3.1d,3.1e) suffice, as temporary bursts in demands can be absorbed over time. In this case, at the beginning of each slot t within frame f, we solve (3.1) with the current demands[3] $\lambda_{ln} = \lambda^t_{ln}$ and the previously determined

[2]Recall that the superscript f indicates parameters/variables corresponding to frame f.

[3]Recall that the superscript t indicates parameters/variables corresponding to slot t.

service placement $x_{lm} = x^f_{lm}$ for the scheduling variable y^t_{lnm}, which is then used to schedule requests probabilistically in this slot.

3.3.2.3 Request Scheduling under Hard Resource Constraints

If the requests cannot be postponed, e.g., for services with hard deadlines, then we must impose hard resource constraints such that all the requests scheduled to the edge clouds in slot t can be finished within the same slot (the unscheduled requests will be routed to the remote cloud for processing). The corresponding optimization problem can be formulated as (3.2) (N: natural numbers):

$$\max \sum_{l\in L}\sum_{n\in N}\sum_{m\in N} z_{lnm} \tag{3.2a}$$

$$\text{s.t.} \sum_{m\in N} z_{lnm} \le \lambda_{ln}, \qquad \forall l \in L, n \in N, \tag{3.2b}$$

$$\sum_{l\in L} \kappa_l \sum_{m\in N} z_{lnm} \le K_n, \qquad \forall n \in N, \tag{3.2c}$$

$$\sum_{l\in L} \omega_l \sum_{n\in N} z_{lnm} \le W_m, \qquad \forall m \in N, \tag{3.2d}$$

$$z_{lnm} \le a_{lnm} x_{lm} \lambda_{ln}, \qquad \forall l \in L, n \in N, m \in N, \tag{3.2e}$$

$$z_{lnm} \in N, \qquad \forall l \in L, n \in N, m \in N. \tag{3.2f}$$

Optimization (3.2) is similar to (3.1) under a fixed feasible service placement x_{lm}, after replacing $\lambda_{ln} y_{lnm}$ by a new variable z_{lnm}. The only difference is that we now impose an integer constraint (3.2f), which means that instead of only specifying the expected number of type-(l, n) requests to schedule to edge cloud m (i.e., $\lambda_{ln} y_{lnm}$), we specify the exact number (i.e., z_{lnm}). At the beginning of each slot t, we solve (3.2) with the demand $\lambda_{ln} = \lambda^t_{ln}$ and the service placement $x_{lm} = x^f_{lm}$ (f: the current frame) for the scheduling variable z^t_{lnm}, which is used to schedule requests deterministically in this slot. The deterministic scheduling ensures that instead of satisfying the communication/computation capacities on the average as in (3.1d,3.1e), we now satisfy them strictly, which ensures that all the scheduled requests can finish within the slot. Note also that since we solve optimization problem (3.2) under a given feasible service placement $x_{l,m}$, constraints (3.1c) and (3.1g) are lifted as the service placement already satisfies those two constraints.

Discussion: At each decision epoch of service placement, we only know the average request rates over the next frame, and thus cannot impose the hard resource constraints. Therefore, soft constraints are assumed for the shadow scheduling problem to evaluate the objective (3.1a) under a given service placement. While our optimization formulation shares similarities with [27], there are several critical changes. First, while [27] assumes full knowledge of the requests, we only assume knowledge of the expected request rates. Accordingly, our objective becomes the expected rate of served requests, and our scheduling decision becomes probabilistic. Moreover, while [27] allows the service placement to change completely every time, we limit it to incremental adjustments by imposing a budget constraint. Probabilistic scheduling relaxes the integer constraints on scheduling variables, thus invalidating previous hardness results. Meanwhile, the added constraint introduces a potential cause of hardness (verified in Theorem 3.4.1).

3.4 Complexity Analysis

The service placement problem (3.1) is a *mixed integer linear program (MILP)*, and the request scheduling problem is a *linear program (LP)* under soft constraints and an *integer linear program (ILP)* under hard constraints. While LP can always be solved in polynomial time [89], MILP and ILP can both be NP-hard [90]. We thus need to understand the complexity of our instances of the MILP/ILP problem.

3.4.1 Complexity of Service Placement

The service placement problem (3.1) is related to, but different from several known problems in the literature, including the knapsack problem, the *data placement problem (DPP)* [91], the *generalized assignment problem (GAP)* [92], the *distributed caching problem (DCP)* [93]. These problems can all be seen as special cases of the *separable assignment problem (SAP)* [93]. SAP considers packing items into bins under general packing constraints that can model both dedicated and non-dedicated resources. For example, if items represent requests and bins represent edge clouds, then SAP can model service placement where requests for the same service can share a service replica, while each consuming a dedicated share of computation resource and bandwidth. However, SAP requires all the resources consumed for serving a

request to be with a single edge cloud. This requirement is satisfied by [94], but not our problem.

We analyze the complexity of (3.1) by considering important special cases. In this optimization problem, there are four types of resource constraints: the R-constraint (3.1c), the K-constraint (3.1d), the W-constraint (3.1e), and the B-constraint (3.1g). In practice, the arrival rate must be lower than the service rate, and therefore the actual application of our solution will need to set K and W to be strictly smaller than the physical communication capacity and the physical computation capacity for each edge cloud. For example, it is well known that for an M/M/1 queue with arrival rate λ and service rate μ, having $\lambda \leq \epsilon^{1/C}\mu$, where $0 < \epsilon < 1$ and $C > 1$, ensures that $P[\text{Queue length} \geq C] \leq \epsilon$. This implies that if we model the downlink and server queue as M/M/1 queues with service rates K' and W', respectively, then setting $K = \epsilon^{1/C}K'$ and $W = \epsilon^{1/C}W'$ guarantees that the probability for queue length to be at least C is no more than ϵ.

3.4.1.1 Having B-constraint Only

Consider the special case where the edge clouds and the services are homogeneous (although having B-constraint only gives the same formulation for homogeneous and heterogeneous scenarios), and R, W and K are large enough that they are unconstrained, i.e., $R \geq |L|$ (i.e., every edge cloud can store all the services), $W \geq \sum_{n\in N}\sum_{l\in L}\lambda_{ln}$ and $K \geq \max_{n\in N}\sum_{l\in L}\lambda_{ln}$. Then, the MILP in (3.1) changes to:

$$\max \sum_{l\in L}\sum_{n\in N}\lambda_{ln}\sum_{m\in N} y_{lnm} \tag{3.3a}$$

$$\text{s.t. } (3.1b), (3.1f), (3.1g), \tag{3.3b}$$

$$x_{ln} \in \{0,1\}, y_{lnm} \in [0,1], \qquad \forall l \in L, n \in N, m \in N. \tag{3.3c}$$

Theorem 3.4.1. *The B-constraint alone makes the problem NP-hard.*

Proof. We prove the NP-hardness of (3.3) by a reduction from the 0-1 knapsack problem: given a set of k items, each with value v_i and weight w_i ($i = 1, ..., k$), select a subset $\mathcal{S}'$ such that $\sum_{i\in\mathcal{S}'} v_i$ is maximized while $\sum_{i\in\mathcal{S}'} w_i \leq \Omega$, for a given size Ω of the knapsack. The problem is well-known to be NP-hard [95].

Construction: For each item i, construct a service l_i with total demands $\sum_{n\in N}\lambda_{l_in} = v_i$ and the placement cost $c_{l_in} = w_i, \forall n \in N$. Let $B = \Omega$ and $a_{lmn} \equiv 1$.

Claim: The optimal service placement of (3.3) gives the optimal solution to a knapsack problem.

Proof of the claim: The optimal service placement places at most one replica among all the edge clouds. Therefore, the scheduling decision is to simply schedule all the requests for service l_i to edge cloud n, if $\exists n \in N$ with $x_{l_in} = 1$; or, not schedule any of these requests if $x_{l_in} = 0, \forall n \in N$. Let $\mathcal{S}'$ be the set of indices of all the placed services under the optimal solution to (3.3). Then, the expected number of served requests equals $\sum_{i\in\mathcal{S}'} v_i$, and $\sum_{i\in\mathcal{S}'} w_i \leq B = \Omega$. Selecting all the items corresponding to the services placed by the optimal solution of (3.3) provides the optimal solution to the knapsack problem. □

Remark: Proving NP-hardness for the special case shows that the problem is NP-hard in the general case as well.

3.4.1.2 Having R-constraint Only

Here we consider the special case in which the edge clouds and the services are homogeneous, and W, K and B are large enough to be unconstrained, i.e., $W \geq \sum_{n\in N}\sum_{l\in L}\lambda_{ln}$, $K \geq \max_{n\in N}\sum_{l\in L}\lambda_{ln}$, and $B \geq \sum_{l\in L}\sum_{n\in N} c_{ln}$. In this case, the MILP in (3.1) becomes:

$$\max \sum_{l\in L}\sum_{n\in N}\lambda_{ln}\sum_{m\in N} y_{lnm} \tag{3.4a}$$

$$\text{s.t. } (3.1b), (3.1f), (3.3c), \tag{3.4b}$$

$$\sum_{l\in L} x_{ln} \leq R, \qquad \forall n \in N. \tag{3.4c}$$

Theorem 3.4.2. *The R-constraint alone makes the problem NP-hard.*

Proof. We prove the hardness by showing that the optimization (3.4) can be reduced to the 2-Disjoint Set Cover ($2DSC$) problem, which is proved to be NP-complete [96]. Given a bipartite graph $\mathcal{G} = (\mathcal{A}, \mathcal{B}, \mathcal{E})$, with edges $\mathcal{E}$ between two disjoint vertex sets $\mathcal{A}$ and $\mathcal{B}$, $2DSC$ determines whether there exist two disjoint sets $\mathcal{B}_1, \mathcal{B}_2 \subset \mathcal{B}$, such that $|\mathcal{B}_1| + |\mathcal{B}_2| = |\mathcal{B}|$ and

$\mathcal{A} = \cup_{b\in\mathcal{B}_1}\mathcal{N}(b) = \cup_{b\in\mathcal{B}_2}\mathcal{N}(b)$, where $\mathcal{N}(b)$ $(\forall b \in \mathcal{B})$ is the set of neighbors of node b.

Construction: Denote $\mathcal{A}$ by $\{a_1, ..., a_I\}$ and $\mathcal{B}$ by $\{b_1, ..., b_J\}$. WLOG, assume $I \leq J$. Construct J edge clouds $N = \{n_1, ..., n_J\}$, each with $R = 1$. Construct two services $L = \{l_1, l_2\}$, each with a unit of demand in the first I edge clouds, i.e., $\lambda_{ln_i} = 1, \forall i \in \{i, ..., I\}, l \in \{l_1, l_2\}$. Note that $\lambda_{ln_i} = 0, \forall i > I$. For each $i \in \{1, ..., I\}$ and $j \in \{1, ..., J\}$, we allow edge cloud n_j to serve requests of type (l_1, n_i) and (l_2, n_i), if and only if $(a_i, b_j) \in \mathcal{E}$, i.e., $a_{l_k n_i n_j} = 1,\ k = \{1, 2\}$, if $(a_i, b_j) \in \mathcal{E}$, otherwise $a_{l_k n_i n_j}$ is zero.

Claim: $2DSC$ is feasible if and only if the optimal value of (3.4) for the above instance is $2\mathcal{I}$.

Proof of the claim: If $2DSC$ is feasible, then storing l_1 at edge clouds corresponding to $\mathcal{B}_1$ and l_2 at the remaining edge clouds will serve all the requests. If there is a service placement that serves all the requests, then $\mathcal{B}_1 = \{b_i \in \mathcal{B} : n_i \text{ stores } l_1\}$, and $\mathcal{B}_2 = \mathcal{B}\backslash\mathcal{B}_2$ is a feasible solution to $2DSC$. Figure 3.3 illustrates the proof. In this figure, solution to $2DSC$ is $\mathcal{B}_1 := \{b_1, b_2\}$, and $\mathcal{B}_2 := \{b_3, b_4\}$ while solution to (3.4) is $x_{l_1 n_1} = x_{l_1 n_2} = 1$, $x_{l_2 n_3} = x_{l_2 n_4} = 1$, and $x_{ln} = 0$ otherwise. □

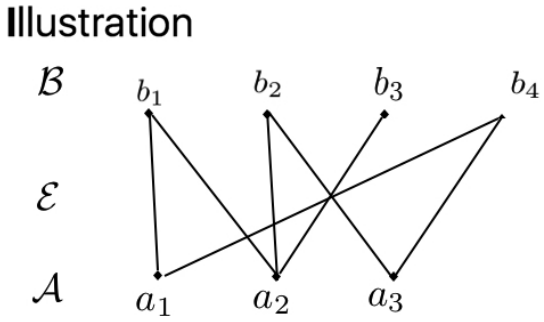


Figure 3.3: Illustration of 2DSC.

3.4.1.3 Removing R- and B-constraints

Lemma 3.4.1. *Removing R- and B-constraints makes the problem polynomial-time solvable.*

Proof. If R_n $(\forall n \in N)$ and B are both large enough, i.e., $\min_{n\in N} R_n \geq |L|$ (every edge cloud can store all the services) and $B \geq \sum_{l\in L}\sum_{n\in N} c_{ln}$, the

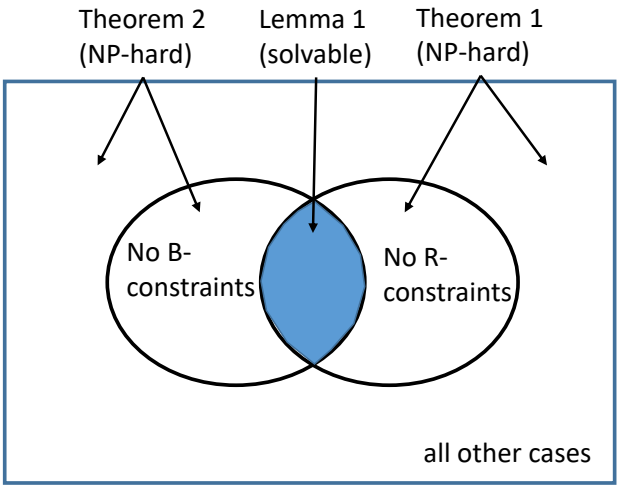


Figure 3.4: Complexity of service placement (3.1).

optimal solution to x_{ln} is trivially $x_{ln} \equiv 1$ ($\forall l \in L$ and $n \in N$). Under this service placement, constraints (3.1c,3.1g) in (3.1) disappear, and constraint (3.1f) changes to $y_{lnm} \leq a_{lnm}$ ($\forall l \in L, n \in N, m \in N$). Removing the constraints (3.1c,3.1g) reduces the original problem (3.1) into a linear program (LP), which is polynomial-time solvable [89]. □

3.4.1.4 Summary of All Cases

Together, Theorems 3.4.1, 3.4.2 and Lemma 3.4.1 cover all the cases. By Theorem 3.4.1, the solvable instances must be cases without the B-constraint. By Theorem 3.4.2, the solvable instances must also be cases without the R-constraint. On the other hand, Lemma 3.4.1 shows that all the cases without either of B- or R-constraint are polynomial-time solvable. Therefore, the colored region in Fig. 3.4 captures all the solvable cases of (3.1).

3.4.2 Complexity of Request Scheduling

Under soft resource constraints, the request scheduling problem ((3.1) with given x_{lm}) is an LP and hence polynomial-time solvable. Under hard resource constraints, however, the problem has a different complexity, as analyzed below.

3.4.2.1 General Case

Theorem 3.4.3. *Under hard resource constraints, the request scheduling problem (3.2) is generally NP-hard.*

Proof. We reduce the *partition problem* to our problem (3.2). Given a set of positive integers $A = \{t_1, \ldots, t_m\}$, the partition problem is the task of deciding whether A can be partitioned into two subsets A_1 and A_2, such that $\sum_{t_i \in A_1} t_i = \sum_{t_j \in A_2} t_j$. This problem is known to be NP-complete.

We construct an equivalent instance of the request scheduling problem as follows. We construct two edge clouds n_1 and n_2, each having unlimited communication capacity, unlimited storage capacity, and a computation capacity of $W = \frac{1}{2} \sum_{t_i \in A} t_i$. For each $t_i \in A$, we construct a request for a service l_i with computation requirement $\omega_{l_i} = t_i$, submitted to edge cloud n_1. Suppose that each edge cloud hosts all the services $l_1, \ldots, l_m$, and is allowed to serve any request. For this instance, (3.2) reduces to ($[m]\{1, \ldots, m\}$):

$$\max \sum_{i=1}^{m} \sum_{j=1}^{2} z_{l_i n_1 n_j} \tag{3.5a}$$

$$\text{s.t.} \sum_{j=1}^{2} z_{l_i n_1 n_j} \leq 1, \qquad \forall i \in [m], \tag{3.5b}$$

$$\sum_{i=1}^{m} t_i z_{l_i n_1 n_j} \leq W, \qquad \forall j \in [2], \tag{3.5c}$$

$$z_{l_i n_1 n_j} \in \{0, 1\}, \qquad \forall i \in [m], j \in [2]. \tag{3.5d}$$

Since $W = \frac{1}{2} \sum_{t_i \in A} t_i$, if A cannot be partitioned into two subsets of equal sum, then the sum for one of the subsets must be greater than W. This implies that we cannot serve every request while satisfying constraint (3.5c), and hence the optimal value of (3.5a) must be smaller than m. If A can be partitioned into subsets A_1 and A_2 of equal sum, then we must have $\sum_{t_i \in A_1} t_i = \sum_{t_i \in A_2} t_i = W$. Then setting $z_{l_i n_1 n_j} = 1$ if and only if $t_i \in A_j$ ($j = 1, 2$) gives a feasible solution to (3.5) with an objective value of m. Therefore, the partition problem has a solution if and only if all the requests can be served in the above instance of the request scheduling problem. □

Remark: The proof of Theorem 3.4.3 holds even if we require the edge clouds to be homogeneous, i.e., $K_n \equiv K$ and $W_n \equiv W$ ($\forall n \in N$). Thus,

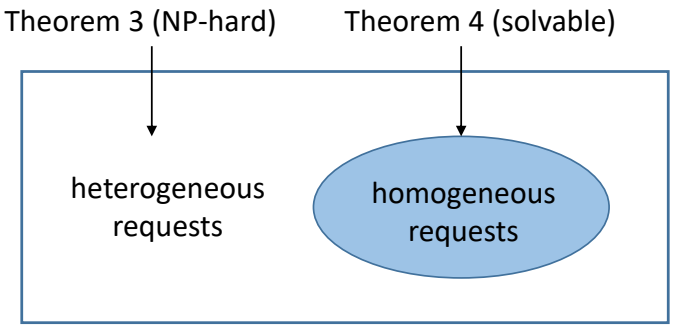


Figure 3.5: Complexity of request scheduling under hard constraints (3.2).

the request scheduling problem under hard resource constraints is NP-hard as long as the requests have heterogeneous resource demands.

3.4.2.2 Homogeneous Special Case

But what if the requests are homogeneous (i.e., $\kappa_l \equiv \kappa$, $\omega_l \equiv \omega$)? We show that the problem is no longer NP-hard in this case.

Theorem 3.4.4. *In the special case when all the requests have identical communication and computation demands, the scheduling problem (3.2) is polynomial-time solvable.*

We prove this theorem by developing a polynomial-time optimal solution in Section 3.5.3.1.

3.4.2.3 Summary of All Cases

Together, Theorems 3.4.3 and 3.4.4 characterize the complexity of the request scheduling problem under hard constraints in all cases, as illustrated in Fig. 3.5.

3.5 Algorithms

We now develop efficient algorithms for the service placement problem and the request scheduling problem separately.

3.5.1 Approximation Algorithm for Service Placement

Due to the NP-hardness of finding the optimal service placement in general (Section 3.4.1), we seek efficient service placement algorithms with approximation guarantees.

3.5.1.1 Conversion to Set Function Optimization

We start by reformulating our problem as a set function optimization problem. Let $S \subseteq L \times N$ denote the set of selected single-service placements, where $(l, n) \in S$ means to place a replica of service l at edge cloud n. Let $\Omega(S)$ denote the optimal objective value of (3.1) for a fixed $\mathbf{x}$ given by $x_{ln} = 1$ if and only if $(l, n) \in S$. This can be calculated by solving the following (shadow) request scheduling problem, where $\mathbb{1}_{l,m}$ is the indicator function:

$$\max \sum_{l \in L} \sum_{n \in N} \lambda_{ln} \sum_{m \in N} y_{lnm} \tag{3.6a}$$

$$\text{s.t. } (3.1b), (3.1d), (3.1e), \tag{3.6b}$$

$$y_{lnm} \leq a_{lnm} \mathbb{1}_{(l,m) \in S}, \qquad \forall l \in L, n \in N, m \in N, \tag{3.6c}$$

$$y_{lnm} \in [0, 1], \qquad \forall l \in L, n \in N, m \in N. \tag{3.6d}$$

After that, we can rewrite the service placement problem as:

$$\max \Omega(S) \tag{3.7a}$$

$$\text{s.t.} \sum_{l:(l,n) \in S} r_l \leq R_n, \qquad \forall n \in N, \tag{3.7b}$$

$$\sum_{(l,n) \in S} c_{ln} \leq B, \tag{3.7c}$$

$$S \subseteq L \times N, \tag{3.7d}$$

where $S_n L \times \{n\}$ is the set of all possible single-service placements at edge cloud n.

First, we prove that, under certain conditions, the objective function of (3.7) has a desirable property.

Definition 3.5.1 ([97]). *A set function $f : 2^{\mathbf{x}} \to \mathcal{R}$ is monotone increasing if $\forall\ S_1 \subseteq S_2 \subseteq \mathbf{x}$, $f(S_1) \leq f(S_2)$. Moreover, the function $f(.)$ is sub-modular if $\forall S_1 \subseteq S_2 \subseteq \mathbf{x}$ and $e \in \mathbf{x} \backslash S_2$, $f(\{e\} \cup S_1) - f(S_1) \geq f(\{e\} \cup S_2) - f(S_2)$.*

Lemma 3.5.1. *The objective function $\Omega(S)$ in (3.7a) is a monotone sub-modular function for all feasible S if $\kappa_l \equiv \kappa$ $(\forall l \in L)$, and*

1. *$\lfloor R_n r_l \rfloor \leq 1$ for all $n \in N$ and $l \in L$, or*
2. *$W_m \geq \sum_{l \in L} \omega_l \sum_{n \in N} \lambda_{ln}$ for all $m \in N$.*

Proof. It is easy to see that $\Omega(S)$ is monotone, as expanding S will relax the constraint (3.6c), hence enlarge the solution space for (3.6) and increase its optimal objective value.

To show that $\Omega(S)$ is sub-modular, we need to show that for any sets $S_1, S_2 \subseteq L \times N$ and any $(l_1, n_1) \in (L \times N) \setminus S_2$, such that $S_1 \subseteq S_2$ and $S_2 \cup \{(l_1, n_1)\}$ is feasible, the following relationship holds

$$\Omega(S_1 \cup \{(l_1, n_1)\}) - \Omega(S_1) \geq \Omega(S_2 \cup \{(l_1, n_1)\}) - \Omega(S_2). \tag{3.8}$$

Suppose that $\mathbf{y}^{(0)}$ and $\mathbf{y}^{(2)}$ are the optimal scheduling solutions according to (3.6) under service placements S_1 and S_2, respectively. Moreover, suppose that $\mathbf{y}^{(1)}$ and $\mathbf{y}^{(3)}$ are the optimal scheduling solutions under service placements $S_1 \cup \{(l_1, n_1)\}$ and $S_2 \cup \{(l_1, n_1)\}$, respectively, that minimize the request rate scheduled to the replica (l_1, n_1), i.e., minimizing $\sum_{n \in N} \lambda_{l_1 n} y_{l_1 n n_1}$. We can then decompose the objective function as:

$$\Omega(S_1) = \sum_{(l,m) \in S_1} \sum_{n \in N} \lambda_{ln} y^{(0)}_{lnm}, \tag{3.9}$$

$$\Omega(S_1 \cup \{(l_1, n_1)\}) = \sum_{(l,m) \in S_1} \sum_{n \in N} \lambda_{ln} y^{(1)}_{lnm} + \sum_{n \in N} \lambda_{l_1 n} y^{(1)}_{l_1 n n_1}, \tag{3.10}$$

$$\Omega(S_2) = \sum_{(l,m) \in S_2} \sum_{n \in N} \lambda_{ln} y^{(2)}_{lnm}, \tag{3.11}$$

$$\Omega(S_2 \cup \{(l_1, n_1)\}) = \sum_{(l,m) \in S_2} \sum_{n \in N} \lambda_{ln} y^{(3)}_{lnm} + \sum_{n \in N} \lambda_{l_1 n} y^{(3)}_{l_1 n n_1}. \tag{3.12}$$

Due to this decomposition, we have

$$\text{LHS of (3.8)} = \sum_{(l,m) \in S_1} \sum_{n \in N} \lambda_{ln} (y^{(1)}_{lnm} - y^{(0)}_{lnm}) + \sum_{n \in N} \lambda_{l_1 n} y^{(1)}_{l_1 n n_1}, \tag{3.13}$$

$$\text{RHS of (3.8)} = \sum_{(l,m) \in S_2} \sum_{n \in N} \lambda_{ln} (y^{(3)}_{lnm} - y^{(2)}_{lnm}) + \sum_{n \in N} \lambda_{l_1 n} y^{(3)}_{l_1 n n_1}. \tag{3.14}$$

The first term in (3.13) is the difference in the request rate served by replicas in S_1 after/before placing the replica (l_1, n_1). Under condition (1) or (2) in the lemma, there is no contention of computation resources between replicas, and hence replicas in S_1 can still process requests scheduled to them under $\mathbf{y}^{(0)}$. Meanwhile, as the communication demands κ_l are the same for all types of requests, dropping requests originally scheduled to S_1 to admit requests to be scheduled to (l_1, n_1) will not improve the objective value of (3.6). Thus, the first term in (3.13) is zero. Similarly, the first term in (3.14) is also zero. The second term in (3.13,3.14) is the minimum request rate served by the replica (l_1, n_1) under an optimal scheduling, in the presence of replicas S_1 and S_2, respectively. Again, as there is no computation resource contention between replicas, requests that used to be served by replicas in S_1 under service placement $S_1 \cup \{(l_1, n_1)\}$ can still be served there after adding replicas in $S_2 \setminus S_1$, but these added replicas may offload some requests that used to be served by the replica (l_1, n_1). Therefore, $\sum_{n\in N} \lambda_{l_1 n} y^{(1)}_{l_1 n n_1} \geq \sum_{n\in N} \lambda_{l_1 n} y^{(3)}_{l_1 n n_1}$. This proves (3.8) and hence the sub-modularity of $\Omega(S)$. □

The constraints of (3.7) also have a desirable property.

Definition 3.5.2 ([98])**.** *Let X be a universe of elements. Consider a collection $\mathcal{I} \subseteq 2^X$ of subsets of X. $(X, \mathcal{I})$ is called an independence system if: (a) $\emptyset \in \mathcal{I}$, and (b) if $Z \in \mathcal{I}$ and $Y \subseteq Z$, then $Y \in \mathcal{I}$ as well. The subsets in $\mathcal{I}$ are called independent; for any set S of elements, an inclusion-wise maximal subset T of S that is in $\mathcal{I}$ is called a basis of S.*

Definition 3.5.3 ([98])**.** *Given an independence system $(X, \mathcal{I})$ and a subset $S \subseteq X$, the rank $r(S)$ is defined as the cardinality of the largest basis of S, and the lower rank $\rho(S)$ is the cardinality of the smallest basis of S. The independence system is called a p-independence system (or a p-system) if $\max_{S\subseteq X} \frac{r(S)}{\rho(S)} \leq p$.*

Lemma 3.5.2. *The constraints (3.7b)-(3.7d) form a p-independence system for $p = \left\lceil \frac{\max\ c_{ln}}{\min_{c_{ln}>0}\ c_{ln}} \right\rceil + \left\lceil \frac{\max\ r_l}{\min_{l:r_l>0}\ r_l} \right\rceil$.*

Proof. By Definition 3.5.1, $(L \times N, \mathcal{I})$, where $\mathcal{I} \subseteq 2^{L\times N}$ is a set of all feasible solutions to (3.7) is an independent system, as $S = \emptyset$ is a feasible service placement, and the subset of any feasible service placement remains feasible. Consider any $S \subseteq L \times N$ and any two maximal feasible service placements $S_1,\ S_2 \subseteq S_1$. To add a pair $(l, n) \in S_2 \setminus S_1$ to S_1, we need to take out a set

Algorithm 1: Greedy Service Placement based on Shadow Scheduling (GSP-SS)

Input: Input parameters of (3.1)
Output: Service placement $\mathbf{x}(x_{lm})_{l\in L, m\in N}$

$S \leftarrow \emptyset$;
while $\exists\ (l,n) \in (L \times N)\ \backslash S$ such that $S \cup \{(l,n)\}$ satisfies (3.7b)-(3.7d) **do**
  $(l^*, n^*) \leftarrow \arg\max_{(l,n):\ S\cup\{(l,n)\} \text{ satisfies (3.7b)-(3.7d)}} \Omega(S \cup \{(l,n)\})$;
  $S \leftarrow S \cup \{(l^*, n^*)\}$;
Convert S to its vector representation $\mathbf{x}$;

S' of pairs form S_1, such that $(S1\backslash S') \cup \{(l,n)\}$ remains a feasible service placement. The set S' contains at most $\lceil \frac{\max\ r_l}{\min_{l:r_l>0}\ r_l} \rceil$ pairs from $\{l\} \times N$ corresponding to removing service replicas from edge cloud n to satisfy (3.7b), and at most $\lceil \frac{\max\ c_{ln}}{\min_{c_{ln}>0}\ c_{ln}} \rceil$ other pairs that correspond to removing service replicas with non-zero placement costs to satisfy (3.7c). Note that in the worst case all the existing service replicas under S_1 at edge cloud n have zero placement cost, and hence we need to remove replicas at other edge clouds to satisfy the budget constraint (3.7c). Repeating this swap to each pair in $S_2\backslash S_1$ shows that we reduce the number of placed service replicas by at most p-fold in modifying S_1 into S_2. Since the above holds for any $S \subset L \times N$ and any maximal independent subsets of S, the constraints (3.7b)-(3.7d) form a p-independent system. □

Combining Lemmas 3.5.1 and 3.5.2 gives the following result.

Theorem 3.5.1. *Under the conditions in Lemma 3.5.1, greedily optimizing (3.7) (Algorithm 1) yields a $1/(1+p)$-approximation for (3.1), where $p = \lceil \frac{\max\ c_{ln}}{\min_{c_{ln}>0}\ c_{ln}} \rceil + \lceil \frac{\max\ r_l}{\min_{l:r_l>0}\ r_l} \rceil$.*

Proof. From [97], for maximizing a monotone sub-modular function subject to a p-system constraint, the greedy algorithm has an approximation ratio of $1/(1+p)$. □

3.5.1.2 Algorithm

Algorithm 1 gives the pseudo code for the greedy algorithm. Note that it differs from the simple greedy heuristic in that each evaluation of $\Omega(\cdot)$ requires solving an instance of the shadow scheduling problem (3.6) using LP.

3.5.1.3 Complexity

There are $O(|N| \times \frac{R_{max}}{r_{min}})$ iterations in Algorithm 1, where $R_{max} = \max_{n \in N} R_n$ and $r_{min} = \min_{l \in L: r_l > 0} r_l$. For each iteration, the algorithm considers $O(|L| \times |N|)$ single service placements, and for each single service placement, we need to evaluate the objective function by solving an $O(|L| \times |N|^2)$-variable $O(|L| \times |N|^2)$-bit input LP, which takes $O(|N|^{11} \times |L|^{5.5})$ time [99]. Therefore, the overall complexity of Algorithm 1 is $O(|N|^{13} \times |L|^{6.5} \times \frac{R_{max}}{r_{min}}) = O(|N|^{13} \times |L|^{6.5})$. We expect this complexity to be acceptable in practice, as this algorithm is only run once per frame, and the frame length will be at the scale of changes in request rates (usually tens of minutes or longer).

3.5.2 Optimal Request Scheduling under Soft Constraints

Given the number of requests of each type observed at the beginning of a slot and the service placement determined at the beginning of the current frame, the request scheduling problem under soft constraints is identical to (3.6), which can be solved by a generic LP solver in $O(|N|^{11} \times |L|^{5.5})$ time (see complexity analysis for Algorithm 1). We then use the resulting y_{lnm} to perform probabilistic scheduling, where a type-(l, n) request will be scheduled to each edge cloud $m \in N$ with probability y_{lnm}. All the scheduling decisions in a frame are based on the same service placement, while the decisions in different slots can differ due to variations in request rates within the frame.

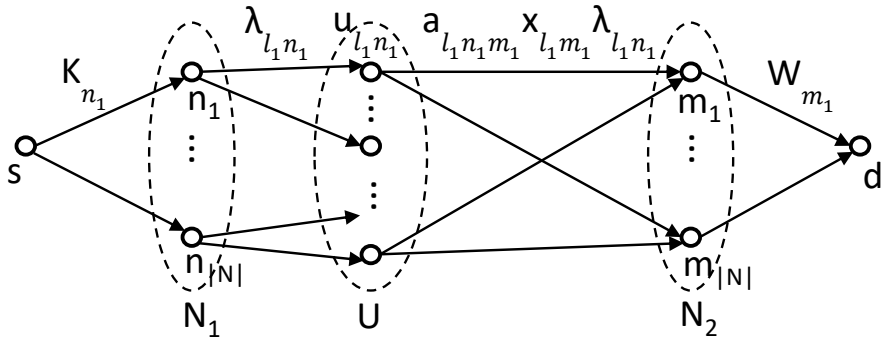


Figure 3.6: Auxiliary graph $\mathcal{G}$ for request scheduling.

3.5.3 Optimal and Heuristic Request Scheduling under Hard Constraints

3.5.3.1 Optimal Algorithm for Homogeneous Requests

Consider the special case where all the requests have identical communication and computation demands. Without loss of generality[4], assume $\kappa_l \equiv 1$, $\omega_l \equiv 1$, and K_n and W_n are integers for all $n \in N$. We will show that in this special case, (3.2) can be converted to a maximum flow problem in an auxiliary graph, and is thus polynomial-time solvable by existing maximum flow algorithms.

Graph construction: Given parameters of (3.2) (including the service placement x_{lm} of the current frame), we construct an auxiliary graph $\mathcal{G}$ as in Fig. 3.6. The nodes in $\mathcal{G}$ consist of a source s, a destination d, a set of nodes U in 1-1 correspondence with the types of requests $\{(l,n)\}_{l\in L, n\in N}$, and two sets of nodes N_1 and N_2, each in 1-1 correspondence with the edge clouds. Node s is connected to each node $n \in N_1$ by a directed link of capacity K_n, and each node $m \in N_2$ is connected to node d by a directed link of capacity W_m. Moreover, each node $n \in N_1$ is connected to each node $u_{ln} \in U$ (representing type-(l,n) requests) by a directed link of capacity λ_{ln}, and each node $u_{ln} \in U$ is connected to each node $m \in N_2$ by a directed link of capacity $a_{lnm}x_{lm}\lambda_{ln}$.

Conversion to a maximum flow problem: We will show that for homogeneous requests, the problem of request scheduling under hard resource

[4]We redefine K_n as the maximum number of requests that an edge cloud can communicate with its covered users in a slot (for input/output), and W_n as the maximum number of requests that an edge cloud can process in a slot.

Algorithm 2: Maximum Flow-based Request Scheduling (MFRS)

input : Input parameters of (3.2), assuming $\kappa_l \equiv 1$, $\omega_l \equiv 1$, and K_n and W_n are integers ($\forall n \in N$)

output: Request scheduling $\mathbf{z} = (z_{lnm})_{l \in L, n, m \in N}$

1 $\mathcal{G} \leftarrow$ auxiliary graph as in Fig. 3.6;

2 compute the maximum integral flow from s to d in $\mathcal{G}$;

3 **foreach** $(l, n, m) \in L \times N \times N$ **do**

4 $\quad z_{lnm} \leftarrow$ flow rate on link (u_{ln}, m) in $\mathcal{G}$;

constraints is equivalent to a maximum flow problem in $\mathcal{G}$.

Theorem 3.5.2. *For homogeneous requests, the optimal value of (3.2) equals the maximum flow between s and d in $\mathcal{G}$, and the optimal solution is to set z_{lnm} to the flow rate on link (u_{ln}, m) under the maximum integral flow from s to d.*

Proof. First, an s-to-d flow satisfies the link capacities in $\mathcal{G}$ if and only if the corresponding $\mathbf{z} = (z_{lnm})_{l \in L, n, m \in N}$ satisfies constraints (3.2b)–(3.2e). This is because if the rate on link (u_{ln}, m) represents the number of type-(l, n) requests that are served by edge cloud m, then by flow conservation, the flow rate on link (s, n) represents the number of served requests that are submitted to edge cloud n, the rate on link (n, u_{ln}) represents the number of served requests of type (l, n), and the rate on link (m, d) represents the total number of requests served by edge cloud m. Thus, by construction, satisfying the capacities of these links is equivalent to satisfying the corresponding constraints in (3.2b)–(3.2e). If we impose a further integral flow constraint, i.e., the flow rate on every link must be an integer, then constraint (3.2f) is also satisfied. Moreover, by the *Integral Flow Theorem* [100], there exists an integral flow between s and d that achieves the maximum flow rate, as the link capacities in $\mathcal{G}$ are all integers. Thus, the optimal objective value of (3.2) equals the maximum integral s-to-d flow, which in turn equals the maximum s-to-d flow. □

Algorithm: By Theorem 3.5.2, we develop a scheduling algorithm called *Maximum Flow-based Request Scheduling (MFRS)*, shown in Algorithm 4. We can leverage existing maximum flow algorithms to implement line 2. In particular, the Ford-Fulkerson algorithm [100] has guaranteed termination and optimality. More importantly, for a graph with integral link capacities,

Algorithm 3: LP Relaxation-based Request Scheduling (LRRS)

input : Input parameters of (3.2)
output: Request scheduling $\mathbf{z} = (z_{lnm})_{l \in L, n, m \in N}$

1 $\mathbf{z}' \leftarrow$ optimal solution to the LP relaxation of (3.2);
2 $\widetilde{\lambda}_{ln} \leftarrow \lambda_{ln}$ for all $l \in L, n \in N$;
3 $\widetilde{K}_n \leftarrow K_n$ for all $n \in N$;
4 $\widetilde{W}_m \leftarrow W_m$ for all $m \in N$;
5 **foreach** $(l, n, m) \in L \times N \times N$ **do**
6 $\quad z_{lnm} \leftarrow \min\Big(\text{round}(z'_{lnm}),\ \min(a_{lnm} x_{lm} \widetilde{\lambda}_{ln}, \lfloor \frac{\widetilde{K}_n}{\kappa_l} \rfloor, \lfloor \frac{\widetilde{W}_m}{\omega_l} \rfloor)\Big)$;
7 $\quad \widetilde{\lambda}_{ln} \leftarrow \widetilde{\lambda}_{ln} - z_{lnm}$;
8 $\quad \widetilde{K}_n \leftarrow \widetilde{K}_n - \kappa_l \cdot z_{lnm}$;
9 $\quad \widetilde{W}_m \leftarrow \widetilde{W}_m - \omega_l \cdot z_{lnm}$;

this algorithm gives an integral solution, i.e., only sending an integral amount of flow per link. The optimality of this algorithm is implied by Theorem 3.5.2.

Corollary 3.5.2.1. *For homogeneous requests, MFRS (Algorithm 4) maximizes the number of requests served by the edge clouds.*

Complexity: It is easy to see that constructing $\mathcal{G}$ (line 1) takes $O(|L| \cdot |N|^2)$ time, and converting the maximum flow solution to a scheduling solution (lines 3–4) also takes $O(|L| \cdot |N|^2)$ time. It is known that for integral link capacities, the Ford-Fulkerson algorithm has complexity $O(|E| \cdot \phi)$, where $|E|$ is the number of links and ϕ is the maximum flow. In our case, $|E| = O(|L||N|^2)$ and $\phi \leq \min(\sum_{n \in N} K_n, \sum_{m \in N} W_m, \sum_{l \in L} \sum_{n \in N} \lambda_{ln})$. Therefore, the overall complexity of Algorithm 4 is $O(|L||N|^2 \min(\sum_{n \in N} K_n, \sum_{m \in N} W_m, \sum_{l \in L} \sum_{n \in N} \lambda_{ln}))$.

Remark: We note that Algorithm 4 extends our previous algorithm *Optimal Request Scheduling* in [27], which requires both the requests and the edge clouds to be homogeneous.

3.5.3.2 Heuristic Algorithm for Heterogeneous Requests

In the general case where requests for different services can have different communication/computation demands, we resort to LP relaxation, i.e., replacing the integer constraint (3.2f) by a linear constraint $z_{lnm} \geq 0$. The

key is how to round the fractional solution to this LP relaxation to a feasible integral solution to (3.2). To this end, we propose *LP Relaxation-based Request Scheduling (LRRS)*, shown in Algorithm 9. LRRS sequentially rounds each fractional scheduling variable z'_{lnm} to the nearest integer while staying within the constraints of (3.2). This is achieved by maintaining the *residual* number of requests $\widetilde{\lambda}_{ln}$, the *residual* communication capacity $\widetilde{K}_n$, and the *residual* computation capacity $\widetilde{W}_m$. For each triple (l, n, m), $\min(a_{lnm}x_{lm}\widetilde{\lambda}_{ln}, \lfloor \frac{\widetilde{K}_n}{\kappa_l} \rfloor, \lfloor \frac{\widetilde{W}_m}{\omega_l} \rfloor)$ is the maximum number of type-(l, n) requests that can be scheduled to edge cloud m without violating any constraint or changing any existing scheduling decision. Thus, line 6 results in a best-effort approximation of the optimal fractional solution while enforcing the hard constraints of (3.2).

Complexity: Line 1 of Algorithm 9 is the same as request scheduling under soft constraints, whose complexity is $O(|L|^{5.5} \times |N|^{11})$ (see Section 3.5.2). The rounding takes $O(|L| \times |N|^2)$ time, as there are $O(|L| \times |N|^2)$ iterations and each iteration (lines 6–9) takes a constant time. Thus, LRRS has a complexity of $O(|L|^{5.5} \times |N|^{11})$.

3.6 Extension to Multi-frame Optimization

So far we have only considered the optimizations within one frame, with the assumption that the solutions will be repeatedly applied in each frame. However, for recurrent workloads, it is possible to predict the request rates for a larger time window (e.g., 24 hours) that contains multiple frames, each being a time interval with constant request rates. In this case, the frame-by-frame optimization framework for service placement can incur sub-optimality, as it neglects the correlation across frames, in the sense that the cost of placing a replica of service l at a given edge cloud depends on where service l was placed in the previous frame. To capture the correlation, we need to jointly optimize the service placement across all the predictable frames.

Let F be the set of frames for which request rate prediction is available. Our objective is to maximize the expected number of requests served over all the frames:

$$\max \sum_{f\in F} T_f \sum_{l\in L}\sum_{n\in N} \lambda_{ln}^f \sum_{m\in N} y_{lnm}^f \tag{3.15a}$$

$$\text{s.t.} \sum_{m\in N} y_{lnm}^f \le 1, \qquad \forall l\in L, n\in N, f\in F, \tag{3.15b}$$

$$\sum_{l\in L} x_{lm}^f r_l \le R_m, \qquad \forall m\in N, f\in F, \tag{3.15c}$$

$$\sum_{l\in L} \lambda_{ln}^f \kappa_l \sum_{m\in N} y_{lnm}^f \le K_n, \qquad \forall n\in N, f\in F, \tag{3.15d}$$

$$\sum_{l\in L} \omega_l \sum_{n\in N} \lambda_{ln}^f y_{lnm}^f \le W_m, \qquad \forall m\in N, f\in F, \tag{3.15e}$$

$$y_{lnm}^f \le a_{lnm} x_{lm}^f, \qquad \forall l\in L, n,m\in N, f\in F, \tag{3.15f}$$

$$\sum_{l\in L}\sum_{n\in N} x_{ln}^f \min_{n'\in N_+} \left(c_{ln'n} x_{ln'}^{f-1} + c_{\max}(1-x_{ln'}^{f-1})\right) \le B, \qquad \forall f\in F, \tag{3.15g}$$

$$x_{ln}^f \in \{0,1\}, y_{lnm}^f \ge 0, \qquad \forall l\in L, n,m\in N, f\in F. \tag{3.15h}$$

This formulation is similar to the single-frame formulation (3.1), but the scope is extended to multiple frames. The real difference is the non-linear constraint (3.15g), where $c_{\max} > \max_{l,n',n} c_{ln'n}$ is a large constant. Essentially, $\min_{n'\in N_+}(c_{ln'n}x_{ln'}^{f-1}+c_{\max}(1-x_{ln'}^{f-1}))$ is the minimum cost of placing a replica of service l at edge cloud n in frame f, which depends on the service placement in frame $f-1$.

The multi-frame optimization (3.15) is a *mixed integer non-linear program (MINP)* that is even harder than (3.1). Given a service placement $(x_{ln}^f)_{f\in F,l\in L,n\in N}$, the remaining optimization is still an LP in $(y_{lnm}^f)_{f\in F,l\in L,n,m\in N}$. Thus, GSP-SS (Algorithm 1) still applies, where we iteratively place one replica at a time in a selected frame, subject to constraints (3.15c, 3.15g), to maximize the objective value of the corresponding LP.

3.7 Performance Evaluation

We have evaluated the performance of the proposed algorithms using both synthetic and trace-driven simulations under soft and hard constraints. For

the synthetic cases we show the impact of varying constraints and characteristics of service requests to show that our algorithms are robust. For the trace cases, we show how frame and slot sizes should be set to achieve desirable performance.

3.7.1 Benchmarks

To assess the performance of the proposed service placement algorithm, we use the following benchmarks:

1. *the optimal solution* of (3.1) using an MILP solver (MATLAB `intlinprog`);
2. *LP-relaxation with rounding*, which first solves the LP relaxation of (3.1), and then rounds the placement variables to $\{0, 1\}$, subject to R- and B-constraints;
3. *top-k service placement*, which sequentially considers each edge cloud $m \in N$, computes the total demand for each service l that can be scheduled to m, defined as $\Lambda_{lm} = \sum_{n \in N} \lambda_{ln} a_{lnm}$, and then places services at m in the descending order of Λ_{lm} until reaching R_m or exhausting the budget.

To assess the performance of the proposed request scheduling algorithm under hard constraints, we use the following benchmarks:

1. *the optimal solution* of (3.2) using an ILP solver (MATLAB `intlinprog`);
2. *greedy request scheduling*, which sequentially considers each triple $(l, n, m) \in L \times N \times N$ and schedules as many type-(l, n) requests to edge cloud m as possible, without violating any constraint in (3.2) or changing any existing scheduling decision.

We note that no benchmark is needed for request scheduling under soft constraints, as the problem is an LP and hence polynomial-time solvable.

3.7.2 Results on Service Placement

3.7.2.1 Synthetic simulation

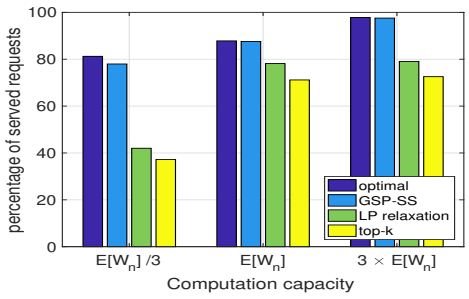
For synthetic simulations, we show results under two settings.

Setting 1: First, we set $|N| = 6$ and $|L| = 100$. We initially draw the values for R_n, K_n and W_n ($\forall n \in N$) uniformly from the intervals $[24, 36]$, $[16, 24]$ and $[32, 48]$, respectively. We then set these values differently based on a representative set of applications.[5] Assuming that the edge clouds are associated with hexagon cells arranged into two rows, we set the costs of replicating a service from an edge cloud k hops away or the remote cloud to $0.2k$ and 2, respectively, and the budget B to $0.2 \cdot |N| \cdot |L|$. We set a_{lnm} such that each request can only be served by edge clouds within 2 hops of the edge cloud it is submitted to. The arrival rate of request l is obtained as $\lambda_{ln} = \lambda_n p_{ln}$, where λ_n (total request rate in edge cloud n) is drawn randomly from the interval $[3, 5]$. For p_{ln} (popularity of service l in edge cloud n), we draw a random subset of services L_n, and set $p_{ln} \propto i_l^{-\alpha}$ for each $l \in L_n$, where i_l is the rank of l in L_n, and $\alpha = 0.5$ is the skewness parameter of Zipf's distribution. We initialize the system by randomly placing $|L|/8$ services. For each service l, κ_l, ω_l and r_l are drawn uniformly from $[0.5, 1]$. All results are averaged over 50 Monte Carlo runs. In the above settings, we set communication resource to be the most restrictive, followed by storage and then computation. This is to model the resource demands of data-intensive applications. Besides the above default settings, we will also explore the parameter space by varying these parameters one by one in order to evaluate the impact of each parameter.

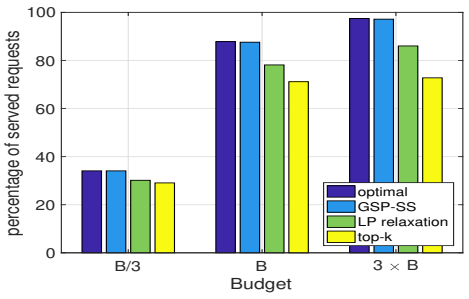
Figs. 3.7 (a-d) illustrate the effect of increasing various resource parameters on the percentage of served requests, including the computation capacity W_n, the service placement budget B, the storage capacity R_n, and the communication capacity K_n. As expected, an increase in the resource capacities leads to a higher percentage of served requests. This trend is more obvious in Figs. 3.7 (b-c), as these resources directly affect the set of feasible service placements. When comparing the performance of different algorithms under the same resource capacities, we observe that GSP-SS considerably outperforms LP-relaxation with rounding and top-k. Furthermore, it is very close to the optimal solution. We have verified that GSP-SS achieves over 90% of the optimal performance, i.e., the ratio of served requests when using GSP-SS versus the optimal solution is greater than 0.9 on the average. Similar observations have been made in the other simulations as well.

We further vary parameters of request generation. Fig. 3.7 (e) shows

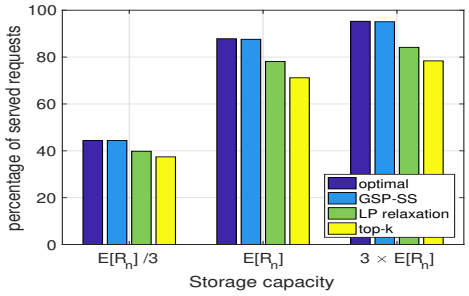
[5]The values of κ_l and K_n are in KBps, r_l and R_n in TB, and ω_l as well as W_n in Mflops/s.



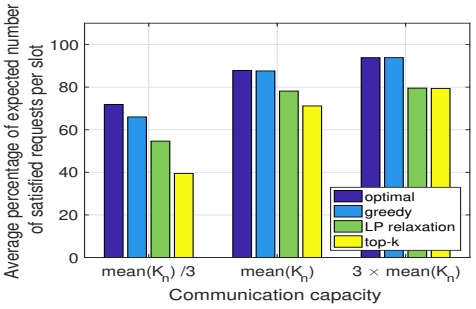
(a) varying W



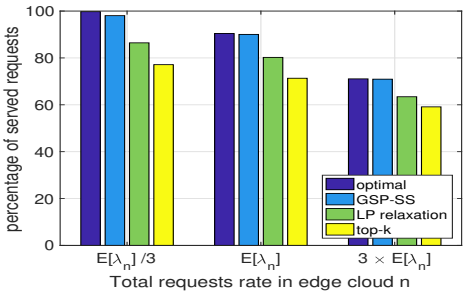
(b) varying B



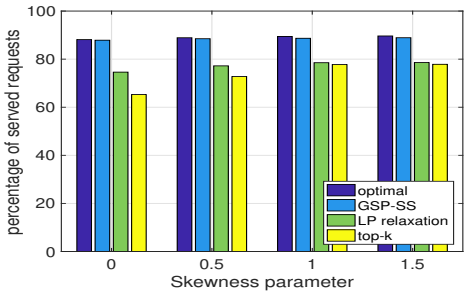
(c) varying R



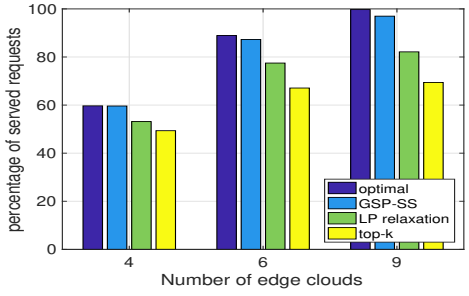
(d) varying K



(e) varying λ



(f) varying α



(g) varying $|N|$

Figure 3.7: Performance evaluation for service placement under the synthetic simulation setup in Section 3.7.2.

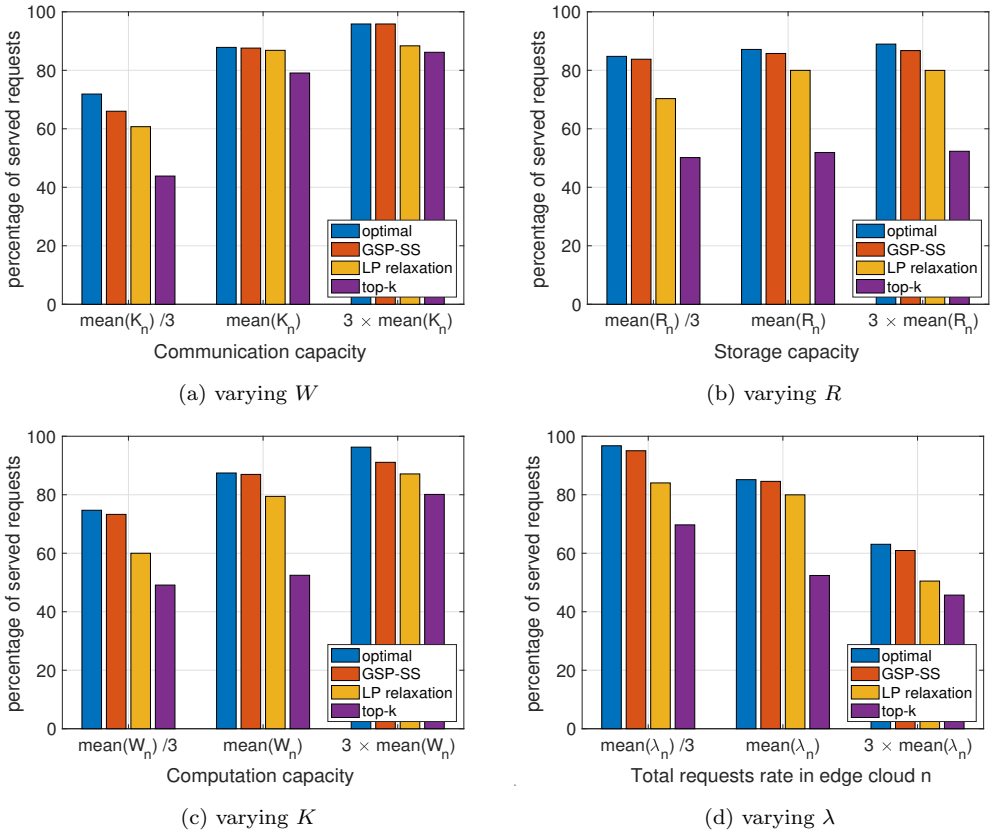
that as we increase the average request rate, the percentage of served requests decreases notably due to the contention of resources. Increasing λ_n (the total request rate in edge cloud n) $3\times$ and $9\times$, respectively, we go from 97.54% of satisfied requests to 88.42% and 69.49%, respectively, for our optimal solution. When it comes to the greedy solution, the level of satisfied requests drops from 95.88% to 88.04% and 69.36%, respectively. This is reasonable as in every step we increase the total amount of requests. Fig. 3.7 (f) shows that as we increase the skewness of service popularities (by increasing α), the optimal solution and GSP-SS remain the same, while the baselines (LP relaxation with rounding and top-k) improve slightly. This is because the increased skewness causes the requests to be more and more concentrated on a few popular services, making service placement easier.

Finally, we vary the number of edge clouds $|N|$ in Fig. 3.7 (g). As expected, the more edge clouds, the more resources, and hence the performance improves. However, we see from Figs. 3.7 (a-d) that similar improvements can be achieved by increasing the capacities of existing edge clouds or the service placement budget. This shows the effect of resource pooling.

We note that our simulation setup does not satisfy the condition in Theorem 3.5.1 as the κ_l's are different, and thus the theoretical approximation guarantee does not apply. Nevertheless, we have observed empirically that GSP-SS always yields near-optimal performance.

Setting 2: Next, we consider scenarios in which the values of the input parameters are at the same order of magnitude as those corresponding to specific highly popular data-intensive services [101], [102], i.e., video analytics and ultra-reliable virtual reality.

Figs. 3.8 (a-d) depict the percentage of served request for different values of communication capacity, arrival rate, storage and computation capacity. The communication capacity of the edge clouds is chosen uniformly on $[20, 30]$ Mbps, computation capacity is uniform on $[320, 480]$ Mflops/s, whereas storage capacity is chosen uniformly from the range $[24, 36]$ TB. The storage requirement for the services is chosen uniformly from the interval $[0.5, 1]$ TB, bandwidth requirement is uniform in $[5, 10]$ Mbps and computation requirement is uniform in $[50, 100]$ Mflops/s. We see that the trend and the comparison between different algorithms are qualitatively the same as those in Fig. 3.7. In particular, the proposed algorithm GSP-SS performs nearly as well as the optimal and notably better than the benchmarks of LP-relaxation with rounding and top-k.



(a) varying W

(b) varying R

(c) varying K

(d) varying λ

Figure 3.8: Performance evaluation for service placement under highly popular data-intensive service input.

3.7.2.2 Trace-driven simulation

We cross-validate our observations in a more realistic scenario driven by traces. For the trace-driven simulation, we extract user and edge cloud locations from real mobility traces and cell tower locations. We use the *taxicab* traces from [103], by extracting the traces of 36 users over a 520-minute period with location updates of 1 and 10 minutes. We assign users into Voronoi cells based on cell tower locations obtained from `http://www.antennasearch.com`, from which we select a subset of 6 cell towers that are at least 9.5 km apart to represent the locations of edge clouds.

User requests are generated from a wireless trace from [104], containing transmission timestamps generated by 5 different applications from 36 wireless devices. We associate each device with a user in the taxicab trace, and

duplicate each trace 5 times to obtain $|L| = 25$ services. As each timestamp in the original trace represents a single packet, we stretch the time axis by 60 (by treating the time unit as 'minute' instead of 'second') to simulate the arrival process of service requests. The obtained request rates range from 288,935 to 650,415, with a mean of 521,070 (requests/slot).

For each edge cloud, we randomly choose a storage capacity R_n of 3-6 TB, a communication capacity of 16-48 Mbps (i.e., $K_n \in [0.12, 0.36]$ GB/slot), unless stated otherwise, and a computation capacity of 50-100 Gflops/s (i.e., $W_n \in [3, 6]$ Tflops/slot), unless stated otherwise.

The other parameters are as before, except that the units of κ_l and ω_l change to GB/slot and Tflops/slot.

Setting Frame and Slot duration: The frame and slot durations are engineering decisions based on the characteristics of the system and desired performance. The frame duration is the time during which a deployment of services on servers remains constant. The re-deployment of services has a cost which in our system is constrained by budget B. How often to move services is driven by how stable the system is in terms of user mobility and user request characteristics (e.g., rate, resource requirements). A system operator will know the expected dynamics of a system based on trends determined over time. As trends change, the duration of the frames can change.

Slot duration is set based on the scheduling delay and job execution time in a system. In our system, jobs are expected to complete their processing within one slot. Therefore, slot duration should be set as small as possible so that jobs can complete so that scheduling delays are small.

To set the frame duration for the trace evaluation, we plotted the performance of the system for different frame and slot durations. Fig. 3.9 illustrates the effect of varying the frame size. In this scenario, the slot duration is 1 min. Fig. 3.9 depicts the percentage of served requests vs. frame size. As can be seen from Fig. 3.9, the performance starts deteriorating considerably for frames that are longer than 30 slots. Choosing a shorter frame provides only a slightly higher percentage of served requests, but increases the cost. Therefore we choose a frame duration of 30 minutes (slots in this case).

Next, we look at the impact of the slot duration on the performance. We consider a frame length of 30 minutes, with 5 different slot durations: 1, 2, 3, 5 and 10 minutes. This means that the frames consist of 30, 15, 10, 6 and 3 slots, respectively. The values of K and W are scaled with the duration of the slot so overall system resources are kept constant. Fig. 3.10

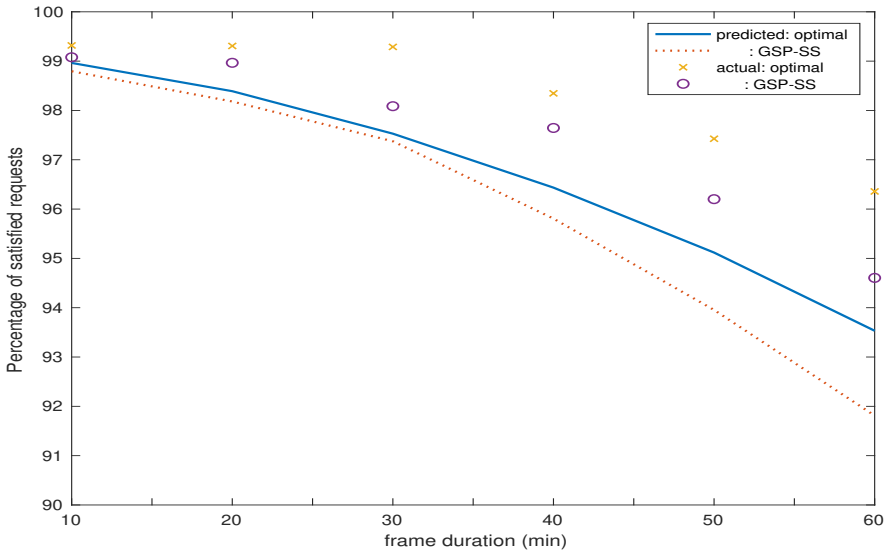


Figure 3.9: Varying the frame size.

illustrates the percentage of served requests vs. slot duration. As can be seen from Fig. 3.10, the performance is almost completely insensitive to the slot duration. Since we are interested in providing the shortest possible delay, we choose the slot duration to be 1 minute.

In this set of experiments we use a frame duration of 30 minutes and slot duration of 1 minute as described above.

Having shown the impact of slot and frame duration on the performance of the system, we proceed with looking at how the number of users varies in a given cell over time, with users leaving and coming into the cell. Slots are 1 minute, and a frame consists of 30 slots. Fig.3.11 shows the results. On the same plot, we also show the actual arrival rate of the requests for services in the cell and the predicted arrival rate of those requests. Note that there is a similar trend in the behavior of the three parameters considered here.

Results: Fig. 3.12 shows the performance of each algorithm over time. 'Predicted' values are the predicted percentage of served requests when solving (3.1) at the beginning of each frame. 'Actual' values are the actual percentage of requests served in each slot under soft constraints, obtained by solving (3.6) for the requests arrived in that slot and the service placement of the corresponding frame. Fig. 3.12 shows that GSP-SS closely approximates the optimal not only in the predicted performance but also in the actual

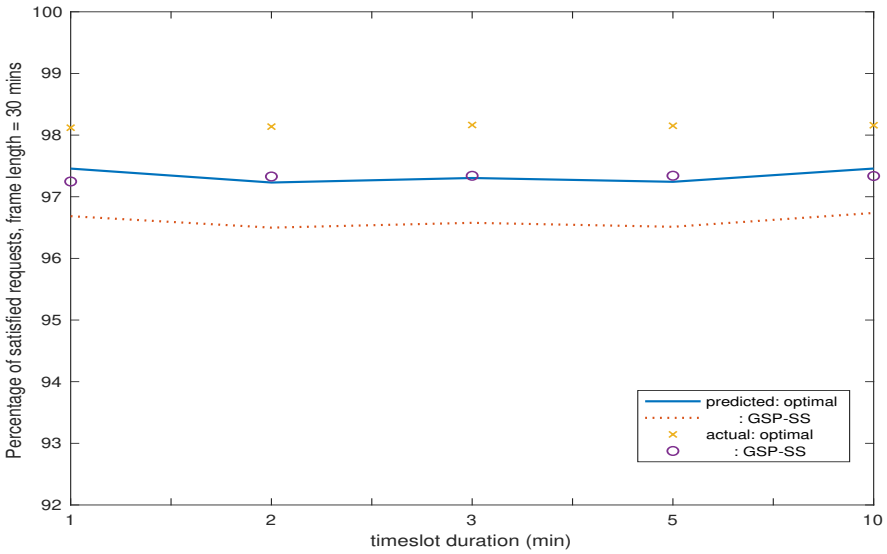


Figure 3.10: Varying the slot duration.

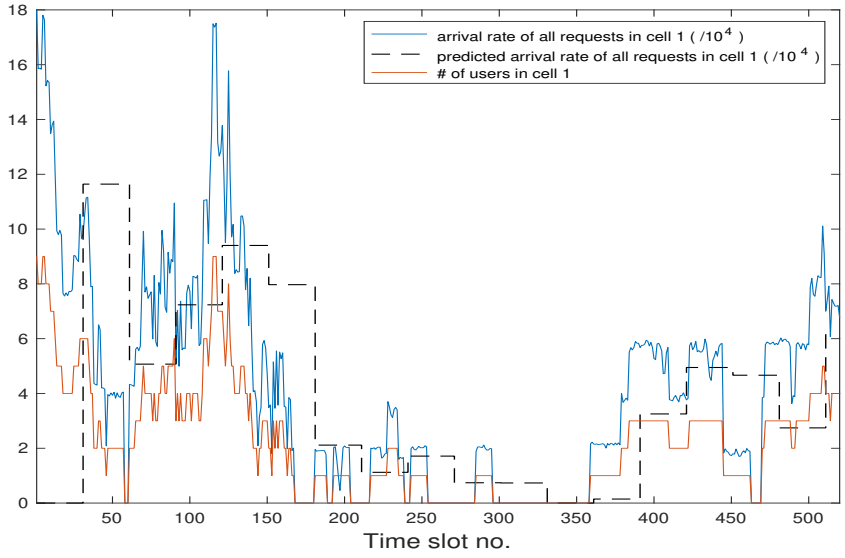


Figure 3.11: The variability of the user and requests in a cell performance, while outperforming the baselines.

3.7.3 Results on Request Scheduling

3.7.3.1 Soft Constraints

Fig. 3.12 ('actual') already shows the performance achieved by probabilistic scheduling under soft resource constraints. Since the optimal probabilistic schedule is not hard to compute (by solving (3.6)), the focus here is to understand to what extent this probabilistic schedule adheres to the resource constraints.

Given that probabilistic scheduling only satisfies the K-constraint (3.1d) and the W-constraint (3.1e) *on the average*, it is possible that the scheduled requests temporarily exceed the communication/computation capacity of an edge cloud. To understand the extent of capacity violations, we evaluate

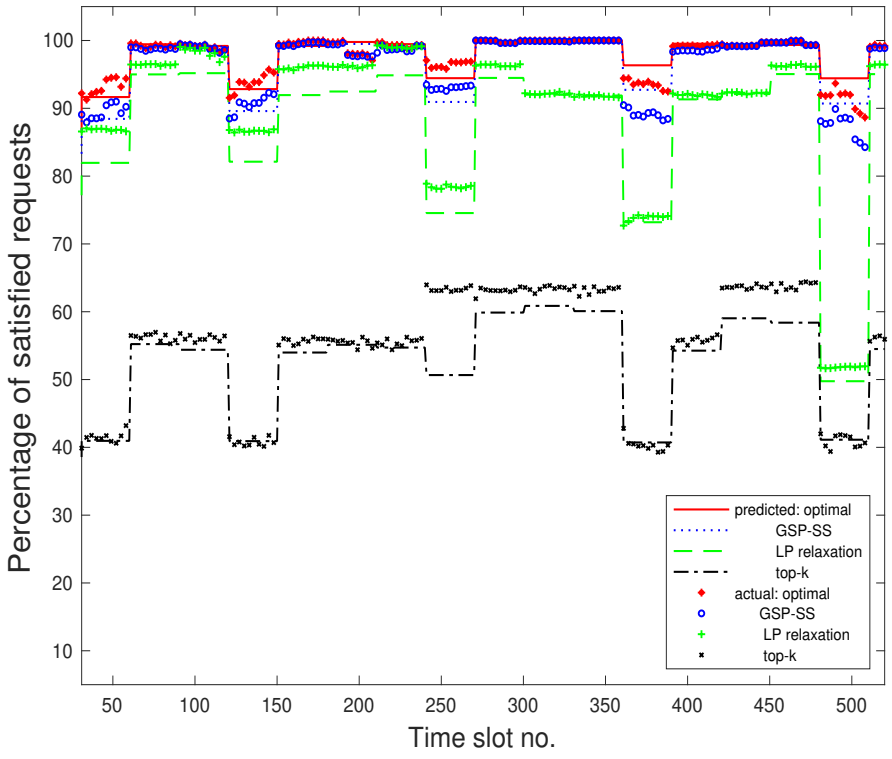


Figure 3.12: Performance evaluation in trace-driven simulation under soft constraints.

both the frequency and the severity of capacity violations. Table 3.2 shows the results for each type of resource (computation/communication) at each edge cloud. These results are obtained under the default parameter setting for synthetic simulations specified in Section 3.7.2. As can be seen from Table 3.2, there are capacity violations in about 5% of the slots, and in these slots, the amount by which the capacities are exceeded is about 10%. These are moderate violations, which justifies the use of probabilistic scheduling and soft resource constraints for services that are not highly delay-sensitive.

Table 3.2: Capacity Violations (%)

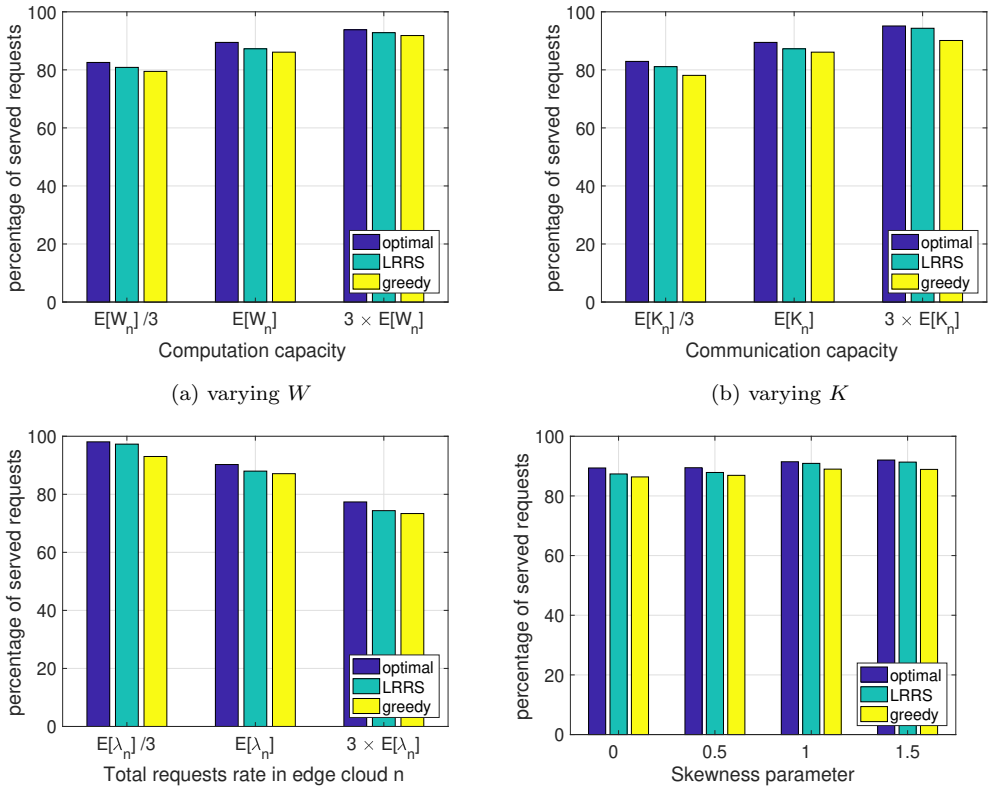
Edge cloud #	**1**	**2**	**3**	**4**	**5**	**6**
% of time K violated	6.33	4.64	2.85	2.86	4.66	2.28
% of time W violated	1.29	1.65	3.26	4.15	3.59	0.95
Amount K violated	3.05	11.48	2.47	7.68	5.61	7.84
Amount W violated	10.69	0.72	9.61	2.87	0.51	2.90

3.7.3.2 Hard Constraints

Similar to Fig. 3.7, we use synthetic simulations to evaluate the impacts of different input parameters for request scheduling under hard resource constraints. All the results are obtained under the optimal service placement.

The results are shown in Fig. 3.13, where (a) shows the impact of increasing the computation capacity (W), (b) shows the impact of increasing the communication capacity (K), (c) shows the impact of increasing the rate of requests (λ), and (d) shows the impact of increasing the skewness parameter (α). In every plot, we compare the proposed LRRS algorithm (Algorithm 9) with the optimal solution and the greedy solution that are explained in Section 3.7.1. Note that MFRS (Algorithm 4) requires $\kappa_l = \omega_l = 1$ for all $l \in L$, which is not satisfied here. While the impacts of these parameters are similar to those observed in Fig. 3.7, these results differ from Fig. 3.7 in that there is not much difference between different algorithms. In particular, the greedy request scheduling already approximates the optimal request scheduling, and LRRS performs in between. This observation indicates that the key performance differentiator is the service placement algorithm, and it suffices to use a simple algorithm for request scheduling.

We now show results for hard constraints using the same trace inputs as described above. As in Fig. 3.12, we show in Fig. 3.14 both the 'predicted'



(a) varying W

(b) varying K

(c) varying λ

(d) varying α

Figure 3.13: Performance evaluation for request scheduling under hard constraints.

percentage of served requests, computed at the beginning of each frame based on the predicted request rates, and the 'actual' percentage of served requests in each slot. The difference from Fig. 3.12 is that the 'actual' values are computed by LRRS under hard resource constraints (while the 'predicted' values are the same as in Fig. 3.12). Comparing Fig. 3.12 and 3.14, we see that imposing hard resource constraints *does not* significantly reduce the percentage of served requests, while having the advantage that every scheduled request is guaranteed to finish within one slot. This justifies the use of deterministic scheduling and hard resource constraints for real-time services.

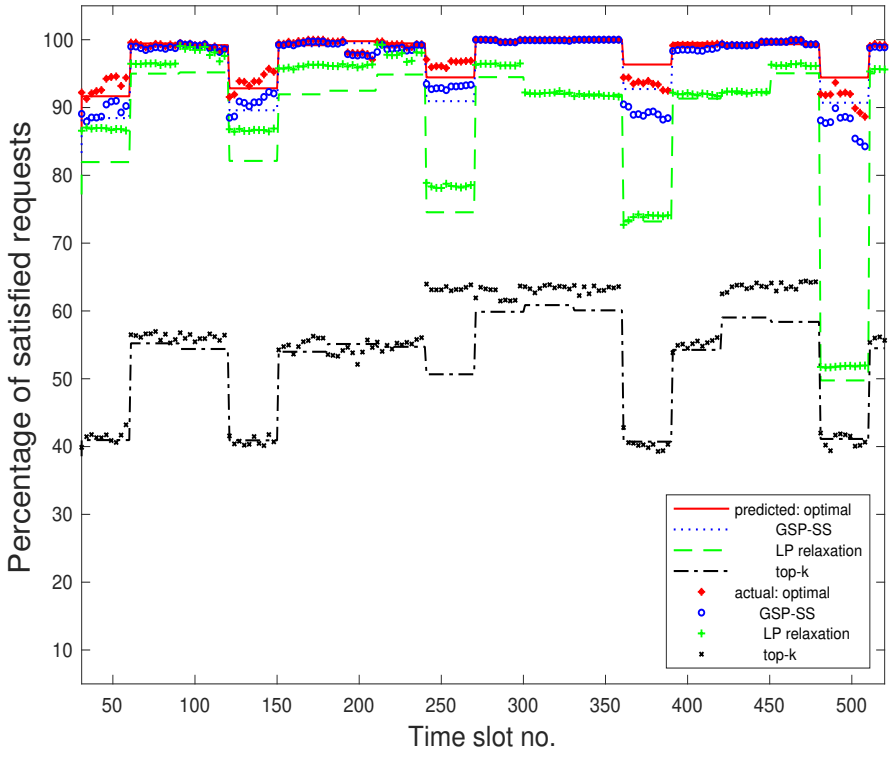


Figure 3.14: Performance evaluation in trace-driven simulation under hard constraints (scheduling done by LRRS).

3.7.4 Results on Multi-frame Extension

Although $|F| = 2$ suffices in this evaluation, we have observed that if we increase the resource contention, a larger prediction window can help. For example, if we double the request rates, then the predicted (actual) percentage of served requests will become 73.63% (72.11%) for $|F| = 1$, 74.37% (72.90%) for $|F| = 2$, and 79.11% (78.50%) for $|F| = 3$.

Finally, we evaluate the extended GSP-SS for the multi-frame optimization (3.15). Fig. 3.15 shows the performance based on request prediction over an $|F|$-frame sliding window. We skip the other algorithms in this evaluation, as top-k performs the same as in Fig. 3.12, the optimal solution is hard to compute due to nonlinearity of (3.15), and LP relaxation does not apply. As in Fig. 3.12, 'predicted' values are based on the request rates predicted at the

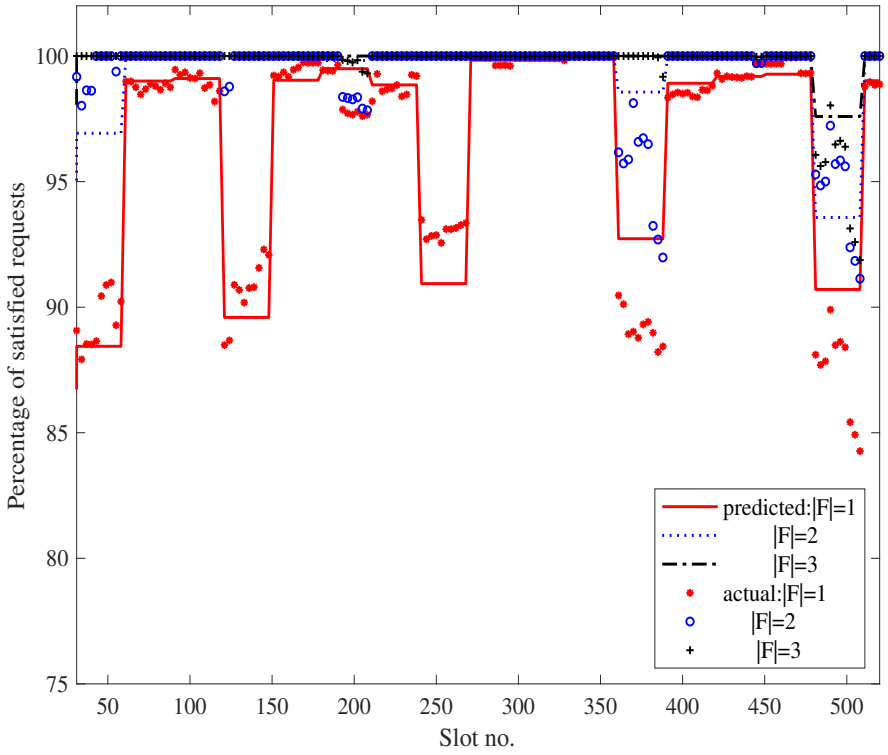


Figure 3.15: Performance of extended GSP-SS for multi-frame optimization in trace-driven simulation.

beginning of each window, and 'actual' values are based on the actual request rates in each slot. We see that prediction over a larger window improves the performance of GSP-SS in terms of the actual values, and 2-frame prediction appears to be sufficient.

3.8 Conclusion

We proposed a two-time-scale solution for joint service placement and request scheduling in a system of networked edge clouds under communication, computation, and storage constraints. We not only proved the NP-hardness of the problem in the general case, but also characterized its complexity in all the special cases. By combining the greedy heuristic with shadow request

scheduling, we developed a polynomial-time service placement algorithm, which was proved to give a constant approximation ratio under certain conditions. We further showed that the problem of request scheduling under hard resource constraints, although NP-hard in general, can be solved in polynomial time if all the requests demand the same amounts of communication and computation resources, in which case we developed a polynomial-time optimal solution based on the maximum flow algorithm. Extensive simulations showed that the key performance differentiator is the service placement algorithm, and the proposed service placement algorithm achieves near-optimal performance.

Chapter 4

Budget-Constrained Reinforcement of SCADA for Cascade Mitigation

We study the impact of coupling between the communication and the power networks as it affects a SCADA-based preventive control system. Today power grids use power lines to carry control information between components in the grid and a control center using power line carrier communication (PLCC). Thus a failure in the power grid will cause a failure in the control network and may reduce the capability of preventive control that in turn increases the risk of cascading failures. We pose the problem of allocating a limited number of non-PLCC communication links (e.g., microwave links) that are immune to failures in the power grid to maximize our controllability over the grid under power system failures, so as to maximize the total demand served at the end of cascade. By formulating the problem as a nonlinear integer programming problem, we establish its hardness and identify a generic heuristic that can find an approximate solution within controllable time. We further develop a domain-specific heuristic that utilizes both graph-theoretic and power system information to achieve similar performance as the generic heuristic at a much lower computational complexity. Our evaluations based on a $2,383$-bus Polish system demonstrate that only a few non-PLCC links, when placed correctly, can substantially improve the robustness of the grid as measured by the total demand served at the end of cascade.

4.1 Introduction

Cascading failures in power grids have led to widespread socio-economic disruptions [68]. Therefore, it is of high importance to improve our understanding of this phenomenon and develop defense mechanisms.

In this chapter, we focus on the interaction between the power grid and the control network during cascading failures. The control network monitors elements of the power grid, and issues command to some of these elements when failures occur to mitigate a cascade. Any degradation that limits the ability of the control network to either monitor or control elements in the power grid will increase the risk of a larger cascade of failures. Ideally, the control network, consisting of sensors/actuators, communication links, routers, and a controller, should be deployed independently of the power grid, with battery backup for all its components. However, this is an expensive solution, especially for deploying dedicated communication links to connect the controller to all the elements in the power grid.

For this reason, utility companies have used power lines to carry control information using the technique of *power line carrier communication (PLCC)* [105, 106, 107, 108, 109]. PLCC enables communication between substations using low [110, 111], medium [111, 112], or high voltage [113, 114] power lines. PLCC links are much less expensive than non-PLCC (i.e., dedicated) links [115, 116, 117], but will fail when power lines fail if no backup communication medium is used, thereby degrading connectivity of the control network. In this chapter, we examine the problem of reinforcing a fully PLCC-based communication network with the addition of a limited number of reliable more expensive dedicated non-PLCC communication links.

Traditionally, PLCC was used for one-way communication to monitor, control, and tele-protect the power grid [118]. In the 1980s and early 1990s, after studies regarding the implementation of the Supervisory Control and Data Acquisition (SCADA) in both Europe and the United States, bi-directional communication by PLCC was invented [115]. The strong growth of PLCC data services and its economic benefit has made it a promising component of information infrastructure [109, 119, 120, 121]. The mixture of PLCC and other technologies extends the utilization of PLCC for power generation and control [122]. While it may seem that the advances of non-PLCC communications to fiber-optic technology [123, 124] eliminate the need for PLCC, PLCC links remain an essential component of modern power grids [110],

such as the Siemens PowerLink PLCC system [116] and General Electric T&D Power Utilities [125], due to their cost savings.

In this chapter, we show that a carefully designed communication network containing a few strategically placed non-PLCC links together with PLCC links can provide almost the same protection against cascading failure as a dedicated communication network with 100% non-PLCC links at a fraction of the cost.

4.1.1 Summary of Contributions

In this chapter, we consider cascading failures in a coupled system of a power grid with a SCADA-based communication network that is geographically co-located with the power grid, through which a Control Center (CC) collects data from sensors and dispatches preventive control commands to generators and loads. Our contributions are:

1) We propose to combine the cost efficiency of PLCC links and the reliability of non-PLCC links in designing the communication network [126], posed as an optimization of maximizing the total power demand served after cascade by selecting a limited number of non-PLCC links, leaving the rest as PLCC.

2) As the demand served after cascade is not an explicit function of the decision variables, we propose a proxy objective function capturing the controllability of nodes in the grid, weighted by their importance in the system topology and the contribution of generation/load. We formulate the underlying optimization as a nonlinear programming (NLP) problem, which is proved to be NP-hard. We then apply a generic heuristic and develop a domain-specific heuristic that explicitly enhances the controllability of the generators and the loads most likely to be needed in preventive control.

3) We evaluate the proposed algorithms on a $2,383$-bus Polish power system. The results show that (i) the algorithms designed to maximize the proxy objective function can effectively increase the demand served after cascade, (ii) a small number of properly placed non-PLCC links together with (unreliable) PLCC links can achieve almost the same performance as a perfectly reliable communication network, and (iii) while performing similarly as a properly configured generic heuristic, the proposed domain-specific heuristic is significantly faster [127].

Roadmap: Section 4.2 provides background information about cascading failures and preventive control. Section 4.3 formulates our problem of

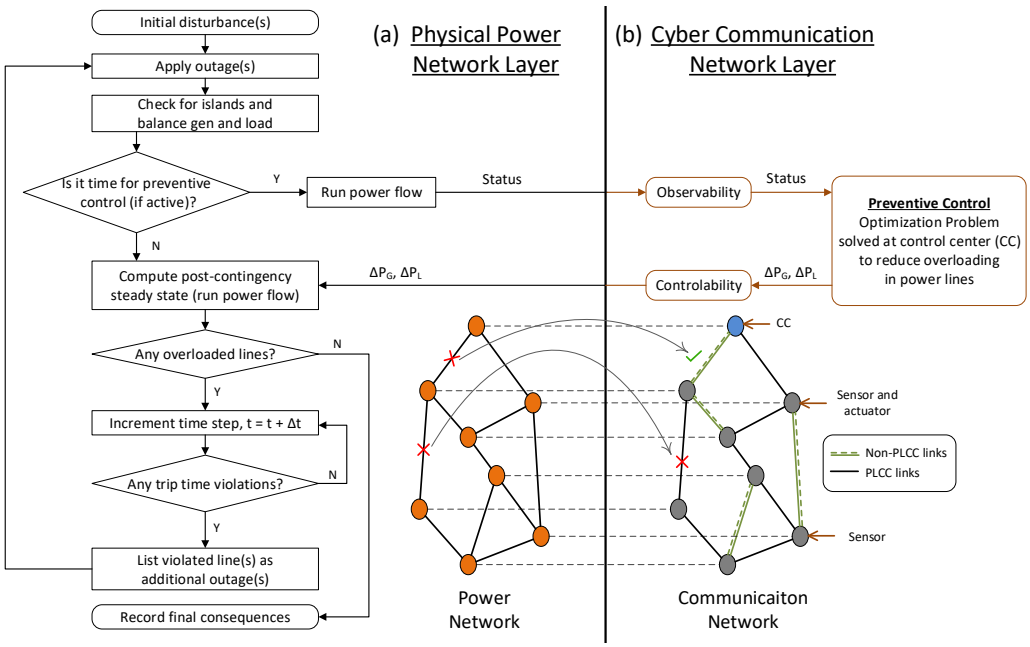


Figure 4.1: Flow chart highlighting the modeling of the coupled cascading failure in power and communication networks.

budget-constrained design of communication links and presents our proposed algorithms. Section 4.4 evaluates our algorithms against benchmarks. Finally, Section 4.5 concludes the chapter.

4.2 Background and Motivation

4.2.1 Modeling Coupled Cascading Failure

DC-QSS models [68, 69, 70, 83], which neglect resistive losses and assume a uniform voltage profile, are computationally efficient and thus suitable for statistical analysis of large-scale systems. Therefore we use DC-QSS models of cascades in this chapter. Figure 4.1 illustrates the model of cascading failure in a coupled power and communication network.

After the impact of the initial outages, the power grid may be segmented into islands. The generation and load in each island are balanced. In order to achieve the balance, either generation is curtailed or the load is reduced uniformly across all generation or load nodes, respectively. If an island does not

have any generators, then a complete blackout is assumed in the island. The overloaded branches, which result from the updated line flows, are tripped according to the tripping delays of overcurrent relays, meaning that for a particular line to trip, it must remain overloaded for the duration of its trip time. Once more lines are tripped, the power grid may be segmented into further islands, where load and generation need to be re-balanced, and the entire process is repeated until there are no potential line trips, i.e, no overloaded lines in the network. At this stage, the cascade propagation comes to an end.

As described earlier, we consider a geographically collocated SCADA-based communication network that connects the CC to sensors and actuators. This is a common assumption in work on power-communication overlays [70]. The connections between the CC and these sensors/actuators can be affected by failures in the power grid if PLCC communication is used under the assumption that no backup communication is deployed. Therefore, the cascading failure propagates in a coupled manner through both power and communication networks.

4.2.2 Modeling Preventive Control

A preventive controller at the CC is introduced, as illustrated in Fig. 4.1, which is assumed to have updated information of all the elements in the power grid, e.g., line flows, breaker status, and power output/consumption of generators and load centers. Based on this information, the controller tries to alleviate line overloading by issuing control commands. Both the sensing information and the control commands are communicated via a communication network.

The objective of the preventive controller is to stop cascade propagation by reducing the overloading in lines, which is defined as $\mathbf{L}_{over}$. This objective is achieved by solving the following optimization problem [70]:

$$\min_{\Delta P_G, \Delta P_L} \; -\mathbf{1}^\mathsf{T}\mathbf{\Delta P}_L + \lambda^\mathsf{T}\mathbf{L}_{over} \tag{4.1a}$$

$$\text{s.t. } \mathbf{\Delta P}_G - \mathbf{\Delta P}_L = \mathbf{B}\mathbf{\Delta}\theta, \quad \Delta\theta_i = 0, \forall i \in \Omega_{ref} \tag{4.1b}$$

$$\Delta L_{ij} = \frac{\Delta\theta_i - \Delta\theta_j}{x_{ij}}, \qquad \forall\ i, j \in \mathcal{M} \tag{4.1c}$$

$$|\mathbf{L}_{\mathcal{M}} + \Delta\mathbf{L}| \leq \mathbf{L}_{max} + \mathbf{L}_{over}, \quad \mathbf{L}_{over} \geq \mathbf{0} \tag{4.1d}$$

$$-(\mathbf{P}_G)_{\mathcal{M}} \leq (\mathbf{\Delta P}_G)_{\mathcal{M}} \leq \mathbf{0} \tag{4.1e}$$

$$-(\mathbf{P}_L)_{\mathcal{M}} \leq (\mathbf{\Delta P}_L)_{\mathcal{M}} \leq \mathbf{0} \tag{4.1f}$$

$$(\mathbf{\Delta P}_G)_{\overline{\mathcal{M}}} = \mathbf{0}, (\mathbf{\Delta P}_L)_{\overline{\mathcal{M}}} = \mathbf{0} \tag{4.1g}$$

Here, the elements of vectors $\mathbf{P}_G$ and $\mathbf{P}_L$ represent the generation and load power at each bus, respectively, and Δ represents the change in these quantities. The objective function (4.1a) minimizes the total amount of load shedding ($-\mathbf{1}^\mathsf{T}\mathbf{\Delta P}_L$) and the weighted sum of all overloads such that no further controls on generators or loads can improve the objective. λ is the uniform weight vector. Similarly, a change in the phase angle at the ith bus is defined as $\Delta\theta_i$. The matrix $\mathbf{B}$ is the admittance matrix of the network. The variables $\mathbf{L}$ and $\mathbf{L}_{max}$ represent the actual power flow and its allowable maximum value, respectively. The subscript $\mathcal{M}$ implies that the corresponding quantities belong to the measurable set, and the subscript $\overline{\mathcal{M}}$ implies the opposite. To get more details of the optimization problem (1), please refer to [70]. As presented in Fig. 4.1, line status and branch flows $\mathbf{L}_{\mathcal{M}}$ are taken as inputs to the optimization problem, and the solution gives load shedding and generation reduction values as outputs.

A communication network with 100% non-PLCC links is cascade-free. In this case, $\overline{\mathcal{M}} = \emptyset$ and the preventive control produces the best possible performance. However, in the presence of PLCC-based links the coupled cascade propagation is affected in a complex manner because power line failure impacts the preventive control. The larger the number of such link failures, the larger the set $\overline{\mathcal{M}}$. This implies that more sensors become unobservable and more actuators become uncontrollable. This limits the effectiveness of cascade prevention, which in turn exacerbates the loss of controllability in a closed-loop fashion.

In practice, preventive control algorithms will run at regular intervals, and there will be a delay in line tripping. To evaluate these effects, we run preventive control optimization following the first outage for every 30 s, which

is the shortest possible time for SCADA-based preventive control systems, and take the tripping delay into account.

4.2.3 Motivating Example

To understand the potential value of preventive control in mitigating cascading failures, we evaluate the performance of a network with preventive control. Considering the Polish network during winter $1999 - 2000$ [128] as a test system, which includes $2,383$ buses and $2,896$ branches, we randomly fail 5% of the buses in the power grid and calculate the served power after cascade propagation in scenarios (i) having a network of 100%-PLCC links versus (ii) no communication network. Results in Table I show superior performance in presence of a communication network.

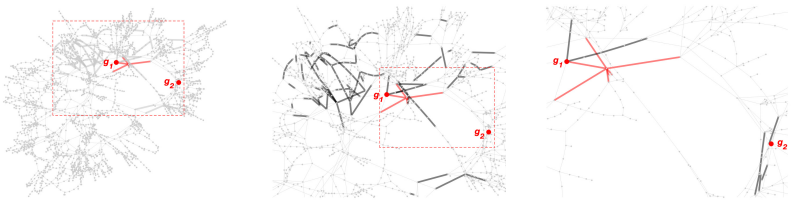
We use a specific example in Fig. 4.2 to illustrate why a suitably designed communication network can help preventive control to mitigate cascade. The red links in Fig. (4.2a) are the failures initiating the cascade, which prompt the overloading and tripping of the black links in Figs. (4.2b-c), respectively. In Fig. (4.2b), there is no communication network. In Fig. (4.2c), the CC communicates with and controls the two red generators.

Without control (Fig. (4.2b)), the cascade causes the loss of 134 lines with the residual power 13653.30 MW. The active power injections of the left (g_1) and the right (g_2) red generators after the cascade are 96.90 MW and 643.0 MW, respectively. With control of only the two nodes g_1 and g_2 (Fig. (4.2c)), e.g., by connecting them to the CC through non-PLCC links, the cascade only causes the loss of 10 lines with the residual power 24460.20 MW(99.60% of the initial power). This is achieved by changing the active power injections at generators g_1 and g_2 to 56.34MW and 645.13MW, respectively, which prevents line overload while satisfying energy conservation.

This example not only demonstrates the benefit of preventive control, but also shows that much of the benefit can be achieved by controlling a small number of critical nodes.

Table 4.1: The effect of preventive control on cascade in (i) 100%-PLCC links and (ii) no communication network.

Scenario	no. of cascade steps	no. of tripped lines	residual power
(i)	5	308	22472.14 MW
(ii)	20	504	7790.8 MW



(a) (b) (c)

Figure 4.2: The effect of preventive control on cascade propagation. (a) initial failures in the Polish grid (red links), (b) cascading failures (black links) without control (as a zoom-in of the rectangle in (a)), (c) cascading failures (black links) with control of the two red nodes (as a zoom-in of the rectangle in (b)).

4.3 Budget-Constrained Reinforcement of Communication Network

We develop algorithms to design the communication network as a mix of PLCC and non-PLCC links under a budget constraint to facilitate the mitigation of cascading failures through preventive control. To rule out other impacts, we assume that all the communication nodes (including sensors, actuators, and relays) have battery backups and are hence immune to the cascade and cannot be affected by failures in the power grid.

4.3.1 Problem Formulation

We formulate the problem as an optimization of non-PLCC link placement under a budget constraint. As PLCC links are much less expensive than non-PLCC links [115, 116, 117], we start with a baseline where all the communication links are PLCC, and then turn a selected subset of links into non-PLCC links to improve the robustness against cascades. To capture resource limitations, we impose a budget B on the number of non-PLCC links, but our solution can be easily extended to incorporate heterogeneous costs of non-PLCC links.

Ideally, we want to maximize the total demand served when the cascade stops. This objective function, however, faces the challenge that it is not an explicit function of the placement of non-PLCC links. To address this challenge, we propose to use a proxy objective function as follows.

We model the power grid as an undirected graph $\mathcal{G}(\mathcal{N}, \mathcal{L})$ with no self-loops or multiple edges. Let P^i, $\forall i \in \mathcal{N}$ denote the set of all possible paths between the CC and node i. Modeling each path $m \in P^i$ as a set of links it traverses, the total reliability of the network [129], measured by the expected number of nodes connected to the CC after (initial) failure, is defined as (assuming nodes do not fail):

$$\sum_{i \in \mathcal{N}} [1 - \prod_{m \in P^i} (1 - \prod_{l \in m} \rho_l)], \tag{4.2}$$

where ρ_l denotes the reliability (i.e., complement of failure probability) of link l. Let $R_l \in \{0, 1\}$ indicate whether link l is a non-PLCC link, and p_1/p_0 denote the reliability of non-PLCC/PLCC link, respectively (assuming $p_1 > p_0$). Then $\rho_l = R_l p_1 + (1 - R_l) p_0$. We formulate the optimization of non-PLCC links as follows:

$$\max \sum_{i \in \mathcal{N}} \gamma_i (1 - \prod_{m \in P^i} (1 - \prod_{l \in m \cap \mathcal{L}} (R_l p_1 + (1 - R_l) p_0)) \tag{4.3a}$$

$$\text{s.t.} \sum_{l \in \mathcal{L}} R_l \leq B, \tag{4.3b}$$

$$R_l \in \{0, 1\}, \qquad \forall l \in \mathcal{L}. \tag{4.3c}$$

In words, (4.3) aims at selecting up to B non-PLCC links to maximize the controllability after (initial) failure, measured by the expected total weight of all the nodes remaining connected to the CC. Intuitively, the more nodes the CC is connected to, the better it can observe/control the grid, and hence the better it can mitigate the cascading of failures. However, not all the nodes are equally important, and hence we use the weight $\gamma_i \geq 0$ to reflect the importance of observing and controlling node i.

Remark: Intuitively, γ_i should reflect both the topological importance (e.g., centrality) and the service importance (e.g., power injection) of node i. We find that defining γ_i as "the betweenness centrality (BC) of node i" $\times$ "the real power injected at node i" yields the best performance (see Fig. 4.4), where BC of a node is the frequency that it appears on the shortest paths between all pair of nodes in the graph [130].

4.3.2 Complexity Analysis

We prove that (4.3) is NP-hard by a reduction from the Steiner tree problem.

Theorem 4.3.1. *Problem (4.3) is NP-hard.*

Proof. The (graph) Steiner tree problem takes as input an undirected graph $\mathcal{G}_0$ with non-negative edge weights and a subset of vertices called *terminals*, and seeks to find a tree that is a subgraph of $\mathcal{G}_0$ with minimum weight to connect all the terminals. The decision problem associated with the Steiner tree problem is "whether exists a solution to the Steiner tree problem with integer edge weights, such that the total weight of the Steiner tree is no greater than a given natural number k". This problem is one of Karp's 21 NP-complete problems [131].

Using the above problem, we will show that the decision problem of "whether there is a feasible solution to (4.3) that connects the CC to all the non-zero-weight nodes by non-PLCC links" is NP-hard.

The NP-hardness of this decision problem implies the NP-hardness of a special case of the optimization problem (4.3) for $p_1 = 1 > p_0$, as otherwise we can solve the optimization problem and compare the achieved objective value with $\sum_{i\in\mathcal{N}} \gamma_i$. As each node i with $\gamma_i > 0$ contributes $\leq \gamma_i$ to the objective value, with "=" achieved only if it is connected to the CC via a perfectly reliable path (consisting of only non-PLCC links), it is easy to see that the optimal objective value equals $\sum_{i\in\mathcal{N}} \gamma_i$ if and only if there is a feasible solution to (4.3), under which every non-zero-weight node is connected to the CC by a path of non-PLCC links, thus solving the decision problem.

Construction: We construct $\mathcal{G}$ according to $\mathcal{G}_0$, except that each edge in $\mathcal{G}_0$ of weight e (a positive integer) is represented by a tandem of e links in $\mathcal{G}$. The budget B is set to k. One of the terminals is set as the CC, and the other terminals as nodes with non-zero weights. The rest nodes have zero weight.

Claim: The answer to the Steiner tree decision problem is "yes" if and only if the answer to the decision version of the above-constructed instance of (4.3) is "yes".

Proof of the claim: If the decision problem associated with the Steiner tree gives "yes", i.e., there is a tree $\mathcal{T}_0$ in $\mathcal{G}_0$ with total weight no more than k that connects all the terminals, then the corresponding tree $\mathcal{T}$ in $\mathcal{G}$ will connect the CC with all the non-zero-weight nodes while covering no more than B links. Hence, setting $R_l = 1$ for links in $\mathcal{T}$ and $R_l = 0$ otherwise will connect the CC to all the non-zero-weight nodes by non-PLCC links within

budget B. Conversely, if the CC can reach all the non-zero-weight nodes through no more than B non-PLCC links, then $\mathcal{G}$ must contain a tree $\mathcal{T}$ of no more than B links that contains the CC and all the nodes with non-zero weights. Then, the corresponding tree $\mathcal{T}_0$ in $\mathcal{G}_0$ must be a Steiner tree with a total weight no greater than k that connects all the terminals. □

4.3.3 Algorithm Design

The NP-hardness of the optimal solution to (4.3) motivates our search for efficient heuristics.

4.3.3.1 Generic heuristic

We first apply a generic heuristic algorithm that is a *genetic algorithm* [132], that belongs to a non-deterministic class of algorithms that provide suboptimal solutions in controllable time. It works by modifying a population of possible solutions repeatedly such that the population evolves toward an optimal solution. At each step, the genetic algorithm arbitrarily picks solutions from the current population to be parents and produces the children for the next step. Due to the possibly exponential complexity in enumerating all possible paths, we limit P^i for each $i \in \mathcal{N}$ to a set of up to N simple paths [133] from the CC to node i with length $\leq L$, where L and N are design parameters that will be tuned later (see Fig. 4.5).

4.3.3.2 Domain-specific heuristic

As shown later (Fig. 4.5), the generic heuristic needs to search a large solution space to achieve reasonable performance, which is computationally expensive. This motivates us to develop the following alternative that uses domain-specific insights. As the ultimate objective is to ensure that the re-dispatch-based preventive control can effectively mitigate the propagation of cascading failure, we will focus solely on maximizing the controllability of the generators and the loads whose re-dispatching is most likely to be needed during preventive control – given the budget constraint. Although this method appears more complicated, we will show that it is much more computationally efficient than the generic heuristic in Section 4.3.3.1 while achieving almost the same performance in terms of served loads.

To this end, we propose a two-part algorithm, shown in Algorithms 4 and 5. In the first algorithm, we follow the steps described below to identify subgraphs and candidate non-PLCC links. The second algorithm selects the non-PLCC links.

Identify subgraphs (Step 1): Ideally, extensive planning studies should give us the information regarding subgraphs that are likely to form during cascading failures. In absence of this information for the system under consideration, we propose a simple heuristic that is motivated by the intuition to partition the power grid by clustering nodes around each high-degree generator/load into a subgraph, which is likely to form an island during cascade. Then, we try to find candidate non-PLCC links connecting the CC to the actuators in each subgraph. Alternative graph clustering algorithms can be found in [134] – exploration of such algorithms are beyond the scope of this work.

To that end, we first sort the generation and load buses according to their degree, then we build subgraphs around them consecutively. We consider the subgraphs formed by all nodes that can be reached in a pre-determined number of hops from the root node (the radius of the neighborhood). We ignore the nodes that are already included by an existing subgraph. This procedure is repeated until we cover all the nodes. In this context, we use words 'subgraph' and 'area' (a power grid domain-specific terminology) interchangeably.

Algorithm 4: Candidate Non-PLCC Link and Candidate Node Selection

input : Power grid data, location of Control Center (CC)
output: Set of candidate non-PLCC links $\mathcal{L}_n$ and candidate control nodes $\mathcal{N}_n$, and subgraphs $\mathcal{V}$

1 **Step** 1**:** Identify subgraphs based on the degree of load and generation nodes and a pre-determined hop count h. G_i/L_i: total generation/load in the ith subgraph;
2 **Step** 2: Within the ith subgraph – If $G_i > L_i (L_i > G_i)$, then choose all generator nodes (load nodes) with degree > 1 as candidate control nodes. If $G_i = L_i$, choose all load nodes (degree > 1) in this subgraph as candidate control nodes. Set of such nodes are $\mathcal{N}_n$;
3 **Step** 3: Solve the problem of finding the tree of shortest paths on graphs to connect the CC to all candidate control nodes from Step 1. Set of such links are $\mathcal{L}_n$;
4 return $\mathcal{L}_n$, $\mathcal{N}_n$, $\mathcal{V}$;

Algorithm 5: Non-PLCC Link Selection

input : Budget B, $\mathcal{L}_n$, $\mathcal{N}_n$, $\mathcal{V}$, location of CC, power grid data
output: Set of non-PLCC links $\mathcal{L}_c$

Calculate subgraph weights: Define k_i: no. of nodes of the ith subgraph $\in \mathcal{V}$ connected to the remaining portion of the grid. Calculate weights $w_i = min(L_i, G_i)/k_i$;
Sort subgraphs: Order the subgraphs in descending order of weights;
$\mathcal{L}_c \leftarrow \emptyset, i \leftarrow 1, \mathcal{G}_c \leftarrow \emptyset$;
while $|\mathcal{L}_c| \leq B$ **do**
    while $i \leq |\mathcal{V}|$ **do**
        if *the ith subgraph does not contain the CC* **then**
            Connect one Gateway node $g \in \mathcal{N}_n$ in the subgraph to one Gateway node $g \in \mathcal{N}_n$ in a subgraph containing the CC using minimum number of non-PLCC links $l \in \mathcal{L}_n$;
            $i \leftarrow i + 1$, $\mathcal{L}_c \leftarrow \mathcal{L}_c \cup \{l\}, \mathcal{G}_c \leftarrow g$;
        else
            $i \leftarrow i + 1$
    return i;
    $i \leftarrow 1$;
    while $i \leq |\mathcal{V}|$ **do**
        if *the ith subgraph contains the CC* **then**
            Calculate betweenness centrality (BC) of nodes within the area w.r.t. the gateway nodes $\in \mathcal{G}_c$ and the CC. Assign non-PLCC links $l \in \mathcal{L}_n$ incident on the node with highest BC;
            $i \leftarrow i + 1$, $\mathcal{L}_c \leftarrow \mathcal{L}_c \cup \{l\}, \mathcal{G}_c \leftarrow g$;
        else
            $i \leftarrow i + 1$
    return i;
    $i \leftarrow 1$;
    while $i \leq |\mathcal{V}|$ **do**
        if *the ith subgraph does not contain the CC* **then**
            Calculate BC of nodes within the area w.r.t. the gateway nodes $\in \mathcal{G}_c$ and the candidate control nodes. Assign non-PLCC links $l \in \mathcal{L}_n$ incident on the node with highest BC;
            $i \leftarrow i + 1$, $\mathcal{L}_c \leftarrow \mathcal{L}_c \cup \{l\}, \mathcal{G}_c \leftarrow g$;
        else
            $i \leftarrow i + 1$
    return i;
return $\mathcal{L}_c$;

Figure 4.3 shows the graph of the Polish system with subgraphs marked on it with different colors, where roots are specified by increasing the sizes of their markers. Nodes within the same subgraph are at most 10 hops away

from the root.

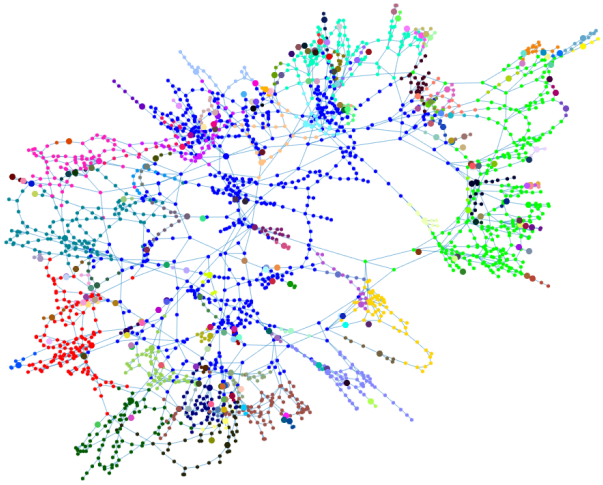


Figure 4.3: Subgraphs of the system for the domain-specific heuristic, $h = 10$.

Find candidate non-PLCC links and candidate control nodes (Steps 2, 3): This is a two-step process. Keeping in mind that generation and load nodes need to be connected to the CC for receiving preventive control commands (see, Section 4.2.2), we aim to find a subset of such nodes that are candidates for this. In Step 2, we find the subgraphs where $G_i \neq L_i$ – these areas affect line flows in external areas. Since we always reduce generation and load, we choose generator nodes as candidates when $G_i > L_i$ and vice-versa. The reason behind neglecting nodes with degree 1 is that outage of a line connected to this node will result in disconnection of the generator/load, thereby rendering the non-PLCC communication useless. In Step 3, we solve the problem of finding the tree of shortest paths on graphs to connect the CC to all candidate control nodes.

Algorithm 5 aims to select the non-PLCC links from the candidate control nodes based on graph-theoretic and domain-driven metrics. The steps are described next:

Calculate subgraph weights (Line 1) and sort subgraphs (Line 2): The proposed algorithm prioritizes subgraphs based on weights w_i, which depend on the value of the generation or load, whichever is smaller ($min(L_i, G_i)$),

and the connectivity between this subgraph and the rest of the grid (k_i). The logical argument is as follows: **(a)** higher $w_i = min(L_i, G_i)/k_i$ implies that the subgraph has a relatively lower connectivity with other subgraphs and thus a higher probability to form an island during cascade, while **(b)** a subgraph with a higher w_i also contains relatively more generation and load, and is thus more critical to control in the case of islanding to prevent further cascade propagation within the subgraph. We define a gateway node as one of the candidate control nodes connecting the particular subgraph to another gateway node in another subgraph. We use the term 'gateway' because this particular subgraph joins rest of the system through this node via a non-PLCC link. Thus, we connect a gateway node of a subgraph with high w_i but not the CC, to a gateway node in a subgraph containing the CC. If there are multiple options to do this, then the shortest path is chosen.

Establish non-PLCC links (Lines $3 - 28$*):* For establishing non-PLCC links within each area, we use BC measures considering the CC, gateway nodes and candidate control nodes as shown in Algorithm 5.

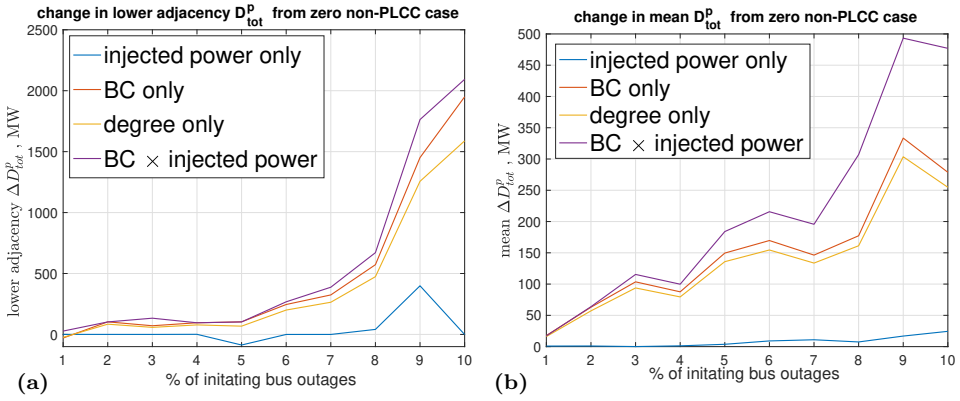


Figure 4.4: Performance evaluation in terms of change in D^p_{tot} with respect to 100% PLCC case for different node weights (non-PLCC links are placed by the generic heuristic under $L = 200$, $N = 50$, and $B = 17$).

4.4 Performance Evaluation

We evaluate the proposed solutions on the Polish network during winter $1999 - 2000$ peak condition from Matpower [128]. This system includes $2,383$ buses, $2,896$ branches, and 327 generators. We consider initial bus

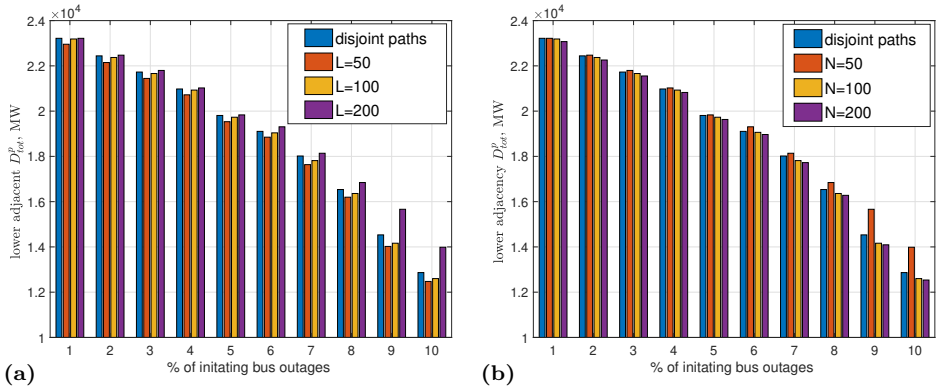


Figure 4.5: Performance evaluation for the generic heuristic under different design parameters: (a) $N = 50$ & varying L, (b) $L = 200$ & varying N ($B = 17$ in both cases).

outages varying from $1\% - 10\%$ of all the buses. For each failure scenario, 500 random sets of node (i.e., bus) outages have been considered. The CC is situated on bus 7 of the Polish network, which is one of the highest degree nodes. We set p_0 and p_1, the reliability of PLCC and non-PLCC links in (4.3) to 0.99 and 0.9999, respectively [135].

Impact of node weight definition: We start by comparing the performance under different definitions of node weight γ_i.

We compare four different definitions of weights: (i) power injection, (ii) degree, (iii) BC, and (iv) power injection $\times$ BC. Under each definition, we solve (4.3) by the generic heuristic under $L = 200$, $N = 50$ and $B = 17$. Figs. 4.4(a-b) show the lower adjacency, the smallest data point that is not an outlier in the plot which is 1st quartile - 1.5$\times$inter-quartile range, and the mean of the change in total post-contingency demand served (ΔD^p_{tot}) with respect to the case of 100% PLCC links, where $D^p_{tot} = 24558$ MW before any failure. The results show that representing a node weight by "the BC of the node" $\times$ "the real power injected at the node" performs the best, as it considers both the topological and the service importance of the node. We also calculated the reliability of the network as defined in (5.2) under the non-PLCC links placed under each of these objectives for different definitions of weights. The results follow the same trend as Figs. 4.4(a-b), but are not shown in the images. This leads us to conclude that a higher reliable network leads to higher post-contingency served demand after cascade.

In the sequel, we will use this definition of weight for the generic heuristic.

Configuration of generic heuristic: We then compare the performance of the generic heuristic under different limits on the length (L) and the number (N) of paths between each node and the CC. For comparison, we also evaluate a variation of this heuristic that requires the paths in each P^i ($i \in \mathcal{N}$) to be disjoint with each other (without limitation on path length). Fig. 4.5 shows the performance in mitigating cascades in terms of the lower adjacency of the improvement in post-contingency demand served (similar comparison has been observed for the mean and other percentiles). The results indicate that when using the generic heuristic, allowing overlap between paths, and choosing a smaller N and a larger L improve the efficacy in mitigating cascades. The reason lies in the limitation of genetic algorithm where the solution quality may deteriorate with the increase of problem size.

Configuration of domain-specific heuristic: Next, we evaluate the performance of the domain-specific heuristic under different settings of the design parameter h, the maximum hop count in generating subgraphs in Algorithm 4. Fig. 4.6 shows the impact of the h, evaluated by the lower adjacency and the mean of ΔD^p_{tot} as in Fig. 4.4. The results suggest that the parameter h significantly affects the performance of the resulting control system as it determines the number and sizes of the generated subgraphs. As it can be seen, the performance starts deteriorating for h that are larger than 10. Choosing a smaller h, doesn't provide a higher percentage of served demands as it excessively increases the number of candidate control nodes/candidate Non-PLCC links. For this particular power grid, we find that setting $h = 10$ leads to better performance, which will be the setting used for this heuristic in the sequel.

Under the above setting, the Polish system is divided into 227 subgraphs by Algorithm 1. Table. 4.2 summarizes different statistics of these subgraphs, including their sizes (defined as the number of nodes within the particular subgraph) and the variables k_i and w_i used by Algorithm 2 in placing non-PLCC links. Recall that k_i is the number of nodes connecting the ith subgraph to the remaining portion of the grid, and $w_i = min(L_i, G_i)/k_i$ where G_i/L_i is the total generation/load in the ith subgraph.

We have also evaluated a variation of the domain-specific heuristic, where the subgraph weight is defined as $u_i = \alpha w_i + (1-\alpha)v_i$, where $v_i := |G_i - L_i| \times k_i$

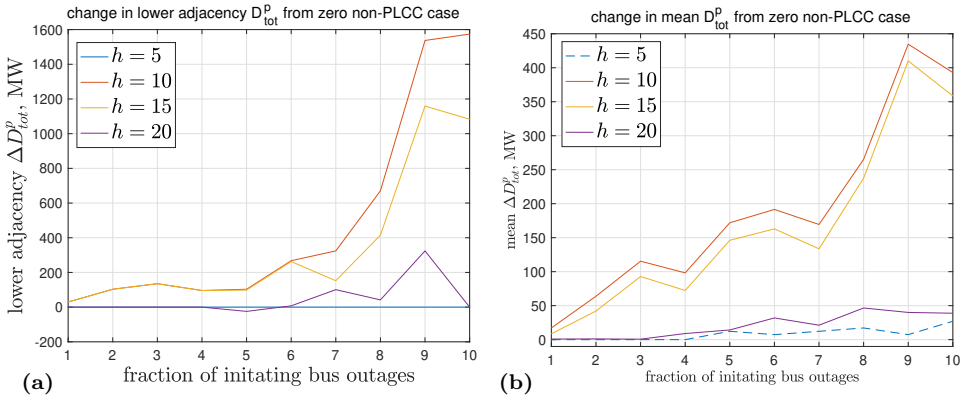


Figure 4.6: Performance evaluation for the domain-specific heuristic under different design parameters ($B = 17$).

Table 4.2: Statistical measures of variables in the domain-specific heuristic under $B = 17$

Variable	min	max	mean	median	STD
size	1	596	10.5	1	46.2
k_i	1	139	2.47	1	9.89
w_i	0	83.16	3.816	0	12.26

and the variable $\alpha = (0, 1]$ determines the relative importance of weights w_i and v_i. The intuitive motivation is that a subgraph with higher v_i has a higher probability to remain connected to other subgraphs and a higher impact due to higher $|G_i - L_i|$ (excess generation or load). The results, which are omitted due to space limitation, indicate that setting $\alpha = 1$ leads to better performance.

Overall comparison: Finally, Fig. 4.7 compares the performance of all the algorithms in terms of the change in total post-contingency demand served ΔD^p_{tot} with respect to the case of 100% PLCC links ('$B = 0$'). In addition to the proposed heuristics and the baseline of randomly selecting non-PLCC links ('Random'), we consider an intuitive benchmark of allocating non-PLCC links by ranking the nodes in descending order of their BC and selecting all the links incident to each node as non-PLCC links until the budget runs out ('BC'). Different statistical measures of ΔD^p_{tot} are shown.

Results show that while both of the proposed heuristics outperform random selection, the BC-based benchmark performs slightly better than the

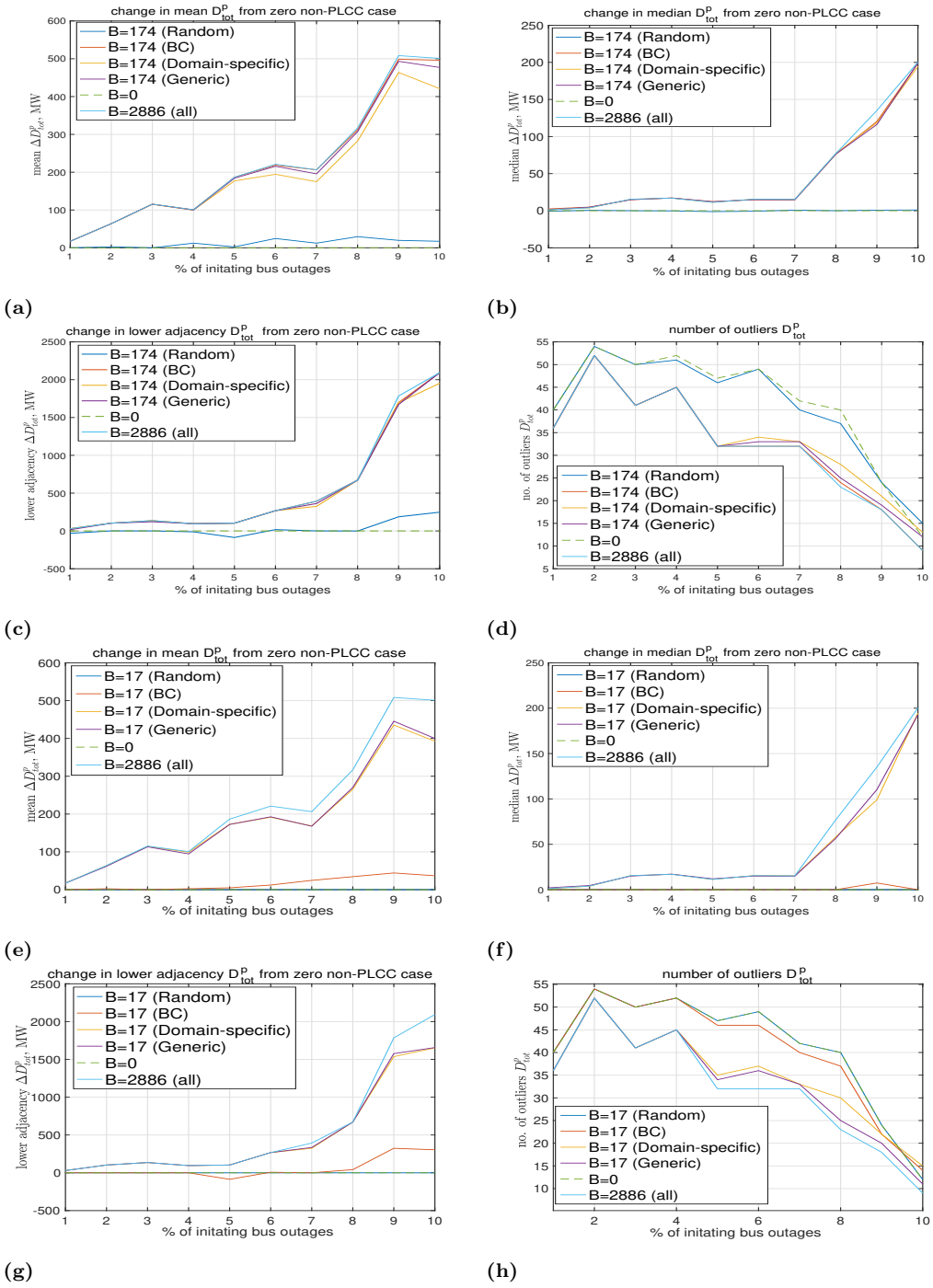


Figure 4.7: Performance evaluation of change in D^p_{tot} with respect to 100% PLCC case (non-PLCC links are placed by different methods under (a-d) $B = 174$, (e-h) $B = 17$ ($L = 200$ and $N = 50$ in the generic heuristic and $h = 10$ in the domain-specific heuristic).

proposed heuristics when the budget is sufficiently high (e.g., $B = 174$ as in Figs. 4.7 (a-d)). However, for a lower budget (e.g., $B = 17$ as in Figs. 4.7 (e-h)), the BC-based benchmark degrades severely, and the proposed heuristics perform much better. This demonstrates the importance of jointly considering the topology information and the power system information in selecting non-PLCC links, instead of solely based on the topology information as in the BC-based method. Moreover, with suitably tuned parameters ($L = 200$, $N = 50$), the generic heuristic can slightly outperform the domain-specific heuristic ($h = 10$), justifying the choice of objective function in (4.3).

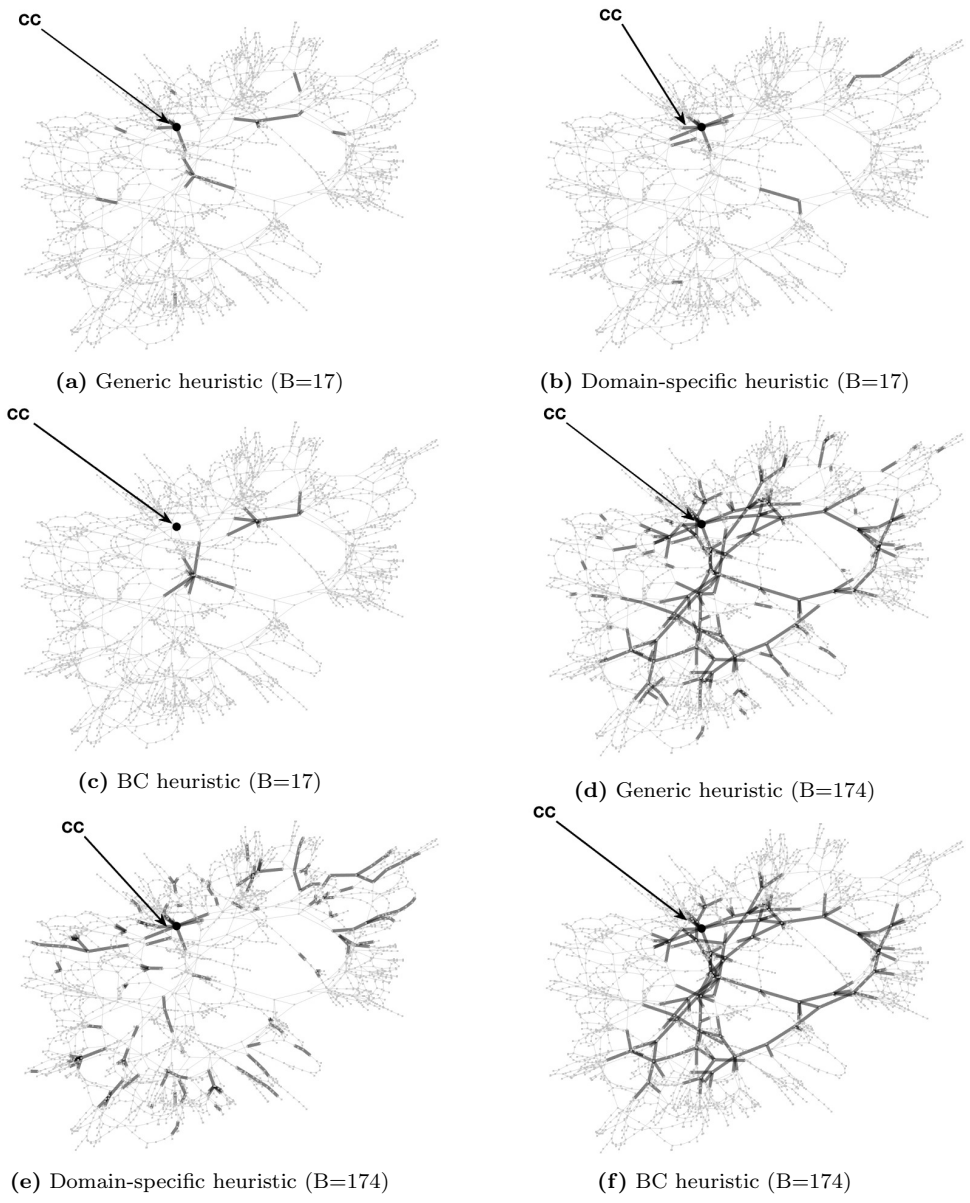
However, we note that the domain-specific heuristic runs significantly faster than the generic heuristic (with an average running time of 32.5 s compared to 2441 s). Furthermore, it can be seen from Fig. 4.7(f) that using as few as 17 non-PLCC links, the proposed heuristics can serve a median post-contingency demand that is close to the ideal case where all the $2,886$ links are non-PLCC. On the other hand, the performance deteriorates significantly with simplistic placement of these non-PLCC links (e.g., 'Random', 'BC'). This result signals the importance of placing non-PLCC links properly when under a budget constraint.

We also see from Fig. 4.7(d,h) that the comparison in terms of the number of outliers (extremely rare cases) is in line with the comparison in terms of the other statistics: a design leading to statistically higher demand served also has fewer outliers. We note that the upper adjacency, the largest data point that is not an outlier in the plot which is 3rd quartile + 1.5×inter-quartile range, demonstrates insignificant variations among different designs, which are not shown here since they represent outages that are non-critical.

Fig. 4.8 shows the communication layer of the system under different designs with a budget of 17 and 174, respectively, before imposing any failure. The plots visually describe the different design principles followed by each heuristic: the BC-based heuristic (Fig. 4.8(c,f)) builds a "backbone" of non-PLCC links, which works well when there is sufficient budget but poorly when the budget is highly limited; in contrast, the proposed heuristics strategically place non-PLCC links at the weakest parts of the network and leverage PLCC links when there is sufficient connectivity.

4.5 Conclusion

We study the impact of coupling between the communication and the power networks as it affects a SCADA-based preventive control system that leverages power line carrier communication (PLCC) to reduce the cost of deploying the communication network. As the failure of a power transmission line will fail the piggybacked PLCC link, we focus on improving the robustness of such a control system against cascading failures by allocating a limited number of non-PLCC links that are immune to power grid failures, which is formulated as a nonlinear integer programming problem. We establish the NP-hardness of the optimal solution and propose two heuristics that achieve different tradeoffs between the computational efficiency and the efficacy in mitigating cascades. Our evaluations based on a $2,383$-bus Polish network demonstrate the promising result that a control system using only a few strategically-placed non-PLCC links and PLCC links elsewhere can achieve almost the same efficacy in mitigating cascading failures as a much more expensive control system that only employs non-PLCC links.



(a) Generic heuristic (B=17)

(b) Domain-specific heuristic (B=17)

(c) BC heuristic (B=17)

(d) Generic heuristic (B=174)

(e) Domain-specific heuristic (B=174)

(f) BC heuristic (B=174)

Figure 4.8: Graphs showing the non-PLCC links selected by different heuristics. Dark edges: non-PLCC, light edges: PLCC, CC: control center.

Chapter 5

Improvement of SCADA-based Preventive Control Under Budget Constraints

Catastrophic disasters in real-world systems, such as large-scale blackouts in power grids, are usually triggered by minor incidents, which culminate in a complex cascading failure in an interdependent system. Because the loss of a power transmission line disrupts the control information piggybacked on the line, failures in the power network may consequently disrupt monitoring and control of the system. Hence, reliable functioning of the communication network in support of monitoring and control is vital to ensure that the re-dispatch-based preventive control effectively restricts cascade propagation. In this chapter, we address this issue by proposing a novel scheme in designing the communication network comprised of both power line carrier communication (PLCC) links and non-PLCC (e.g., microwave) links in preparation of possible failures under the lack of knowledge of system topology and a budget constraint on the communication link deployment cost. First, we characterize the fundamental hardness of our problem. Next, we develop a solvable Mixed-integer linear programming (MILP)-based algorithm, which attains a constant-factor approximation under certain conditions. Finally, we show via simulations on the IEEE 118-bus system that the proposed algorithm achieves superior performance in terms of enabling more accurate topology estimation and more served demand in the face of cascades.

5.1 Introduction

As discussed in the previous chapter, cascading failure is the process in an interconnected system in which damage or loss in one part of the system triggers damage or loss throughout the system.

The efficient and secure operation of power systems relies heavily on the associated SCADA system. The advantage of PLCC is that it is less expensive than dedicated communication links (including wireless links) [35, 136], which makes it a vital component of the SCADA system. One major limitation of PLCC links is their unreliability due to power line failures. This is caused by the open circuit problem. Situations such as open switches or disconnected power lines caused by disruptive events in the physical grid will lead to the loss of communication in certain parts of the grid, which results in less observability and controllability by the CC. This in turn makes it more difficult for the CC to halt the propagation of failure in the physical grid, which can then cause more failures in the PLCC links and hence further loss of observability and controllability. To break this harmful cycle without incurring exorbitant costs, it is crucial to judiciously combine PLCC and more reliable (and expensive) non-PLCC links in constructing the SCADA system, which is the goal of this chapter.

5.1.1 Background & Contribution

Communication links – PLCC vs. non-PLCC: PLCC links provide a means of supporting the SCADA system at a lower cost than using dedicated communication links, but PLCC has its own drawbacks. Power lines are primarily designed to be a transmission medium for electrical energy; hence, they may not be as suitable as data network media in terms of reliability, controllability, and security for data communication applications. Due to the complexity of transmitting data reliably over power lines, such systems also incur a nontrivial cost, albeit typically less than installing dedicated physically-separate communication media.

Due to the transmission properties of power lines, transmitted signals tend to attenuate over distance. Noise is added to the system by various loads and switching devices. PLCC links also radiate signals so the PLCC link itself has to meet current EMC (Electromagnetic compatibility) regulation limits to control radio signal interference [137]. To overcome the losses resulting

from physical characteristics of power lines, repeaters are used to transmit signals over long distances, which re-amplify and re-package the signal to restore its previous signal level [138]. PLCC technology is being deployed mainly in the MV (medium voltage) distribution substations that have more suitable communication channels due to lower attenuation, time-invariant behavior, simplified network configurations without any branches, and fewer noise sources [139]. In MV topologies, the positions of repeaters are fixed at distribution substations [139].

Since power lines are already deployed in power grid, deployment costs of PLCC links are largely confined to connecting repeaters and modems to the existing electrical grids. In this work, for the sake of simplicity, we consider the average cost of placing PLCC links per meter as calculated in [35]. For non-PLCC links, fiber, copper, and microwave have been considered. Among these non-PLCC technologies, microwave provides the best flexibility and cost savings [35].

Admittance matrix for preventive control: The CC requires knowledge of the admittance matrix of the power grid to run the preventive control. In a power system with N buses, where each bus is connected to the other buses through transmission lines, an $N \times N$ admittance matrix (a.k.a. **B**-matrix) describes the nodal admittances of the various buses. Admittance, expressed in the unit "mho", is the reciprocal of impedance, which is a complex number that measures how easily an element will allow a current to flow. Note that the impedance of a circuit element is the ratio of the phasor voltage across the element to the phasor current through the element [140]. Most prior works in the domain of coupled power and communication networks assume constant availability of the **B**-matrix [70]. In practice, the **B**-matrix is completely known before the cascade but only partially known as the cascade propagates. This is because some buses may lose their connectivity to the CC if the power lines that fail are used to implement PLCC links, the failure of which causes these buses and their incident lines to become unobservable to the CC.

This chapter presents a procedure to design the control communication network of a power grid under PLCC as well as non-PLCC links budget constraint, using an objective that represents the expected observability/controllability of the grid (to the CC) after initial failure. Our empirical evaluation shows that the proposed design effectively improves the accuracy of **B**-matrix estimation as well as the demand served after cascade.

The previous chapter assumes that all lines are equipped with PLCC

links. It proposes algorithms to select a set of PLCC lines accompanied by non-PLCC links with a budget constraint on such links. Moreover, it deals with an idealistic assumption that all lines' breaker status, i.e., the **B**-matrix, are known. Therefore, the resource allocation focus is solely on maximizing the controllability of generators and loads, given the budget constraint of non-PLCC links. However, this chapter examines the DC-QSS model of cascading failures in a power grid coupled with a SCADA-based communication network, while:

1. The ideological assumption of the known **B**-matrix is relaxed, so we rely solely on the status of observable breakers. The new assumption poses new challenges, e.g., the preventive control requires estimating the **B**-matrix. It also realistically captures the impact of the loss of observations due to communication links failing because without full sensor readings, the **B**-matrix cannot be fully reconstructed. The estimation of the admittance matrix can be divided into two parts; first the islands formed after failures must be detected, and second the connectivity of unobservable lines/nodes within islands must be estimated. The effectiveness of the preventive control depends upon the accuracy of both.

2. While ensuring observability of all nodes at the CC before cascade, we consider the total cost of all communication technologies, including both PLCC and non-PLCC link types. The reason is that deploying PLCC links also incurs a cost, and it is not realistic to assume all links are PLCC.

Optimization for resource allocation: There are several well-known techniques for solving resource allocation problems. The most widely used optimization technique is linear programming (LP) [38]. However, many problems of interest cannot be exactly formulated as LP, in which cases researchers often attempt to approximate the non-linear constraints/objective function of their problem by linear functions and then solve the modified problem. Mixed-integer linear programming (MILP) is a generalization of LP such that some of the variables are constrained to be integers, while other variables are allowed to be non-integers. MILP is often used to solve large and complex optimization problems, as it balances imprecision from linearization while taking advantage of the well-defined global optima and

efficient commercial-grade solvers that are generally unavailable for mixed-integer non-linear programming formulations [141, 142, 143].

This chapter develops an approximation solution to the proposed resource allocation problem by relaxing the original non-linear programming problem into a MILP (see Section 5.4).

Our main contributions of this chapter are as follows:

- In designing the communication network, we intend to mix the cost efficiency of PLCC links with the reliability of non-PLCC links. The optimization problem is formulated as maximizing an objective capturing the controllability of nodes in the grid, which is weighted by the node's importance in system topology and its generation/load contribution. The solution is a set of communication links, either PLCC or non-PLCC, in a coupled power grid with a geographically co-located SCADA-based communication network, albeit with a budget constraint limiting the number and type of links (either PLCC or non-PLCC). We formulate the underlying optimization as non-linear programming (NLP).

- By analyzing the complexity of our NLP problem, we prove that it is generally NP-hard. We derive a constant-factor approximation guarantee for the proposed formulation, computable by mixed-integer linear programming (MILP).

- We perform extensive evaluations on IEEE 118-bus power system. The proposed algorithm consistently outperforms baselines while it effectively (i) estimates system topology in the absence of fully known **B**-matrix, (ii) increases the demand served after cascade.

5.1.2 Research Gap and Challenges

Table 5.1 summarizes literature on cascading failure in the power grid in presence/absence of control network, indicating the interaction between the power grid and control network as well as the authors' assumption about the type of communication links. Real-world examples [151, 152, 65] emphasize the significance of understanding the interdependency between the coupled systems of the power grid and communication network to prevent cascading failures from causing massive blackouts. The majority of the existing works

Literature	Dependency	Comm. link
[69, 144, 145]	none (standalone grid)	-
[146, 147, 148]	one-way (control $\rightarrow$ grid)	non-PLCC
[70, 83, 64, 84, 149, 150]	two-way (control $\leftrightarrow$ grid)	non-PLCC
[127]	two-way (control $\leftrightarrow$ grid)	PLCC & non-PLCC

Table 5.1: Literature on cascading failure in power grid in presence/absence of control network

focus on the interdependency between the control network and the power system under the simplistic assumption of secure and full connectivity between all the nodes and the control center via non-PLCC links such as fiber optic links. The important questions of "*how to build that secure connectivity*" and "*what will happen in the absence of the secure and full connectivity*" of the control network have remained unanswered. Our work addresses this gap by characterizing the impact of having a realistic control network coupled to the power grid on the algorithms for detecting and stopping cascades, and optimizing the tradeoff between the reliability and the cost of constructing such a control network.

To the best of our knowledge, we are the first to study the design of control networks comprised of PLCC and non-PLCC links while considering the communication needs during cascade mitigation. The main challenges in solving this problem are: (i) the effect of control network design on the propagation of cascade is highly non-linear and non-explicit, and (ii) the solution space for the design is discrete and can be very large for large grids. In this chapter, we present an optimization-based approach to tackle both challenges that is shown to effectively support control algorithms in topology estimation and preventive control in the face of cascading failure.

5.2 System Model & Motivation

This chapter studies cascading failure in a coupled power grid with a geographically co-located SCADA-based communication network. In summary, a Control Center (CC) gathers data from sensors/actuators and dispatches

preventive control commands to generators and loads. This section provides specifics of the cascading failure and preventive control model of the power grid.

5.2.1 System Model

Here, we model the power grid as a undirected graph $G = (V, E)$ with no self-loops or multiple edges, where the V and E are the buses and the transmission lines, respectively. We also consider a geographically co-located SCADA-based communication network, albeit with a budget constraint limiting the number and type of links. Refer to Fig. 4.1, which shows the interaction between the power system and the communication network during cascade. After applying initial disturbance-based outages, we detect islands in the power grid and balance the generation and load in each of them. The balance is obtained either by curtailing the generation across all generators or reducing the load over all load nodes uniformly. A complete blackout in an island results from the lack of at least one active generator within the island.

If the preventive controller is activated, all the known power grid information, including line flows, the status of breakers, power output of generators and consumption at load centers, are communicated through communication links, either by power lines (PLCC links) or dedicated communication infrastructures (non-PLCC links). Once all the relevant information is passed on to the preventive controller, the CC tries to reduce line overloading by issuing control commands, which are communicated back to the power grid.

5.2.2 Preventive Control & Topology Estimation

To prevent cascade propagation, centralized generation redispatch and load shedding can be deployed using the control network. The admittance matrix (**B**-matrix) provides the topology of the power grid, as well as the information needed for the load/generation reduction or the power flow study of buses. Although the complete system information is available before the outage, it is only partially known as the cascade proceeds. Due to interdependency between cyber and physical layers of the grid, failure in the power grid during a cascade leads to outages in the communication network, which gradually decreases the observable areas. Hence, in the absence of a fully known admittance matrix, we need to estimate it. In the following, we focus on the

preventive control and estimation of the power grid admittance matrix ($\hat{\mathbf{B}}$) in detail.

5.2.2.1 Example Application 1 - Preventive Control

The CC houses the preventive controller in the geographically co-located SCADA-based communication network and by running the optimization problem (4.1) in previous chapter at regular intervals, minimizes the sum of total amount of load shedding and weighted overloads in the power grid.

5.2.2.2 Example Application 2 - Topology Estimation

The admittance matrix is of critical importance because it is used to analyze the data needed in the load/generation reduction or the power flow study of buses; for example, breaker statuses indicate tripped lines in the power system. In summary, the **B**-matrix explains the admittance and the topology of the power system. In the absence of a fully known **B**-matrix, formulation (4.1) takes $\hat{\mathbf{B}}$ (the estimated admittance matrix of the power grid), proposed in [1] as input, such that (4.1b) is replaced with

$$\mathbf{\Delta P}_G - \mathbf{\Delta P}_L = \hat{\mathbf{B}}\mathbf{\Delta}\theta, \quad \Delta\theta_i = 0, \forall i \in \Omega_{ref} \tag{5.1a}$$

As the cascade propagates, multiple line outages may happen in sequence, which leads to the formation of many islands within the power grid. Authors in [1] first identify the buses forming the connected components and then detect further line outages within the individual islands. Their topology identification algorithm uses power system measurements and observable breaker statuses. Also, their proposed estimation is verified at every step of the cascading failure. As the estimation methodology of the admittance matrix is outside the scope of this chapter, we skip the detailed explanation of it and refer the audience to [1]. We use this method of estimating the admittance matrix in preventive control of cascading failure, in a closed-loop fashion.

The principle of energy conservation in each island requires the existence of at least one active generator to produce power and at least one non-zero load to consume the power. The island will be dead if there is no active generator or non-zero load within it. As this chapter focuses on the re-dispatch control sub-problem, it necessitates an accurate island estimation. To that

end, at least one generator and one load must be observable within that particular island. The results in [1] show that the identification is accurate in more than 95% of cases when at least 50% of nodes are observable in each island (including at least one generator and one load bus). Hence, for the purpose of preventive control, we focus on ensuring observability of generator and load nodes from the CC's points of view that can provide these two objectives: (i) observing at least one active generator and one non-zero load in each island and (ii) observing at least 50% of nodes in each island. Satisfying these objectives helps identify the islands, followed by identifying the $\hat{\mathbf{B}}$ within each island.

These two focus areas are complementary, but neither is solely enough to ensure maximum served demand. The former approaches the cascade prevention purely from a controllability standpoint, while the latter considers the sub-problem from the observability side. For the sake of simplicity, we refer to islands consisting of at least one active generator and one non-zero load as valuable islands. Highly observable islands are valuable islands with at least 50% nodes observable by the CC.

5.2.3 Motivating Experiment

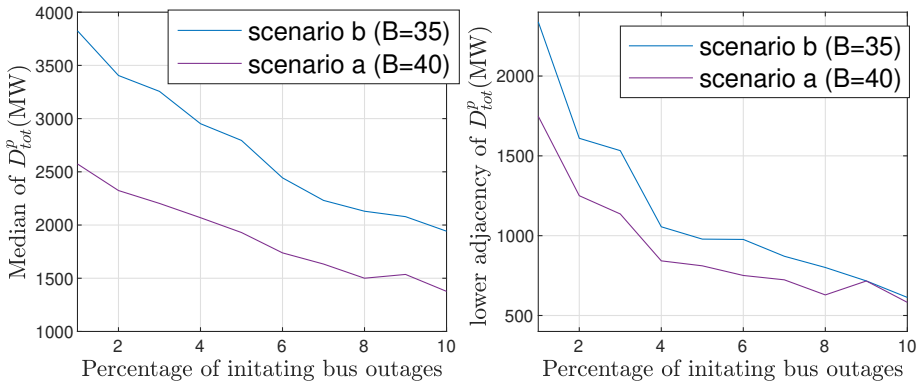


Figure 5.1: Performance evaluation of scenarios (a) randomly selecting PLCC links under $\mathcal{B} = 40$ and (b) mixture of suitably placed PLCC and non-PLCC links under $\mathcal{B} = 35$, in terms of median and lower adjacency of D^p_{tot}.

To understand the potential value of preventive control in mitigating cascading failures, we evaluate the performance of a network with preventive

control. Considering the IEEE 118-bus system[153] as a test system, which includes 118 buses, 186 branches, and 54 generators. We randomly fail 5% of the buses in the power grid and calculate the served power after cascade propagation in scenarios (i) having no communication network versus (ii) a network of 100%-PLCC links and (iii) a network of 100%-non-PLCC links. In scenarios (ii) and (iii), the communication networks are geographically co-located with the power grids and have the same topology as power grids. Assuming each PLCC link costs 0.3 and non-PLCC costs 0.6, the total cost for the communication network is $\mathcal{B} = 55.8$ and $\mathcal{B} = 111.6$ for scenarios (ii) and (iii), respectively. In this experiment, 100 random cases were tested. Results show that the mean of residual power after cascade propagation are 2041.3 MW (47.57% of the initial power), 2492.1 MW (58.08% of the initial power), and 3038.3 MW (70.81% of the initial power) for scenarios (i), (ii), and (iii), respectively. As it was expected superior performance is achieved with a pure non-PLCC communication network, albeit at a higher cost.

In the next example, we show that by strategically-placing non-PLCC and PLCC links, we achieve better efficacy in mitigating cascading failures compared with a more expensive communication network that only employs PLCC links. We describe the algorithms we use to determine this placement in the remainder of the chapter; here we are just summarizing the large

Table 5.2: Percentage of (i) valuable and (ii) highly observable islands in different scenarios (a) randomly selecting PLCC links under $\mathcal{B} = 40$ and (b) mixture of suitably placed PLCC and non-PLCC links under $\mathcal{B} = 35$.

	scenario (a)	scenario (b)
(i)	16.38	32.97
(ii)	3.01	26.34

Table 5.3: Percentage of estimation accuracy in different scenarios (a) randomly selecting PLCC links under $\mathcal{B} = 40$ and (b) mixture of suitably placed PLCC and non-PLCC links under $\mathcal{B} = 35$ for (i) valuable, (ii) highly observable and (iii) slightly observable islands.

	scenario (a)	scenario (b)
(i)	39.54	87.48
(ii)	51.67	93.55
(iii)	33.11	52.20

benefit of a good solution. We randomly fail 1%-10% of the buses in the power grid and calculate the served power after cascade propagation in scenarios (a) having a network of randomly placed PLCC links under budget $\mathcal{B} = 40$ versus (b) communication network that is mixture of appropriately placed PLCC and non-PLCC links under budget $\mathcal{B} = 35$. In this experiment, 100 random cases were tested. Results in Fig.5.1 show superior performance in the latter scenario, although its budget is tighter.

In support of topology estimation and the accuracy of the $\hat{\mathbf{B}}$ (the estimated admittance matrix of the power grid), Table 5.2 and Table 5.3 demonstrate the importance of installing the appropriate type of links in the communication network. Table 5.2 compares the percentage of valuable and highly observable islands, where valuable islands consist of at least one active generator and one non-zero load. Highly observable islands are the valuable islands with at least 50% nodes observable by the CC, otherwise they are considered as slightly observable islands (the valuable islands with less than 50% nodes observable by the CC). Furthermore, Table 5.3 checks the percentage of correctly identified islands.

These examples demonstrate the potential value of preventive control and the role of types of links in mitigating cascading failures, jointly considering the budget.

5.3 Problem Formulation

Without a properly designed communication network, a SCADA-based system cannot work adequately. All supervisory control and data acquisition aspects of the SCADA system rely entirely on the communication system to provide a conduit for data flow.

This chapter's main improvement in providing sufficient observability and controllability for cascade prevention is observing and mitigating cascades in valuable islands. Also, to guarantee the CC's communication with these islands, it is crucial to correctly place PLCC and non-PLCC links, as failure in the power line fails the PLCC link piggybacked on the line. Moreover, we assume all the communication nodes, i.e., sensors, actuators, and relays, have battery backups and are immune from power grid failures.

Table 5.4: Table of notations

Notation	meaning
$\mathbf{B}$	admittance matrix
$\hat{\mathbf{B}}$	estimated admittance matrix
V	set of nodes in the power grid
E	set of links in the power grid
$\mathcal{C}$	set of communication link platforms
w_c	cost of communication link type c
s_e^c	binary variable shows type c of communication link e
f_{ij}^k	flow of commodity k from node i to node j
$\mathcal{B}$	communication links budget
P_i	the set of all possible paths between the CC and node i
γ_i	the importance of observing/controlling node i by the CC
p_c	reliability of communication link type c

5.3.1 Underlying Optimization Problem

We formulate the problem as link placement optimization in the communication network, considering both PLCC and non-PLCC links, albeit under a budget of $\mathcal{B}$ to capture the resource constraints. In this way, we combine the cost efficiency of PLCC links and the robustness against cascades of non-PLCC links. Although cascade prevention aims to maximize the total demand served after the cascade, this objective function is not an explicit function of link placement. To address this challenge, we propose using an objective function as follows, which is the same objective function in previous chapter.

Without loss of generality, we assume that the given power grid topology $G = (V, E)$ is a undirected graph with no self-loops or multiple edges. Let $P_i,\ \forall i \in V$ denote the set of all possible paths between the CC and node i. By modeling each path $m \in P_i$ as a set of traversing links, the total reliability of the network is defined as follows (assuming nodes never fail), where ρ_e denotes the reliability of link e.

$$\sum_{i \in V} \left(1 - \prod_{m \in P_i} (1 - \prod_{e \in m} \rho_e) \right), \tag{5.2}$$

Let the binary variables $s_e^c \in \{0, 1\}$ indicate selection of link e by type $c \in \mathcal{C}$, and p_c denote the reliability of type c link (PLCC or non-PLCC).

Reliability of link e is defined as $\rho_e := \sum_{c\in\mathcal{C}} s_e^c p_c$. Thus, the link placement problem (5.3) maximizes the total reliability of the network under a given budget of PLCC and non-PLCC links, while ensuring each node is observable by the CC. The principal used notations are described in Table 5.4.

$$\max \sum_{i\in V} \gamma_i \left(1 - \prod_{m\in P_i} (1 - \prod_{e\in m} \sum_{c\in\mathcal{C}} s_e^c p_c)\right) \tag{5.3a}$$

$$\text{s.t.} \quad \sum_{e\in E} \sum_{c\in\mathcal{C}} s_e^c . w_c \leq \mathcal{B} \tag{5.3b}$$

$$\sum_{c\in\mathcal{C}} s_e^c \leq 1, \forall e \in E \tag{5.3c}$$

$$\sum_{j\in V} f_{ij}^k - \sum_{j\in V} f_{ji}^k = \begin{cases} 1 & i = CC \\ -1 & i = k \\ 0 & \text{otherwise} \end{cases} \quad \forall i \in V, \forall k \in V \setminus \{CC\} \tag{5.3d}$$

$$f_{ij}^k \leq \sum_{c\in\mathcal{C}} s_e^l, \forall e = \{(i,j) \vee (j,i)\} \in E, k \in V \setminus \{CC\} \tag{5.3e}$$

$$f_{ij}^k \geq 0, \forall e = \{(i,j) \vee (j,i)\} \in E, k \in V \setminus \{CC\} \tag{5.3f}$$

$$s_e^c \in \{0,1\}, \forall e \in E, \forall c \in \mathcal{C} \tag{5.3g}$$

Here, binary variables $s_e^c \in \{0,1\}$ indicate the selection of type c for link e and the variable f_{ij}^k for each arc (i,j) symbolizes the flow of commodity k from node i to node j. Briefly, the Non-Linear Programming (NLP) (5.3) aims at selecting PLCC and non-PLCC links up to budget $\mathcal{B}$ to maximize the expected total weight of all the nodes remaining connected to the CC after the initial failure. Intuitively, the more nodes connected to the CC, the better the CC observes and controls the grid, and hence mitigates the failure cascading properly. However, not all the nodes are equally important; therefore, we use the weight γ_i to consider the importance of observing and controlling node i. Constraints (5.3b) and (5.3c) make sure that the budget is not exceeded, and for each link at most one type of communication link is selected, respectively. Constraints (5.3d)-(5.3f) guaranty the observability of each node by the CC before any failure occurs, which is based on the link constrained Steiner tree problem in undirected graphs, described in the following.

Definition 5.3.1 ([154]). *Given the undirected graph $G(V, E)$ with non-negative edge cost, Link Constrained Steiner Tree problem of V determines the Steiner tree $T = (V, E(T))$ rooted at a source node with minimum cost and such that the number of edges $E(T) \in E$ is less than or equal to a given threshold.*

Graph construction: Given the undirected graph $G(V, E)$, construct the bi-directed graph $\mathcal{H} =(V, \mathcal{A})$ such that an edge $e \in E$ incident to nodes i and j, is replaced by two arcs, that is, $\{(i, j), (j, i)\} \in \mathcal{A}$. Considering a commodity k for each node $V \setminus \{CC\}$, the variable f_{ij}^k for each arc $(i, j) \in \mathcal{A}$, represents the flow of commodity k from node i to node j.

Claim: The solution of link constraint Steiner tree problem on graph $\mathcal{H} =(V, \mathcal{A})$ provides a solution to the constraints (5.3d)-(5.3f) on graph $G = (V, E)$.

Proof of the claim: The Steiner tree $T = (V, E(T))$ of graph $\mathcal{H}$ obtains a Steiner arborescence rooted at node CC, containing a directed path from node CC to every other terminal node in $V \setminus \{\text{CC}\}$. The variable s_e^c takes value equal to 1 if corresponding edge $e \in E(T)$ and 0 otherwise.

5.3.2 Complexity analysis

Consider the special case that the costs of different communication links are $\{1, 0\}$ for non-PLCC and PLCC links, respectively. Since PLCC costs nothing, without loss of generality we can assume that $\sum_{c\in\mathcal{C}} s_e^c = 1$ for every link e, thus constraint (5.3c) is unnecessary. Moreover, constraints (5.3d)-(5.3f) are no longer needed as there is a communication path between the CC and every other node (assuming the power grid topology is connected before failure through PLCC links). Thus, the NLP formulation (5.3) changes to (5.4) by keeping equations (5.3a), (5.3b) and (5.3g).

$$\max \sum_{i\in V} \gamma_i \left(1 - \prod_{m\in P_i} (1 - \prod_{e\in m} \sum_{c\in\mathcal{C}} s_e^c p_c)\right) \tag{5.4a}$$

$$\text{s.t.} \sum_{e\in E} \sum_{c\in\mathcal{C}} s_e^c \leq \mathcal{B} \tag{5.4b}$$

$$s_e^c \in \{0, 1\} \qquad \forall e \in E, \forall c \in \mathcal{C} \tag{5.4c}$$

Theorem 5.3.1. *Problem* (5.4) *is NP-hard.*

Proof. Formulation (5.3) is a generalization of the NP-hard optimization in Chapter 4. □

NP-hardness of the special case (5.4), proves (5.3) is NP-hard too.

5.4 Approximation Algorithm

In the previous section, we proposed the link placement problem (5.3), which maximizes the total reliability of the network from the CC's point of view, based on jointly considering the topology and power system information metrics. In this section, we explore an efficient solvable solution, which derives a constant-factor approximation algorithm for the formulation (5.3) that is computable by MILP.

Optimization link placement problem (5.3) aims at maximizing (5.3a), which is the root of non-linearity. Let $p_{\min} := \min_{c\in\mathcal{C}} p_c$ and $p_{\max} := \max_{c\in\mathcal{C}} p_c$ denote the minimum and maximum reliability per link, where p_c shows the reliability of link type c.

Assume $V = V_1 \cup V_2 \cup V \setminus \{V_1 \cup V_2\}$ such that $V_1 := \{i \in V : \sum_{m\in P_i} \prod_{l\in m} p_{max} \leq 1\}$ and $V_2 := \{i \in V : \prod_{l\in m_{2,\max}} p_{min} \geq 1\}$ can be any subsets of V, where $|m_{2,\max}|$ is the maximum hop count per path over all the paths in $\bigcup_{i\in V_2} P_i$. For the sake of simplicity, let $A_l = \sum_{c\in\mathcal{C}} s_l^c p_c$. Hence, the objective function (5.3a) changes to (5.5):

$$\begin{aligned} \max \sum_{i\in V_1} \gamma_i \left(1 - \prod_{m\in P_i} (1 - \prod_{l\in m} A_l)\right) + \\ \sum_{i\in V_2} \gamma_i \left(1 - \prod_{m\in P_i} (1 - \prod_{l\in m} A_l)\right) + \\ \sum_{i\in V\setminus\{V_1\cup V_2\}} \gamma_i \left(1 - \prod_{m\in P_i} (1 - \prod_{l\in m} A_l)\right) \end{aligned} \tag{5.5}$$

Theorem 5.4.1. *The optimal solution to* (5.6) *yields a* $(1-\epsilon)$*-approximation for* (5.3) *under the condition of* $p_{min}^{|m_{3,\max}|} \geq p_{max}^{|m_{3,\min}|} (1 - \frac{1}{e})|P_{max}|$*, where*

$\epsilon := (e-(e-1)(1-p_{\min})|m_{1,\max}|)^{-1}$ *such that* $|m_{1,\max}|$ *denotes the maximum hop count per path over all the paths in* $\bigcup_{i\in V_1} P_i$. *Moreover,* $|m_{3,\min}|$ *defines the minimum hop count per path over all the paths in* $\bigcup_{i\in V\setminus\{V_1\cup V_2\}} P_i$ *and* $|P_{max}| := \max_{i\in V\setminus\{V_1\cup V_2\}} |P_i|$.

$$\begin{aligned} &\max\ UB + UB' + UB" \\ &s.t.\quad (5.3b)-(5.3g), \end{aligned} \tag{5.6}$$

where UB, UB' *and* $UB"$ *are the upper-bounds of the first, the second, and the third terms of* (5.5), *respectively, defined as:*

$$UB := \sum_{i\in V_1}\gamma_i \sum_{m\in P_i}\left((1-(1-\frac{1}{e})\sum_{l\in m}(1-A_l)\right) \tag{5.7}$$

$$UB' := \sum_{i\in V_2}\gamma_i \tag{5.8}$$

$$UB" := \sum_{i\in V\setminus\{V_1\cup V_2\}}\gamma_i\left(1-\prod_{m\in P_i}(1-\prod_{l\in m}p_{max})\right) \tag{5.9}$$

PROOF OF THEOREM 5.4.1:

Proof.

Definition 5.4.1 ([155])**.** *Goemans-Williamson inequality gives the following bound* (5.10) *for any sequence of* $y_i \in [0,1],\ i \in \{1,...,n\}$*:*

$$(1-\frac{1}{e})\min\{1,\sum_{i=1}^{n} y_i\} \le 1-\prod_{i=1}^{n}(1-y_i) \le \min\{1,\sum_{i=1}^{n} y_i\} \tag{5.10}$$

By applying (5.10) to the first term in (5.5), for all nodes in V_1 we have

$$\begin{aligned} \sum_{i\in V_1}\gamma_i(1-\frac{1}{e})\min\{1,\sum_{m\in P_i}\prod_{l\in m}A_l\} &\le \sum_{i\in V_1}\gamma_i \\ \left(1-\prod_{m\in P_i}(1-\prod_{l\in m}A_l)\right) &\le \sum_{i\in V_1}\gamma_i\min\{1,\sum_{m\in P_i}\prod_{l\in m}A_l\}. \end{aligned} \tag{5.11}$$

As for $i \in V_1 : \sum_{m \in P_i} \prod_{l \in m} A_l \leq \sum_{m \in P_i} \prod_{l \in m} p_{max} \leq 1$, so $\min\{1, \sum_{m \in P_i} \prod_{l \in m} A_l\} = \sum_{m \in P_i} \prod_{l \in m} A_l$. Thus,

$$\begin{aligned} (1 - \frac{1}{e}) \sum_{i \in V_1} \gamma_i \sum_{m \in P_i} \prod_{l \in m} A_l \leq \sum_{i \in V_1} \gamma_i \times \\ \left(1 - \prod_{m \in P_i} (1 - \prod_{l \in m} A_l)\right) \leq \sum_{i \in V_1} \gamma_i \sum_{m \in P_i} \prod_{l \in m} A_l. \end{aligned} \tag{5.12}$$

Let $|m_{1,\max}|$ denotes the maximum hop count per path over all the paths in $\bigcup_{i \in V_1} P_i$. Define

$$\epsilon := \left(e - (e-1)(1 - p_{\min})|m_{1,\max}|\right)^{-1}. \tag{5.13}$$

If $(1 - p_{\min})|m_{1,\max}| < 1$, then $\sum_{l \in m}(1 - A_l) < 1$ for all $m \in \bigcup_{i \in V_1} P_i$, $\epsilon \in [\frac{1}{e}, 1)$, and

$$\epsilon \geq \left(e - (e-1) \sum_{l \in m} (1 - A_l)\right)^{-1}, \quad \forall m \in \bigcup_{i \in V_1} P_i. \tag{5.14}$$

By applying (5.10) to (5.12), we have

$$\begin{aligned} LB := (1 - \frac{1}{e}) \sum_{i \in V_1} \gamma_i \sum_{m \in P_i} \left(1 - \sum_{l \in m} (1 - A_l)\right) \leq \\ \sum_{i \in V_1} \gamma_i \left(1 - \prod_{m \in P_i} (1 - \prod_{l \in m} A_l)\right) \leq \\ \sum_{i \in V_1} \gamma_i \sum_{m \in P_i} \left((1 - (1 - \frac{1}{e}) \sum_{l \in m} (1 - A_l)\right) := UB. \end{aligned} \tag{5.15}$$

From (5.14), we get

$$\begin{aligned} (1 - \frac{1}{e}) \left(1 - \sum_{l \in m} (1 - A_l)\right) \\ \geq (1 - \epsilon) \left(1 - (1 - \frac{1}{e}) \sum_{l \in m} (1 - A_l)\right), \end{aligned} \tag{5.16}$$

which implies

$$LB \geq (1-\epsilon)\sum_{i\in V_1}\gamma_i \sum_{m\in P_i}\left(1-(1-\frac{1}{e})\sum_{l\in m}(1-A_l)\right) \\ = (1-\epsilon)UB. \tag{5.17}$$

Hence,

$$(1-\epsilon)UB \leq \sum_{i\in V_1}\gamma_i\left(1-\prod_{m\in P_i}(1-\prod_{l\in m}A_l)\right)\leq UB. \tag{5.18}$$

By applying (5.10) to the second term in (5.5), for all nodes in V_2 we have

$$\sum_{i\in V_2}\gamma_i(1-\frac{1}{e})\min\{1,\sum_{m\in P_i}\prod_{l\in m}A_l\}\leq\sum_{i\in V_2}\gamma_i\times \\ \left(1-\prod_{m\in P_i}(1-\prod_{l\in m}A_l)\right)\leq\sum_{i\in V_2}\gamma_i\min\{1,\sum_{m\in P_i}\prod_{l\in m}A_l\}. \tag{5.19}$$

As for all nodes in V_2 , $\sum_{m\in P_i}\prod_{l\in m}A_l \geq \prod_{l\in m_{2,\max}}p_{min}\geq 1$, so $\min\{1,\sum_{m\in P_i}\prod_{l\in m}A_l\}=1$. Thus,

$$(1-\frac{1}{e})\sum_{i\in V_2}\gamma_i\leq\sum_{i\in V_2}\gamma_i\left(1-\prod_{m\in P_i}(1-\prod_{l\in m}A_l)\right)\leq \\ \sum_{i\in V_2}\gamma_i := UB'. \tag{5.20}$$

From (5.14), $\frac{1}{e}\leq\epsilon$. Hence,

$$(1-\epsilon)UB'\leq\sum_{i\in V_2}\gamma_i\left(1-\prod_{m\in P_i}(1-\prod_{l\in m}A_l)\right)\leq UB'. \tag{5.21}$$

Considering the third term in (5.5), for all nodes in $V\setminus\{V_1\cup V_2\}$ we have (5.22), where $|m_{3,\max}|$ denotes the maximum hop count per path over all the paths in $\bigcup_{i\in V\setminus\{V_1\cup V_2\}}P_i$.

$$
\begin{aligned}
LB" := \sum_{i \in V \setminus \{V_1 \cup V_2\}} \gamma_i \prod_{l \in m_{3,\max}} p_{min} \leq \\
\sum_{i \in V \setminus \{V_1 \cup V_2\}} \gamma_i \left(1 - \prod_{m \in P_i} (1 - \prod_{l \in m} A_l)\right) \leq \\
\sum_{i \in V \setminus \{V_1 \cup V_2\}} \gamma_i \left(1 - \prod_{m \in P_i} (1 - \prod_{l \in m} p_{max})\right) := UB"
\end{aligned}
\tag{5.22}
$$

Special case: Consider the condition of $p_{min}^{|m_{3,\max}|} \geq p_{max}^{|m_{3,\min}|} \, (1 - \frac{1}{e})|P_{max}|$, where $|m_{3,\min}|$ defines the minimum hop count per path over all the paths in $\bigcup_{i \in V \setminus \{V_1 \cup V_2\}} P_i$ and $|P_{max}| := \max_{i \in V \setminus \{V_1 \cup V_2\}} |P_i|$.

In this special case, by applying binomial expansion, we get

$$
\begin{aligned}
UB" = \sum_{i \in V \setminus \{V_1 \cup V_2\}} \gamma_i \left(1 - \prod_{m \in P_i} (1 - \prod_{l \in m} p_{max})\right) \\
\leq \sum_{i \in V \setminus \{V_1 \cup V_2\}} \gamma_i \left(1 - (1 - \prod_{l \in m_{3,\min}} p_{max})^{|P_i|}\right).
\end{aligned}
\tag{5.23}
$$

So,

$$
\begin{aligned}
\sum_{i \in V \setminus \{V_1 \cup V_2\}} \gamma_i \left(1 - (1 - \prod_{l \in m_{3,\min}} p_{max})^{|P_i|}\right) < \\
\sum_{i \in V \setminus \{V_1 \cup V_2\}} \gamma_i \left(1 - (1 - |P_i| \prod_{l \in m_{3,\min}} p_{max})\right) = \\
\sum_{i \in V \setminus \{V_1 \cup V_2\}} \gamma_i \, |P_i| \, p_{max}^{|m_{3,\min}|} \leq (1 - \frac{1}{e})^{-1} \times \\
\sum_{i \in V \setminus \{V_1 \cup V_2\}} \gamma_i \, p_{min}^{|m_{3,\max}|} = (1 - \frac{1}{e})^{-1} \, LB".
\end{aligned}
\tag{5.24}
$$

From (5.14), $\frac{1}{e} \leq \epsilon$. Hence,

$$(1-\epsilon)UB" \leq \sum_{i \in V \setminus \{V_1 \cup V_2\}} \gamma_i \left(1 - \prod_{m \in P_i} (1 - \prod_{l \in m} A_l)\right) \leq UB". \tag{5.25}$$

Finally, by (5.18), (5.21) and (5.25), we have

$$(1-\epsilon)(UB + UB' + UB") \leq \text{ (5.3a)} \leq UB + UB' + UB". \tag{5.26}$$

□

5.5 Performance Evaluation

This section demonstrates the performance of the proposed approximation algorithm on an electric power system.

5.5.1 Benchmarks and metrics

To assess the performance of the proposed algorithm, we use the following benchmarks:

- *The approximation algorithm* (5.6), using a MILP solver (MATLAB `intlinprog`);
- *LP-relaxation with rounding*, which first solves the LP-relaxation of proposed approximation algorithm (5.6), and then rounds the link selection variables to $\{0, 1\}$, subject to constraints (5.3b)-(5.3c);
- $(1 + \varepsilon, 1)$-*approximation* algorithm of CMST, which is explained in part 5.5.1.1 in the following;
- $(1, 1 + \varepsilon)$-*approximation* algorithm of CMST, which is explained in part 5.5.1.1 in the following;
- *BC method*, which is explained in part 5.5.1.2 in the following;
- *Random*, which randomly selects links and their type until exhausting the budget or assuring the CC's connectivity to all nodes.

5.5.1.1 (α, β)-Approximation of CMST

The general NP-hardness of MILP approximation algorithm (5.6) motivates us to develop the following alternative solution by focusing solely on the graph-theoric metric of the network. In this method, we form a special case of the constrained minimum spanning tree (CMST) problem based on the power grid topology. Then, we apply an (α, β)-approximation of it, albeit with a budget constraint. Although this method differs from (5.3) in the objective function, we will show that it is polynomial-time solvable while achieving considerably better performance compared to random in terms of served loads and accuracy of $\hat{\mathbf{B}}$. Recalling that the solution of (5.3) is a spanning tree, the output of this method is also a spanning tree. This method reveals that having a connected communication network such that the CC is connected to all nodes, does not necessarily guarantee the best performance, as shown later in Fig. 5.7.

Let $\mathcal{H}' = (V, \mathcal{A}')$ be a undirected graph and W is a positive integer. Assume two different non-negative functions, weight and cost, associate on edges in $\mathcal{A}'$.

Definition 5.5.1 ([156])**.** *The constrained minimum spanning tree (CMST) problem on graph $\mathcal{H}'$ is to identify a minimum total cost spanning tree on graph $\mathcal{H}'$, such that the total weight is at most W.*

Definition 5.5.2 ([156])**.** *Given two positive real numbers α and β, an (α, β)-approximation algorithm for the CMST problem on graph $\mathcal{H}'$ is defined as a polynomial-time algorithm that returns a solution with the total weight at most α times the bound W, and the total cost at most β times the total cost of the optimal solution for the CMST problem.*

Graph construction: Given the power grid topology $G(V, E)$, construct graph $\mathcal{H}' = (V, \mathcal{A}')$, such that an edge $e \in E$ is replaced by two edges with associated weights equal implementation cost of PLCC and non-PLCC link. Assign each edge in $\mathcal{A}'$ the cost equals $1 - p_c$, where p_c is the reliability of edge e with type c.

Claim: Assume W equals $\mathcal{B}$ (the budget of communication links), and $\varepsilon > 0$ is a positive constant. Then, $(1, 1+\varepsilon)$-approximation algorithm [156] and $(1+\varepsilon, 1)$-approximation algorithm [157] of the CMST problem on graph $\mathcal{H}'$, provide $(1, 1+\varepsilon)$ and $(1+\varepsilon, 1)$-approximation solutions of problem (5.3)

on graph G, respectively.

Proof of the claim: The approximation solution $T' = (V, E(T'))$ on graph $\mathcal{H}' = (V, \mathcal{A}')$ obtains a spanning tree rooted at node CC, containing a directed path from node CC to every other node in $V \setminus \{\text{CC}\}$. The variable s_e^c takes value equal to 1 if corresponding edge $e \in E(T')$ and 0 otherwise.

Complexity: Given undirected graph $\mathcal{H}' = (V, \mathcal{A}')$, the time complexity of two polynomial time $(1, 1+\varepsilon)$-approximation and $(1+\varepsilon, 1)$-approximation algorithms of CMST on graph $\mathcal{H}'$ are $O((|V| \log\log |V| + |V|^{1/((1+\varepsilon)^\zeta - 1)} \log\log(1+\varepsilon))(|\mathcal{A}'| \log^2 |V| + |V| \log^3 |V|))$ [156], and $O(|V|^{O(\frac{1}{\varepsilon})}(|\mathcal{A}'| \log^2 |V| + |V| \log^3 |V|))$ [157] for any constant $\varepsilon > 0$, respectively. $\zeta > 0$ is a constant. As we construct graph $\mathcal{H}' = (V, \mathcal{A}')$ from the power grid topology $G(V, E)$, hence $|\mathcal{A}'| = 2|E|$.

5.5.1.2 BC method

Given the power grid topology $G(V, E)$, this method sequentially considers each link $e \in E$ and computes the value of BC_e equal to the maximum betweenness centrality of its two endpoints. The betweenness centrality of a node is the frequency that it appears on the shortest paths between all pairs of nodes in the graph [130]. Then, in the descending order of BC_e, it selects non-PLCC links until reaching 5% of the total budget (the proposed approximation algorithm (5.6) utilizing almost the same non-PLCC links), and PLCC links until exhausting the budget or assuring the CC's connectivity with all other nodes.

Remark: The BC method gives priority to links incident to nodes with higher betweenness centrality. Intuitively, these nodes have substantial influence over the information passing between other nodes. Thus, the goal is to maintain as many paths from these elements in the power grid to the CC.

To demonstrate the effectiveness of the proposed algorithm, we use two metrics supporting different applications: the ratio of valuable and highly observable islands in support of topology estimation, and the statistical measurements of load served after failure cascade in support of preventive control. Recall that valuable islands are islands consisting of at least one active generator and one non-zero load, and highly observable islands are the valu-

able islands with at least 50% nodes observable by the CC. In the former metric, the total numbers of formed islands are normalized by the 2-norm method [158]. For the latter metric, we show the lower adjacency and the median of the change in total post-contingency demand served (D^p_{tot}). The lower adjacency is the smallest data point that is not an outlier in the plot, which is 1st quartile - 1.5×inter-quartile range. D^p_{tot} before any failure is 4291 MW. Moreover, to examine the accuracy of the $\hat{\mathbf{B}}$, we check the percentage of correctly identified islands in some scenarios and eventually compare misses and false alarms in identifying the connected/tripped lines within the islands.

5.5.2 Simulation Setup

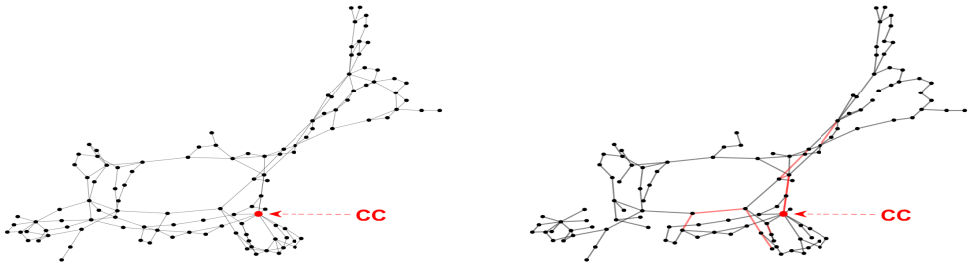
We evaluate the proposed solution on the IEEE 118-bus system[153], including 118 buses, 186 branches, and 54 generators. We study cases with initial bus outages varying from $1\% - 10\%$ of the total buses. For each case, 100 random sets of node outages have been considered, such that all result in cascade. The CC is situated on bus 49, one of the highest degree nodes in the power grid system. The preventive control optimization and the delay in line tripping are both set to 80 seconds.

It is difficult to get data on the implementation cost of different types of communication links. In general, fiber and then microwave links have the highest implementation cost, while PLCC links are the cheapest [159]. Authors in [35] and [136] estimate the implementation cost of PLCC, microwave, and fiber as \$123, \$241 and \$450 per meter, respectivey, where in this work, we normalize these costs. For the sake of simplicity, we consider all non-PLCC links as microwave, and set the cost of non-PLCC links twice the cost of PLCC links, which are 0.6 and 0.3, respectively. Note that this parameter setting implies that all links are assumed to have the same length, but our solution can be easily extended to incorporate heterogeneous lengths.

We set reliability, p_c, of PLCC and non-PLCC links to 0.99 and 0.9999 [135]. Therefore, equations (5.21) and (5.25) hold for special cases $V_2 = \{\emptyset\}$, $|P_{max}| = 1$ and $|m_{3,\max}| \leq 45$.

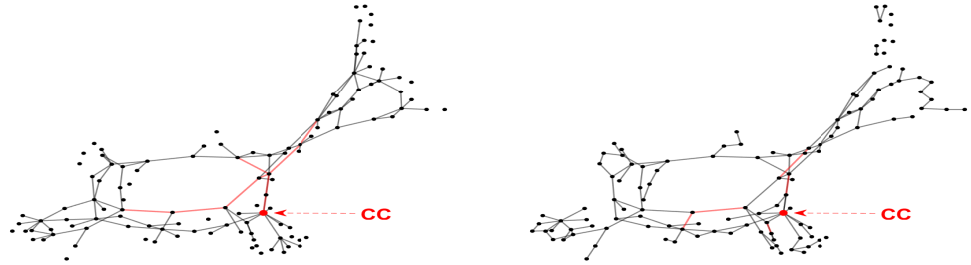
5.5.3 Results

To provide a general understanding of different algorithms, we depict design outcomes of the benchmarks discussed in part 5.5.1 in Fig. 5.2. Figure 5.2(a) shows the IEEE 118-bus power grid topology, and Figs. 5.2(b-g) offer the



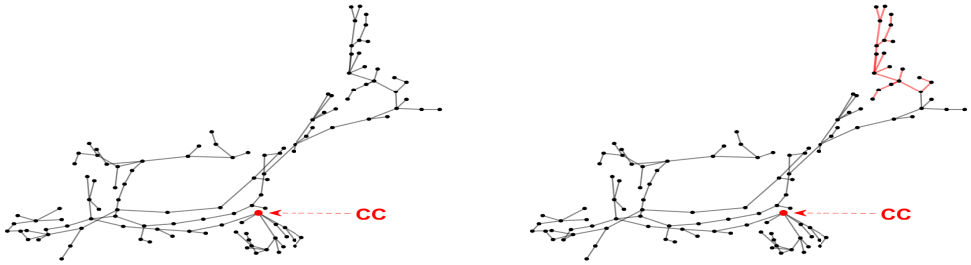
(a) Power grid topology

(b) Approximation method



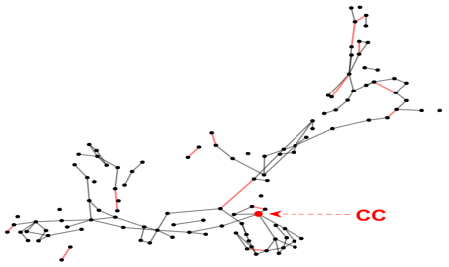
(c) BC method

(d) LP-relaxation method



(e) (1,1.02)-approximation of CMST

(f) (1.02,1)-approximation of CMST



(g) Random method

Figure 5.2: Graphs showing (a) the power grid topology and (b-g) the communication network obtained from different design methods. Black edges: PLCC, red edges: non-PLCC, CC: control center

communication layers of the IEEE 118-bus power system under different designs before imposing any failure. The budget is set to 40. It is sensible to conclude that different link selection policies lead to picking various links as PLCC and non-PLCC, which further affect the performance of cascade prevention. Next, we will explain the settings used in these benchmarks to produce the corresponding communication networks in more detail.

5.5.3.1 Setting Design Parameters

Comparison under node weight definition: Recalling that γ_i represents the importance of observing node i by the CC, we examine both the topological (centrality and degree) and the service (power injection) importance of node i. Moreover, we impose upper-bounds L and N on the length and number of paths between the CC and any particular node. This is justified because the throughput of a flow drops as the hop count increases.

We compare performance of the approximation algorithm (5.6) under four different definitions of weights: (i) power injection $\times$ BC (Betwenness centrality), (ii) BC, (iii) degree, and (iv) power injection of nodes solely. Upper-bounds $\mathcal{B} = 40$, $N = 5$ and $L = 20$ are also imposed.

The results in Fig. 5.3 and Table 5.5 show that the node weight definition of "the BC of the node $\times$ the real power injected at the node" attains the best performance in both the total load served after cascade and topology estimation, as it considers both the topological (BC) and the service (power injection) importance of node i. For the rest of the results in this section, we will use this definition of weight for the approximation algorithm.

Table 5.5: The effect of different node weight definition on approximation algorithm on percentage of (i) valuable and (ii) highly observable islands under $\mathcal{B} = 40$, $N = 5$ and $L = 20$.

	BC $\times$ power	BC	degree	power
(i)	54.55	48.82	48.30	47.07
(ii)	42.48	38.01	37.60	36.80

Configuration of proposed approximation algorithm: We compare performance of the proposed approximation algorithm under different limits on number (N) of paths in Fig. 5.4 and Table 5.6 between each node and the

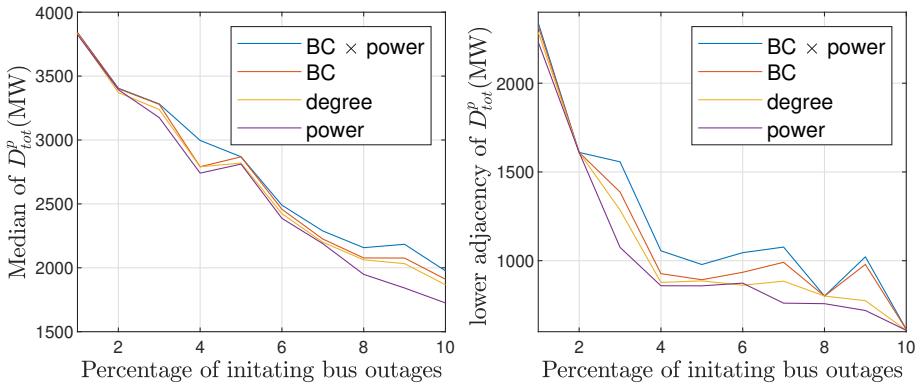


Figure 5.3: Performance evaluation of approximation algorithm in terms of median and lower adjacency of D^p_{tot} for different node weight definition under $\mathcal{B} = 40$, $N = 5$ and $L = 20$.

CC. Figure 5.5 and Table 5.7 show the effect of length (L) of paths between each node and the CC in the approximation algorithm. We also evaluate the approximation algorithm such that the paths in each P_i ($i \in V$) are disjoint with each other, without limitation on path length and number. The results note that topology estimation and cascade prevention achieve the best performance by allowing overlap between paths, picking a smaller N and a larger L. For the rest of the results, we set $N = 5$ and $L = 20$ in the approximation algorithm. Also, Fig. 5.2(b) shows the designed communication network of the approximation algorithm for IEEE 118-bus system under these settings. In this figure, budget and node weight definition are considered 40 and "the BC of the node × the real power injected at the node", respectively.

Table 5.6: The effect of varying N on percentage of (i) valuable and (ii) highly observable islands under $\mathcal{B} = 40$ and $L = 20$.

	disjoint path	$N = 15$	$N = 10$	$N = 5$
(i)	42.83	48.82	51.79	54.78
(ii)	34.04	37.91	39.84	42.66

Comparison under different budget: In Fig. 5.6 and Table 5.8, we show the proposed approximation algorithm's performance under different budgets. As expected, the higher budget performs the best, as it produces a better-connected communication graph with more links, especially non-PLCC links.

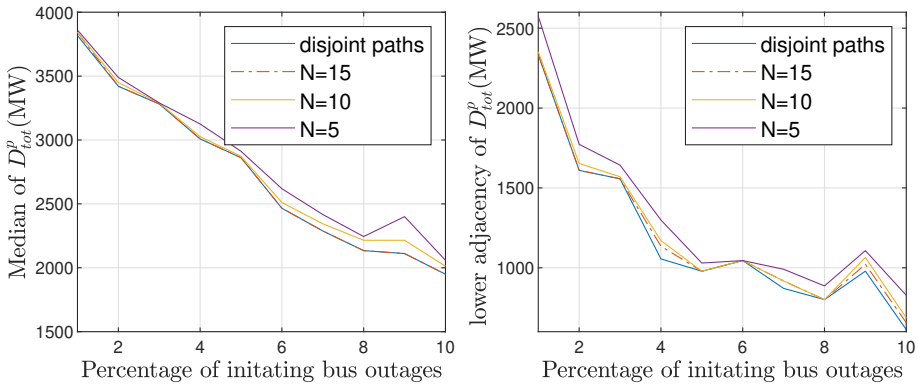


Figure 5.4: Performance evaluation of approximation algorithm in terms of median and lower adjacency of D^p_{tot} for different N under $\mathcal{B} = 40$ and $L = 20$.

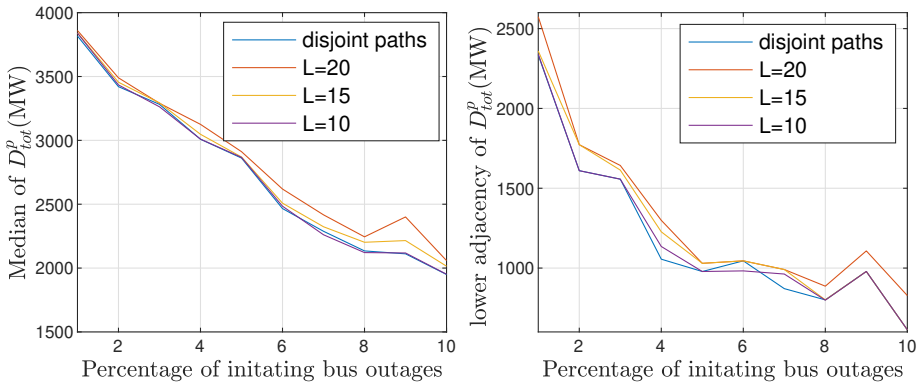


Figure 5.5: Performance evaluation of approximation algorithm in terms of median and lower adjacency of D^p_{tot} for different L under $\mathcal{B} = 40$ and $N = 5$.

Table 5.7: The effect of varying L on percentage of (i) valuable and (ii) highly observable islands under $\mathcal{B} = 40$ and $N = 5$.

	disjoint path	$L = 20$	$L = 15$	$L = 10$
(i)	45.85	58.65	47.41	46.05
(ii)	36.44	45.68	37.41	36.64

To compare the proposed method with existing solutions where all the communication links are considered as non-PLCC, we compare the results from the motivating experiments (Section 5.2.3) with Fig. 5.6 under 5% ini-

tial bus outages. The existing solutions require a budget of $\mathcal{B} = 111.6$, and the mean value of the served power after cascade propagation is 3038.3 MW (70.81% of the initial power). Meanwhile, using the proposed technique, we can achieve a mean served power after cascade propagation of 2993.75 MW (69.78% of the initial power) at a budget of $\mathcal{B} = 50$. This comparison shows that our solution can significantly reduce the cost of constructing the communication network with little negative impact on the efficacy of preventive control.

We also calculate the network's reliability as defined in (5.3a), under different budget scenarios. The results are 1.96, 0.84, 0.75 and 0.51 for budgets of 50, 45, 40 and 35, respectively. The results follow the same trend as Figs. 5.6 and note that a more reliable network delivers higher post-contingency served demand after cascade. The nodes' weights are normalized here.

To analyze the accuracy of the $\hat{\mathbf{B}}$, Table 5.9 provides the percentage of islands that have been correctly identified for different budget scenarios. For each scenario, 100 random cases were tested. In this table, the "slightly observable" islands are the valuable islands that are not highly observable; in other words, they include less than 50% nodes observable by the CC. As was expected, a higher budget leads to higher accuracy of island detection obtained from a more resilient communication network.

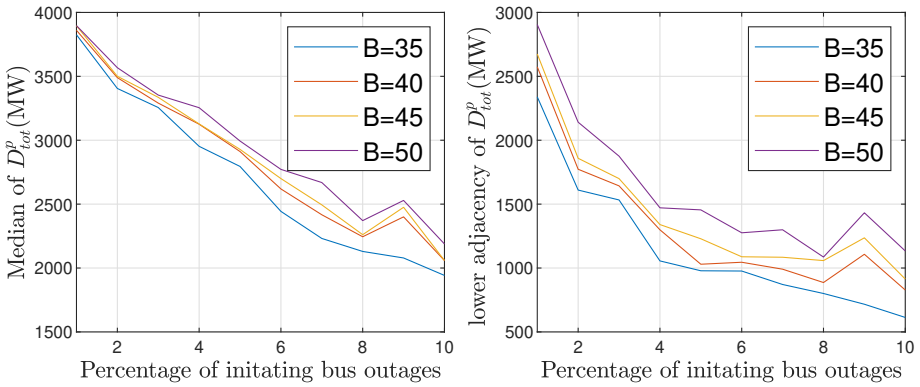


Figure 5.6: Performance evaluation of proposed approximation algorithm in terms of median and lower adjacency of D^p_{tot} for different $\mathcal{B}$ under $N = 5$ and $L = 20$.

Configuration of (α, β)-approximation of CMST algorithm: Recalling that

Table 5.8: The effect of varying budget in proposed approximation algorithm on percentage of (i) valuable and (ii) highly observable islands under $N = 5$ and $L = 20$.

	$\mathcal{B} = 35$	$\mathcal{B} = 40$	$\mathcal{B} = 45$	$\mathcal{B} = 50$
(i)	32.97	43.40	52.74	64.80
(ii)	26.34	33.80	38.76	46.71

Table 5.9: The effect of varying budget in proposed approximation algorithm on percentage of estimation accuracy for (i) valuable, (ii) highly observable and (iii) slightly observable islands (under $N = 5$ and $L = 20$).

	$\mathcal{B} = 35$	$\mathcal{B} = 40$	$\mathcal{B} = 45$	$\mathcal{B} = 50$
(i)	87.48	90.87	95.99	99.57
(ii)	93.55	95.79	98.38	99.82
(iii)	52.20	67.23	87.32	98.57

the $(1 + \varepsilon, 1)$-approximation of CMST algorithm allows an extra budget of $\varepsilon\mathcal{B}$, hence, larger ε supplies a larger budget. We evaluated the communication network of $(1 + \varepsilon, 1)$-approximation of CMST algorithm under varying ε, which are not shown in the chapter. Results indicate that larger ε returns a graph with more links, especially more non-PLCC links; hence it performs better. To make the result of $(1 + \varepsilon, 1)$-approximation of CMST algorithm comparable with other benchmarks, we fix ε to a small value of 0.02.

We also found that the $(1, 1 + \varepsilon)$-approximation of CMST design is not sensitive to ε. The reason is that this method focuses on minimizing the cost of the constructed MST such that the total weight is at most $\mathcal{B}$. In words, this algorithm groups links based on their cost, defined as "1− reliability of the link". Then, in ascending order of the costs, it constructs a MST from the links' groups of not bigger than that particular cost. The cost of a non-PLCC link is smaller than PLCC; however, the MST of all non-PLCC links doesn't satisfy the weight budget constraint. So the $(1, 1+\varepsilon)$-approximation of CMST algorithm gives the same MST of PLCC links, independently of ε.

Figures 5.2(e/f) depicts the designed communication networks of $(1, 1.02)$/ $(1.02, 1)$-approximation of CMST algorithm for IEEE 118-bus system. Budget is 40 in these figures.

Overall comparison of all algorithms:

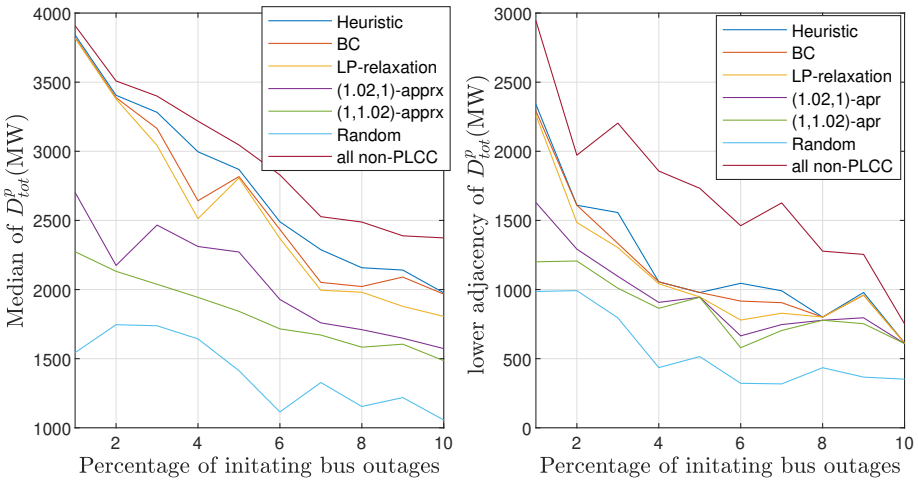


Figure 5.7: Performance evaluation of all algorithms in terms of median and lower adjacency of D^p_{tot} under $\mathcal{B} = 40$, ($N = 5$ and $L = 20$ for approximation).

Table 5.10: Performance of different methods in terms of (i) percentage of valuable and (ii) percentage of highly observable islands under $\mathcal{B} = 40$, ($N = 5$ and $L = 20$ for approximation).

	Approximation	BC	LP-relaxation
(i)	43.46	38.95	39.15
(ii)	33.85	27.20	23.18
	(1.02,1)-aprx of CMST	(1,1.02)-aprx of CMST	Random
(i)	38.10	33.63	18.93
(ii)	15.39	10.23	3.05

5.5.3.2 Overall Comparison of All Algorithms

Finally, Fig. 5.7 and Table 5.10 compare the performance of all algorithms in terms of total post-contingency demand served D^p_{tot} and the percentage of valuable/highly observable islands in the power grid after cascade. Results show the proposed approximation algorithm consistently exceeds all the baselines, which emphasizes the importance of strategically placing links considering both the system topology and its generation/load contribution. The second best algorithm, the BC method, outperforms the LP-relaxation with rounding benchmark and works notably better than $(1, 1.02)/(1.02, 1)$-

Table 5.11: Performance evaluation of all algorithms in terms of percentage of estimation accuracy in (i) valuable, (ii) highly observable and (iii) slightly observable islands under $\mathcal{B} = 40$, ($N = 5$ and $L = 20$ for approximation method).

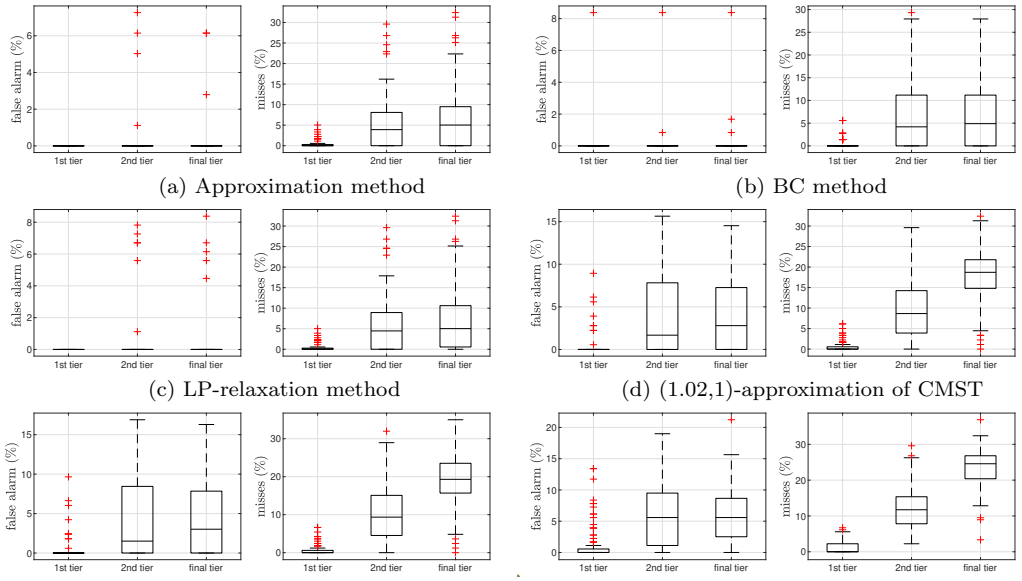
	Approximation	BC	LP-relaxation
(i)	92.55	90.16	88.89
(ii)	96.22	94.23	93.62
(iii)	70.00	66.67	64.86
	(1.02,1)-aprx of CMST	(1,1.02)-aprx of CMST	Random
(i)	72.33	61.62	42.42
(ii)	89.62	74.29	60.40
(iii)	63.92	54.69	33.16

approximation of CMST algorithms due to placing non-PLCC links at the network's weakest parts. In $(1, 1.02)/(1.02, 1)$-approximation of CMST designs, although there is not a notable gap in the percentage of valuable islands with respect to the LP-relaxation and BC methods, Only a small portion of these islands are highly observable. The reason is that these approximation of CMST algorithms create a spanning tree; hence the created graph is not well-connected and can be broken into subtrees simply in failure occurrence. Furthermore, the $(1, 1.02)$-approximation of CMST graph only uses PLCC links that are not immune to failure, while $(1.02, 1)$-approximation of CMST picks non-PLCC links without any specific strategy. We have also added the "upper bound" of 100% non-PLCC case with no budget constraint. This case gives the potential room of improvement on top of the proposed algorithm.

Table 5.11 gives the percentage of correctly identified islands for different methods. For each scenario, 100 random cases were tested, and 2% of initial node outages are considered. Results are in line with Table 5.10 which higher observability leads to higher accuracy of island identification.

Figure 5.8 illustrates the increase in the percentage of misses and false alarms as the cascade proceeds from one to two to the final tier of line outages. Misses are the cases in which certain lines are out but are identified as healthy. We compare these values for all benchmarks. For each scenario, 100 random cases and 2% of initial node outages are considered here. By comparing the first and final tier of line outages in all scenarios, it is logical to conclude that as the cascade progresses, the error accumulates, which further diminishes the accuracy of $\hat{\mathbf{B}}$. Furthermore, the approximation method out-

performs the other methods, especially against the random method, which highlights the importance of a well-designed communication network. Taking into account Figs. 5.7, 5.8 and Table 5.10 together, it seems that different methods follow the same trend in these two figures and the table. The origin of this stems from the resilience of the communication network against failures, both at the initiation and during the cascade. The more the communication network is observable by the CC, the higher the percentage of valuable/highly observable islands, and the more accurate $\hat{\mathbf{B}}$ is. This, in turn, leads to better results in term of total post-contingency demand served since the preventive controller produces a more accurate result in a closed-loop manner while solving (4.1), which depends on $\hat{\mathbf{B}}$.



(a) Approximation method (b) BC method

(c) LP-relaxation method (d) (1.02,1)-approximation of CMST

(e) (1,1.02)-approximation of CMST (f) Random method

Figure 5.8: Performance evaluation of different methods in terms of percentage of misses and false alarms of line outage identification under $\mathcal{B} = 40$, ($N = 5$ and $L = 20$ for proposed approximation method).

5.6 Conclusion

We study the impact of coupling between the power grid and a SCADA-based communication network. We combine the cost efficiency of PLCC links and

the reliability of non-PLCC links and allocate a limited number of communication links while focusing on improving the robustness of such control system against cascading failures under the assumption of uncertain knowledge of failure and system topology. We not only proved the NP-hardness of the proposed solution in the general case, but we also developed a polynomial-time algorithm giving a constant approximation ratio under certain conditions. Extensive simulations on the 118-IEEE bus power system showed that the proposed algorithm achieves efficacy in estimating the system topology and different statistical measures of total demand served at the end of the cascade.

Chapter 6

Conclusion and Future Works

This chapter summarizes our research contributions toward resource allocation in distributed systems to properly serve the applications of interest under resource capacity and operational budget constraints. We then outline several interesting future directions of the works in this dissertation.

6.1 Summary of Contributions

This dissertation provides solutions to selected problems in optimally determining the allocation of resources in both edge cloud computing networks and power grid networks with a SCADA-based communication network. Mobile edge cloud computing is a new paradigm to provide cloud computing capabilities at the edge of pervasive radio access networks in close proximity to mobile users. A smart grid is a modernized power grid that uses information and communication technologies to collect information and dispatch control commands to the power grid. This information is used to remotely regulate the production and distribution of electricity or adjust power consumption to save energy and reduce losses.

In the following, we summarize the main contributions of each chapter.

- **Chapter 3:** In this chapter, we propose a two-time scale framework for joint service placement and request-scheduling in a mobile edge computing environment and formulate the underlying optimization as a mixed-integer linear program (MILP) that jointly considers dedicated and amortized resources.

By analyzing the complexity in carefully selected special cases, we not only prove that our problem is generally NP-hard but also characterize all the cases that are polynomial-time solvable and identify the root cause of hardness.

Then, by reformulating our problem as a set function optimization, we develop a greedy service placement algorithm based on shadow request scheduling computed by a linear program (LP). By proving that our objective function is monotone submodular under certain conditions and our constraints form a p-independence system, we derive a constant-factor approximation guarantee for the proposed algorithm.

We extend the problem formulation under hard resource constraints and substantially improve the complexity analysis for the request scheduling subproblem under hard constraints. We prove that in the special case where all the requests demand the same amount of communication and computation resources, the request-scheduling subproblem under hard constraints can be converted to a maximum-flow problem in a carefully constructed auxiliary graph, based on which we develop a polynomial-time algorithm that is provably optimal.

We show that both our formulation and our algorithm can be extended to exploit request-prediction over multiple frames.

In the end, we perform extensive performance evaluations via synthetic and trace-driven simulations. The proposed algorithm consistently outperforms baselines while achieving over 90% of the optimal performance in all the evaluated cases, even when the approximation guarantee does not hold.

- **Chapter 4:** We study the impact of coupling between the communication and the power networks as it affects a SCADA-based preventive control system under the assumption that loss of a power transmission line disables PLCC links.

  In this chapter, we model the problem of allocating non-PLCC communication links to maximize the total demand served at the end of cascade failure under a budget constraint on the number of non-PLCC links for a given power system with a given control center location.

  We propose a heuristic to solve this problem that takes into account both graph-theoretic and power-system information.

The performance of the proposed algorithm is evaluated using the DC-QSS model on a 2383-bus Polish network. The proposed method demonstrates superior performance as quantified by different statistical measures of total demand served at the end of cascade.

- **Chapter 5:** We improve the formulation in Chapter 4 by combining the cost efficiency of PLCC links and the reliability of non-PLCC links. We allocate a limited number of both types of communication links while focusing on improving the robustness of the control system against cascading failures under the assumption of uncertain system topology and knowledge of failure.

  We prove the NP-hardness of the proposed solution in the general case. Then, we developed an algorithm giving a constant approximation ratio under certain conditions.

  Next, we show the superior performance of the communication network designed by the proposed algorithm in facilitating topology estimation and demand satisfaction in the event of cascading failure by performing extensive simulations on the 118-IEEE bus power system.

6.2 Conclusion & Future Works

Communication networks have profound impacts on the development of our science and society, enabling enhanced data streaming, automation, and communications. However, limited budgets and rapid increases in demand have necessitated more efficient use of resources. This dissertation addresses optimized algorithms to allocate resources in mobile edge computing and the smart grid applications.

Mobile edge computing provides computation and storage resources for applications for networking nearby end-users, typically within or at the edge of operator networks. For data-heavy ultra-low latency applications, such as autonomous drones or remote telesurgery, even with 5G, sending data constantly back to the cloud will be costly and deteriorate the customer experience. However, with edge technology, full stream transmission of data is required only as far as a local edge, and only what is necessary is streamed and stored in the centralized cloud. Accordingly, the combination of 5G and edge computing is essential. These enhancements will enable organizations to harness huge amounts of data to drive advanced analytical and artificial

intelligence programs and support mission-critical services that require Ultra-Reliable, Low Latency Communication (URLLC). As future work, research opportunities in developing resource allocation solutions for data-intensive applications that infuse the power of the edge clouds directly into the 5G network can be sought.

A smart grid is an electricity network enabling a two-way flow of electricity and data with digital communications technology enabling control algorithms to control issues caused by malicious attacks, element failure, and disturbances provoked by nature or humans. This purpose is served by remotely regulating the production and distribution of electricity or adjusting power consumption to save energy and reduce losses. Therefore, connectivity of the power grid components to a communication network in a reliable fashion is essential. Most modern SCADA systems use a variety of communication options within one system to meet their needs. Typically, there is no one-size-fits-all solution, so the SCADA system should be designed carefully to fit the system's needs. Nevertheless, budget is a limiting factor in designing a communication network. Our study shows that strategic placement of non-PLCC and PLCC links is crucial in the efficacy of power grids in mitigating cascading failures. Moreover, it is important to remember that these technologies are not mutually exclusive. Different types of Non-PLCC and PLCC options can be used alone or in tandem, depending on the system's size and nature. Extension to more complex models is left to future works.

With the emerging 5G technology, the need for extensive cabling in power grids is relaxed, especially in distribution systems. The diversity of monitoring and controlling services requires different levels of security, latency, rate, reliability, etc. 5G network slicing may be used to satisfy these demands, which enables future research on the appropriate allocation of network bandwidth to different slices of 5G networks, such that maximum connectivity of the power grid's components is ensured, and the diverse services of industrial control and information collection are served.

Bibliography

[1] Sai Gopal Vennelaganti et al. *Topology Estimation Following Islanding and its Impact on Preventive Control of Cascading Failure*. arXiv: 2104.06473. 2021. arXiv: 2104.06473 [eess.SY].

[2] Mahadev Satyanarayanan. "The Emergence of Edge Computing". In: *Computer* 50 (2017), pp. 30–39.

[3] Y. C. Lee et al. "Profit-Driven Service Request Scheduling in Clouds". In: *2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*. 2010, pp. 15–24.

[4] Pavel Mach and Zdenek Becvar. "Mobile Edge Computing: A Survey on Architecture and Computation Offloading". In: *IEEE Communications Surveys & Tutorials* 19.3 (Mar. 2017), pp. 1628–1656.

[5] Shiqiang Wang et al. "Dynamic Service Migration in Mobile Edge-Clouds". In: *IFIP Networking*. May 2015.

[6] M. Satyanarayanan et al. "The Role of Cloudlets in Hostile Environments". In: *IEEE Pervasive Computing* 12.4 (Oct. 2013), pp. 40–49.

[7] F. Bonomi et al. "Fog Computing and its Role in the Internet of Things". In: *MCC*. 2012.

[8] T. Taleb and A. Ksentini. "Follow Me Cloud: Interworking Federated Clouds and Distributed Mobile Networks". In: *IEEE Network* 27.5 (Sept. 2013), pp. 12–19.

[9] S. Wang et al. "Dynamic Service Placement for Mobile Micro-Clouds with Predicted Future Costs". In: *IEEE Transactions on Parallel and Distributed Systems* 28.4 (Apr. 2017), pp. 1002–1016.

[10] T. Houser and P. Masters. *The world's second largest blackout*. 2017. URL: https://rhg.com/research/%20puerto-rico-hurricane-maria-worlds-second-largest-blackout/.

[11] D. Grozev et al. “Experimental study of Cloud Computing based SCADA in Electrical Power Systems”. In: *2016 XXV International Scientific Conference Electronics (ET)*. 2016, pp. 1–4.

[12] T. Houser and P. Masters. *Supervisory Control and Data Acquisition (SCADA) Systems,* 2004. URL: https://www.cedengineering.com/userfiles/SCADA%5C%20Systems.pdf,.

[13] K. Ha et al. *Adaptive VM Handoff across Cloudlets*. Technical Report CMU-CS-15-113. June 2015. URL: https://www.cs.cmu.edu/~satya/docdir/CMU-CS-15-113.pdf.

[14] A. Ksentini, T. Taleb, and M. Chen. “A Markov Decision Process-based Service Migration Procedure for Follow Me Cloud”. In: *IEEE ICC*. 2014.

[15] Shiqiang Wang et al. “Mobility-induced Service Migration in Mobile Micro-Clouds”. In: *IEEE MILCOM*. Oct. 2014.

[16] T. Taleb, A. Ksentini, and P. Frangoudis. “Follow-Me Cloud: When Cloud Services Follow Mobile Users”. In: *IEEE Transactions on Cloud Computing* (2018), pp. 1–1.

[17] Kiryong Ha et al. “You can Teach Elephants to Dance: Agile VM Handoff for Edge Computing”. In: *ACM/IEEE Symposium on Edge Computing (SEC)*. Oct. 2017.

[18] M. Jia, J. Cao, and W. Liang. “Optimal Cloudlet Placement and User to Cloudlet Allocation in Wireless Metropolitan Area Networks”. In: *IEEE Transactions on Cloud Computing* PP.99 (2015), pp. 1–1. ISSN: 2168-7161. DOI: 10.1109/TCC.2015.2449834.

[19] Z. Xu et al. “Efficient Algorithms for Capacitated Cloudlet Placements”. In: *IEEE Transactions on Parallel and Distributed Systems* 27.10 (Oct. 2016), pp. 2866–2880. ISSN: 1045-9219. DOI: 10.1109/TPDS.2015.2510638.

[20] A. Ceselli, M. Premoli, and S. Secci. “Mobile Edge Cloud Network Design Optimization”. In: *IEEE/ACM Trans. on Netw.* 25.3 (2017).

[21] *Open Edge Computing*. URL: http://www.openedgecomputing.org/.

[22] *OpenFog Consortium*. URL: https://www.openfogconsortium.or.

[23] *ETSI ISG on Multi-access Edge Computing (MEC)*. URL: http://www.etsi.org/technologies-clusters/technologies/multi-access-edge-computing.

[24] L. Wang et al. "Online Resource Allocation for Arbitrary User Mobility in Distributed Edge Clouds". In: *IEEE ICDCS*. 2017. DOI: 10.1109/ICDCS.2017.30.

[25] H. Tan et al. "Online Job Dispatching and Scheduling in Edge-Clouds". In: *IEEE INFOCOM*. May 2017.

[26] H. Casanova et al. "Heuristics for scheduling parameter sweep applications in grid environments". In: *Proceedings 9th Heterogeneous Computing Workshop (HCW 2000) (Cat. No.PR00556)*. 2000, pp. 349–363.

[27] T. He et al. "It's Hard to Share: Joint Service Placement and Request Scheduling in Edge Clouds with Sharable and Non-sharable Resources". In: *IEEE ICDCS*. July 2018.

[28] Zeineb Rejiba, Xavi Masip, and E. Marin-Tordera. "A Survey on Mobility-Induced Service Migration in the Fog, Edge, and Related Computing Paradigms". In: *ACM Computing Surveys* 52 (Sept. 2019), pp. 1–33. DOI: 10.1145/3326540.

[29] T. Taleb and A. Ksentini. "Follow me cloud: interworking federated clouds and distributed mobile networks". In: *IEEE Network* 27.5 (2013), pp. 12–19.

[30] T. Taleb et al. "Mobile Edge Computing Potential in Making Cities Smarter". In: *IEEE Communications Magazine* 55.3 (2017), pp. 38–43.

[31] A. Aissioui et al. "On Enabling 5G Automotive Systems Using Follow Me Edge-Cloud Concept". In: *IEEE Transactions on Vehicular Technology* 67.6 (2018), pp. 5302–5316.

[32] W. Bao et al. "Follow Me Fog: Toward Seamless Handover Timing Schemes in a Fog Computing Environment". In: *IEEE Communications Magazine* 55.11 (2017), pp. 72–78.

[33] R. Bruschi et al. "Move with Me: Scalably Keeping Virtual Objects Close to Users on the Move". In: *2018 IEEE International Conference on Communications (ICC)*. 2018, pp. 1–6.

[34] C. Puliafito et al. “Companion Fog Computing: Supporting Things Mobility Through Container Migration at the Edge”. In: *2018 IEEE International Conference on Smart Computing (SMARTCOMP)*. 2018, pp. 97–105.

[35] Fariba Aalamifar. “Viability of powerline communication for smart grid realization”. In: (2012).

[36] V. Farhadi et al. “Service Placement and Request Scheduling for Data-intensive Applications in Edge Clouds”. In: *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*. 2019, pp. 1279–1287.

[37] Yanchen Liu, Myung J. Lee, and Yanyan Zheng. “Adaptive Multi-Resource Allocation for Cloudlet-Based Mobile Cloud Computing System”. In: *IEEE Transactions on Mobile Computing* 15.10 (2016), pp. 2398–2410. DOI: 10.1109/TMC.2015.2504091.

[38] Narendra Karmarkar. “A New Polynomial-Time Algorithm for Linear Programming-II”. In: *Combinatorica* 4 (Dec. 1984), pp. 373–395. DOI: 10.1007/BF02579150.

[39] Kanapathippillai Cumanan et al. “Joint Beamforming and User Maximization Techniques for Cognitive Radio Networks Based on Branch and Bound Method”. In: *IEEE Transactions on Wireless Communications* 9.10 (2010), pp. 3082–3092. DOI: 10.1109/TWC.2010.072610.090898.

[40] Attahiru S. Alfa et al. “Mixed-integer programming based techniques for resource allocation in underlay cognitive radio networks: A survey”. In: *Journal of Communications and Networks* 18.5 (2016), pp. 744–761. DOI: 10.1109/JCN.2016.000104.

[41] P. Cheng et al. “A Distributed Algorithm for Optimal Resource Allocation in Cognitive OFDMA Systems”. In: *2008 IEEE International Conference on Communications*. 2008, pp. 4718–4723. DOI: 10.1109/ICC.2008.884.

[42] Qianxi Lu et al. “Optimal Subcarrier and Power Allocation under Interference Temperature Constraints”. In: *2009 IEEE Wireless Communications and Networking Conference*. 2009, pp. 1–5. DOI: 10.1109/WCNC.2009.4917870.

[43] Chiu-Han Hsiao et al. "Optimization-Based Resource Management Algorithms with Considerations of Client Satisfaction and High Availability in Elastic 5G Network Slices". In: *Sensors* 21.5 (2021). ISSN: 1424-8220. DOI: 10.3390/s21051882. URL: https://www.mdpi.com/1424-8220/21/5/1882.

[44] Laiping Zhao et al. "Online Virtual Machine Placement for Increasing Cloud Provider's Revenue". In: *IEEE Transactions on Services Computing* 10.2 (2017), pp. 273–285. DOI: 10.1109/TSC.2015.2447550.

[45] Renchao Xie et al. "Dynamic Channel and Power Allocation in Cognitive Radio Networks Supporting Heterogeneous Services". In: *2010 IEEE Global Telecommunications Conference GLOBECOM 2010*. 2010, pp. 1–5. DOI: 10.1109/GLOCOM.2010.5683562.

[46] Giuseppe Portaluri and Stefano Giordano. "Power efficient resource allocation in cloud computing data centers using multi-objective genetic algorithms and simulated annealing". In: *2015 IEEE 4th International Conference on Cloud Networking (CloudNet)*. 2015, pp. 319–321. DOI: 10.1109/CloudNet.2015.7335329.

[47] Amirhossein Feizi Ashtiani and Samuel Pierre. "Secrecy Based Resource Allocation for D2D Communication Using Tabu Search Algorithm". In: *2019 IEEE Canadian Conference of Electrical and Computer Engineering (CCECE)*. 2019, pp. 1–5. DOI: 10.1109/CCECE43985.2019.9052395.

[48] S. Wang, M. Zafer, and K. K. Leung. "Online Placement of Multi-Component Applications in Edge Computing Environments". In: *IEEE Access* 5 (2017), pp. 2514–2533.

[49] L. Tong, Y. Li, and W. Gao. "A hierarchical edge cloud architecture for mobile computing". In: *IEEE INFOCOM 2016*. Apr. 2016.

[50] D. Gonçalves et al. "Proactive Virtual Machine Migration in Fog Environments". In: *2018 IEEE Symposium on Computers and Communications (ISCC)*. 2018, pp. 00742–00745.

[51] Gy orgy Dán and Niklas Carlsson. "Dynamic Content Allocation for Cloud-assisted Service of Periodic Workloads". In: *IEEE INFOCOM*. Apr. 2014.

[52] Ssamta Shukla et al. “Hold’em Caching: Proactive Retention-Aware Caching with Multi-path Routing for Wireless Edge Networks”. In: *ACM Mobihoc*. July 2017.

[53] I-Hong Hou et al. “Asymptotically Optimal Algorithm for Online Reconfiguration of Edge-clouds”. In: *ACM MobiHoc*. July 2016.

[54] Sem Borst, Varun Gupta, and Anwar Walid. “Distributed Caching Algorithms for Content Distribution Networks”. In: *IEEE INFOCOM*. Apr. 2010.

[55] Mostafa Dehghan et al. “On the Complexity of Optimal Request Routing and Content Caching in Heterogeneous Cache Networks”. In: *IEEE/ACM Transactions on Networking* 25.3 (June 2017).

[56] A. Aral and T. Ovatman. “A Decentralized Replica Placement Algorithm for Edge Computing”. In: *IEEE Transactions on Network and Service Management* 15.2 (June 2018), pp. 516–529.

[57] T. Ouyang, Z. Zhou, and X. Chen. “Follow Me at the Edge: Mobility-Aware Dynamic Service Placement for Mobile Edge Computing”. In: *2018 IEEE/ACM 26th International Symposium on Quality of Service (IWQoS)*. 2018, pp. 1–10.

[58] Marc Barcelo et al. “The Cloud Service Distribution Problem in Distributed Cloud Networks”. In: *IEEE ICC*. 2015.

[59] Jaime Llorca et al. “Optimal Content Distribution and Multi-resource Allocation in Software Defined Virtual CDNs”. In: *AIRO ODS*. Sept. 2017.

[60] Rahul Urgaonkar et al. “Dynamic service migration and workload scheduling in edge-clouds”. In: *Performance Evaluation* 91 (Oct. 2015), pp. 205–228.

[61] Hao Feng et al. “Approximation Algorithms for the NFV Service Distribution Problem”. In: *IEEE INFOCOM*. Apr. 2017.

[62] Yuval Rochman, Hanoch Levy, and Eli Brosh. “Resource Placement and Assignment in Distributed Network Topologies”. In: *IEEE INFOCOM*. Apr. 2013.

[63] A. A. Haghighi, S. Shah Heydari, and S. Shahbazpanahi. “Dynamic QoS-Aware Resource Assignment in Cloud-Based Content-Delivery Networks”. In: *IEEE Access* 6 (2018), pp. 2298–2309.

[64] Ye Cai et al. “Cascading Failure Analysis Considering Interaction Between Power Grids and Communication Networks”. In: *IEEE Transactions on Smart Grid* 7.1 (2016), pp. 530–538. DOI: 10.1109/TSG.2015.2478888.

[65] Xiaoyuan Fan et al. “Coordination of Transmission, Distribution and Communication Systems for Prompt Power System Recovery after Disasters:Report – Grid and Communication Interdependency Review and Characterization of Typical Communication Systems”. In: (Mar. 2019). DOI: 10.2172/1526728. URL: https://www.osti.gov/biblio/1526728.

[66] M. Eppstein and P. Hines. “A “Random Chemistry” algorithm for identifying collections of multiple contingencies that initiate cascading failure”. In: *2013 IEEE Power Energy Society General Meeting*. 2013, pp. 1–1.

[67] Benjamin Carreras et al. “Critical Points and Transitions in an Electric Power Transmission Model for Cascading Failure Blackouts”. In: *Chaos (Woodbury, N.Y.)* 12 (Jan. 2003), pp. 985–994. DOI: 10.1063/1.1505810.

[68] Sakshi Pahwa, Caterina Scoglio, and Antonio Scala. “Abruptness of Cascade Failures in Power Grids”. In: *Scientific reports* 4 (Jan. 2014), p. 3694. DOI: 10.1038/srep03694.

[69] J. Yan et al. “Cascading Failure Analysis With DC Power Flow Model and Transient Stability Analysis”. In: *IEEE Transactions on Power Systems* 30.1 (2015), pp. 285–297.

[70] Mert Korkali et al. “Reducing Cascading Failure Risk by Increasing Infrastructure Network Interdependence”. In: *Scientific Reports* 8 (Mar. 2017), p. 46959. DOI: 10.1038/srep46959.

[71] D. Bienstock. “Adaptive online control of cascading blackouts”. In: *2011 IEEE Power and Energy Society General Meeting*. 2011, pp. 1–8.

[72] Dusko Nedic et al. “Criticality in a Cascading Failure Blackout Model”. In: *International Journal of Electrical Power Energy Systems* 28 (Mar. 2006), pp. 627–633. DOI: 10.1016/j.ijepes.2006.03.006.

[73] M. Almassalkhi and I. Hiskens. "Model-predictive cascade mitigation in electric power systems with storage and renewables, Part I: Theory and implementation". In: *2015 IEEE Power Energy Society General Meeting*. 2015, pp. 1–1.

[74] M. R. Almassalkhi and I. A. Hiskens. "Model-Predictive Cascade Mitigation in Electric Power Systems With Storage and Renewables—Part II: Case-Study". In: *IEEE Transactions on Power Systems* 30.1 (2015), pp. 78–87.

[75] S. Mei et al. "A Study of Self-Organized Criticality of Power System Under Cascading Failures Based on AC-OPF With Voltage Stability Margin". In: *IEEE Transactions on Power Systems* 23.4 (2008), pp. 1719–1726.

[76] Q. Chen and L. Mili. "Composite Power System Vulnerability Evaluation to Cascading Failures Using Importance Sampling and Antithetic Variates". In: *IEEE Transactions on Power Systems* 28.3 (2013), pp. 2321–2330.

[77] Jian Li et al. "A cascading failure model based on AC optimal power flow: Case study". In: *Physica A: Statistical Mechanics and its Applications* 508 (May 2018). DOI: 10.1016/j.physa.2018.05.081.

[78] W. Ju, K. Sun, and R. Yao. "Simulation of Cascading Outages Using a Power-Flow Model Considering Frequency". In: *IEEE Access* 6 (2018), pp. 37784–37795.

[79] M. H. Athari and Z. Wang. "Stochastic Cascading Failure Model With Uncertain Generation Using Unscented Transform". In: *IEEE Transactions on Sustainable Energy* 11.2 (2020), pp. 1067–1077.

[80] D. Fabozzi and T. Van Cutsem. "Simplified time-domain simulation of detailed long-term dynamic models". In: *2009 IEEE Power Energy Society General Meeting*. 2009, pp. 1–8.

[81] J. Song et al. "Dynamic Modeling of Cascading Failure in Power Systems". In: *IEEE Transactions on Power Systems* 31.3 (2016), pp. 2085–2095.

[82] S. K. Khaitan, Chuan Fu, and J. McCalley. "Fast parallelized algorithms for on-line extended-term dynamic cascading analysis". In: *2009 IEEE/PES Power Systems Conference and Exposition*. 2009, pp. 1–7.

[83] D. Z. Tootaghaj et al. "Mitigation and Recovery From Cascading Failures in Interdependent Networks Under Uncertainty". In: *IEEE Transactions on Control of Network Systems* 6.2 (2019), pp. 501–514.

[84] Marzieh Parandehgheibi, Eytan Modiano, and David Hay. "Mitigating cascading failures in interdependent power grids and communication networks". In: *2014 IEEE International Conference on Smart Grid Communications (SmartGridComm)*. 2014, pp. 242–247. DOI: 10.1109/SmartGridComm.2014.7007653.

[85] Vajiheh Farhadi et al. "Service Placement and Request Scheduling for Data-Intensive Applications in Edge Clouds". In: *IEEE/ACM Transactions on Networking* 29.2 (2021), pp. 779–792. DOI: 10.1109/TNET.2020.3048613.

[86] Yazhou Hu et al. "Workload prediction for cloud computing elasticity mechanism". In: *Proc. IEEE International Conference on Cloud Computing and Big Data Analysis (ICCCBDA)*. 2016.

[87] G. Kecskemeti et al. "Cloud workload prediction based on workflow execution time discrepancies". In: *Cluster Computing* 22.3 (2019).

[88] A. Khan et al. "Workload characterization and prediction in the cloud: A multiple time series approach". In: *Proc. IEEE Network Operations and Management Symposium*. 2012.

[89] D. Bertsimas and J. Tsitsiklis. *Introduction to linear optimization*. Athena Scientific, 1997.

[90] M. Conforti, G. Cornuejols, and G. Zambelli. *Integer programming*. Springer, 2014.

[91] Ivan Baev, Rajmohan Rajaraman, and Chaitanya Swamy. "Approximation Algorithms for Data Placement Problems". In: *SIAM Journal on Computing* 38.4 (2008).

[92] David B. Shmoys and Éva Tardos. "An Approximation Algorithm for the Generalized Assignment Problem". In: *Mathematical Programming* 62.1-3 (Feb. 1993), pp. 461–474.

[93] Lisa Fleischer et al. "Tight Approximation Algorithms for Maximum General Assignment Problems". In: *ACM-SIAM SODA*. Jan. 2006.

[94] K. Poularakis et al. “Joint Service Placement and Request Routing in Multi-cell Mobile Edge Computing Networks”. In: *IEEE INFOCOM*. Apr. 2019.

[95] H. Kellerer, U. Pferschy, and D. Pisinger. *Knapsack problems*. Springer, 2004.

[96] M. Cardei and D-Z. Du. “Improving Wireless Sensor Network Lifetime through Power Aware Organization”. In: *Wireless Networks* 11.3 (2005), pp. 333–340.

[97] M.L. Fisher, G.L. Nemhauser, and L.A. Wolsey. “An Analysis of Approximations for Maximizing Submodular Set Functions – II”. In: *Math. Prog. Study* 8 (1978), pp. 73–87.

[98] A. Gupta et al. “Constrained non-monotone submodular maximization: Offline and secretary algorithms”. In: *Lecture Notes in Computer Science (LNCS)* 6484.12 (2010).

[99] Gilbert Strang. “Karmarkar’s Algorithm and its Place in Applied Mathematics”. In: *The Mathematical Intelligencer* (1987).

[100] Bernhard Korte and Jens Vygen. “Network Flows”. In: *Combinatorial Optimization*. Berlin, Germany: Springer, 2000. Chap. 8, pp. 153–184.

[101] Mohammed Saad Elbamby et al. “Toward Low-Latency and Ultra-Reliable Virtual Reality”. In: *IEEE Network* 32.2 (2018). DOI: 10.1109/MNET.2018.1700268. URL: https://doi.org/10.1109/MNET.2018.1700268.

[102] Z. Lu et al. “On-demand video processing in wireless networks”. In: *Proc. IEEE ICNP*. 2016.

[103] Michal Piorkowski, Natasa Sarafijanovic-Djukic, and Matthias Grossglauser. *CRAWDAD dataset epfl/mobility (v. 2009-02-24)*. Feb. 2009. URL: http://crawdad.org/epfl/mobility/20090224.

[104] A. Selcuk Uluagac. *CRAWDAD dataset gatech/fingerprinting (v.2014-06-09)*. https://crawdad.org/gatech/fingerprinting/20140609/isolatedtestbed. traceset: isolatedtestbed. June 2014. DOI: 10.15783/C78G67.

[105] H. Hadlach et al. “Modeling of a Smart Grid Monitoring System using Power Line Communication”. In: *2017 International Renewable and Sustainable Energy Conference (IRSEC)*. 2017, pp. 1–4.

[106] K. Ali et al. “Distributed Spectrum Sharing for Enterprise Powerline Communication Networks”. In: *2018 IEEE 26th International Conference on Network Protocols (ICNP)*. 2018, pp. 367–377. DOI: 10.1109/ICNP.2018.00052.

[107] A. Majumder and James Caffery. “Power line communication: An overview”. In: *Potentials, IEEE* (2004).

[108] S. Galli, A. Scaglione, and Z. Wang. “For the Grid and Through the Grid: The Role of Power Line Communications in the Smart Grid”. In: *Proceedings of the IEEE* 99.6 (2011), pp. 998–1027.

[109] D. Healey H. Akkermans and H. Ottosson. “THE REPORT ON TRANSMISSION OF DATA OVER THE ELECTRICITY POWER LINES.” In: *AKMC Spectrum EnerSearch,* (1998).

[110] R. Alaya and R. Attia. “Narrowband Powerline Communication Measurement and Analysis in the Low Voltage Distribution Network”. In: *2019 International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*. 2019, pp. 1–6. DOI: 10.23919/SOFTCOM.2019.8903668.

[111] A. Mengi, S. Ponzelar, and M. Koch. “The ITU-T G.9960 broadband PLC communication concept for smartgrid applications”. In: *2017 IEEE International Conference on Smart Grid Communications (SmartGridComm)*. 2017, pp. 492–496. DOI: 10.1109/SmartGridComm.2017.8340715.

[112] S. Canale et al. “Optimal Planning and Routing in Medium Voltage PowerLine Communications Networks”. In: *IEEE Transactions on Smart Grid* 4.2 (2013), pp. 711–719.

[113] R. Pighi and R. Raheli. “On multicarrier signal transmission for high-voltage power lines”. In: *International Symposium on Power Line Communications and Its Applications, 2005.* 2005, pp. 32–36.

[114] D. Hyun and Y. Lee. “A Study on the Compound Communication Network over the High Voltage Power Line for Distribution Automation System”. In: *2008 International Conference on Information Security and Assurance (isa 2008)*. 2008, pp. 410–414.

[115] K. W. Louie et al. “Discussion on Power Line Carrier Applications”. In: *2006 Canadian Conference on Electrical and Computer Engineering*. 2006, pp. 655–658.

[116] *Power network telecommunication PowerLink – power line carrier system*. URL: www.siemens.com.

[117] X. Zhang et al. "Design and implementation of power line carrier intelligent home appliance control system". In: *2011 International Conference on Mechatronic Science, Electric Engineering and Computer (MEC)*. 2011, pp. 158–161.

[118] P. A. Brown. "Power line communications—past present and future". In: *in Proceedings of the International Symposium on Power Line Communicatons and Its Applications (ISPLC '99),* (1999).

[119] B. S. Sushma et al. "Performance Characterization of Broadband Powerline Communication for Internet-of-Things". In: *2019 International Conference on Wireless Communications Signal Processing and Networking (WiSPNET)*. 2019, pp. 146–151. DOI: 10/WiSPNET45539.2019.9032857.

[120] N. Graf, I. Tsokalo, and R. Lehnert. "Validating broadband PLC for smart grid applications with field trials". In: *2017 IEEE International Conference on Smart Grid Communications (SmartGridComm)*. 2017, pp. 497–502. DOI: 10.1109/SmartGridComm.2017.8340662.

[121] K. Ali et al. "Boosting powerline communications for ubiquitous connectivity in enterprises". In: *2016 IEEE 24th International Conference on Network Protocols (ICNP)*. 2016, pp. 1–2. DOI: 10.1109/ICNP.2016.7784453.

[122] Subhra Sarkar and Palash Kundu. "A Proposed Method of Load Scheduling and Generation Control Using GSM and PLCC Technology". In: Jan. 2015, 47 (5.)–47 (5.) DOI: 10.1049/cp.2015.1643.

[123] Q. Wang et al. "Framework for vulnerability assessment of communication systems for electric power grids". In: *IET Generation, Transmission Distribution* 10.2 (2016), pp. 477–486.

[124] L. Xu et al. "Robust Routing Optimization for Smart Grids Considering Cyber-Physical Interdependence". In: *IEEE Transactions on Smart Grid* 10.5 (2019), pp. 5620–5629.

[125] *Telecommunications for power utilities*. URL: https://www.ge.com/digital/sites/default/files/download-assets/Utilities-Communications-Brochure-GE.pdf.

[126] "Supervisory Control and Data Acquisition (SCADA) Systems". In: *Technical information bulletin 04-1, National Communications Systems* (2004). URL: https://www.cedengineering.com/userfiles/SCADA%5C%20Systems.pdf.

[127] Vajiheh Farhadi et al. "Budget-Constrained Reinforcement of SCADA for Cascade Mitigation". In: *2021 International Conference on Computer Communications and Networks (ICCCN)*. 2021, pp. 1–9. DOI: 10.1109/ICCCN52240.2021.9522250.

[128] URL: https://www.matpower.org.

[129] Cigre Report. "Requirements and performance of packet switching networks with special reference to telecontrol". In: (1991). URL: https://cigreindia.org/CIGRE%5C%20Lib/Tech.%5C%20Brochure/.

[130] Jennifer Golbeck. "Chapter 21 - Analyzing networks". In: *Introduction to Social Media Investigation*. Ed. by Jennifer Golbeck. Boston: Syngress, 2015, pp. 221–235. ISBN: 978-0-12-801656-5. DOI: https://doi.org/10.1016/B978-0-12-801656-5.00021-4. URL: http://www.sciencedirect.com/science/article/pii/B9780128016565000214.

[131] H. Promel and Angelika Steger. "The Steiner Tree Problem: A Tour Through Graphs Algorithms and Complexity". In: (Jan. 2002).

[132] Darrell Whitley. "A Genetic Algorithm Tutorial". In: *Statistics and Computing* 4 (1994), pp. 65–85.

[133] URL: https://xlinux.nist.gov/dads//HTML/simplepath.html.

[134] Satu Schaeffer. "Graph Clustering". In: *Computer Science Review* 1 (Aug. 2007), pp. 27–64. DOI: 10.1016/j.cosrev.2007.05.001.

[135] A. E. Jahromi and Z. B. Rad. "Optimal topological design of power communication networks using genetic algorithm". In: *Scientia Iranica* 20 (2013), pp. 945–957.

[136] US department of Energy. "Advanced Metering Infrastructure and Customer Systems". In: (2016). URL: https://www.energy.gov/sites/prod/files/2016/12/f34/AMI%5C%%2020Summary%5C%20Report-09-26-16.pdf.

[137] Petr Musil et al. "Simulation-Based Evaluation of the Performance of Broadband over Power Lines with Multiple Repeaters in Linear Topology of Distribution Substations". In: *Applied Sciences* 10 (2020), p. 6879.

[138] Hendrik Ferreira. *1. HC Ferreira, L Lampe, JE Newbury and TG Swart, Editors: "Power Line Communications," John Wiley and Sons, UK, ISBN 978-0-470-740309, June 2010.* Jan. 2010.

[139] William C. Black. "Data transmission through distribution transformers without bypass components". In: *ISPLC2010*. 2010, pp. 13–17. DOI: 10.1109/ISPLC.2010.5479922.

[140] Jun Ushida et al. "Immittance matching for multidimensional open-system photonic crystals". In: 68.15, 155115 (Oct. 2003), p. 155115. DOI: 10.1103/PhysRevB.68.155115. arXiv: cond-mat/0306260 [cond-mat.mtrl-sci].

[141] Robert Bixby. "A Brief History of Linear and Mixed-Integer Programming Computation". In: *Documenta Mathematica* (Jan. 2012).

[142] Jan Kronqvist and Andreas Lundell. "Convex Minlp – An Efficient Tool for Design and Optimization Tasks?" In: *Computer Aided Chemical Engineering* (2019).

[143] Cody Ruben et al. "Optimal PMU Allocation on Smart Grids: a MILP Model considering Minimal Current Measurements". In: *2018 IEEE Power Energy Society General Meeting (PESGM)*. 2018, pp. 1–5. DOI: 10.1109/PESGM.2018.8586515.

[144] Wasseem H. Al-Rousan, Caisheng Wang, and Feng Lin. "A Discrete Event Theory Based Approach for Modeling Power System Cascading Failures". In: *2019 IEEE Power Energy Society General Meeting (PESGM)*. 2019, pp. 1–5. DOI: 10.1109/PESGM40551.2019.8974071.

[145] Paul Hines, Ian Dobson, and Pooya Rezaei. "Cascading power outages propagate locally in an influence graph that is not the actual grid topology". In: *2017 IEEE Power Energy Society General Meeting*. 2017, pp. 1–1. DOI: 10.1109/PESGM.2017.8273859.

[146] Vetrivel Subramaniam Rajkumar et al. "Cyber Attacks on Power System Automation and Protection and Impact Analysis". In: *2020 IEEE PES Innovative Smart Grid Technologies Europe (ISGT-Europe)*. 2020, pp. 247–254. DOI: 10.1109/ISGT-Europe47291.2020.9248840.

[147] Ming Ni and Manli Li. "Reliability Assessment of Cyber Physical Power System Considering Communication Failure in Monitoring Function". In: *2018 International Conference on Power System Technology (POWERCON)*. 2018, pp. 3010–3015. DOI: 10.1109/POWERCON.2018.8601964.

[148] Zhengcheng Dong, Meng Tian, and Jiaqi Liang. "Cascading failures of spatially embedded cyber physical power system under localized attacks". In: *2018 37th Chinese Control Conference (CCC)*. 2018, pp. 6154–6159. DOI: 10.23919/ChiCC.2018.8483691.

[149] Mahshid Rahnamay-Naeini and Majeed M. Hayat. "Cascading Failures in Interdependent Infrastructures: An Interdependent Markov-Chain Approach". In: *IEEE Transactions on Smart Grid* 7.4 (2016), pp. 1997–2006. DOI: 10.1109/TSG.2016.2539823.

[150] Rezoan A. Shuvro et al. "Modeling cascading-failures in power grids including communication and human operator impacts". In: *2017 IEEE Green Energy and Smart Systems Conference (IGESSC)*. 2017, pp. 1–6. DOI: 10.1109/IGESC.2017.8283461.

[151] A. Berizzi. "The Italian 2003 blackout". In: *IEEE Power Engineering Society General Meeting, 2004.* 2004, 1673–1679 Vol.2. DOI: 10.1109/PES.2004.1373159.

[152] Min Ouyang. "Review on modeling and simulation of interdependent critical infrastructure systems". In: *Reliability Engineering & System Safety* 121 (2014), pp. 43–60. ISSN: 0951-8320. DOI: https://doi.org/10.1016/j.ress.2013.06.040. URL: https://www.sciencedirect.com/science/article/pii/S0951832013002056.

[153] URL: https://icseg.iti.illinois.edu/ieee-118-bus-system/.

[154] Luigi Di Puglia Pugliese et al. "An Algorithm to Find the Link Constrained Steiner Tree in Undirected Graphs". In: *Mathematical Software – ICMS 2016*. Ed. by Gert-Martin Greuel et al. Cham: Springer International Publishing, 2016, pp. 492–497.

[155] Stratis Ioannidis and Edmund Yeh. *Jointly Optimal Routing and Caching for Arbitrary Network Topologies*. arXiv:1708.05999. 2017. arXiv: 1708.05999 [cs.NI].

[156] Yen Chen. "Polynomial Time Approximation Schemes for the Constrained Minimum Spanning Tree Problem". In: *Journal of Applied Mathematics* 2012, Special Issue (Mar. 2012). DOI: 10.1155/2012/394721.

[157] R. Ravi and M. Goemans. "The Constrained Minimum Spanning Tree Problem (Extended Abstract)". In: *SWAT*. 1996.

[158] Eric W Weisstein. *Norm.* URL: https://mathworld.wolfram.com/Norm.html.

[159] D. J. Marihart. "Communications technology guidelines for EMS/SCADA systems". In: *IEEE Transactions on Power Delivery* 16.2 (2001), pp. 181–188. DOI: 10.1109/61.915480.

Vita

Vajiheh Farhadi

EDUCATIONS

- Ph.D., Computer Science & Engineering, Pennsylvania State University.
- M.Sc., Electrical Engineering, Yazd University, Iran.
- B.Sc, Biomedical Engineering, Amirkabir University of Technology, Iran.

PUBLICATIONS

- V. Farhadi, S. G. Vennelaganti, T. He, N. R. Chaudhuri and T. L. Porta, "Budget-Constrained Reinforcement of SCADA for Cascade Mitigation," *2021 International Conference on Computer Communications and Networks (ICCCN)*, 2021, pp. 1-9.
- V. Farhadi, F. Mehmeti, T. He, T. L. Porta, H. Khamfroush, S. Wang, K. S. Chan, and K. Poularakis, "Service Placement and Request Scheduling for Data-Intensive Applications in Edge Clouds," in *IEEE/ACM Transactions on Networking*, April 2021, vol. 29, no. 2, pp. 779-792.
- V. Farhadi, F. Mehmeti, T. He, T. L. Porta, H. Khamfroush, S. Wang, and K. S. Chan, "Service Placement and Request Scheduling for Data-intensive Applications in Edge Clouds," *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, 2019, pp. 1279-1287.
- V. Farhadi and G. Mirjalily,"Optimal Resource Assignment in Wireless Mesh Networks", *Fourth International Conference on Computational Intelligence and Information Technology (CIIT)*, 2014, pp. 14-21.
- V. Farhadi and G. Mirjalily, "Cross-layer optimal resource assignment with fairness and load balancing in wireless mesh networks," *7'th International Symposium on Telecommunications (IST'2014)*, 2014, pp. 1064-1070.
- V. Farhadi, S. G. Vennelaganti, T. He, N. R. Chaudhuri and T. L. Porta, "Improvement of SCADA-based Cascade Prevention Control Under Budget Constraint", In *IEEE Transactions on Network Science and Engineering (TNSE)*, 2021, revised manuscript under review.
- V. Farhadi, T. He, N. R. Chaudhuri and T. L. Porta, "Communication network capacity enhancement in power grids with 5G multi-tenancy network slicing", 2022, under preparation.