School of Electrical Engineering, Computing and
Mathematical Sciences

Multi-Stage Network Upgrade for Green Software Defined Networking

Lely Hiryanto

0000-0002-9441-5551

This thesis is presented for the Degree of
Doctor of Philosophy
of
Curtin University

June 2022

Declaration

To the best of my knowledge and belief this thesis contains no material previously published by any other person except where due acknowledgment has been made.

This thesis contains no material which has been accepted for the award of any other degree or diploma in any university.

Signature:

Date: 17-June-2022

"*I am dedicating this thesis to my beloved father for his endless love and support. I love and miss you beyond words.*"

"*What will be, will be, what will not, will not, just keep moving forward.*"

— AUTHOR

Abstract

Each router/switch (l-switch) in a *legacy* network performs both network control and data forwarding. In Software Defined Networking (SDN), the network control is performed by an SDN *controller* while each SDN-switch (s-switch) only forwards data packets. Thus, SDN offers more flexibility in network configuration and management than legacy networks. Despite its benefits, some network operators remain hesitant to completely upgrade their networks into SDN, among others, due to significant capital investment, lack of SDN standards, and possible service disruption. To this end, some operators prefer to upgrade their network to SDN over multiple *stages* (in years). Another important concern of operators is network energy consumption. Current networks tend to over-provision resources to satisfy demands during peak hours. To save energy, unused links are turned off during off-peak hours. Network traffic rerouting has been used to maximize the number of idle links that can be turned-off. In addition, optimal placement of controllers in a network has been shown to further minimize the number of turned-on links, and hence save energy.

This thesis addresses three versions of a *novel* problem, called Green Multi Stage Upgrade (GMSU), to upgrade a legacy network into SDN. The three versions, namely GMSU-1, GMSU-2, and GMSU-3, consider legacy networks that support IEEE 802.1AX technology, where each link contains multiple cables, and each idle cable can be switched off. Each version aims to replace a set of l-switches

with *s*-switches over a given number of stages. The aim is to maximally turn off idle cables that are adjacent to *s*-switches to save energy. Each of the three versions considers the following five parameters: (i) a total budget (in $) and a maximum budget per stage, (ii) a switch upgrade cost and depreciation in upgrade cost, (iii) a set of traffic demands and growing traffic sizes, (iv) a predefined path delay constraint, and (v) a maximum link utilization (MLU) threshold.

This thesis first addresses GMSU-1 that considers a single path for parameter (iv) to route each data traffic between switches. It formulates GMSU-1 as an Integer Linear Program (ILP) that can be used to obtain *optimal* solution. It also proposes a heuristic algorithm to solve the problem. Then, this thesis describes GMSU-2 that considers two scenarios of multi-path routing for parameter (iv): non-link-disjoint and link-disjoint. For each scenario, it formulates one Mixed Integer Program (MIP) and a heuristic solution to solve GMSU-2. Finally, this thesis presents GMSU-3 that addresses controller placement. GMSU-3 spends the budget in parameter (i) to deploy both *s*-switches and controllers, and uses the following extra parameters: (vi) the number of control-packet-per-second (*cpps*) transmitted by each *s*-switch, and its increase rate, and (vii) capacity (in *cpps*) and deployment cost of each controller. This thesis formulates an MIP and develops a heuristic solution to solve GMSU-3.

This thesis uses five actual network topologies to evaluate all proposed solutions in terms of their energy saving, number of upgraded switches, number of traffic flows that pass by *s*-switches, and number of deployed controllers. The evaluation results show that all solutions effectively reduce the energy usage of the upgraded network. Further, all heuristic solutions run significantly faster than their corresponding ILP/MIP while producing energy savings that are off only by 5%. Finally, our solutions have been shown to perform favorably against some existing works.

Acknowledgements

This work would not have been possible without the constant support, guidance, and assistance of many people. First and foremost, I would like to express my deepest gratitude to my supervisor, Dr. Sie Teng Soh, for his invaluable advice, continuous support, and generous guidance during my PhD study. His levels of patience, knowledge, and ingenuity are something I will always keep aspiring to. To my co-supervisor, Associate Professor Mihai Lazarescu, thank you for your support on my study. Additionally, I am grateful to Associate Professor Kwan-Wu Chin at the University of Wollongong and Dr. Duc-Son Pham (Sonny), for their valuable suggestions, constant encouragement, and great feedback on my papers. My appreciation goes to the anonymous reviewers and editors of conferences and journals for their comments and suggestions which have certainly increased the quality of my publications.

Very special thanks to the Australia Awards in Indonesia administered by the Department of Foreign Affairs and Trade (DFAT), Australian Government, for the scholarship awards, financial resources, and merits bestowed to me to complete my study. Much appreciation to the Faculty of Information Technology, Tarumanagara University, Jakarta, Indonesia, for the incessant support for me to undertake this PhD journey. Special thanks to Julie Craig, Ian Tsen, Raquel Illiano, and Mandy Wan from Curtin International Sponsored Student Unit (ISSU), for their professional support during my study. I profusely thank my thesis com-

mittee, Associate Professor Wan-Quan Liu, Associate Professor Aneesh Krishna, Dr. King-Sun Chan, Dr. Senjian An, and Dr. Kit-Yan Chan for their significant feedback to improve my work. Thanks to all staffs in School of Electrical Engineering, Computing, and Mathematical Sciences (EECMS) for their help in administration affairs.

Getting through my thesis required more than academic support, and I have many people to thank for lending their ears and being there for me over the past four years. Very special thanks to Rebecca Santi, Benedicta Santoso, Arnold Kaudin, Imelda Tandra, Manlika Ratchagit, Bau Dilam Ardyansyah, Sanyulandy Leowalu, Yuni Yuningsih, Christofori Nastiti, Popy Yuniar, Valentina Utari, Ni Wayan Septarini, Zakaria Victor Kareth, Antoni Liang, and Starsha Odelia Wijaya, for the wonderful and encouraging conversations. Thanks to my research mates in Building 314 Level 4, Lara, Gaurav, Suparna, Ameer, Dath, and Tranh for a pleasant working environment.

I cannot thank you enough to anonymous paramedics and all good Samaritans for their wholehearted action in saving my life at the accident site. Special thanks for nurses and doctors in the emergency department of Royal Perth Hospital and Physiotherapists in Bentley Health Service for helping me gain full recovery.

Finally, I must express my very profound gratitude to my husband and my daughter for their unfailing support and everlasting encouragement through the process of writing this thesis. This accomplishment would not have been possible without them. Thank you.

Contents

List of Figures

List of Tables

Published Works

This thesis is based upon several works that have been presented at conferences, published in journals, and submitted to a conference over the course of the author's PhD. They are listed in chronological order as follows.

1. **Lely Hiryanto**, Sieteng Soh, Kwan-Wu Chin, Mihai Lazarescu, "Green Multi-Stage Upgrade for Bundled-Link SDNs with Budget Constraint," in *29th International Telecommunication Networks and Applications Conference (ITNAC)*, Auckland, New Zealand, Nov. 2019. The paper received the Student Travel Grant.

2. **Lely Hiryanto**, Sieteng Soh, Kwan-Wu Chin, Mihai Lazarescu, "Green Multi-Stage Upgrade for Bundled-Link SDNs with Budget and Delay Constraints," in *IEEE Transactions on Green Communications and Networking*, vol. 5, issue 3, pp. 1410–1425, Sept. 2021.

3. **Lely Hiryanto**, Sieteng Soh, Kwan-Wu Chin, Duc-Son Pham, Mihai Lazarescu, "Green Multi-Stage Upgrade for Bundled-Links SDN/OSPF-ECMP Networks," in *IEEE International Conference on Communications (ICC)*, Montreal, Canada, Jun. 2021. The paper received the Student Grant.

4. **Lely Hiryanto**, Sieteng Soh, Kwan-Wu Chin, Duc-Son Pham, Mihai Lazarescu, "Multi-Path Routing in Green Multi-Stage Upgrade for Bundled-Links SDN/ OSPF-ECMP Networks," in *IEEE Access*, vol. 9, pp. 99073–99091, July 2021.

5. **Lely Hiryanto**, Sieteng Soh, Kwan-Wu Chin, Duc-Son Pham, Mihai Lazarescu, "Green Multi-Stage Upgrade of Switches and Controllers for Bundled-Links SDNs," submitted (under review) to *IEEE Access*.

The copyright information to reuse the published works has been provided in Appendix.

Table 1. Attribution statement for all published works

	Conception and Problem Design	**Network Model**	**Problem Solution**	**Evaluation**	**Interpretation and Discussion**	**Final Approval**
Co-Author 1 (Sieteng Soh)	✓	✓	✓		✓	✓
Co-Author 1 Acknowledgement: I acknowledge that these represent my contribution to the above research output Signed:						
Co-Author 2 (Kwan-Wu Chin)		✓			✓	✓
Co-Author 2 Acknowledgement: I acknowledge that these represent my contribution to the above research output Signed:						
Co-Author 3 (Duc-Son Pham)					✓	✓
CoAuthor 3 Acknowledgement: I acknowledge that these represent my contribution to the above research output Signed:						
Co-Author 4 (Mihai Lazarescu)					✓	✓
Co-Author 4 Acknowledgement: I acknowledge that these represent my contribution to the above research output Signed:						

Acronyms

General	
BDDP	Broadcast Discovery Domain Protocol
CAPEX	CApital EXpenditure
cpps	control-packet-per-second
D2CIF	Directed Two Commodity Integral Flow
EBI	East-Bound Interface
EEGAH	Energy Efficient Genetic Algorithm for Hybrid-SDN
GA	Genetic Algorithm
IEEE	the Institute of Electrical and Electronics Engineers
IGCPA	Improved Genetic Controller Placement Algorithm
ILP	Integer Linear Program
LP	Linear Program
LLDP	Link Layer Discovery Protocol
LS	Local Search
LSA	Link State Advertisement
MIP	Mixed Integer Program
MKP	Multiple Knapsack Problem
MLU	Maximum Link Utilization
MNL	Most Non-controlled Links first
MPLS	Multi-Protocol Label Switching
NBI	North-Bound Interface
OPEX	OPerational EXpenditure
OSPF	Open Shortest Path First
OSPF-ECMP	OSPF with Equal Cost Multi-Paths
OSPF-TE	OSPF with Traffic Engineering
SBI	South-Bound Interface
SDN	Software Defined Networking
TC	Traffic Controllability
TE	Traffic Engineering
WBI	West-Bound Interface
Problem 1	
GMSU-1	Green Multi-Stage Upgrade - Problem 1

GU-1	heuristic solution for GMSU-1
ILP-1	ILP solution for GMSU-1
Problem 2	
GMSU-2	Green Multi-Stage Upgrade - Problem 2
GU-2a	heuristic solution for GMSU-2 with non-link-disjoint paths
GU-2b	heuristic solution for GMSU-2 with link-disjoint paths
MIP-2a	MIP solution for GMSU-2 with non-link-disjoint paths
MIP-2b	MIP solution for GMSU-2 with link-disjoint paths
Problem 3	
GMSU-3	Green Multi-Stage Upgrade - Problem 3
GU-3	heuristic solution for GMSU-3
MIP-3	MIP solution for GMSU-3

Notation

Notation	Definition
$G^0(V, E)$	A *legacy* network with $\|V\|$ nodes and $\|E\|$ links.
b_{uv}	The bundle size, i.e., number of cables of link $(u, v) \in E$.
γ	The cable capacity (in bytes).
c_{uv}	The capacity of link (u, v) in total.
π_{uv}	The propagation delay (in seconds) of link (u, v).
$U_{\max}$	The threshold of maximum link utilization.
T	The total number of upgrade time periods or stages.
V^t	The set of s-switches at stage t.
$\mathcal{V}^T$	The percentage of l-switches that have been upgraded to s-switches over T stages.
$G^t(V, E)$	An upgraded network with $\|V\|$ nodes and $\|E\|$ links at stage t.
B	The maximum budget over T stages.
B^t	The maximum budget at each stage t.
ρ	The decrease rate of switch upgrade cost per stage.
p_u^t	The upgrade cost for node $u \in V$ at stage t.
μ_d	The increase rate of traffic size for each demand d per stage.
D^0	The set of traffic demands in $G^0(V, E)$.
D^t	The set of traffic demands in $G^t(V, E)$.
$(s_d, \tau_d, \omega_d^t)$	A traffic demand in D^t from node s_d to τ_d with traffic size ω_d^t.
σ	Delay multiplier.
$\delta_{\max,d}$	The maximum path delay requirement for demand d.
$\mathcal{P}_d$	The set of alternative paths within $\delta_{\max,d}$ for demand d.
R^0	The set of paths within $\delta_{\max,d}$ to route all demands in $G^0(V, E)$.
R^t	The set of paths within $\delta_{\max,d}$ to route all demands in $G^t(V, E)$.
n_{uv}^t	The number of *on*-cables in link $(u, v) \in E$.
f_{uv}^t	The total traffic volume of link (u, v) at stage t.
ε^t	The energy saving at stage t.
ε_T	The average energy saving over T stages.
x_u^t	An indicator set to 1 if switch u is upgraded at stage t.

Chapter 1

Introduction

Software Defined Networking (SDN) is an attractive solution that simplifies network resource management. In a legacy network, each router/switch or l-switch must perform two main tasks: to compute forwarding or routing rules and to forward packets. Further, each l-switch generates rules individually based on its *local* network information. In contrast, SDN decouples both tasks, whereby an SDN-switch or s-switch forwards packets according to forwarding rules computed by an SDN *controller*. The controller generates the rules using *global* network information, and hence offers more flexibility in network configuration and management. This leads to improved network quality of services and efficiency, as well as helping promote open networking [3]–[5]. As a result, SDN is being adopted by enterprise networks such as Google [6], Microsoft public cloud [7], NTT cloud gateway [8], and IBM public cloud [9] to name a few.

Operators that plan to upgrade their networks into SDN must consider a number of issues. The first issue is the significant capital investment required to upgrade their *legacy* network to an SDN and retrain personnel to manage the upgraded network. The second issue is that an SDN is an in-progress technology in which users have less confidence compared to legacy networks that use proven and

mature technologies, especially in terms of their scalability, reliability, robustness, and security [3]. More specifically, the scalability, resiliency, and fault tolerance issues of using multiple controllers are still under active investigations [5]. Further, SDN is still lack of standardization, i.e., there is no a single protocol that can be used by different developers/vendors to provide communication among controllers and between controller and SDN applications [10]. Up to this end, one protocol that has been standardized and commonly used by different vendors is OpenFlow [10], which provides communication between controller and s-switches. As a consequence, operators are more likely to upgrade their l-switches over multiple periods or *stages* (in years), creating a so-called *hybrid* SDN that contain a mix of l-switches and s-switches. Specifically, at each stage, operators replace a subset of strategically selected l-switches with s-switches [3].

Another important issue for network operators is energy saving [4], [11]. Current networks are known to over-provision network resources, e.g., link bandwidth, to satisfy traffic demands during peak hours; thus, networks are under-utilized during off-peak periods [12]. Consequently, as shown in [12], [13], some network resources, e.g., links, should be powered down to conserve energy.

To this end, researchers [12]–[20] have used the following three approaches to turn off a maximum number of idle links: a bundled link technology IEEE 802.1AX, network traffic rerouting, and energy-aware controller placement. The IEEE 802.1AX [21] creates the so-called *bundled*-links containing multiple physical cables. Advantageously, during off-peak hours, each unused cable that is equipped with IEEE 802.3az [22] can be switched off [23]. Network traffic rerouting has been used to maximize the number of idle links and/or switches that can be powered off. It steers the traffic to either one single alternative path [13], [18] or multi-paths [12], [14]–[17] subject to some network performances, e.g., delay. The authors in [19], [20] have shown that the locations of controllers

have a significant impact on energy consumption. This motivates research into energy-aware controller placement [19] and traffic rerouting among s-switches and between s-switches and their associated controller [20]. However, the existing efforts in [19], [20] consider controller placement and traffic routing in a *pure* SDN that consists of only s-switches.

This thesis addresses three versions of a novel problem to upgrade a legacy network to SDN in *multi-stages*. The problem, called Green Multi-Stage Upgrade (GMSU), considers upgrading a legacy network that support IEEE 802.1AX technology. More specifically, it replaces a set of l-switches with s-switches over a given number of planning stages. The aim is to turn off a maximum number of idle cables that are adjacent to s-switches to save energy. The first version of GMSU considers l-switches that support Open Shortest Path First routing protocol (OSPF) to route each traffic demand. On the other hand, the second version of GMSU assumes l-switches that support OSPF-Equal-Cost-Multi-Paths routing protocol (OSPF-ECMP) which equally splits and routes traffic via a set of shortest paths. Finally, the third version of GMSU addresses controller placement when upgrading l-switches that support OSPF.

1.1. Aim and Approach

The aims of our work in this thesis are as follows.

Aim 1 – To propose and study a novel network upgrade problem called GMSU-1 that upgrades an OSPF network to SDN over multi-stages. It considers five parameters: (i) a total budget and a maximum budget per stage, (ii) a switch upgrade cost and its depreciation rate, (iii) a set of traffic demands and increasing rate of traffic volume per demand, (iv) a pre-defined maximum path delay, and

(v) an MLU threshold. GMSU-1 uses a single path for parameter (iv) to route each *data traffic* among switches. This thesis formulates the problem as an ILP, called ILP-1, that can be used to provide optimal solution for GMSU-1. It also proposes a heuristic solution called GU-1 to solve the problem.

Aim 2 − To propose and study a novel network upgrade problem called GMSU-2 that upgrades an OSPF-ECMP network to SDN over multi-stages. As in GMSU-1, GMSU-2 considers parameters (i) to (v). For parameter (iv), GMSU-2 considers multi-path routing to route data traffic demands. It considers two multi-path routing scenarios: a) non-link-disjoint and b) link-disjoint. This thesis formulates one MIP as the optimal solution for each routing scenario: MIP-2a for scenario-1 and MIP-2b for scenario-a. In addition, it develops two heuristic solutions: GU-2a for scenario-a and GU-2b for scenario-b.

Aim 3 − To propose and study a novel network upgrade problem called GMSU-3. The goal of GMSU-3 is to upgrade l-switches that support OSPF and deploy controllers over multi-stages. GMSU-3 uses the budget in parameter (i) to deploy both s-switches and controllers. For parameter (iv), it requires a shortest path for each data traffic and up to two link-disjoint paths for each *control traffic* between an s-switch and its associated controller. In addition, GMSU-3 considers two additional parameters: (vi) the number of control-packet-per-second (*cpps*) sent by each switch to a controller, and (vii) capacity (in *cpps*) and deployment cost of each controller. This thesis formulates GMSU-3 as an MIP, called MIP-3, to obtain the optimal solution. It also proposes a heuristic solution called GU-3.

1.2. Contributions and Significance

The main contributions and significance of this thesis are as follows.

Contribution

1. It introduces the first version of GMSU called GMSU-1 to maximize energy saving by switching off idle cables. GMSU-1 upgrades l-switches in an OSPF network to s-switches over multi-stages. Thus, each data traffic is routed via a single path. This thesis proves the NP-hardness of GMSU-1 and formulates an ILP called ILP-1 that can be used to obtain the optimal solution. It also proposes an efficient heuristic algorithm called GU-1 and analyses its time complexity.

2. It presents the second version of GMSU called GMSU-2 to upgrade switches in a legacy network that supports OSPF-ECMP to SDN over multi-stages. GMSU-2 considers two multi-path routing scenarios: a) non-link-disjoint and b) link-disjoint. Further, GMSU-2 considers *link cost setting* to ensure that each l-switch splits traffic equally and routes them via a set of shortest paths. The thesis shows the NP-hardness of GMSU-2 and uses two MIPs, i.e., MIP-2a for scenario-1 and MIP-2b for scenario-2, to formulate GMSU-2 and obtain optimal solutions. It also proposes two heuristic algorithms called GU-2a for scenario-1 and GU-2b for scenario-2. This thesis outlines the time complexity of the two algorithms as well as a proof of their correctness.

3. It presents the third version of GMSU called GMSU-3. GMSU-3 considers switches upgrade and controller placement over multi-stages for a legacy network that supports OSPF. Further, it uses up to two link-disjoint paths to route each control traffic demand. One path is used to route traffic, while

the other serves as backup. The thesis analyses the NP-hardness of GMSU-3 and formulates GMSU-3 as an MIP called MIP-3 which is used as the optimal solution. Further, the thesis develops a heuristic solution called GU-3. The heuristic upgrades a set of l-switches and deploy a new controller only when there are no existing controllers with sufficient capacity and at least a path within delay tolerance and MLU. The thesis also outlines the time complexity of GU-3.

Significance

1. Addressing GMSU-1, GMSU-2, and GMSU-3 problems is significant for network operators that (i) aim to save energy consumed by idle resources to reduce the overall network operation cost, and (ii) have a limited budget or aim to reduce cost to upgrade their network subjects to the network meeting delay requirement and growing volume of traffic demands. We envisage that solutions to the problems would be appealing to some operators who prefer to upgrade their network over multi-stages to minimize the risks that arise from adopting a new technology such as SDN.

2. Solving GMSU-2 is important for operators that prefer to maintain the same multi-path routing services to users.

3. Addressing GMSU-3 is crucial for operators who seek for optimal deployment of s-switches and controllers under maximum delay and growing volume of traffic among switches and between s-switches and controllers.

4. The performance evaluation using five actual network topologies shows that each solution of GMSU-1, GMSU-2, and GMSU-3 produces significant energy saving.

1.3. Thesis Organisation

The contents of each chapter in this thesis are as follows.

Chapter 2: Background

Chapter 2 first describes Energy Saving in Communication networks and Software Defined Networking. It then provides a review of related work on upgrading legacy networks to SDN. Next, it outlines the system model and notations that are used in this thesis. It also provides a brief overview of the problem addressed in this thesis. Lastly, the chapter presents the simulation environment used to evaluate the performance of all proposed algorithms in this thesis.

Chapter 3: Multi-Stage Upgrade for Green SDN/OSPF Networks

Chapter 3 firstly introduces GMSU-1 and provide an example to illustrate the problem. It then presents the mathematical model of GMSU-1 in an ILP called ILP-1 and discusses the NP-hardness analysis of the problem. Next, it describes the proposed heuristic solution for the problem called GU-1. It also includes the analytical analysis of GU-1 with respect to the budget increase, the planning stages, the number of remaining l-switches, and its time complexity. Finally, the chapter evaluates the performance of ILP-1 and GU-1.

Chapter 4: Multi-Stage Upgrade for Green SDN/OSPF-ECMP Networks

Chapter 4 describes GMSU-2 and provides an example to illustrate the problem. It outlines two multi-path routing scenarios, i.e., non-link-disjoint and link-disjoint, and the mathematical model for each of the two scenarios in MIP, i.e., MIP-2a and MIP-2b, respectively. The chapter then presents one heuristic solu-

tion for each of the respective two scenarios, namely GU-2a and GU-2b. Further, it includes the time complexity of both heuristics as well as a proof of their correctness. Finally, Chapter 4 provides the performance evaluation of MIP-2a, MIP-2b, GU-2a, and GU-2b.

Chapter 5: Multi-Stage Upgrade of Switches and Controllers Placement for Green SDN/OSPF Networks

Chapter 5 discusses GMSU-3 and its illustration. It provides the mathematical model of the problem in an MIP called MIP-3 and the NP-hardness of GMSU-3. It then outline the proposed heuristic solution called GU-3 and a proof of its time complexity. Finally, the chapter presents the performance evaluation of both solutions.

Chapter 6: Conclusion

Chapter 6 summarises this thesis and discusses possible areas of future research.

This thesis includes Appendix A that contains copyright information from IEEE conferences and journals in which the author has published.

Chapter 2

Background

This chapter is divided into five sections that contains background materials that are used in Chapters 3, 4, and 5. Section 2.1 describes Open Shortest Path First (OSPF) routing protocol. Section 2.2 introduces energy saving in communication networks and Section 2.3 discusses Software Defined Networking (SDN). Section 2.4 provides related work and research gaps that motivates the work in this thesis. Section 2.5 presents the system models and notations that are used throughout this thesis. Additional notations that are used only in specific chapters are described in their corresponding chapters. Section 2.6 briefly discusses the key problem addressed in this thesis. Section 2.7 presents the network topologies, the value of all parameters included in GMSU, and platform used in the thesis. Finally, Section 2.8 summarises this chapter.

2.1. OSPF Routing Protocol

The Internet comprises of a massive number of autonomous systems operated by different network operators. OSPF is a standard interior gateway protocol for routing traffic within an autonomous system [24]. It is a link-state routing that

requires each link to have a distance metric or a *link cost* that is either the link's interface bandwidth or the physical distance or the communication delay between its end nodes [25]. OSPF uses the link cost to find a path that has the *least cost* called a *shortest path*. The cost of a path is calculated by taking the sum of cost of links that form the path.

Each OSPF router has a collection of Link State Advertisements (LSAs) called Link-State Database (LSDB). Each LSA sent by a router mainly contains information such as the state of its links and their cost. Using LSDB, each router runs Dijkstra's algorithm [24] to construct a tree of shortest paths with the router as the root. The tree gives the shortest path from the router to each destination. The path is stored in the router's routing table and the routing decision is made through the table.

Meanwhile, OSPF-ECMP is an OSPF that supports Equal Cost Multi-Paths (ECMP) [26] that considers multi shortest-paths routing. In OSPF-ECMP, each traffic from a source to a destination is equally split and routed over all of its shortest paths. The goal is to balance links load [25]. Note that in practice the number of shortest paths is pre-defined [24]. OSPF-ECMP has been shown to have a fast reaction to a link failure by switching traffic from the shortest path with failed link to the other available shortest paths [27].

This thesis considers upgrading a legacy network that supports OSPF protocol. More specifically, GMSU-1 and GMSU-3 in Chapter 3 and Chapter 5, respectively, consider a legacy network that supports OSPF. On the other hand, GMSU-4 in Chapter 4 assumes a legacy network that adopts OSPF-ECMP.

2.2. Energy Saving in Communication Networks

Over-provisioning network resources consume unnecessarily more energy that increases network operational cost and global CO_2 emission [11], [28]. In terms of cost, saving energy is important because, as reported by [29], energy expenditure for the commercial sector between 2020 and 2025 is expected to increase by 3.29% on average.

Network providers usually over-provision resources in their network infrastructure to satisfy customer demands in case of network congestion and some possible failures. More specifically, they typically design their network to address traffic demands in peak hours, which are significantly higher than in off-peak times. A report in [30] on the traffic flow over 40 North American and 25 European Internet Service Providers reveals that the difference between traffic rate in peak and off-peak hours is around 60%. Further, the average link utilization of large backbone networks is estimated only between 30% and 40% [12], with a significant number of idle resources such as links and switches [12], [31].

A recent report on Green House Gas Emission (GHGE) footprint [32] shows that ICT activities are predicted to contribute 14% of the global GHGE, which is equivalent to 5100 to 5300 million metric tones of CO_2 by 2040 [32]. Further, according to [28], [32], more than half of the ICT energy usage comes from telecommunication network and data center infrastructure. The aforementioned findings on significant amount of idle network resources and the network infrastructure to CO_2 emission emphasize the importance of reducing network's energy consumption.

Following the seminal work in [33], numerous efforts to reduce energy consumption in network infrastructures have been reported in the literature [11], [34], [35]. The authors in [33], among others, have suggested using (i) energy

aware network components, e.g., line cards (links), to enable the sleep mode when they are idle or carrying a low traffic rate, and (ii) energy-aware traffic engineering to maximize unused network components that can be switched off. This thesis uses both suggested approaches (i) and (ii) in the context of SDN and each link contains multiple cables. More specifically, it considers a network that uses bundled link technology with energy saving feature, i.e., IEEE 802.1AX link consisting of IEEE 802.3az cables, and network traffic rerouting to maximize total number of unused cables that can be turned-off. The details are presented in the following Section 2.2.1 and Section 2.2.2, respectively.

2.2.1. Energy-Efficient Bundled Link Technology

The IEEE has standardized a bundled link technology, i.e., IEEE 802.1AX, where each link is formed from multiple *cables* of the same size and type [12], [36]. The bundled link model is commonly used in practice for large networks because it provides easy way to upgrade network capacity, i.e., by merely adding new cables alongside the existing ones [12]. A link remains *active* if there is at least one cable that is carrying traffic. Advantageously, a link only needs to switch on the minimum number of cables required to carry traffic flows [12]. Specifically, each unused cable in a bundled link equipped with IEEE 802.3az cables [22] is placed into *sleep* mode to save energy. Note that a cable in sleep mode requires only 10% of the total energy used when in active mode. Further, the conversion time from sleep to active mode, and vice versa, takes only a few microseconds, e.g., 2.88µs and 4.48 µs, respectively [22] for a 10 Gbps-IEEE 802.3az. Thus, a cable in sleep mode can be reverted to active mode when its link's capacity is insufficient to carry the current traffic demands.

Some prior works, e.g., in [12], [13], and [14], have addressed energy saving

in large backbone networks that use the bundled links of IEEE 802.1AX. The authors of [37] propose two types of cables in an energy-efficient bundled link, namely, (i) cables with different energy levels and (ii) cables with sleep mode. They find that type (ii) provides more energy saving. This thesis considers SDNs that use the bundled link of type (ii). Another work in [23] considers traffic load distribution among IEEE 802.3az cables in a bundled link to minimize their utilization and thus, the unused cables can be put to sleep mode. It proposes a water-filling algorithm that a cable is used to carry traffic if all current used cables has been used at their maximum capacity. In this thesis, the traffic distribution among cables in a bundled link follows the mechanism proposed in [23].

2.2.2. Network Traffic Rerouting

Traffic Engineering (TE) aims to adjust existing network routing by measuring and analyzing network traffic. The goal is to optimize the utilization of network resources and improve network performance [38], [39]. TE has been extensively employed among others in the Internet Protocol (IP) and Multi-Protocol Label Switching (MPLS) based networks [39]. Network traffic routing plays a key role in TE. It selects a set of paths to optimize some network performances, e.g., load balancing and energy saving.

The selection of paths, among others, considers two important Quality of Service parameters, i.e., maximum link utilization (MLU) and communication delay. Since MLU defines the maximum traffic volume on a link, minimizing each link's MLU can prevent packet loss due to link congestion [40], [41]. The reason is because each link will have residual capacity to handle sudden increase in traffic demands or *flash crowd traffic*. There are four types of delay in a network: propagation delay, transmission delay, queuing delay, and processing delay.

This thesis considers only the propagation delay that is mainly determined by the geographical distance between a source and a destination over the transmission speed [42]. It is reasonable because in a network that spans over a large geographical distance, propagation delays are more significant than the other communication delays [43].

To this end, some solutions for energy-aware traffic rerouting, e.g., in [12]–[15], and [16], have been successfully applied to *legacy* networks. Some solutions used a single path routing [13], while others employed the multi-path routing [12], [14]–[16]. Multi-paths can share common links (*non-link-disjoint paths*) or have no common link (*link-disjoint paths*). Link-disjoint paths provide path resiliency against link failures [44]. This thesis considers both single path routing and multi-path routing that must satisfy the required propagation delay and MLU. Moreover, GMSU-2 problem, discussed in Chapter 4, considers link-disjoint paths.

2.3. Software Defined Networking

A switch in *legacy* networks is responsible for (i) constructing the routing table to control the network and (ii) forwarding packets to the next switch based on the routing rules in the table. The rules in the table are generated per destination from *local* network information using protocols such as OSPF [45]. Since the protocols are usually embedded in the switch's firmware [35], [46], the legacy networks lack of flexibility in configuring and managing network resources. Thus, it is difficult for a network administrator to quickly make any necessary changes to the network to adapt with new and more sophisticated requirements.

In contrast, each switch in SDN performs only the forwarding role, while its controlling role is assigned to a software-based centralized controller. The controller uses global network information to generate rules to control the for-

warding behaviour of *s*-switches. As a result, SDN can greatly improve network programmability and manageability, thereby reducing vendor dependency and promoting open networking, e.g., OpenFlow [47].

SDN has been adopted by some big companies, e.g., Google [6], Microsoft public cloud [7], and NTT cloud gateway [8] for their proprietary network architectures. Google has reported that their deployed SDN provides improved network fault tolerance and link utilization [7]. Similarly, Microsoft uses Software-driven WAN to achieve high utilization for its inter-data-center networks [8]. Further, NTT via OpenFlow is able to efficiently program and control their networks. The following Sections 2.3.1 and Section 2.3.2 provide a discussion on the architecture of SDN and the network upgrade problem from legacy network to SDN, respectively.

2.3.1. SDN Architecture and Protocols

Figure 2.1 shows the architecture of SDN which comprises of data, control, and application planes [48]. The following three sections describe in details the respective three planes.

2.3.1.1. Data Plane

Data plane consists of SDN switches or *s*-switches. An *s*-switch can be a software-based (virtual) switch such as Open vSwitch (OVS) [49] or a hardware-based switch such as Cisco catalyst 9000 Series with OpenFlow-hybrid mode [50]. A virtual switch can run typically in single-core one-GHz CPU [48]. An *s*-switch is composed of (i) a *South-Bound Interface* (SBI), e.g., OpenFlow [47], to communicate with SDN controller, (ii) a set of flow tables to evaluate and take action on each incoming packet of a traffic flow, and (iii) a packet processing function,

where a virtual switch uses a packet processing software while hardware switch embedded the function in Ternary Content Addressable Memories (TCAMs) of Layer 3 and Content Addressable Memories (CAMs) at Layer 2 [48].

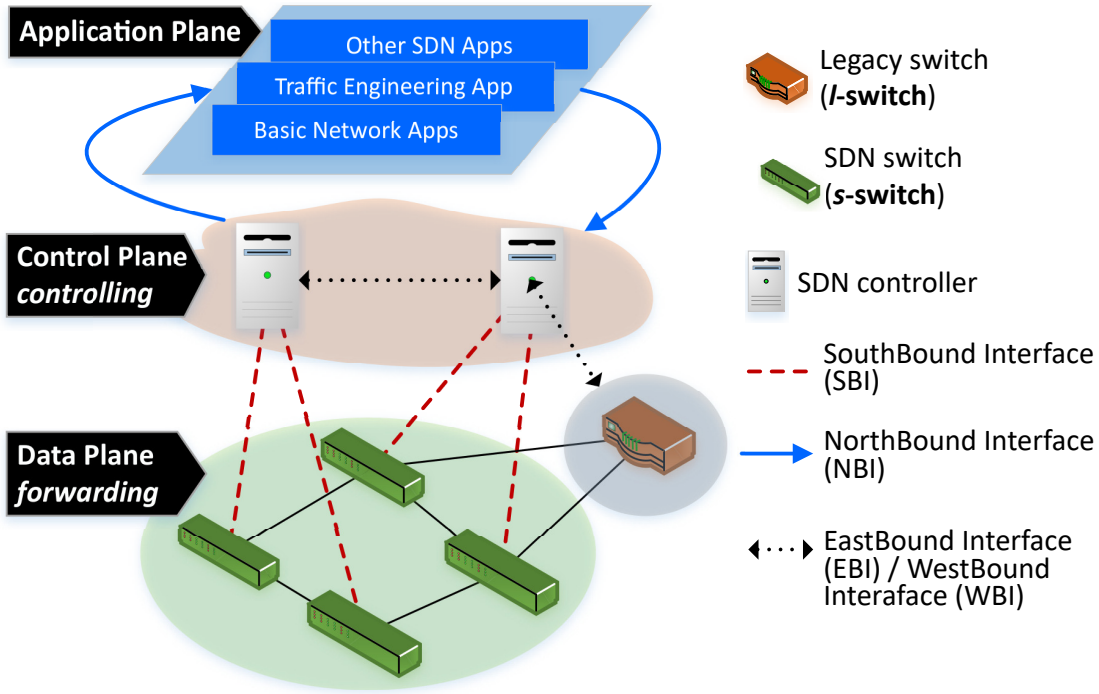


Figure 2.1. SDN architecture that consists of three planes: data, control, and application.

Some types of *s*-switches, e.g., OpenFlow switches [40] and OpenFlow-hybrid switches [51], can be used to communicate with other *s*-switches as well as *l*-switches. All *l*-switches inform their state via their LSAs that are intercepted and forwarded by OpenFlow switches to SDN controller which will extract the information in LSAs [40]. OpenFlow-hybrid switches support OpenFlow operation and normal Ethernet switching operation in forwarding packets [51]. This type of *s*-switch allows any incoming packet to be processed according to OpenFlow protocol or legacy switching/routing/VLAN protocol. The switch can be used to manipulate the routing process in an *l*-switch, e.g., distributing the ad-

justed link cost setting by SDN controllers to the *l*-switches [41], [52]. In this thesis, each *s*-switch is an OpenFlow-hybrid switch.

2.3.1.2. Control Plane

Control plane is a logically centralized controller which can be implemented physically as a single centralized controller or distributed controllers [53]. Example of existing controllers include ONOS [54], Ryu [55], and OpenDayLight (ODL) [56]. A controller is basically a Network Operating System (NOS) with an SBI, a *North-Bound Interface* (NBI), and a set of core network modules [48]. It is typically implemented on a high performance machine, e.g., eight-core two-GHz CPU [48]. A controller uses SBI to send messages to *s*-switches and provide information related to *s*-switches to the controller [10]. The interface allows network topology discovery, defines network flows, and implements request sent by the application plane [10]. A controller uses NBI to inform application plane about events that occur in the network. The events may include a request packet received by the controller from *s*-switches or some changes in the network topology, e.g., link failure. Core modules of a controller includes end-user device discovery, network device discovery, network device topology management, and flow management [48].

A controller maintains the global network topology by periodically requesting its managed *s*-switches to send their connectivity status. A typical mechanism for topology discovery is by flooding Link Layer Discovery Protocol (LLDP) and Broadcast Domain Discovery Protocol (BDDP) packets [40], [47]. Any *s*-switch that receives the packets will forward them back to the controller.

In addition to SBI, each of the physically distributed controllers uses two additional interfaces: (i) *East-Bound Interface* (EBI), and (ii) *West-Bound Interface* (WBI). EBI is used by a controller to share information with the other

controllers. The controllers are organized in either hierarchical model or flat model [10]. Any solution for each of the two models has its own designed EBI. Hierarchical model arranges the controllers into multiple layers of services, e.g., Espresso [57]. On the other hand, all controllers in flat model provide the same services, e.g., ONOS [54]. A controller uses WBI to exchange information with l-switches [10], [53], e.g., BTSDN [58]. Further, OSPF-TE can be used by a controller to discover connectivity between l-switches and obtain the real traffic measurement from the switches [59].

As discussed later in Section 2.3.2.3, an optimized controller placement [60] is needed to achieve a particular network performance. The controller placement problem is to determine the required number and location of controllers as well as their association with s-switches. The work in Chapter 5 addresses the controller placement problem to maximize energy saving.

2.3.1.3. Application Plane

Application plane contains basic and specific network applications. The examples of basic applications include a graphical user interface for managing the controller, a media access control address learning, and routing. Traffic Engineering and Network Security are some of the specific applications [5], [35], [48]. Note that all works in Chapters 3, 4, and 5 develop traffic routing solutions for TE application. Their goals are to maximize energy saving.

Each application in the plane communicates with controller through an NBI, e.g., REST API [10]. In general, each application listens to events such as request for rules for new traffic flows and newly discovered devices such as end user devices and switches [48]. A controller that receives any of the events will invoke the application's callback function. The application then uses the global network information collected by controllers to response accordingly to the event, e.g.,

generate routing and forwarding rules for the new traffic flows. Moreover, the applications may invoke their callback function according to other external input such as from intrusion detection system.

2.3.2. Upgrading Legacy Network into SDN

Upgrading a legacy network to SDN consists of three main tasks: (i) switch upgrade, (ii) network traffic routing, and (iii) controller placement. The following Sections 2.3.2.1, 2.3.2.2, and 2.3.2.3 provide details for the respective three tasks.

2.3.2.1. Switch Upgrade

Switch upgrade involves replacing an l-switch with an s-switch. A legacy network can be converted into a *pure* SDN by directly replacing each l-switch with an s-switch. Some network operators may face two main concerns in adopting *pure* SDNs [3], [5]. First, a significant budget is needed to purchase and set up SDN devices, including the cost to train personnel to manage the SDN. Second, the new or in-progress SDN technologies have not been rigorously tested, unlike those of legacy networks.

The aforementioned concerns lead to the recently proposed *hybrid* SDNs that contain both s-switches and l-switches [3], [5]. Here, l-switches are selectively upgraded subject to the available budget over multiple stages that span several years. The goal is to maximize the benefits of SDNs using the available budget, while keeping the trusted l-switches in the networks. More specifically network upgrade in stages is preferable because, over time, SDN technology is maturing and its cost is decreasing. This thesis considers upgrading a legacy network to *hybrid* SDN over multi-stages.

2.3.2.2. SDN Traffic Routing

Traffic demands in SDN includes *data traffic* which contains data packets between switches as well as *control traffic* between *s*-switch and controller and among controllers. Packets in control traffic between an *s*-switch and its controller is called *control packets* and can originate from the *s*-switch or its controller. For example, an *s*-switch generates control traffic when requesting forwarding rules for a new data traffic flow or when sending its status. On the other hand, a controller generates control packets when sending forwarding rules and maintenance information as well as synchronizing network information with the other controllers.

The *control traffic* between an *s*-switch and its controller as well as among controllers can either use dedicated links, i.e., *out-band channel* or share the same links used by switches, i.e., *in-band channel*. In-band channel is cost-efficient, practical for large networks, and good trade-off between communication performance and infrastructure expenses [61]. On the other hand, out-band channel offers fast and reliable communication [62], [63]. This thesis considers only control traffic between *s*-switches and controllers that uses *in-band channel*.

Both data and control traffic routing in SDN can use a single path or multi-paths. Further, using link-disjoint paths to route control traffic between *s*-switches and controllers provide resiliency against link failures [64]. Our work in Chapter 5 considers two link-disjoint paths to route each control traffic demand. In addition, each selected path to route a traffic can satisfy some performance requirements, e.g., delay and MLU, and can be determined by a controller without relying on link cost as in OSPF and OSPF-ECMP. [39].

In a *hybrid* SDN, the controller and legacy protocols such as OSPF co-exist together. This means *l*-switches control some parts of the network traffic, while other parts are directed by the controller [3]. OSPF that supports TE, i.e., OSPF-

TE [65], [66], is used by controllers to gather the network information such as link cost from l-switches [59]. OSPF-TE distributes additional link information such as link propagation delay, delay variation, link loss, residual bandwidth, available bandwidth, and utilized bandwidth. Thus, the link information can be used by an SDN controller to obtain the real traffic measurement from each l-switch [59]. Simple Network Management Protocol (SNMP) is another alternative to collect real time traffic information [40]. Further, OSPF-TE can be used to influence the routing decision of the l-switches by re-setting the cost of each link [41], [52]. This thesis assumes each l-switch supports OSPF-TE. Further, our work in Chapter 4 considers re-setting the cost of each link to ensure that each l-switch follow the OSPF-ECMP protocol.

2.3.2.3. SDN Controller Placement

Physically, control plane consists of either one centralized controller or distributed controllers. Using a single centralized controller is simple. However, a controller has limited capacity to process requests from s-switches. Thus, it is prone to single point of failure issue where the controller can be overloaded, which in turn results in unacceptable switch-to-controller delay. In this case, operators may deploy more controllers to share the load on processing the requests from s-switches. A key design problem is to determine the minimum number of controllers and their location so that their association with s-switches achieves a defined goal, e.g., communication in terms of delay, reliability, and energy saving. This so called controller placement problem is non-trivial [67].

Controllers can be located in a dedicated network [63] or each of them can be placed at the same location of a switch [19], [68]–[72]. On the other hand, placing controllers with s-switches reduces traffic delay and ensures high communication reliability [71]. An s-switch can either be associated to exactly one controller [19],

[69]–[72], or to more than one controllers [64]. Further, a controller can manage a limited number of *s*-switches. The maximum number of control packets per seconds (*cpps*) is often used as the capacity of a controller [68], [70]–[72].

The controller placement addressed in Chapter 5 assumes (i) each controller is located at the same location of an *s*-switch, (ii) each *s*-switch is associated to one controller, and (iii) the number of switches that can be associated to a controller is limited based on its capacity to process the total *cpps* transmitted by the switches.

2.4. Related Work

Table 2.1 summarizes some existing works that are relevant to the works in this thesis. The following four sections discusses them in detail. More specifically, Section 2.4.1 discusses works on energy saving in legacy networks. Section 2.4.2 presents some efforts on energy saving in pure SDNs. Both Section 2.4.3 and Section 2.4.4 consider works that upgrade a legacy network into a *hybrid* SDN. However, the works in Section 2.4.3 are for a single stage upgrade, while those in Section 2.4.4 are for multi-stage.

2.4.1. Energy saving in Legacy networks

The bundled link technology IEEE 802.1AX [21] has been adopted in the backbone networks to provide more flexible bandwidth allocation in each link. More specifically, during off-peak hours, unused cables in each link can be switched off to reduce the overall energy consumption of a network [23], [37]. Prior research such as [12], [13], and [14] aim to maximize energy saving in a legacy network that supports bundled link technology. They propose energy-aware routing to maximize the number of unused cables that can be switched off.

Table 2.1. A summary of related work.

Scheme	References	Scope				
		ES	**SU**	**TR-SP**	**TR-MP**	**CP**
Legacy	[13]	✓	–	✓	×	–
Networks	[12], [14]–[16]	✓	–	×	✓	✓
Pure SDNs	[19], [20]	✓	–	✓	×	✓
Hybrid SDNs (Single stage)	[40], [41]	×	✓	×	✓	×
	[17]	✓	✓	×	✓	×
	[2], [18], [73]	✓	✓	✓	×	×
	[69]	×	✓	✓	×	✓
Hybrid SDNs (multi-stages)	[1]	×	✓	✓	✓	×
	[70]–[72]	×	✓	✓	×	✓
	Our work	✓	✓	✓	✓	✓

ES: Energy Saving SU: Switch Upgrade CP: Controller Placement
TR-SP: Traffic Routing via a Single Path
TR-MP: Traffic Routing via Multi-Paths
–: Not Applicable ×: not addressed ✓: addressed

Fisher *et al.* [12] propose an ILP and the Fast Greedy Heuristic (FGH) to power-off as many unused cables as possible. FGH routes each traffic demand via multi-paths. It iteratively removes one cable from the network and solve the ILP using the pruned network until no feasible solution is obtained.

Lin *et al.* [13], [14] consider energy saving under maximum delay and MLU constraints in backbone networks that support bundled links. Both efforts propose ILP and a set of heuristic solutions to turn off a maximum possible number of unused cables. The authors of [13] assume a single path routing and all links consume the same energy. On the other hand, the authors of [14] use MPLS-based multi-path routing and assume that each link and l-switch have different power usage.

Moulierac *et al.* [16] and Lin *et al.* [15] maximize the energy saving of legacy networks with non-bundled links. The authors of [16] employ OSPF-ECMP based

multi-path routing that satisfies MLU. The authors of [15] show how to save energy *legacy* networks while maintaining link-disjoint multi-paths.

Similar to references [12], [13] and [14], the work in this thesis aims to maximize energy saving by switching off the maximum number of unused cables. However, our work addresses energy saving in SDN.

2.4.2. Energy saving in Pure SDNs

The locations of controllers have been shown to have a significant impact on energy consumption [19], [20]. This in turn motivates research into energy-aware controller placement solutions [19] and traffic routing between *s*-switches and their associated controller [20].

Ruiz-Rivera *et al.* [20] consider *s*-switches and controller associations in a pure SDN. It proposes a heuristic-based energy-aware routing solution called GreCo. GreCo associates an equal number of switches to each of a given set of controllers. It selects a path within the bounded delay between each switch and its controller. It uses the links in selected paths between switches and their controller to carry data traffic between switches, and then turns off unused links. In GreCo, each controller also acts as a forwarder.

The work in [74] addresses the overloaded-link probability of GreCo by disallowing each path carrying traffic among switches to use any controller as a forwarder (except for source or target). The controllers only belong to paths carrying traffic between switches and controllers.

Hu *et al.* [19] finds the best locations to deploy a fixed number of controllers. They propose an Improved Genetic Controller Placement Algorithm (IGCPA) which is an energy-aware controller placement solution for a *pure* SDN. IGCPA uses GreCo to perform association between switches and controllers along with

data and control traffic routing that yields minimum number of unused links.

In Chapter 5, GMSU-3 optimizes controller placement for hybrid SDNs over multi-stage. Similar to [19], the goal is energy saving, but the number of deployed controllers is bounded by a given budget.

2.4.3. Hybrid SDNs - Single Stage Upgrade

Some strategies of traffic routing have been successfully implemented in a *hybrid* SDN. Guo *et al.* [41] have addressed traffic routing problem for a given set of *s*-switches. On the other hand, Hong *et al.* [40] jointly address switch upgrade and traffic routing.

The authors of [41] exploit the advantage of using TE in *hybrid*-SDN and consider multi-path routing for each traffic demand. Given the upgraded switches, they optimize the links weight of *l*-switches and splitting ratio at *s*-switches to minimize MLU of entire *hybrid* SDN. However, they do not bound the delay for each of the multi-paths used to route traffic. The authors develop a greedy algorithm to replace a set of *l*-switches with the highest total traffic load on their outgoing links with *s*-switches. Using the given placement of the *s*-switches, they propose a heuristic algorithm called SDN/OSPF Traffic Engineering (SOTE) to find the cost of each link and traffic splitting ratio at each *s*-switch that give the possible minimum MLU.

The authors of [40] also aim to minimize MLU subject to the maximum percentage of upgraded *l*-switches, maximum capacity of all links, and each traffic demand residing at any *s*-switch must be routed via a primary shortest path and one or more backup paths. They propose a greedy-based heuristic solution to upgrade a given maximum percentage of *l*-switches with the highest number of incoming and outgoing links. Their solution routes each traffic flow that reaches

an *s*-switch: (i) via a path consisting of links with the highest spare capacity, or (ii) is split over some *weighted* multi-paths.

Other works in [2], [17], [73] and [18] have considered a *green hybrid* SDN where the location of a single centralized controller are pre-defined. They use OSPF routing for all *l*-switches. Among them, efforts in [17], [73] and [2] minimize power consumption of *hybrid* SDNs by optimizing network traffic routing with a given set of *s*-switches. On the other hand, the work in [18] minimizes the network's power consumption by optimizing both switch upgrades and the shortest path traffic routing.

Wei *et al.* [17] randomly and uniformly distribute *s*-switches in a *hybrid* SDN. They use the fixed placement of *s*-switches to optimize OSPF-based routing for *l*-switches and multi-path routing for *s*-switches. The goal is to minimize power consumption of links by enabling each *s*-switch to split traffic. The authors of [17] jointly optimize the shortest path routing for *l*-switches via link-cost adjustment and traffic-splitting over multi-paths for *s*-switches. Their energy-aware routing solution, called HEATE, moves traffic flow from the low utilization links to those with the high utilization by re-adjusting the OSPF cost of the links.

Efforts by Wang *et al.* [73], Galán-Jiménez [2], and Huin *et al.* [18] aim to minimize power usage of both *s*-switches and their adjacent links. They use OSPF-based shortest path routing. The authors of [73] select a set of *l*-switches with the highest number of incoming and outgoing links that are not connected to any *s*-switch. They construct groups of minimum spanning trees consisting of all paths that include *s*-switches to produce the routing rules. Any underutilized links and *s*-switches is powered-off, while any unused legacy elements are assumed to stay active.

The authors of [2] propose an Energy Efficient Genetic Algorithm for *Hybrid*-SDN (EEGAH). Given a set of *s*-switches, EEGAH finds the maximum possible

number of s-switches and links adjacent to an s-switch that can be off. EEGAH ensures that the remaining active links can route a given set of traffic demands. EEGAH consider four different criteria: (i) most non-controlled links first (MNL), (ii) least non-controlled links first, (iii) partitioning network into two areas, and (iv) partitioning network into three areas, and (v) random. Their simulation shows that MNL provides higher energy saving than the other criteria (ii) to (v).

The authors of [18] assume OSPF routing for all l-switches and single path routing for all s-switches. They develop a heuristic algorithm called Smooth ENergy Aware Routing (SENAtoR) that upgrades the first m switches in decreasing order of their number of incoming and outgoing links. It finds a set of shortest paths that uses a minimum number of s-switches and links adjacent to an s-switch. Further, it sets any switch with an unused link to reroute its outgoing traffic to another link belonging to a pre-computed path or tunnel with all of its links are active. In addition, SENAtoR includes mechanism of reactivating the turned-off s-switches and links to deal with traffic spikes and link failure by using traffic analysis and estimation.

In addition, Guo *et al.* [69] jointly upgrade l-switches to s-switches and deploy multiple controllers in *hybrid* SDNs in a single planning stage. They consider two objectives: (i) maximize traffic flows that pass s-switches or *traffic controllability* and (ii) minimize the propagation delay between s-switches and controllers. They solve the first problem with objective (i) to produce the placement of s-switches and use the placement to solve the second problem with objective (ii).

The aforementioned works on energy saving in this section consider upgrading a legacy network to a hybrid SDN in a single stage. In addition, the works aim to upgrade only switches; i.e., they do not address controller placement. Further, each link contains only one cable. On the other hand, this thesis considers multi-stage upgrade and networks with bundled link model so that each unused cable

can be individually turned off. In addition, GMSU-3 in Chapter 5 also considers controller placement in the network upgrade.

2.4.4. Hybrid SDNs - Multi-Stage Upgrade

The optimization of switch upgrade and traffic routing over multi-stages has been proposed, among others, by Caria *et al.* [75], Das *et al.* [76], and Poularkis *et al.* [1]. Their aim is to find the maximum number of available paths, which may not be the shortest route, to deliver each traffic demand. Here, a path is deemed *available* if all switches which make routing decision on the path, are s-switches. Further, the three references do not address controller placement.

The authors of [75] limit the number of upgraded l-switches to the ratio between the total number of l-switches and a given upgrade number of stages. They propose ILP-based solution for the problem. The authors of [76] extend the work in [75] by using also a total budget (in $) to constraint the number of upgraded switches over all stages. They consider a fixed upgrade cost (in $) for each l-switch. Further, they propose two greedy solutions that use the ratio and the total budget, respectively, to limit the number of upgraded switches. Each solution greedily selects an l-switch with the highest number of available paths.

Given a total budget (in $), the authors of [1] aim to upgrade l-switches in order to maximize the number of available paths over T stages. They consider an upgrade cost that decreases over time and assume that traffic size (in bytes) increases over multi-stages. A variant of greedy algorithm is proposed to solve the problem. Another work in [77] propose an Ant Colony Optimization Rank-based algorithm to solve the problem. In addition, the authors of [1] aim to maximize traffic controllability, i.e., traffic flows that passes through at least one s-switch, subject to a shortest path routing. They propose *Local Search* algorithm, which

is publicly made available in [78], to solve the later problem.

Some works on *hybrid* SDNs, namely [70], [71], and [72], optimize switch upgrades and controller placement over multi-stages. The authors of [70] consider to solve two sub problems: (i) minimize the cost to setup and operate each controller subject to a given sum of maximum two-round trip propagation delay and processing delay at the controller, i.e., *control delay*, and (ii) minimize the maximum control delay subject to the number of controllers upgraded at each stage produced by solving problem (i). They assume the request packet rates of each s-switch increases at each subsequent stage. The authors of [71] address a similar issue to problem (ii) of [70]. However, the goal is to minimize the failure probability of paths between switches and controllers over multi-stages. The goal subjects to the maximum number of deployed controllers at each stage. The authors of [71] do not bound the maximum path delay. Both works in [70] and [71] assume no backup controller(s), i.e., each s-switch is only associated to one controller. The authors of [72] consider switch upgrades and controller placement that aim to minimize flow-setup delay, path failure, and controllers load. The aim subjects to a given number of controllers and their capacity.

Different from the works in [1], [70]–[72], [75], [76], the works in this thesis aim to maximize energy saving. Further, unlike the work in [1], our works also consider budget constraint at each stage.

2.5. System Model

This section describes system models and notations that will be used in all chapters in the thesis. More specifically, the following five subsections discuss the models for network, traffic flow, path delay, budget, and energy saving. Note that additional models and notations that are specific to each problem, i.e., GMSU-1, GMSU-2, and GMSU-3, will be discussed in their corresponding Chapters 3, 4,

and 5, respectively. The table in page xxxi summarizes the notations used in all models.

2.5.1. Network Model

Let $G^0(V, E)$ be a *legacy* network to be upgraded into an SDN. The network has $|V|$ nodes that represent the location of l-switches, where $|.|$ is the cardinality of a set. There are $|E|$ *directed* links. The set E contains both link (u, v) and (v, u), for $u, v \in V$ and $u \neq v$. Let b_{uv} be the number of cables or the *bundle size* of each link $(u, v) \in E$. All cables have the same capacity γ (in bits per second or bps). Thus, the total capacity of link (u, v) is $c_{uv} = \gamma \times b_{uv}$. Let π_{uv} (in seconds) represent the propagation delay of link (u, v). This thesis assumes the legacy network $G^0(V, E)$ has sufficient capacity to route any traffic demand via its shortest path in each upgrade stage. Further, the delay of each link (u, v) is proportional to the distance between nodes u and v, which is approximately 0.0005 seconds per meter [43].

A network operator aims to upgrade $G^0(V, E)$ into an SDN over one or more time periods or stages, denoted as $T \geq 1$. The duration of each stage t can be determined by the lifetime of a network device, e.g., three to five years [1], or can be in shorter range, e.g., months. At each stage $t \in \{1, 2, \ldots, T\}$, the operator upgrades a set of l-switches to s-switches. This thesis uses V^t to denote the set of s-switches that have been installed up to stage t. Each s-switch is able to communicate with both l-switch and s-switch. Let $\mathcal{V}^T$ denote the percentage of l-switches that have been upgraded into s-switches over T stages, i.e., $\mathcal{V}^T = |V^T|/|V| \times 100\%$. Further, let $G^t(V, E)$ be the resulting network after undergoing an upgrade at stage t. Each link $(u, v) \in E$ in $G^t(V, E)$ is a c-link if it is adjacent to at least one s-switch, i.e., $u \in V^t$ and/or $v \in V^t$; otherwise it is an l-link.

2.5.2. Traffic Flow Model

A traffic flow is a single commodity with a source node, a destination node, and size; these parameters are denoted as s_d, τ_d, and ω_d^t, respectively. Let D^t denote a set of commodities or traffic demands in $G^t(V, E)$, i.e., $D^t = \{(s_d, \tau_d, \omega_d^t) \mid d \in \{1, 2, \ldots, |D^t|\}\}$. Each commodity has either a different source s_d or a different destination τ_d compared to other commodities. Further, this thesis sets $\omega_d^1 = \omega_d^0$, where ω_d^0 is the traffic volume of demand $d \in \{1, 2, \ldots, |D^0|\}\}$ in $G^0(V, E)$ and $D^1 = D^0$. Note that the total number of traffic demands is assumed to remain the same over T stages, i.e., $|D^t|$ is constant. The size of each demand d increases at a rate of $\mu_d \geq 0$ per stage. Thus, $\omega_d^t = \omega_d^0 \times (1 + \mu_d)^{t-1}$. The work in this thesis assumes that the legacy network $G^0(V, E)$ has a sufficient capacity to carry all demands at their maximum volume, i.e., ω_d^T for each demand d.

2.5.3. Routing Model

Each demand $d \in \{1, \ldots, |D^t|\}$ has a set of alternative paths from source node s_d to destination node τ_d. These paths are recorded in a set $P_d = \{P_{d,i} \mid \forall i \in \{1, \ldots, |P_d|\}\}$. The delay over path $P_{d,i}$, denoted by $\delta_{d,i}$ (in seconds), is computed as the sum of propagation delays over all its links, i.e., $\delta_{d,i} = \sum_{(u,v) \in P_{d,i}} \pi_{uv}$. The work in this thesis ignores transmission and queuing delay because propagation delay dominates when a network spans a large geographical distance [43]. Let $\delta_{\min,d}$ be the minimum delay among all paths in P_d, and $\delta_{\max,d}$ be the maximum path delay for demand d. A path $P_{d,i} \in P_d$ is called the *shortest path* if its delay is equal to the minimum delay $\delta_{\min,d} = \min_{P_{d,i} \in P_d}\{\delta_{d,i}\}$. This thesis considers two alternative constraints of maximum path delay: (i) delay that is computed for a given multiplier $\sigma \in [1.0, 2.0]$ used in [14], i.e., $\delta_{\max,d} = \lceil \sigma \times \delta_{\min,d} \rceil$, and (ii) delay that is computed from the network's diam-

eter, i.e., $\delta_{\max,d} = \max_{d\in\{1,\ldots,|D^0|\}}\{\delta_{\min,d}\}$; in words, the maximum delay among the shortest paths for all demands in $G^0(V,E)$. Delay constraint (i) is for users that can use a path of up to $(\sigma-1)\times 100\%$ longer than its original shortest path. On the other hand, delay constraint (ii) considers a network in which users can tolerate a delay that is no longer than the network diameter.

Let $\mathcal{P}_d \subset P_d$ denote a set of paths, each of which has delay within $\delta_{\max,d}$. Finally, let R^0 and R^t denote two sets of all paths used to route each demand d in $G^0(V,E)$ and $G^t(V,E)$, respectively. Each path in either sets has delay within $\delta_{\max,d}$.

2.5.4. Budget Model

Let B be the total budget (in \$) over T stages, and $B^t \leq B$ is the maximum available budget at each stage t. This thesis sets $B^t = B/T$. Any unused budget in stage t, denoted by ΔB^t, can be spent in subsequent stages. Define p_u^t to denote the cost of upgrading or replacing l-switch u with an s-switch that supports bundled-link in stage t. The cost includes the s-switch's purchase price and installation cost. The upgrade cost of switches may vary due to differences in their models and types, e.g., edge or core switches [1]. Notation ρ is used to denote the depreciation rate in upgrade cost of each switch per stage, where $0 \leq \rho < 1$. Hence, the upgrade cost of l-switch u at stage t is given as $p_u^t = p_u^0 \times (1-\rho)^{t-1}$, where p_u^0 is the initial cost. For example, if the l-switch u is replaced at the second stage $t = 2$, s-switch u is purchased and installed at that stage, where the cost is depreciated by rate ρ at that second stage.

2.5.5. Energy saving Model

This thesis computes energy saving according to the total number of *unused* cables, i.e., cables that do not carry traffic are switched off by powering off their line card. Specifically, a cable can be powered off, called *off*-cable, if it is connected to at least one s-switch, i.e., a cable of a c-link. Note that line cards contribute significantly to a router's energy consumption [12]. Thus, without loss of generality, this thesis assumes the energy consumption of a cable is equivalent to that of its line card. Following [12] and [13], each cable that carries traffic, called *on*-cable, consumes the same amount of energy. For example, an *on*-cable with 1% load and another with 100% load consume the same amount of energy. Further, as per [73] and [18], a cable in a c-link can be powered-off when it has zero flow rate. It is possible as each cable supports IEEE 802.3az [22]. Each c-link can turn on an *off*-cable if it has insufficient capacity.

Let f_{uv}^t denote the total volume of traffic demands carried by link (u, v) at stage t. Further, let n_{uv}^t, for $0 \leq n_{uv}^t \leq b_{uv}$, be the required number of *on*-cables to carry traffic volume f_{uv}^t. Also, let $U_{\max}$ be the maximum link utilization threshold, for $0 \leq U_{\max} \leq 1$. Note that this threshold also applies to the maximum capacity of each cable which is $\gamma \times U_{\max}$. Thus, we have $n_{uv}^t = \lceil f_{uv}^t/(\gamma \times U_{\max}) \rceil$ and the maximum capacity of link (u, v) at stage t is given as $c_{uv}^t = (n_{uv}^t/b_{uv}) \times c_{uv} \times U_{\max}$. Without loss of generality, this thesis assumes each l-switch does not turn off unused cables, i.e., the switch does not comply with the IEEE 802.3az [22] standard. Thus, if switch u and v are both l-switches, all cables between the two switches are on, i.e., $n_{uv}^t = b_{uv}$.

The energy saving at stage t, denoted by ε^t, is formally computed as

$$\varepsilon^t = \frac{\sum\limits_{(u,v)\in E} (b_{uv} - n^t_{uv})}{\sum\limits_{(u,v)\in E} b_{uv}}. \tag{2.1}$$

In words, the energy saving ε^t is a ratio between the total number of *off*-cables and the total number of cables. The average energy saving over T stages is then computed as

$$\varepsilon_T = \frac{1}{T} \sum_{t=1}^{T} \varepsilon^t. \tag{2.2}$$

2.6. Overview of Green Multi-Stage Upgrade

This section provides an overview of the problem addressed in this thesis, i.e., Green Multi-Stage Upgrade (GMSU). GMSU consists of three novel problems or versions: GMSU-1, GMSU-2, and GMSU-3. Each version of GMSU is to upgrade a legacy network $G^0(V, E)$ to an SDN $G^T(V, E)$ over a given T planning stages. Further, each version aims to maximize the number of unused cables that can be turned off per Eq. (2.2) to save energy. All versions consider the following three main constraints:

C1 - **Budget** $B^t = B/T$: The total cost to upgrade network $G^{t-1}(V, E)$ to network $G^t(V, E)$ at each stage t does not exceed the maximum budget B^t. GMSU considers depreciation of upgrade cost per stage.

C2 - **Routing**: Each path selected to route traffic from an l-switch or from an s-switch to any destination must have delay that satisfies the pre-defined maximum path delay $\delta_{\max,d}$.

C3 - **Link Capacity**: The total traffic volume of each link at each stage t must

be within the maximum utilization of the link, i.e., MLU threshold $U_{\max}$. GMSU considers the traffic volume of each demand increases at each stage.

Note that each version has its specific requirements for constraint C2. Further, GMSU-3 has additional requirement for constraint C1.

Table 2.2 summarizes the similarities and differences among the three versions of GMSU. In addition to traffic routing, GMSU-1 and GMSU-2 consider switch upgrade while GMSU-3 addresses both switch upgrade and controller placement. In this case, the maximum budget B^t in GMSU-3 must be sufficient to deploy both s-switches and controllers. GMSU-1 and GMSU-3 consider an l-switch that supports OSPF, while GMSU-2 assumes an l-switch that follows OSPF-ECMP rule. For s-switches, GMSU-1 and GMSU-3 consider a single path to route data traffic, while GMSU-2 uses multi-path routing. However, GMSU-3 uses active and backup paths that are link-disjoint to route control traffic. Further, GMSU-3 requires three additional constraints that are related to the controller placement: controller location, its capacity, and its association with s-switches. Chapters 3, 4, and 5 describe the details of GMSU-1, GMSU-2, and GMSU-3, respectively.

2.7. Simulation Environments

This section describes the network topologies, the set of traffic matrices, the value of eight main parameters, and the platform used in each simulation performed in Chapters 3 through 5.

2.7.1. Topologies and Traffic Matrices

The performance of the proposed algorithms in Chapters 3, 4, and 5 are evaluated using five network topologies listed in Table 2.3. Briefly, Abilene was a research

Table 2.2. The summary of similarities and differences between GMSU-1, GMSU-2, and GMSU-3.

Subjects	**Description**	**GMSU Versions**		
		GMSU-1	**GMSU-2**	**GMSU-3**
Goal	Maximize total number of *off*-cables	✓	✓	✓
Upgrading Tasks	Switch upgrade	✓	✓	✓
	Traffic routing			
	- *Data traffic*	Single path	Multi-paths	Single path
	- *Control traffic*	×	×	Active and backup paths
	Controller placement	×	×	✓
Constraints	Budget	deploy s-switches	deploy s-switches	deploy s-switches & controllers
	Routing			
	- *Delay*	✓	✓	✓
	- *Legacy routing*	OSPF	OSPF-ECMP	OSPF
	Link capacity	✓	✓	✓
	Controller placement	×	×	✓

×: not considered ✓: considered

and education network in the USA, GÉANT is a research and education network in Europe, the Deutsche Forschungsnetz network (DFN) is used for science and research in Germany, Deltacom was a regional network carrier in USA, and TATA Communications Limited is an Indian telecommunications company [79]. For Abilene and GÉANT, this thesis uses their actual traffic matrices. As for DFN, Deltacom and TATA, the gravity model [80] is used to generate their synthetic traffic demands because there are no publicly available traffic matrices. Each experiment in Chapters 3, 4, and 5 sets the capacity of each cable to $\gamma = 2.5$ Gbps, and each link contains four cables; thus, each link has a capacity of 10

Gbps.

Table 2.3. List of real topology used for evaluation.

Name	$\|V\|$	$\|E\|$	$\|D\|$	**Traffic**
Abilene [81]	12	30	132	Real
GÉANT [82]	23	74	466	Real
DFN [79]	58	174	3306	Synthetic
Deltacom [79]	113	322	12656	Synthetic
TATA [79]	145	372	20880	Synthetic
Each network has: - Bundle size $b_{uv} = 4$ - Cable capacity $\gamma = 2.5$ Gbps				

2.7.2. Parameters Value

The three versions of GMSU, i.e., GMSU-1, GMSU-2, and GMSU-3, consider the following seven main parameters: the total number of planning stages T, the total budget B, a switch initial upgrade cost p_u^0, a depreciation rate in upgrade cost ρ, a growing rate of traffic volume μ_d per demand $d \in \{1, \ldots, |D|\}$, a predefined path delay constraint $\delta_{\max,d}$, and an MLU threshold $U_{\max}$.

The performance evaluation of the proposed solutions for all versions use the following values for the seven parameters. We consider the number of stages T from one to five stages and various budget values of $B \in \{\$200K, \$400K, \$600K, \$800K, \$1M, \$1.2M\}$. As per [1], the depreciation rate of upgrading a switch is $\rho = 40\%$ and traffic increases at the same rate of $\mu_d = 22\%$ for all demands. Further, each switch is assigned an initial upgrade cost as follows. Using a Normal distribution $\mathcal{N}(2, 0.5)$, the work in this thesis generates a set of $|V|$ real numbers with values between 1.0 and 3.0. It then rounds each generated number to its nearest integer value and associate the integer to each switch $u \in V$. A switch u

is assigned an initial cost p_u^0 of \$50K, \$100K, or \$150K if its associated integer has a value of 1, 2, or 3, respectively. The delay multiplier is set to be 10% longer than each demand's original shortest path $\delta_{\min,d}$, i.e., $\sigma = 1.1$. Thus, the maximum delay of each path to route demand d is $\delta_{\max,d} = \lceil 1.1 \times \delta_{\min,d} \rceil$. Finally, the MLU threshold $U_{\max}$ of each link is set to 80%. Table 2.4 summarizes the seven aforementioned parameters and their values.

Table 2.4. List of parameters and their values for simulation.

Parameter	**Value**
Planning stages T	$\{1, 2, 3, 4, 5\}$
Total Budget B	{\$200K, \$400K, \$600K, \$800K, \$1M, \$1.2M}
Initial switch cost p_u^0	{\$50K, \$100K, \$150K}
Cost depreciation rate ρ	40%
Traffic increase rate μ_d	22%
Delay multiplier σ	1.1
MLU Threshold $U_{\max}$	80%

2.7.3. Platform

All proposed ILP/MIPs are solved by Gurobi Solver [83] and heuristic algorithms are implemented in C++. All experiments are conducted on a 64-bit Windows machine with an Intel-core-i7 CPU @3.60 GHz and 16 GB of memory.

2.8. Chapter Summary

This chapter covers the background of this thesis. It describes OSPF and OSPF-ECMP routing protocol, introduces the energy saving in communication networks, and discusses the current emerging technology called Software Define networking. Further, it provides related literature and identify the research gap that

motivates the work in this thesis. In addition, it presents the system model and notations used in thesis and briefly introduces three versions of our proposed optimization problem called GMSU. Finally, it provides the simulation environments to evaluate all proposed solutions to solve GMSU. The following three chapters provide the details of the three versions of GMSU, i.e., GMSU-1, GMSU-2, and GMSU-3.

Chapter 3

Multi-Stage Upgrade for Green SDN/OSPF Networks

This chapter addresses GMSU-1 problem to upgrade a legacy network that supports bundled links and OSPF to SDN over a given planning stages. GMSU-1 routes each traffic demand via a single path. The problem aims to turn off a maximum number of unused cables that are adjacent to s-switches to save energy. The goal considers constraints on maximum budget at each stage, maximum delay tolerance, and MLU. The layout of this chapter is as follows. Section 3.1 formulates GMSU-1, Section 3.2 describes the proposed heuristic algorithm to solve GMSU-1, Section 3.3 presents the simulation results, and Section 3.4 concludes this chapter. Note that the work in this chapter has been published in [84] and [85].

3.1. Problem Formulation

Section 3.1.1 formally defines GMSU-1 and presents some notations related to single path routing used in this chapter. Note that Section 2.5 discusses the other

models and notations used in this chapter. Section 3.1.2 provides illustration of GMSU-1 and Section 3.1.3 presents its mathematical model as an Integer Linear Programming (ILP). Finally, Section 3.1.4 shows that GMSU-1 is NP-hard.

3.1.1. GMSU-1 Problem Definition

Given a legacy network $G^0(V, E)$ that supports OSPF, GMSU-1 replaces a subset of l-switches in $G^0(V, E)$ with a set of s-switches V^T over a given $T \geq 1$ stages. GMSU-1 aims to maximize the number of unused cables that can be turned off per Eq. (2.2); equivalently, it minimizes the number of *on*-cables. The aim is subject to the following three main constraints:

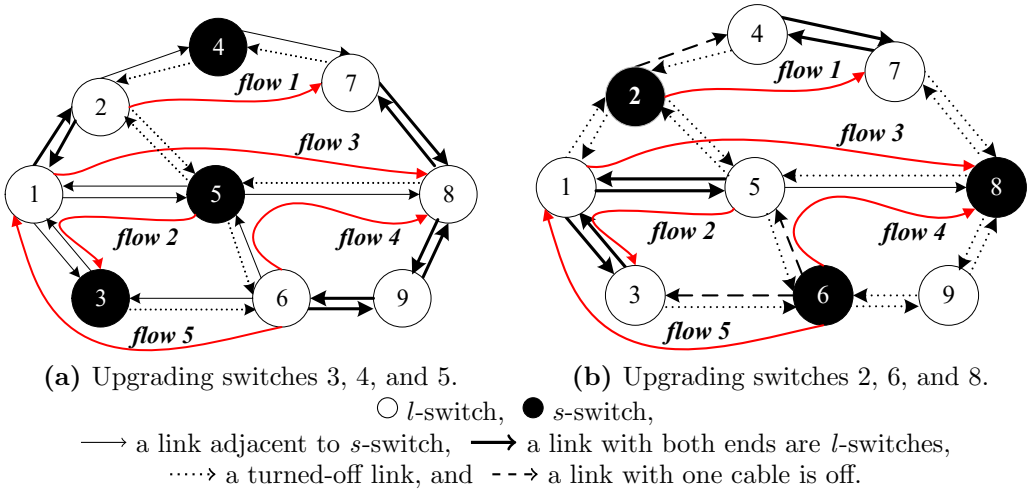
1. **Budget** B: Total cost to upgrade l-switches at each stage t does not exceed the maximum budget B^t. Recall that (i) $\mathrm{B} = \sum_{t=1}^{T} B^t$, and (ii) the upgrade cost of each switch depreciates with a given rate ρ per stage,

2. **Routing**: Each data traffic demand $d \in \{1, \ldots, |D^t|\}$ at each stage t must be routed via a single path, denoted by R_d^t, that satisfies a given path delay constraint $\delta_{\max,d}$.

3. **Link capacity**: The total traffic volume on each link $(u, v) \in V$ at each stage t is within the maximum capacity of the link $c_{uv}^t = (n_{uv}^t / b_{uv}) \times c_{uv} \times U_{\max}$. Here, GMSU-1 considers the traffic volume of each demand d at each stage t, i.e., ω_d^t increases with a rate of μ_d per stage.

3.1.2. GMSU-1 Illustration

To illustrate GMSU-1, consider the topology of a legacy network $G^0(V, E)$ in Figure 3.1. The network has $|V| = 9$ nodes and $|E| = 24$ directed links. Assume the total budget is $B = \$40$ and the upgrade is performed over $T = 2$ stages. The



(a) Upgrading switches 3, 4, and 5. **(b)** Upgrading switches 2, 6, and 8.

Figure 3.1. An illustration of (a) upgrading switch 5 in stage 1 followed by switch 3 and switch 4 in stage 2, and (b) upgrading switch 2 in stage 1 followed by switch 6 and switch 8 in stage 2.

budget for each stage t is initially set to $B^t = \$20$, i.e., $B^1 = B^2 = \$20$. In the first stage, the upgrade cost of each switch u is $p_u^1 = p_u^0 = \$15$ but it is decreased by rate $\rho = 20\%$ in the second stage, i.e., the upgrade cost in stage 2 is reduced to $p_u^2 = \$12$. In stage 1, we can upgrade only one switch. The remaining budget, $\Delta B^1 = \$20 - \$15 = \$5$, is added to the budget of stage 2, which is now equal to $B^2 = \$20 + \$5 = \$25$. This budget allows the upgrade of two more switches. Let each link have a capacity of ten units and the maximum link utilization $U_{\max}$ is 100%. Further, assume there are five flows, i.e., *flow 1*, *flow 2*, *flow 3*, *flow 4*, and *flow 5*, with a traffic volume of 2.4, 1.2, 4.8, 1.2, and 0.6 units, respectively. Assume the volume of each traffic demand is increased by rate $\mu_d = 0\%$. Thus, the sets of traffic demands $D^2 = D^1 = D^0 = \{(s_1 = 2, \tau_1 = 7, \omega_1^2 = 2.4), (5, 3, 1.2), (1, 8, 4.8), (6, 8, 1.2), (6, 1, 0.6)\}$. Figure 3.1a shows that the five flows pass only nine of the 24 links, i.e., 15 links are unused. Note that the following examples consider only energy saving in stage 2.

First, consider solution 1 depicted by Figure 3.1a, where it upgrades switch 5 in stage 1, i.e., $V^1 = \{5\}$, followed by switch 3 and switch 4 in stage 2, i.e.,

$V^2 = \{5, 3, 4\}$. Solution 1 allows seven of the 15 unused links, i.e., $(2, 5)$, $(5, 2)$, $(3, 6)$, $(5, 6)$, $(8, 5)$, $(4, 2)$ and $(7, 4)$ to be turned off, as indicated by the dotted lines, resulting in an energy saving of $\varepsilon^2 = 7/24 \times 100\% = 29.17\%$. Recall that a link can be switched off only if it is connected to an s-switch. As an example, link $(7, 8)$ or $(8, 9)$, shown in bold lines, cannot be turned off.

Figure 3.1b shows an alternative solution 2 that upgrades switch 2 in stage 1, i.e., $V^1 = \{2\}$, followed by switch 6 and switch 8 in stage 2, i.e., $V^2 = \{2, 6, 8\}$. Advantageously, solution 2 allows 14 out of the 15 unused links shown in dotted lines to be turned off. This results in an energy saving of $\varepsilon^2 = 58.33\%$. Notice that except for link $(7, 4)$, each unused link is connected to at least one of the upgraded switches. Solution 2 shows that switch selection is important in order to maximize energy savings.

Let us now consider solution 3, which assumes each directed link (u, v) consists of $b_{uv} = 2$ cables, and thus the topology in Figure 3.1b contains $24 \times 2 = 48$ directed cables, each with a maximum capacity of $\gamma = 5$ units. Each unused cable can be powered off individually. Notice that *flow 3* and *flow 4* pass link $(5, 8)$ and thus both cables of the link are needed to carry $f^2_{(5,8)} = 4.8 + 1.2 = 6$ units of traffic. However, each of the other eight links that is used by the five flows requires only one cable to carry the flows. For example, link $(2, 4)$ needs only one cable to carry *flow 1* that has traffic size $\omega^2_1 = 2.4$. Notice that three of the eight links, i.e., $(2, 4)$, $(6, 5)$ and $(6, 3)$, shown as dashed lines, are connected to at least one s-switch, and thus three cables can be switched off. As in solution 2, there are 14 unused links in solution 3 that are connected to at least one s-switch, and thus $14 \times 2 = 28$ cables can be switched off. Thus, there are $28 + 3 = 31$ unused cables, which is equivalent to an energy saving of $\varepsilon^2 = 31/48 \times 100\% = 64.58\%$. Solution 3 shows that using bundled links can further increase energy saving. Notice that the energy saving is achieved without rerouting any demands.

Lastly, consider solution 4 where a flow can be rerouted via an alternative path that has: (i) a delay no larger than 10% of the delay in its original path, i.e., $\sigma = 1.1.$, and (ii) link load that is within a given maximum capacity. Assume that each link (u, v) has a propagation delay of $\pi_{(u,v)} = 1$ second. Here, rerouting *flow 5* from the original shortest path $(6, 3, 1)$ to an alternative path $(6, 5, 1)$ can further increase energy saving. The reason is because an additional cable of link $(6, 3)$ can be switched off. Notice that rerouting *flow 5* does not violate constraints (i) and (ii). solution 4 shows that traffic rerouting can be used to maximize energy savings.

3.1.3. GMSU-1 Mathematical Model

An ILP called ILP-1 in (3.1) is used to formulate GMSU-1. ILP-1 has three decision variables: (i) the number of *on*-cables n_{uv}^t for each link (u, v), (ii) the binary variable x_u^t set to 1 (0) if node u is upgraded (not upgraded) at stage t, and (iii) the binary variable $y_{d,uv}^t$ to indicate whether demand d is routed ($y_{d,uv}^t = 1$) or not routed ($y_{d,uv}^t = 0$) through link (u, v) at stage t. The objective, i.e., (3.1a), is to minimize the total number of *on*-cables, i.e., cables that carry traffic, over T stages. Thus, objective (3.1a) will maximize the total number of unused cables that can be switched off as per Eq. (2.2) in order to reduce energy usage.

$$\min_{n_{uv}^t, x_u^t, y_{d,uv}^t} \quad \sum_{t=1}^{T} \sum_{(u,v)\in E} n_{uv}^t \tag{3.1a}$$

s.t.

$$\sum_{u\in V} (p_u^t \times x_u^t) \leq \sum_{k=1}^{t} B^k - \sum_{k=1}^{t-1} \sum_{u\in V} (p_u^k \times x_u^k), \tag{3.1b}$$

$$\sum_{t=1}^{T} x_u^t \leq 1, \tag{3.1c}$$

$$\sum_{(u,v)\in E} y^t_{d,uv} - \sum_{(v,u)\in E} y^t_{d,vu} = \begin{cases} 1, & u = s_d \\ -1, & u = \tau_d \\ 0, & u \neq s_d, \tau_d \end{cases}, \tag{3.1d}$$

$$\sum_{(u,v)\in E} y^t_{d,uv} \times \pi_{uv} \leq \delta_{\max,d}, \tag{3.1e}$$

$$\sum_{d\in 1,2,\ldots,|D^t|} \omega^t_d \times y^t_{d,uv} \leq (n^t_{uv}/b_{uv}) \times U_{\max} \times c_{uv}, \tag{3.1f}$$

$$0 \leq n^t_{uv} \leq b_{uv}, \tag{3.1g}$$

$$n^t_{uv} = \max\left(n^t_{uv}, b_{uv} \times \left(1 - \sum_{k=1}^{t} x^k_u - \sum_{k=1}^{t} x^k_v\right)\right), \tag{3.1h}$$

$$y^t_{d,uv}, x^t_u \in \{0,1\}. \tag{3.1i}$$

Constraints (3.1b) and (3.1c) ensure that the upgraded l-switches are those that can maximally turn off their unused cables over T stages. Specifically, constraint (3.1b) ensures that at each stage the total upgrade cost does not exceed the budget for that stage. Constraint (3.1c) ensures each switch u is only upgraded once.

Constraints (3.1d)–(3.1h) ensure a single path that meets a given maximum delay and MLU threshold to route each traffic demand. More specifically, constraint (3.1d) is the standard flow conservation constraint. It ensures that the amount of each flow d into a switch u equals the amount exiting switch u, unless the switch is the source s_d or the destination τ_d. Each flow d that originates from (terminates at) switch u must exit (enter) via one of its outgoing (incoming) link (u,v) ((v,u)), and must not enter (exit) u via any one of its incoming (outgoing) link (v,u) ((u,v)). The constraint also ensures (s_d, τ_d) is connected in order to route each demand d. Let the selected path R^t_d to route demand d is represented as $R^t_d = \{(u,v) \mid y^t_{d,uv} = 1, (u,v) \in E\}$. Constraint (3.1e) ensures that the delay of

each selected path in R_d^t is within a given delay tolerance $\delta_{\max,d}$. The load of each link at stage t is bounded as per constraint (3.1f), meaning that the load must not exceed the capacity of *on*-cables for a given link utilization $U_{\max}$ threshold that is computed as $c_{uv}^t = (n_{uv}^t/b_{uv}) \times c_{uv} \times U_{\max}$. Constraint (3.1g) binds the number of *on*-cables to the bundle size of each link.

Constraint (3.1h) enforces all cables in each l-link to be powered-on because only unused cables in a c-link can be turned off. Note that formulation (3.1h) is a simplified form of its real implementation that utilizes the set of linear formulations (B.1) in Appendix B. Finally, constraint (3.1i) defines decision variables $y_{d,uv}^t$ and x_u^t to be binary.

Note that the aforementioned constraints, except for (3.1c) that applies to each $u \in V$, are repeated for each stage t, where $t \in \{1, 2, \ldots, T\}$. Constraint (3.1d) is for each node $u \in V$ and each traffic demand $d \in \{1, 2, \ldots, |D^t|\}$, while (3.1e) is only for each traffic demand $d \in \{1, 2, \ldots, |D^t|\}$. Constraints (3.1f), (3.1g) and (3.1h) exist for all links $(u, v) \in E$.

Appendix C provides the code implementation of ILP-1 using Gurobi API for C++ [83]. The appendix also includes the result of solving ILP-1 using the small example in Figure 3.1.

3.1.4. GMSU-1 Complexity Analysis

GMSU-1 is related to two well-known problems: (i) simple Directed Two Commodity Integral Flow (D2CIF) [86] - an NP-complete problem, and (ii) 0-1 Multiple Knapsack Problem (MKP) [87] - an NP-hard problem. More specifically, D2CIF is equivalent to a special case of our problem, i.e., when (i) all upgraded l-switches at every stage t are given, and (ii) set D^t contains only two traffic demands. For this case, our problem is only concerned with finding a set of shortest

paths for each traffic demand in D^t that minimizes the number of *on*-cables.

On the other hand, MKP is equivalent to another special case of GMSU-1, i.e., when (i) there is no depreciation in switch upgrade cost, and (ii) the path used to carry each traffic demand $d \in \{1, 2, \ldots, |D^t|\}$ is fixed at each stage t; thus, the number of *on*-cables per link $(u, v) \in E$ is known. Briefly, given m items, each of which has a profit and weight, and T knapsacks, each of which has a maximum weight capacity, MKP aims to select T-disjoint subsets of items that maximize the total profit, subject to each subset having a total weight no more than its knapsack's capacity. Note that the profit and weight of each item in MKP are respectively equivalent to the number of *off*-cables for each switch $u \in V$ and the switch upgrade cost p_u^t. Further, the maximum budget at each stage B^t is the same as a knapsack's capacity in MKP. The goal of our problem is to upgrade T disjoint subsets of l-switches that minimize the total number of *on*-cables over multiple stages T. Instead of total profit in the MKP case, the work in this thesis aims to maximize the total number of *off*-cables. Thus, our optimization problem is at least as hard as MKP. The next section describes the heuristic solution for our optimization problem.

3.2. Heuristic Solution

This section presents the proposed heuristic algorithm for GMSU-1 called GU-1. Section 3.2.1 provides the details of the algorithm, Section 3.2.2 provides an example of how GU-1 works, and Section 3.2.3 presents an analysis of how budget and planning stages impact the network energy saving.

3.2.1. Details of GU-1

Referring to Algorithm 3.1, GU-1 consists of three main parts: 1) initial traffic routing, 2) switch upgrade, and 3) traffic rerouting. Note that while Part 1 is used only at stage $t = 1$, Part 2 is used at the beginning of each stage (in years). On the other hand, traffic rerouting in Part 3, in addition to being computed at the beginning of each stage, can be used whenever there is a significant change in network traffic within the stage, e.g., every week. The following three sections present the details of Part 1, Part 2, and Part 3, respectively.

Algorithm 3.1. : **GU-1 – heuristic solution for GMSU-1**

Input: $G^0(V, E)$, T, B, D^0, p_u^0, $U_{\max}$, μ_d, ρ, σ
Output: V^t, R^t, ε^t

▷ *Part 1: Initial Routing*
1: **for** $(d \in \{1, 2, \ldots, |D^T|\})$ **do**
2: Generate set $\mathcal{P}_d$
3: $R^0 = R^0 \cup \mathcal{P}_{d,1}$
4: $f_{uv}^T = f_{uv}^T + \omega_d^T$ for each $(u, v) \in \mathcal{P}_{d,1}$
5: **end for**
6: $n_{uv}^T = \lceil f_{uv}^T / (\gamma \times U_{\max}) \rceil$ for each $(u, v) \in E$
7: Compute w_u for each $u \in V$ using Eq. (3.2)
8: **for** $(t \in \{1, 2, \ldots, T\})$ **do**
▷ *Part 2: Switch Upgrade*
9: $\{V^t, \Delta B^t, L\} =$ **Selection**(V^{t-1}, B^t)
10: $B^{t+1} = B^{t+1} + \Delta B^t$
▷ *Part 3: Traffic Rerouting*
11: $R^t =$ **GTE**(V^t, R^{t-1}, L)
12: Compute ε^t using Eq. (2.1)
13: $\varepsilon_T = \varepsilon_T + \varepsilon^t / T$
14: **end for**

3.2.1.1. Part 1 – Initial Traffic Routing

Part 1 routes each demand $d \in \{1, 2, \ldots, |D^T|\}$ via a shortest path. It then uses the initial shortest paths of all demands to calculate the total number of unused cables which are adjacent to each switch. More specifically, *Line 2* of Algorithm 3.1 generates a set $\mathcal{P}_d$ that contains the first k paths from s_d to τ_d

in order of increasing delay. This means that $\mathcal{P}_d = (\mathcal{P}_{d,1}, \mathcal{P}_{d,2}, \ldots, \mathcal{P}_{d,k})$, where path $\mathcal{P}_{d,1}$ has the shortest delay. Let $Y_{d,j}$ denote the j-th shortest path from source s_d to destination τ_d for demand d, for $j \in \{1, 2, \ldots, k\}$. GU-1 uses Yen's algorithm [88] to generate each set $\mathcal{P}_d$ as follows: (i) use Dijkstra's shortest path algorithm to generate $Y_{d,1}$ and store it in $\mathcal{P}_d$, and (ii) generate the next shortest path $Y_{d,j}$, for $j \in \{2, 3, \ldots, \text{ up to } k\}$. For each path $Y_{d,j}$ and a node $v \in Y_{d,j}$, for $v \neq \tau_d$, the shortest path from v to τ_d (s_d to v) is called a *spur* (*root*). To generate each path $Y_{d,j}$, Yen's algorithm uses all *spurs* from each node v and all *roots* to each node v in the previously generated path $Y_{d,j-1} \in R_d$. Each *spur* for node v must not pass any node on the *root* of v to avoid a cycle. Further, each *spur* must not branch from v on any link used by any path $Y_{d,m} \in \mathcal{P}_d$, for $m \leq j-1$, with the same *root* to avoid re-generating an existing path in R_d. Each *spur* that satisfies the two aforementioned criteria is appended to the *root* to form a candidate path. All candidate paths are stored in a set $\mathcal{Y}_d$. Path $Y_{d,j}$ is a candidate path in $\mathcal{Y}_d$ with the shortest delay. If the delay of path $Y_{d,j}$ is within $\delta_{\max,d}$, GU-1 adds the path to $\mathcal{P}_d$ and removes it from $\mathcal{Y}_d$. Yen's algorithm then repeats from Step (ii) to generate the next shortest path $Y_{d,j+1}$. Otherwise, GU-1 stops Yen's algorithm because no other shortest path will satisfy delay constraint $\delta_{\max,d}$.

Next, *Line 3* records each shortest path $\mathcal{P}_{d,1} \in \mathcal{P}_d$ into the set of initial routes R^0. *Line 4* computes the total traffic volume f_{uv}^T for each link $(u, v) \in E$ at stage T, for $\omega_d^T = \omega_d^1 \times (1 + \mu_d)^{T-1}$. *Line 6* calculates the number of *on*-cables n_{uv}^T of link (u, v) at stage T that can carry flow size f_{uv}^T. For each switch $u \in V$, *Line 7* computes the total number of its adjacent unused cables w_u as follows:

$$w_u = \sum_{(u,v) \in E} (b_{uv} - n_{uv}^T), \quad u \in V. \tag{3.2}$$

Computing w_u per Eq. (3.2) is based on the following observation:

Observation 3.1. Upgrading a set of l-switches with the highest total number of unused cables at the earliest possible stage can increase energy saving over T stages.

To explain Observation 3.1, recall that the volume of demand d grows at a rate of $\mu_d \geq 0$ for each subsequent stage $t \in \{2, \ldots, T\}$. Thus, if a link (u, v) that has n_{uv}^T number of *on*-cables is able to carry the traffic demand at stage T, these *on*-cables are also sufficient to carry traffic demands at any stage $t < T$. This implies that $n_{uv}^t \leq n_{uv}^{t+1}$ and $(b_{uv} - n_{uv}^t) \geq (b_{uv} - n_{uv}^{t+1})$ for each link (u, v). In this case, the number of unused cables $(b_{uv} - n_{uv}^t)$ at stage t includes the $(b_{uv} - n_{uv}^T)$ unused cables that can remain off in stage $t + 1$. An l-switch u with the highest w_u also has the highest number of unused cables at stage $t < T$.

3.2.1.2. Part 2 – Switch Upgrade

Part 2 uses the total unused cables w_u per switch u to strategically select a set of l-switches that maximize energy saving. For each stage $t \in \{1, \ldots, T\}$, in *Line 9* of Algorithm 3.1, Part 2 calls function **Selection**(V^{t-1}, B^t), shown as Algorithm 3.2 to find an *upgradable* l-switch, which is defined as follows.

Definition 3.1. An l-switch $u \in V - V^t$ is *upgradeable* if (i) the switch has non-zero total number of unused cables, i.e., $w_u > 0$, and (ii) the cost p_u^t to upgrade switch u is at most B^t.

Part 2 uses ratio w_u/p_u^t to upgrade a switch with the highest *off*-cables per cost unit. It starts from the largest ratio w_u/p_u^t in order to increase the number of *off*-cables, and hence, energy saving, over T stages. Function **Selection**() returns a set V^t that contains the selected l-switches that are upgraded up to stage t,

remaining budget $\Delta B^t \geq 0$, and a set L that stores every c-link (u, v) with non-zero traffic flow.

Algorithm 3.2. : **Selection()**

Input: V^{t-1}, B^t
Output: V^t, ΔB^t, L

1: $V^t = V^{t-1}$ and $X = V - V^t$
2: **for** (each $u \in X$ that has $p_u^t \leq B^t$, $w_u > 0$, and $\max_{u \in X}\{w_u/p_u^t\}$) **do**
3: $X = X - u$
4: $V^t = V^t \cup u$
5: $B^t = B^t - p_u^t$
6: **for** ($v \in X$ **and** $(u, v) \in E$) **do**
7: $w_v = w_v - (b_{uv} - n_{uv}^T)$
8: **if** ($n_{uv}^T > 0$) **then**
9: $L = L \cup (u, v)$
10: **end if**
11: **end for**
12: **end for**

In details, *Line 1* of Algorithm 3.2 initializes set V^t with the previous upgraded switches V^{t-1} and uses set X to store all l-switches that yet to be upgraded. Note that V^0 is an empty set. *Line 2* selects one candidate l-switch u that satisfies criteria (i) and (ii) of Definition 3.1 and has the largest w_u/p_u^t. *Lines 3–4* move the selected node u from X into set V^t. *Line 5* computes the remaining budget. *Line 6* considers each l-switch neighbor v of the selected switch u, and *Line 7* reduces the total unused cables w_v by the total cables powered off by switch u. *Lines 8-10* place c-link (u, v) into the set L if some traffic demands pass the link, i.e., $n_{uv}^T > 0$. *Lines 2–12* are repeated until either criterion (i) or (ii) of Definition 3.1 fails. *Line 10* of Algorithm 3.1 then adds the remaining budget ΔB^t to the budget for the next stage.

3.2.1.3. Part 3 – Traffic Rerouting

Part 3 aims to increase the number of *off*-cables, equivalently energy saving, achieved by Part 2, if possible. Part 1 calls function **GTE**(), shown as Algo-

rithm 3.3, *in Line 11*. It checks if it is feasible to switch off one cable in each c-link, starting from a cable that has the smallest used capacity. The reason is because a cable with the smallest used capacity has the smallest traffic to be rerouted, and thus is more feasible to be switched off. This greedy approach is expected to maximize the number of additional *off*-cables. However, switching off a cable is feasible only if each *controllable* path R_d^t that passes through the cable can be rerouted to an alternative path $\mathcal{P}_{d,i} \in \mathcal{P}_d$ that is *routable*. The following Definition 3.2 and Definition 3.3 define a *controllable* path and a *routable* path, respectively.

Definition 3.2. Path R_d^t of traffic demand $d \in \{1, \ldots, |D^t|\}$, is *controllable* only if it passes at least one c-link.

Definition 3.3. An alternative path $\mathcal{P}_{d,i} \in \mathcal{P}_d$ for R_d^t, i.e., $\mathcal{P}_{d,i} \neq R_d^t$, is called *routable* if it can be used to forward demand d at stage t subject to (i) its path delay $\delta_{d,i}$ being less than or equal to the maximum delay $\delta_{\max,d}$, and (ii) adding new traffic volume ω_d^T to the current traffic flow f_{uv}^T on each link $(u,v) \in \mathcal{P}_{d,i}$ does not exceed the link's capacity at stage T, i.e., $(f_{uv}^T + \omega_d^T) \leq c_{uv}^T$, where $c_{uv}^T = (\gamma \times n_{uv}^T \times U_{\max})$.

Definition 3.3 considers ω_d^T, f_{uv}^T, and n_{uv}^T to ensure that each *routable* path can carry traffic demands at any stage $t \leq T$.

The detailed description of function **GTE**() is given as follows. *Line 1* of Algorithm 3.3 initializes each path $R_d^t \in R^t$ at stage t with path $R_{d,1}^{t-1} \in R^{t-1}$ from the previous stage. *Line 1* also makes a temporary copy of set L, denoted as $\mathcal{L}$. *Line 3* then selects a c-link $(u,v) \in \mathcal{L}$ with a cable that has the smallest used capacity r_{uv}. If a link (u,v) has $r_{uv} = 0$, this means that each of its *on*-cables is fully utilized, and thus the used capacity is set to $r_{uv} = \gamma$. *Line 4* finds each path $P_d^t \in P^t$ that passes link (u,v) and stores it in the set Q_{uv}.

Algorithm 3.3. : **GTE()**

Input: V^t, R^{t-1}, L
Output: R^t

1: $R^t = R^{t-1}$ and $\mathcal{L} = L$
2: **while** $(\mathcal{L} \neq \{\})$ **do**
3: Find $(u, v) \in \mathcal{L}$ with the smallest $r_{uv} = (f^T_{uv} - \gamma \times U_{\max} \times \lfloor f^T_{uv}/\gamma \times U_{\max} \rfloor)$
4: Put all $R^t_d \in R^t$ that pass (u, v) in Q_{uv}
5: $n^T_{uv} = n^T_{uv} - 1$
6: **for** $(R^t_d \in Q_{uv}$ **and** $r_{uv} > 0)$ **do**
7: **if** (exist a *routable* path $\mathcal{P}_{d,i}$ in $\mathcal{P}_d$) **then**
8: Replace R^t_d with $\mathcal{P}_{d,i}$
9: $r_{uv} = r_{uv} - \omega^T_d$
10: **end if**
11: **end for**
12: **if** $(r_{uv} > 0)$ **then**
13: Revert back each changed path R^t_d to its previous path
14: $n^T_{uv} = n^T_{uv} + 1$
15: $\mathcal{L} = \mathcal{L} - (u, v)$
16: **else if** $(n^T_{uv} == 0)$ **then**
17: $\mathcal{L} = \mathcal{L} - (u, v)$
18: $L = L - (u, v)$
19: **end if**
20: **end while**
21: Compute w_u for each $u \in V - V^t$ using Eq. (3.2)

Lines 6–11 then check if each path in Q_{uv} can be rerouted when one cable in link (u, v) is switched-off in *Line 5*. More specifically, *Line 7* aims to find a *routable* path $\mathcal{P}_{d,i}$ to reroute each path $R^t_d \in Q_{uv}$. For every successful attempt to find a *routable* path $\mathcal{P}_{d,i}$ for each *controllable* path $R^t_d \in Q_{uv}$, *Line 8* replaces path R^t_d with $\mathcal{P}_d$. Further, *Line 9* reduces the used capacity of the cable being turned off in *Line 5*. However, if each path in Q_{uv} that uses the *off*-cable has no *routable* path, the rerouting process is terminated, i.e., $r_{uv} > 0$. This indicates that shutting down one cable in c-link (u, v) in *Line 5* is not feasible. In this case, *Lines 13-15* revert each rerouted path R^t_d to its original path, turn on the cable in link (u, v), and remove link (u, v) from the set $\mathcal{L}$. On the other hand, if all paths in Q_{uv} that pass the *off*-cable are successfully rerouted, i.e., $r_{uv} \leq 0$, and link (u, v) has no *on*-cable, i.e., $n^T_{uv} = 0$, *Lines 17-18* remove the link from both sets, $\mathcal{L}$ and L. In other words, the link remains off in the remaining upgrade

stages. Delivering all demands using the new routes in R^t affects the number of unused cables for the remaining l-switches. Thus, *Line 21* recalculates the total unused cables w_u for each l-switch u in $V - V^t$. Finally, *Lines 12-13* of Algorithm 3.1 use Eq. (2.1) and Eq. (2.2) to compute energy saving ε^t for each stage t and average energy saving ε_T over T stages.

3.2.2. An Example

Figure 3.1 illustrates how GU-1 upgrades the network depicted in Figure 3.1a over $T = 2$ stages given a total budget of $B = \$40$, meaning that we have $B^1 = B^2 = \$20$. Each switch u has an upgrade cost of $p_u^1 = \$15$ with a depreciation rate of $\rho = 0.2$ per stage, and thus the cost $p_u^2 = \$12$. Each link (u, v) has $b_{uv} = 2$ and each cable has a capacity of $\gamma = 5$ with an MLU threshold of $U_{\max} = 80\%$. The delay of each link π_{uv} is set to one second. Further assume there are five demands in Figure 3.1a, i.e., $D^1 = \{(2, 7, 2), (3, 5, 1), (1, 8, 4), (6, 8, 1), (6, 1, 0.5)\}$, with a growth rate of $\mu = 0.2$ per stage. Thus, $D^2 = \{(2, 7, 2.4), (5, 3, 1.2), (1, 8, 4.8), (6, 8, 1.2), (6, 1, 0.6)\}$.

For this example, we use the first maximum delay constraint, i.e., $\delta_{\max,d} = \lceil \sigma \times \delta_d^0 \rceil$, with delay multiplier $\sigma = 1.1$. As shown in Figure 3.1a, the shortest path delay for each demand is equal to two seconds. As an example, the delay of path $(2, 4, 7)$ is $\delta_1^0 = 2$. Thus, the maximum delay for each path in the set $\mathcal{P}_1$ of k shortest paths is computed as $\delta_{max,1} = \lceil 1.1 \times 2 \rceil = 3$. *Lines 2–3* of Algorithm 3.1 then generate a set of two shortest paths for each demand, e.g., $R_1^0 = \{(2, 4, 7), (2, 5, 8, 7)\}$, $R_4^0 = \{(6, 5, 8), (6, 9, 8)\}$ and $R_5^0 = \{(6, 3, 1), (6, 5, 1)\}$.

The shortest path for each demand R_d^0, e.g., $R_1^0 = (2, 4, 7)$, $R_4^0 = (6, 5, 8)$ and $R_5^0 = (6, 3, 1)$, is used in *Line 4* to compute f_{uv}^T, e.g., $f_{(1,2)}^2 = 0$, $f_{(2,4)}^2 = 2.4$, $f_{(1,5)}^2 = 4.8$, $f_{(5,1)}^2 = 1.2$, $f_{(6,3)}^2 = 0.6$, $f_{(6,5)}^2 = 1.2$, and $f_{(5,8)}^2 = 4.8 + 1.2 = 6$. *Line 6*

obtains each n^T_{uv} for link (u, v), e.g., $n^2_{(1,2)} = 0$, $n^2_{(2,4)} = n^2_{(5,1)} = n^2_{(6,3)} = n^2_{(6,5)} = 1$, and $n^2_{(1,5)} = n^2_{(5,8)} = 2$. *Line 7* then computes w_u of each node u, e.g., $w_1 = 7$, $w_2 = 11$ and $w_5 = 10$.

In *Line 1* of Algorithm 3.2, all nine switches in V are copied into X. Thus, we have $X = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$. Among all switches in X, only switch $v = 2$ can be upgraded at $t = 1$. *Line 7* of Algorithm 3.2 decreases the total unused cables w_u if node u is a neighbour of node $v = 2$, e.g., $w_1 = w_1 - ((b_{(1,2)} - n^2_{(1,2)}) + (b_{(2,1)} - n^2_{(2,1)})) = 3$. *Lines 8-10* obtains six c-links: {(1,2), (2,1), (2,5), (5,2), (4,2), and (2,4)}. The first five c-links have no traffic flow and hence, they are turned-off. Here, we have $L = \{(2, 4)\}$. Thus, in *Line 9* of Algorithm 3.1, function **Selection**() returns $V^1 = \{2\}$ and $B^1 = \$20 - \$15 = \$5$, and $L = \{(2, 4)\}$, while *Line 10* updates $B^2 = \$20 + \$5 = \$25$.

For stage $t = 1$, *Line 1* of Algorithm 3.3 sets R^1 to all paths in R^0, e.g., $R^1_1 = R^0_1 = (2, 4, 7)$, and duplicates set L to $\mathcal{L} = \{(2, 4)\}$. Thus, *Line 3* considers only link $(2, 4)$ with $r_{(2,4)} = 2.4 - 5 \times 0.8 \times \lfloor 2.4/5 \times 0.8 \rfloor = 2.4$. Further, *Line 4* obtains $Q_{2,4} = \{R^1_1 = (2, 4, 7)\}$, and *Line 5* has $n^2_{(2,4)} = 0$. Path $(2, 5, 8, 7)$ in R^1_1 is not *routable* because link $(5, 8)$ will exceed its maximum link utilization when the traffic demand $d = 1$, with $\omega^2_1 = 2.4$, is routed on the path. As a result, *Line 14* sets $n^2_{(2,4)}$ back to one and Line 15 removes link $(2, 4)$ from the set $\mathcal{L}$. The loop or *Lines 2-20* stops because set $\mathcal{L}$ is now empty. *Line 21* updates the value of w_u for all switches except for switch 2, e.g., $w_1 = 3$, $w_4 = 3$ and $w_5 = 6$. For stage 1, there are 11 cables that can be switched off: one cable in link (2,4), and two cables each of links $(1, 2)$, $(2, 1)$, $(4, 2)$, $(2, 5)$ and $(5, 2)$. Thus, *Line 12* of Algorithm 3.1 outputs $\varepsilon^1 = 11/48 \times 100 = 22.92\%$.

For stage $t = 2$ and $B^2 = \$25$, two more switches are selected for upgrade, i.e., $V^2 = \{8, 6\}$. At this stage, we have three additional c-links that carry traffic flow, and thus we have $L = \{(2, 4), (5, 8), (6, 3), (6, 5)\}$. Among these

c-links, link (6,3) has one cable with the smallest used capacity $r_{(3,6)} = 0.6$. Thus, path $R_5^2 = (6, 3, 1)$ is replaced with a *routable* path $(6, 5, 1)$, and we have $n_{(6,3)}^2 = 0$. For link $(6, 5)$, there is no *routable* path for paths $R_4^2 = (6, 5, 8)$ and $R_5^2 = (6, 5, 1)$ because the only paths for R_4^2 and R_5^2, which are $(6, 9, 8)$ and $(6, 3, 1)$, respectively, do not satisfy the maximum link utilization. Notice that $n_{(6,9)}^2 = n_{(9,8)}^2 = n_{(6,3)}^2 = 0$. All controllable paths that pass the remaining c-links, namely path $\{(2, 4, 7)\}$ passing link $(2, 4)$, paths $\{(6, 5, 1), (6, 5, 8)\}$ passing link $(6, 5)$, and paths $\{(1, 5, 8)$ and $(6, 5, 8)\}$ passing link $(5, 8)$, have no alternative path that is routable. Accordingly, for stage $t = 2$, we have a total of 32 *off*-cables, i.e., two cables from each c-link in $\{(1, 2), (2, 1), (2, 5), (5, 2), (4, 2), (3, 6), (6, 3), (5, 6), (6, 9), (9, 6), (7, 8), (8, 7), (8, 5), (8, 9), (9, 8)\}$, and one cable from each c-link in $\{(2, 4), (6, 5)\}$. Hence, we have $\varepsilon^2 = 66.67\%$, and the average energy saving for $T = 2$ is $\varepsilon_2 = (22.92 + 66.67)/2 = 44.8\%$.

3.2.3. Analysis of GU-1

This section presents an analysis of GU-1 with respect to budget increase, upgrade stages, number of remaining l-switches, and time complexity. The following proposition states that function **Selection**() upgrades at least $|V^t|$ switches when budget B^t is increased.

Proposition 3.1. *Let B_1^t and B_2^t be two alternative budgets that are used by **Selection** (X, t) to produce two sets of upgraded switches V_1^t and V_2^t from the set X, respectively. For $B_2^t \geq B_1^t$, we have $V_2^t \supseteq V_1^t$, i.e., each switch in V_1^t is also an upgraded switch in V_2^t. Thus, $|V_2^t| \geq |V_1^t|$.*

Proof. Function **Selection**() upgrades only each l-switch u that can save more energy, i.e., any switch with $w_u > 0$. Further, switches are selected by the function in decreasing order of w_u/p_u^t. Let $V_1^t = (u_1, u_2, \ldots, u_{|V_1^t|})$ be the set of

switches, in decreasing order of w_u/p_u^t, that can be upgraded using budget B_1^t. Since $B_2^t \geq B_1^t$, the function can upgrade at least all switches in V_1^t. Further, the remaining budget $(B_2^t - B_1^t)$ can be used to upgrade other switches u in X that have $w_u > 0$. Thus, budget B_2^t can afford to upgrade a sequence of switches $V_2^t = (u_1, u_2, \ldots, u_{|V_1^t|}, \ldots, u_{|V_2^t|})$, i.e., $V_2^t \supseteq V_1^t$, and $|V_2^t| \geq |V_1^t|$. □

Consider two sets of upgraded switches, V^α and V^β, generated by **Selection**() at stage α and β, respectively. Let X^t be a set that contains l-switches, each of which has $w_u > 0$, at period t; thus upgrading each switch in X^t will increase energy saving. The following proposition shows that, for $\alpha < \beta$, function **Selection**() will upgrade more switches at period β than at period α if at period β the network contains more than V^α number of switches with $w_u > 0$.

Proposition 3.2. *Consider two periods α and β, for $1 \leq \alpha < \beta \leq T$. If (i) $|X^\beta| \geq |V^\beta|$, and (ii) $(|V^\beta| \times p_u^\beta) \leq B^\beta$, then function **Selection()** generates $|V^\beta| \geq |V^\alpha|$.*

Proof. Condition (i) considers there are at least $|V^\beta|$ l-switches that can be upgraded to increase energy savings. Condition (ii) ensures that the allocated budget at period β is sufficient to upgrade $|V^\beta|$ switches, each of which has an upgrade cost of p_u^β. Notice that $p_u^\beta < p_u^\alpha$ since the cost is decreased by a rate of ρ per period. Thus, with a lower upgrade cost, the function can upgrade more switches with budget B^β than with budget B^α. We have $B^\beta \geq B^\alpha$ because initially GU-1 sets $B^\beta = B^\alpha$, and the remaining budget at period α is added to B^β at stage β. □

While **Selection()** can potentially generate more upgraded switches at later stages as per Proposition 3.2, it does not mean that it can always produce more energy savings in later stages. The reason is because network traffic increases by

a rate of μ_d per stage, which may lead to fewer number of switched-off cables. Further, any upgraded l-switch at stage t can reduce the number of remaining l-switches with unused cables to turn-off, as stated by the following Proposition 3.3.

Proposition 3.3. *Let X^t be a set containing l-switches $u \in V$ with $w_u > 0$ at period t. At period $t+1$, we have $|X^{t+1}| \leq |X^t| - |V^t|$.*

Proof. Any selected l-switch $u \in V^t$ can turn off unused cables in link (u, v) and (v, u). Thus, for every $v \in X^t$ and v is directly connected to u, its w_v is decreased by $(b_{uv} - n^T_{uv}) + (b_{vu} - n^T_{vu})$. If we have $w_v = 0$, v is not included in X^{t+1}, i.e., $v \notin X^{t+1}$. Thus, $|X^{t+1}| \leq |X^t| - |V^t|$. □

Thus, longer upgrade stages do not always translate to more energy saving, especially with the increasing traffic size which can reduce the number of unused cables. Finally, the following proposition provides the complexity analysis of GU-1.

Proposition 3.4. *The time complexity of GU-1 is $O(|V|^2|E|^2)$.*

Proof. *Line 2* of Algorithm 3.1 can use Yen's algorithm [88] to generate k-shortest paths. The algorithm takes $O(k|D||V|(|E|+|V|log|V|))$ for all $|D|$ demands. Note that $|D| = |D^t|$ for every $t \in \{1, \ldots, T\}$. *Line 3* takes a constant time of $O(1)$. The calculation of f^T_{uv} for all $|D|$ demands in *Line 4* require $O(|D||E|)$. Note that all values of ω^T_d can be computed in $O(|V|^2)$. *Lines 6–7* require $O(|E|)$. Thus, Part 1 takes in total $O(k|D||V|(|E| + |V|log|V|) + |D||E| + |V|^2 + |E|) = O(k|D||V|(|E| + |V|log|V|))$. In Algorithm 3.2, *Line 1* takes $O(|V|)$, *Line 2* has a run-time of $O(|V|)$, *Lines 3–5* each takes $O(1)$, *Lines 6-11* requires $O(|E|)$, and *Lines 2–12* are repeated at most $|V|$ times. *Line 10* of Algorithm 3.1 takes $O(1)$. Thus, Part 2 takes $O(|V|^2 + |V| + |V||E|) = O(|V||E|)$. Line 1 of Algorithm 3.3 takes $O(|D|)$. In the worst case, the number of c-links in $\mathcal{L}$ is the as same as

the number of links in E. Thus, *Lines 2–20* are repeated $O(|E|)$ times. For each repetition, *Line 3* needs $O(|E|)$, while *Line 4* takes $O(|D||E|)$. *Line 5* takes a constant time. *Line 7* takes $O(k|E|)$ to find a *routable* path among the generated k-shortest path for each demand. *Lines 8–9* each takes a constant time. *Lines 6–11* are repeated in the worst case $O(|D|)$ times, and thus these lines take $O(k|D||E|)$ time. *Line 13* can revert up to $|D|$ paths, and thus its complexity is $O(|D||E|)$. *Lines 14–15* and *Lines 17–18* take $O(1)$, while *Line 21* takes $O(|E|)$. Thus, in total, Part 3 has $O(|E|(|E|+|D||E|+2+k|D||E|+|D||E|+4)+|E|) = O(k|D||E|^2)$ of time complexity. Note that *Lines 8–14* of Algorithm 3.1 are repeated T times and *Lines 12-13* of Algorithm 3.1 needs $O(T|E|)$. Thus, Part 2 and Part 3 have time complexity of $O(T|V||E|)$ and $O(Tk|D||E|^2)$, respectively. Overall, the time complexity of GU-1 is $O(k|D||V|(|E|+|V|log|V|)+T|V||E|+Tk|D||E|^2+T|E|) = O(Tk|D||E|^2)$. Here, we consider $|E| \geq |V|$, $|D| \leq |V|^2$, and T and k are constant, e.g., we use $T = 5$ and $k = 10$ at maximum. Thus, GU-1's time complexity is equal to $O(|V|^2|E|^2)$. □

3.3. Performance Evaluation

We implement GU-1 in C++ and use Gurobi [83] to solve ILP-1. Each experiment is conducted on the platform described in Section 2.7.3. Further, each evaluation uses the five network topologies listed in Table 2.3 and the value for each parameter listed in Table 2.4 (see Section 2.7 for details).

For each demand d, Yen's algorithm [88] is used to generate up to $k = 10$ paths in increasing order of delays so that each path has a delay no longer than $\delta_{\max,d}$. For each of the five networks, the simulation uses three delay constraints: (a) shortest delay $\delta_{\max,d} = \delta_{\min,d}$, (b) delay multiplier $\delta_{\max,d} = \lceil \sigma \times \delta_{\min,d} \rceil$ where $\sigma = 1.1$, and (c) network diameter $\delta_{\max,d} = \max_{d \in \{1,2,\ldots,|D^0|\}} \{\delta_{\min,d}\}$. Let GU-1a,

GU-1b, GU-1c, ILP-1a, ILP-1b, and ILP-1c be GU-1 and ULP-1 that use the respective three delays.

This section is organized as follows. First, Section 3.3.1 evaluates the scalability of GU-1a, ILP-1a, GU-1b, ILP-1b, GU-1c and ILP-1c in terms of their running time (in CPU seconds). Section 3.3.2 analyzes the effect of using longer delays on the energy saving and path delay of each network. Section 3.3.3 and 3.3.4 study the effect of increasing budget and number of stages, respectively, on energy saving and the percentage of upgraded l-switches. Section 3.3.5 evaluates the effect of the following four parameters on energy saving: link's bundle size b_{uv}, cost depreciation rate ρ, traffic increase rate μ_d, and MLU threshold $U_{\max}$. Finally, Section 3.3.6 evaluates the performance of GU-1 and ILP-1 against an existing technique, i.e., the Local Search (LS) [1], in terms of traffic controllability and energy saving.

3.3.1. Running Time Performance

This section evaluates the scalability of GU-1 and ILP-1 in terms of their running time (in CPU seconds). We use a total budget of $B = \$1.2M$ and set the number of stages to $T = 3$. We first consider their run time when each demand is routed via its shortest path. Table 3.1 shows that GU-1a uses less than one second to generate the result for each network. ILP-1a is significantly slower than GU-1a, e.g., 21879.37 versus 0.07 seconds for Deltacom. Further, ILP-1a fails to produce results for TATA because the optimizer ran out of memory.

Next, we aim to evaluate the impact of rerouting each demand onto an alternative path on the running time of GU-1 and ILP-1. As shown in Table 3.1, the run-time of GU-1b and GU-1c is similar, which indicates that the time complexity of function **GTE**() is not significantly affected by the two alternative

delays $\delta_{\max,d}$. However, as shown in columns 2, 4 and 6, GU-1 spends most of its run-time using function **GTE**(). As an example, running GU-1c on TATA, the function **GTE**() accounts for $(483.32 - 0.12)/483.32 \times 100\% = 99.97\%$ of the run-time of GU-1a. The most time consuming part of function **GTE**() is to generate the $k = 10$ shortest paths for each demand. It consumes approximately 89% to 98.53% of the total run time of function **GTE**(). Table 3.1 also shows that ILP-1b and ILP-1c run significantly slower than GU-1b and GU-1c, respectively. For example, for GÉANT, GU-1b runs for 1.32 CPU seconds, while ILP-1b requires 30.31 seconds. Further, while ILP-1b fails to produce results for TATA, ILP-1c produce no results for GÉANT, DFN, and Deltacom after running for more than one week.

Table 3.1. Running time.

Name	**Running Time (in CPU seconds)**					
	GU-1a	**ILP-1a**	**GU-1b**	**ILP-1b**	**GU-1c**	**ILP-1c**
Abilene	0.001	0.06	0.11	0.06	0.12	0.73
GÉANT	0.002	2.3	1.32	30.31	1.3	N/A
DFN	0.01	647.6	26.47	807.62	26.27	N/A
Deltacom	0.07	21879.37	305.58	26400.47	305.1	N/A
TATA	0.12	N/A	468.29	N/A	483.32	N/A

3.3.2. Effect of Rerouting Demands

This section analyzes the effect of rerouting demands via paths with longer delays on energy saving. It considers GU-1b, ILP-1b, GU-1c, and ILP-1c. We set the total allocated budget to $B = \$1.2M$ and use $T = 3$ planning stages. Figure 3.2 shows that GU-1 and ILP-1 with delay multiplier, i.e., GU-1b and ILP-1b, and delay not exceeding the network diameter, i.e., GU-1c and ILP-1c, can increase the energy saving ε_T of each network. As an example, for Abilene, the saving ε_T

of GU-1c and ILP-1c increases from 68.42% to 71.93% and 68.42% to 73.98%, respectively. By allowing longer path delays, both GU-1 and ILP-1 have more alternative paths to reroute traffic, which helps further minimize the total number of *on*-cables. As shown in Figure 3.2, the energy saving produced by GU-1 is close to the optimal saving obtained by ILP-1. More specifically, for Abilene, the energy savings produced by GU-1b and GU-1c are only 1.68% and 2.77%, respectively, off from the saving obtained by ILP-1b and ILP-1c. For GÉANT, DFN, and Deltacom, GU-1b has 1.38%, 2.6%, and 0.63%, respectively, less energy saving than ILP-1b. As shown in Figure 3.2, ILP-1b and ILP-1c fail to produce results for TATA. Moreover, ILP-1c fails to produce results for GÉANT, DFN, and Deltacom.

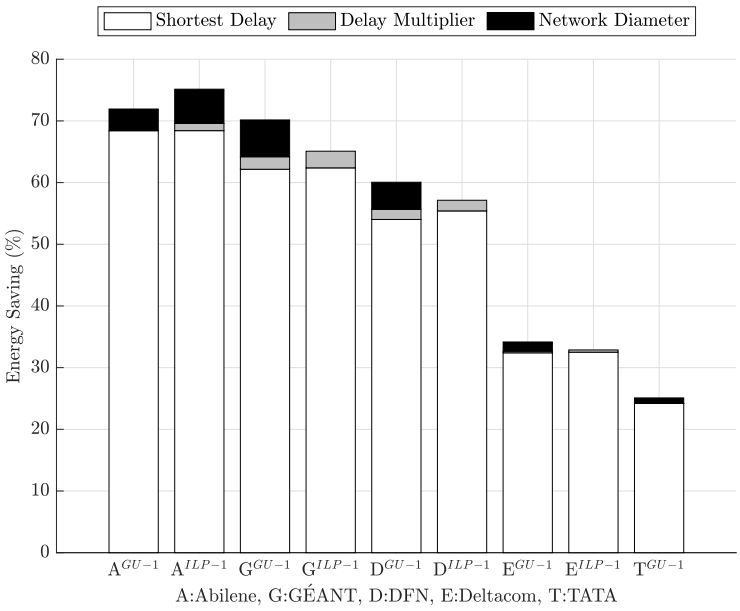


Figure 3.2. Increase in energy saving (ε_T) when using shortest delay $\delta_{\min,d}$, delay multiplier $\lceil \sigma \times \delta_{\min,d} \rceil$, and network diameter $\max_{d \in \{1,\ldots,|D^0|\}} \{\delta_{\min,d}\}$.

Figure 3.2 also shows that both GU-1 and ILP-1 produce a larger increase in energy saving ε_T when using delay constraint (c) as compared to constraint (b).

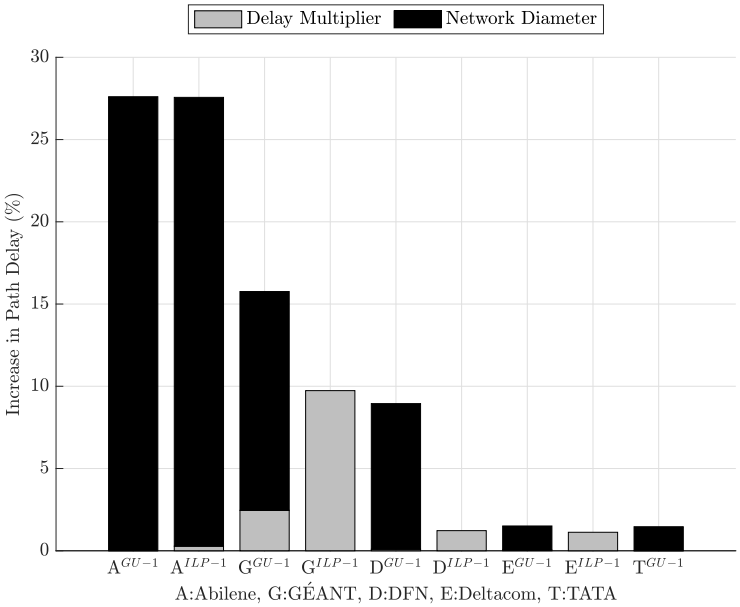


Figure 3.3. Increase in path delay for delay multiplier $\lceil \sigma \times \delta_{\min,d} \rceil$ and network diameter $\max_{d \in \{1,\ldots,|D^0|\}} \{\delta_{\min,d}\}$.

For example, for Abilene, while GU-1b does not increase its saving ε_T, GU-1c increases it by 5.13%. Further, ILP-1b recordes a 1.71% increase in the energy saving ε_T, i.e., from 68.42% to 69.59%, while ILP-1c is able to increase the saving from 68.42% to 73.98%, i.e., an increase by 8.12%. Similarly, for GÉANT, GU-1b and ILP-1b increase the saving ε_T by 3.26% and 4.33%, respectively, i.e., from 62.16% to 64.19% and 62.39% to 65.09%, as compared to an increase of 9.6% in GU-1c. The reason is because there are more paths that satisfy the delay constraint (c) than (b). Thus, GU-1c and ILP-1c have more alternative paths to reroute traffic, allowing them to turn off more cables than GU-1b and ILP-1b, respectively.

Next, let us see how the increase in energy saving ε_T due to the use of delay constraints (b) and (c) affect the average increase in path delay. As shown in Figure 3.3, both constraints (b) and (c) trade-off higher energy savings for longer path delays. Specifically, Figure 3.2 and Figure 3.3 show that we can gain higher

ε_T at the expense of longer delays. Figure 3.3 shows that GU-1c and ILP-1c produce significantly longer path delays than GU-1b and ILP-1b, respectively. In other words, constraint (c) produces higher ε_T than (b) but results in longer path delays compared to (b). Taking Abilene as an example, GU-1c and ILP-1c recorded an increase in path delay of 27.6% and 27.29%, respectively. On the other hand, GU-1b and ILP-1b respectively have an increase of 0% and 0.28%.

For large scale networks, i.e., Deltacom and TATA, GU-1 with both delay constraints (b) and (c) produce a very small increase in energy saving and path delay; see Figure 3.2 and Figure 3.3, respectively. The reason is because budget $B = \$1.2\text{M}$ could only be used to upgrade a small percentage of switches and thus there is a limited number of *routable* paths for each demand. To further analyze the effect of rerouting demands on ε_T and path delay in large scale networks, we rerun our simulation for Deltacom and TATA where we set $B = 5$ M. The larger budget allows GU-1b to upgrade 58.41% and 52.41% of the l-switches in Deltacom and TATA, respectively. It also produce an energy saving ε_T of 64.08% and 56.36%, respectively for both networks. On the other hand, when we use GU-1c, the energy saving ε_T of Deltacom and TATA is increased by 4.28% to 66.82% and 3.42% to 58.29%, respectively. Meanwhile, the path delays for both networks are increased, on average, by 97.58% and 98% respectively.

3.3.3. Effect of Increasing Budget

We now study the impact of increasing budget on energy saving ε_T and the percentage of upgraded switches $\mathcal{V}^T$. We consider six different budgets, i.e., $B \in \{\$200\text{K}, \$400\text{K}, \$600\text{K}, \$800\text{K}, \$1\text{M}, \$1.2\text{M}\}$ and set $T = 3$. This experiment uses GU-1b and ILP-1b and delay multiplier $\sigma = 1.1$. Figure 3.4 shows that both GU-1b and ILP-1b produce a higher energy saving ε_T when there is a large

budget, which allows both solutions to upgrade more switches and power off more cables. However, Figure 3.5 shows that a larger budget does not always make ILP-1b upgrade more l-switches. For example, ILP-1b upgrades 66.67% and 58.33% of l-switches in Abilene using a budget B of \$800K and \$1.2M, respectively. This is because the goal of ILP-1b is to upgrade l-switches to minimize the number of *on*-cables instead of maximizing the number of upgraded switches. Note that our problem considers the upgrade cost of switches may vary; some can be more expensive than the others. Consequently, ILP-1b may not always upgrade more switches for larger budget value B. On the other hand, Figure 3.5 shows that GU-1b consistently upgrades more l-switches for larger budgets. This result is consistent with Proposition 3.1.

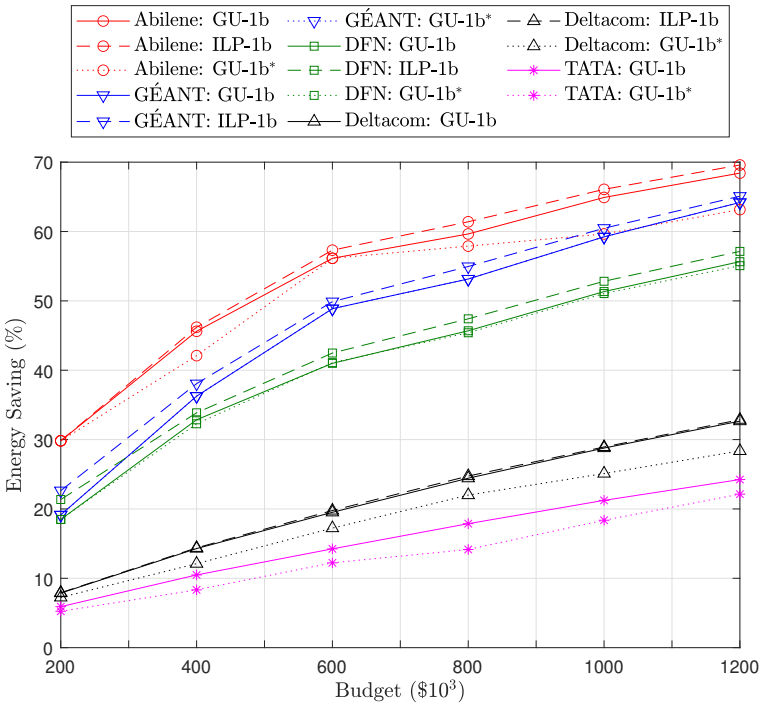


Figure 3.4. Energy saving (ε_T) for various budget B values.

Figure 3.4 also shows that the energy saving ε_T of GU-1b is slightly less

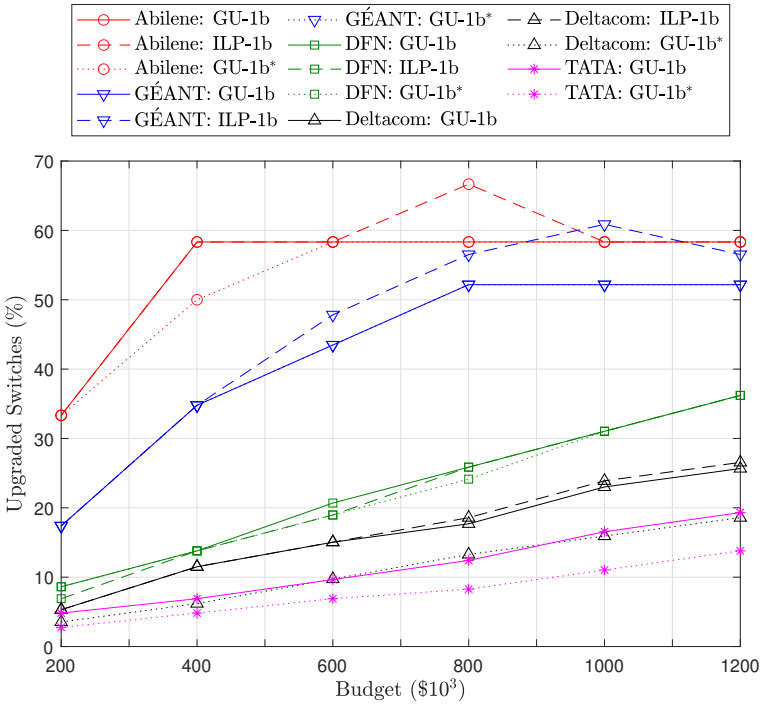


Figure 3.5. The percentage of s-switches ($\mathcal{V}^T$) for various budget B values.

than ILP-1b. More specifically, GU-1b produces energy saving ε_T that is only 1.6%, 4.82%, 4.83%, and 0.85% off from the optimal value for Abilene, GÉANT, DFN, dan Deltacom, respectively. The result is reasonable because GU-1b is a heuristic solution so its energy saving ε_T cannot be larger than that of ILP-1b. Figure 3.4 also shows that GU-1b produces energy saving ε_T of only up to 32.66% and 24.24% for Deltacom and TATA, respectively. These percentages are significantly lower than those of the other three networks, i.e., up to 68.42%, 64.19%, and 55.65% for Abilene, GÉANT and DFN, respectively. The reason is because Deltacom and TATA have a larger number of l-switches to upgrade than the other three networks. Thus, each allocated budget for the two networks can upgrade a significantly smaller percentage of l-switches. As an example, with a budget of B = \$1.2M, GU-1b is able to upgrade seven l-switches of Abilene, meaning GU-1b upgrades 58.33% of the total l-switches. Using the same budget

for Deltacom and TATA, GU-1b upgrades only 25.66% and 19.31% of the total l-switches, respectively. To validate our explanation for Deltacom and TATA, we rerun the experiment with a budget $B = \$5\text{M}$, which is sufficient to upgrade 58.41% and 52.41% of l-switches, respectively. We find that GU-1b produces energy saving ε_T of 64.08% for Deltacom and 56.36% for TATA.

Next, we compare two alternative switch selection criteria, namely w_u/p_u^t versus w_u as used in [84]. We record the resulting energy saving ε_T and the percentage of upgraded switches $\mathcal{V}^T$. Let GU-1b* denote GU-1b that uses criterion w_u. As shown in Figure 3.4, GU-1b gives a higher ε_T than GU-1b*, particularly for Abilene, Deltacom and TATA. As an example, for budget $B = \$1.2\text{M}$, GU-1b produces an energy saving of 68.42%, 32.66%, and 24.24% for Abilene, Deltacom and TATA, respectively. On the other hand, for the respective three networks, GU-1b* results in smaller energy saving of 63.16%, 28.36%, and 22.12%. Further, as shown in Figure 3.5, the percentage of upgraded switches $\mathcal{V}^T$ produced by GU-1b for all networks is never less than that of GU-1b*. The reason is because selecting switches using the ratio w_u/p_u^t allows GU-1b to upgrade a switch with the maximum *off*-cables per cost unit. Thus, a switch with the highest number of unused cables and with the cheapest price is more likely to be selected for upgrade. Moreover, the switch may have many neighbouring switches.

3.3.4. Effect of Increasing Number of Stages

This section analyzes how the number of stages affect the energy saving ε_T and the percentage of upgraded l-switches $\mathcal{V}^T$. We use a budget of $B = \$1.2\text{M}$ and vary T from one to five. This experiment uses GU-1b and ILP-1b with a delay multiplier of $\sigma = 1.1$. Figure 3.6 shows that the energy saving ε_T for Abilene, GÉANT and DFN decreases as T increases. More specifically, for Abilene, GÉANT, and

DFN, GU-1b results in the saving ε_T that ranges from 73.68% to 63.51%, 75.34% to 61.55%, and 59.48% to 56.55%, respectively, when T increases from one to five. GU-1b is able to generate energy saving that is close to that of ILP-1b. For example, the energy saving ε_T of GU-1b for Abilene, GÉANT, DFN, and Deltacom with $T = 5$ is only 2.16%, 2.25%, 2.67%, and 1.27%, respectively, off from the optimal value. In contrast, increasing T from one to five for Deltacom and TATA results in an increased energy saving ε_T from 34.63% to 38.57% and 25.4% to 30.35%, respectively. The reason is because the allocated budget value of $B = \$1.2\text{M}$ is able to upgrade a larger percentage of switches in smaller networks in earlier stages than in the larger networks. Thus, for smaller networks, there are fewer switches in later stages with cables that can be turned off, i.e., switches with $w_u > 0$. Further, as the traffic volume grows, the remaining switches will have a smaller number of *off*-cables so upgrading them does not significantly increase the energy saving ε_T. In contrast, for larger networks, due to cheaper upgrade cost, more switches can be upgraded in later stages, resulting in larger saving ε_T. Note that for these larger networks, increasing traffic volume also decreases the number of cables that can be switched off. However, the effect of increasing traffic volume is less prominent because there are more switches with unused cables to select in larger networks.

Figure 3.7 shows the percentage of upgraded l-switches $\mathcal{V}^T$ for different number of stages T. As shown in the figure, ILP-1b does not always upgrade more switches for larger T values. Recall that the goal of ILP-1b is not to maximize the number of upgraded switches but to maximize energy saving. In contrast, the number of l-switches upgraded by GU-1b never decreases with increasing stages T, which is consistent with Proposition 3.2. For Abilene, GU-1b obtains the same number of upgraded l-switches, i.e., 58.33% of $|V|$ switches when T increases from one to five. The reason is because the budget $B = \$1.2\text{M}$ is sufficient for GU-1b

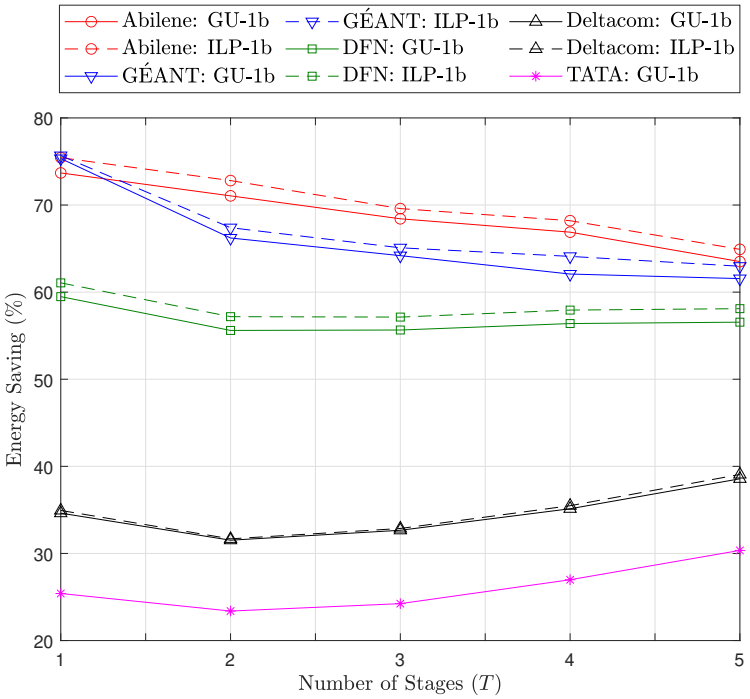


Figure 3.6. Energy saving (ε_T) for various T values.

to upgrade all l-switches of Abilene that have unused cables in only one stage. In contrast, when T value is increased from one to five, GU-1b produces higher percentage $\mathcal{V}^T$, i.e., from 47.83% to 52.17%, 20.69% to 46.55%, 15.93% to 43.36%, and 9.66% to 33.79% of $|V|$ switches for GÉANT, DFN, Deltacom and TATA, respectively. The reason is because for these four larger networks and a budget value of $B = \$1.2\text{M}$, GU-1b requires more than one stage to upgrade all l-switches that have unused cables.

3.3.5. Effect of Other Key System Parameters

This section studies the impact of increasing bundle size b_{uv}, cost depreciation rate ρ, traffic increase rate μ_d, and MLU threshold $U_{\max}$ on energy saving ε_T produced by GU-1b. Each simulation uses the same budget $B = \$1.2\text{M}$ for $T = 3$ stages and link capacity of 10 Gbps. Further, except in Section 3.3.5.1, we set $b_{uv} = 4$.

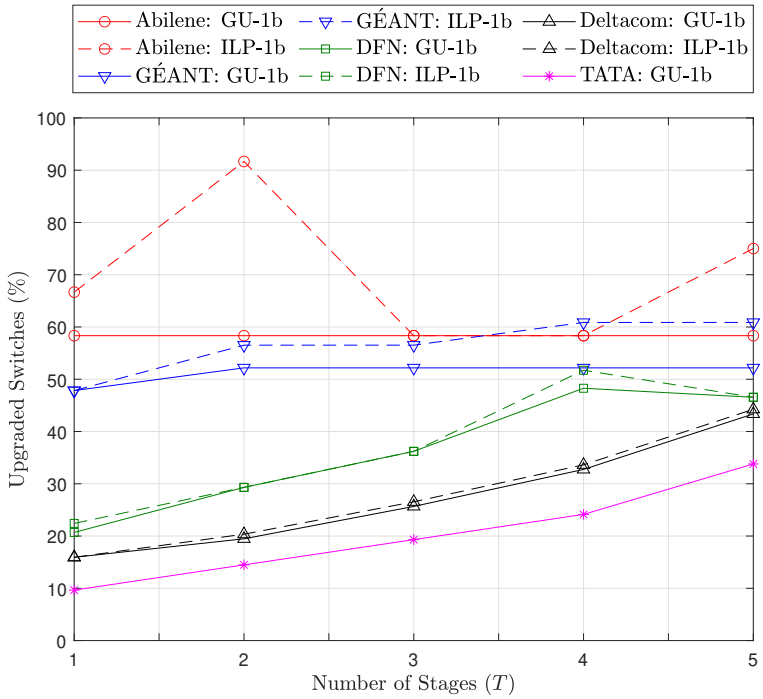


Figure 3.7. The percentage of s-switches ($\mathcal{V}^T$) for various T values.

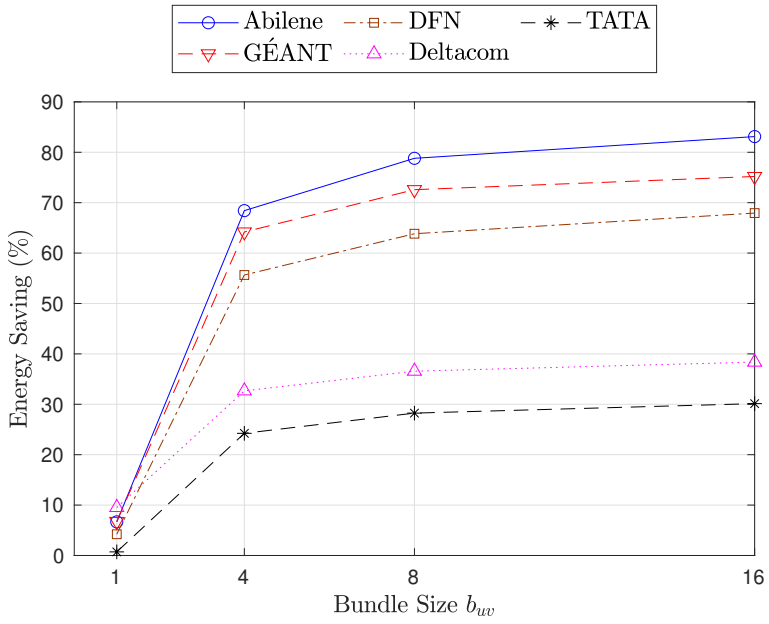


Figure 3.8. Energy saving for various bundle size b_{uv}.

3.3.5.1. Increase in Bundle Size

This evaluation uses four values of bundle size, i.e., $b_{uv} \in \{1, 4, 8, 16\}$. As shown in Figure 3.8, larger bundle size increases energy saving for all networks. For example, in GÉANT and Deltacom, raising the size from one to four cables significantly increases energy saving ε_T from 6.76% to 68.42% and 9.52% to 32.66%, respectively. Note that increasing b_{uv} from one to four decreases cable capacity γ from 10 to 2.5 Gbps. While flow size up to 2.5 uses only one cable for both cases, there are three unused cables that can be switched off for $b_{uv} = 4$, compared to none for $b_{uv} = 1$. However, increasing the size beyond four cables does not significantly improve energy saving. For example, using $b_{uv} = 16$ for GÉANT and Deltacom, the energy saving ε_T of GU-1b is 75.2% and 38.37%, respectively.

3.3.5.2. Increase in cost depreciation rate

This simulation uses three values of depreciation rate, i.e., $\rho \in \{10\%, 20\%, 40\%, 60\%, 70\%\}$. A higher depreciation rate means more switches can be upgraded in the later stages. The reason is because each stage has the same minimum budget $B^t = \$400K$, and the upgrade cost is cheaper at later stages. Thus, GU-1b with a larger depreciation rate produces more energy saving. For example, as shown in Figure 3.9, raising the rate ρ for GÉANT and TATA from 10% to 70% increases energy saving ε_T from 61.37% to 65.77% and 18.95% to 38.67%, respectively. For GÉANT, a budget of $B = \$1.2$M and a rate of $\rho = 10\%$ are sufficient to upgrade 52.17% of l-switches. For TATA, however, the percentage of upgraded switches is only 11.03%, which increases to 45.52% when $\rho = 70\%$. Therefore, the increase in energy saving for TATA is more significant than that for GÉANT.

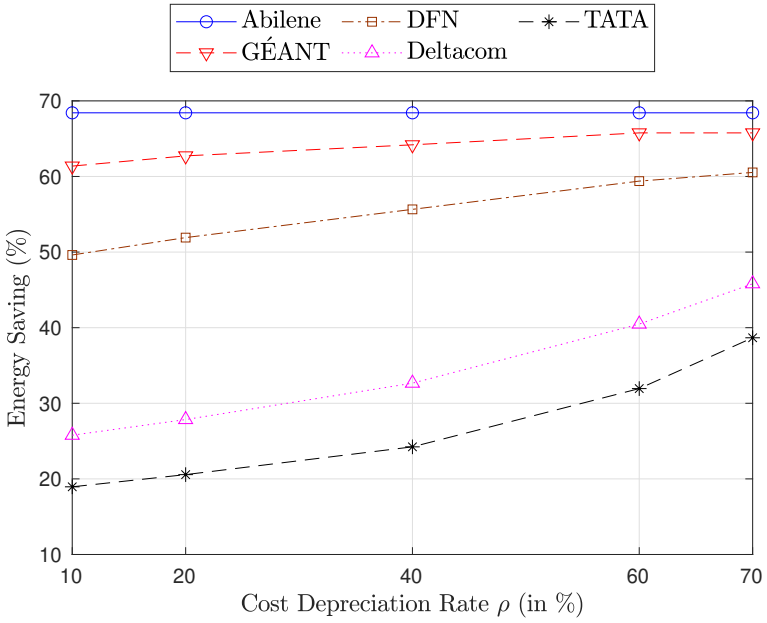


Figure 3.9. Energy saving for various depreciation cost rate ρ.

3.3.5.3. Increase in traffic rate

This evaluation considers two cases: 1) each demand d has the same rate $\mu_d \in \{22\%, 50\%, 60\%, 80\%\}$, and 2) each demand d has a random rate $\mu_d \in [22\%, 80\%]$.

Case 1): a larger increase rate μ_d requires more cables to carry traffic, resulting in fewer unused cables that can be switched off, and hence less energy saving. As shown in Figure 3.10a, for Abilene and GÉANT, GU-1b decreases energy saving ε_T from 68.42% to 65.79% and 64.19% to 59.91%, respectively, when the rate μ_d increases from 22% to 80%.

Case 2): for each network in Table 3.1, we use a Normal distribution with a mean of 40% and standard deviation of 0.1 to generate rate μ_d that has a value between 22% and 80% for each demand d. We generate a random rate μ_d for ten times. Thus, for each network with an initial set of demand D^0, each iteration $j = \{1, 2, \ldots, 10\}$ obtains a set $M_j = \{\mu_1, \mu_2, \ldots, \mu_{|D^0|}\}$. We run GU-1b on each set M_j of each network. Our simulation shows that assigning random rate

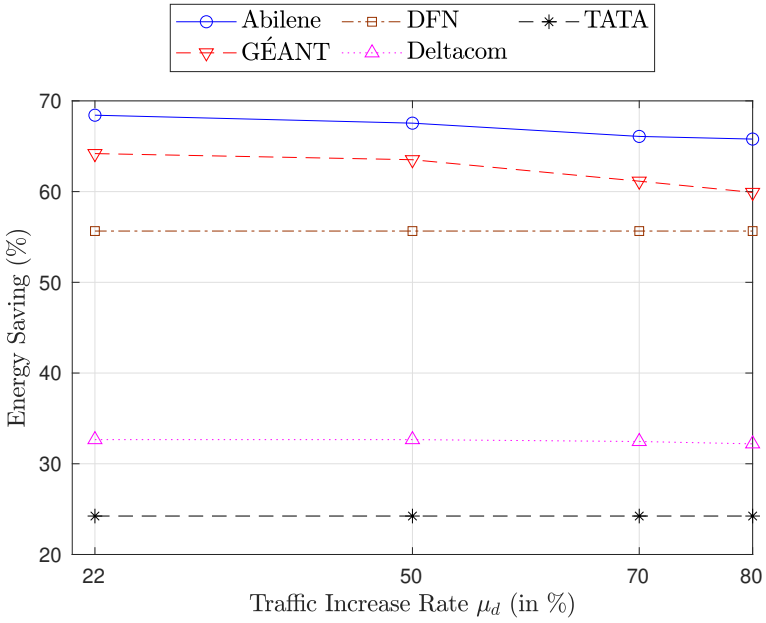
μ_d for each demand d does not significantly affect energy saving. As shown in Figure 3.10b, for Abilene, GU-1b produces the energy saving ε_T of 67.54% for eight sets of M_j, and 67.84% for the other two sets. Similarly, for GÉANT, it produces energy saving ε_T from 62.5% to 63.51%.

3.3.5.4. Increase in MLU Threshold

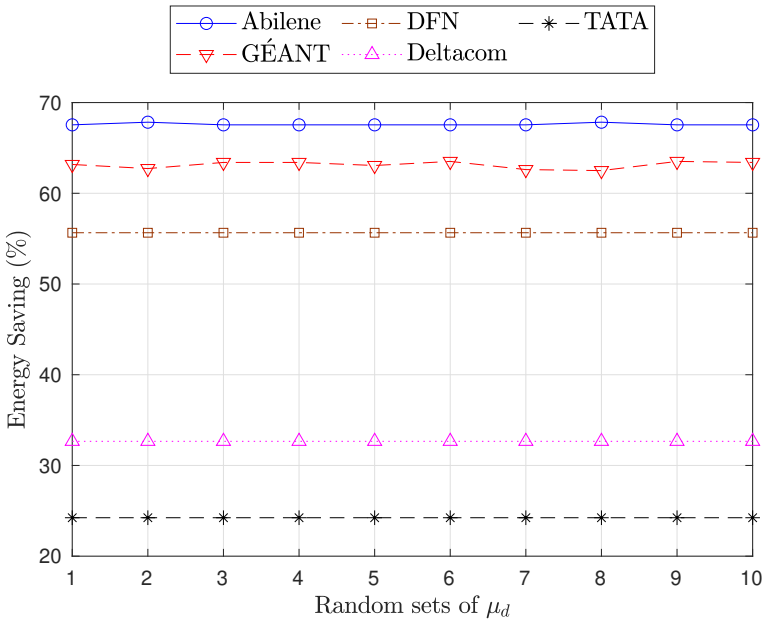
This evaluation uses MLU threshold of $U_{\max} \in \{20\%, 40\%, 60\%, 80\%\}$. Figure 3.11 shows the results of using various MLU thresholds. A larger $U_{\max}$ value results in each cable carrying more traffic, which in turn leads to more unused cables that can be switched-off, resulting in higher energy savings. As an example, for Abilene, GU-1b increases its energy saving ε_T from 53.51% to 68.42% when we increase $U_{\max}$ from 20% to 80%. Note that Figure 3.11 does not show the results for $U_{\max} = 20\%$ for GÉANT because the utilization of some links in the network cannot be less than 20%.

3.3.6. GU-1 and ILP-1 versus Local Search

This section evaluates the performance of GU-1 and ILP-1 against the Local Search (LS) algorithm [1] in terms of traffic controllability (TC) and energy saving. Recall that TC is the ratio between the total traffic volume that passes s-switches and the total traffic volume. Briefly, LS [1] aims to upgrade l-switches over multiple stages subject to a given budget constraint B. However, the goal of LS is to maximize the total TC over $T \geq 1$ stages. LS is allowed to spend its entire budget in one stage. Further, each traffic demand when using LS is routed via the shortest path, and each link consists of only one cable. We use the implementation of LS that is publicly available in [78]. For a fair comparison, we use GU-1a and ILP-1a that also route each demand via its shortest path. We



(a) Traffic rate μ_d)



(b) Random traffic rate μ_d

Figure 3.10. Energy saving for various traffic increase rate.

compute the energy saving for LS from the traffic volume of each link. Specifically, when using LS, for each link with traffic volume ω, it uses $\lceil \omega/2.5 \rceil$ cables.

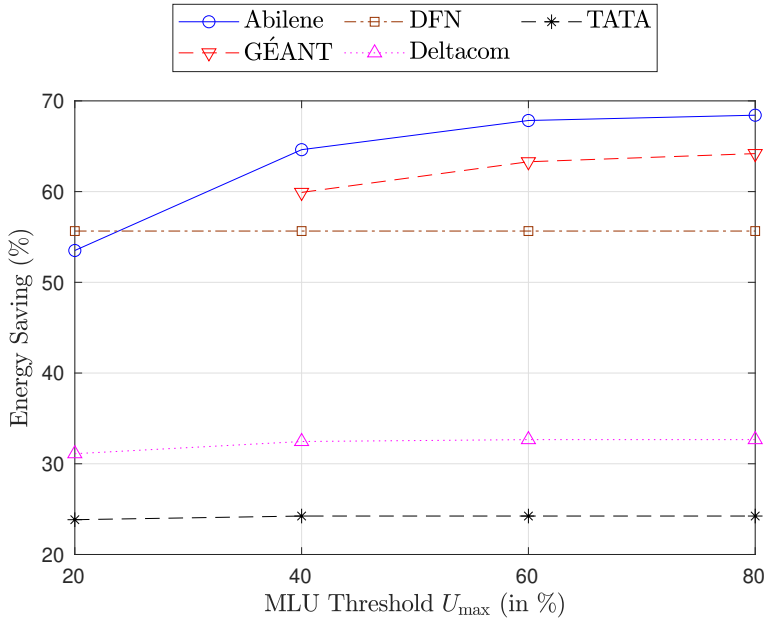


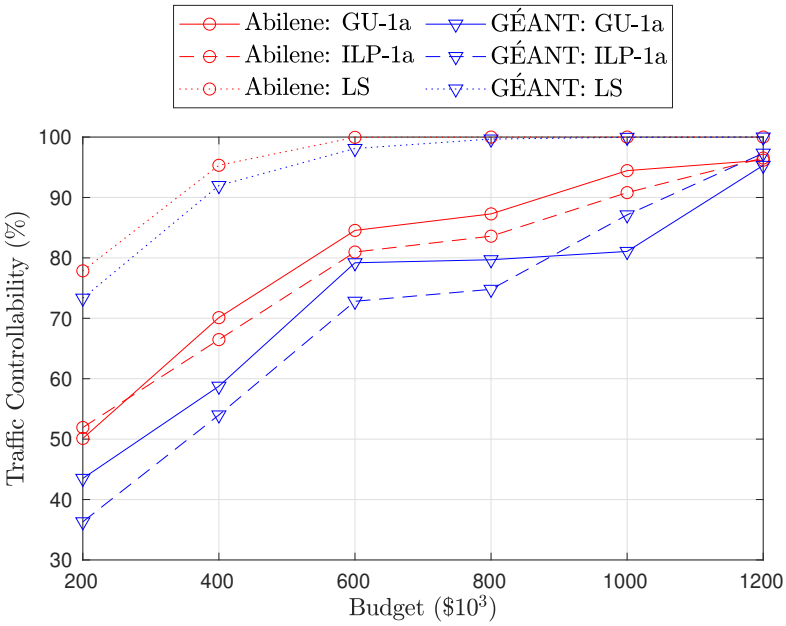
Figure 3.11. Energy saving for various MLU threshold $U_{\max}$.

Each directed link contains four cables with a capacity of 10 Gbps.

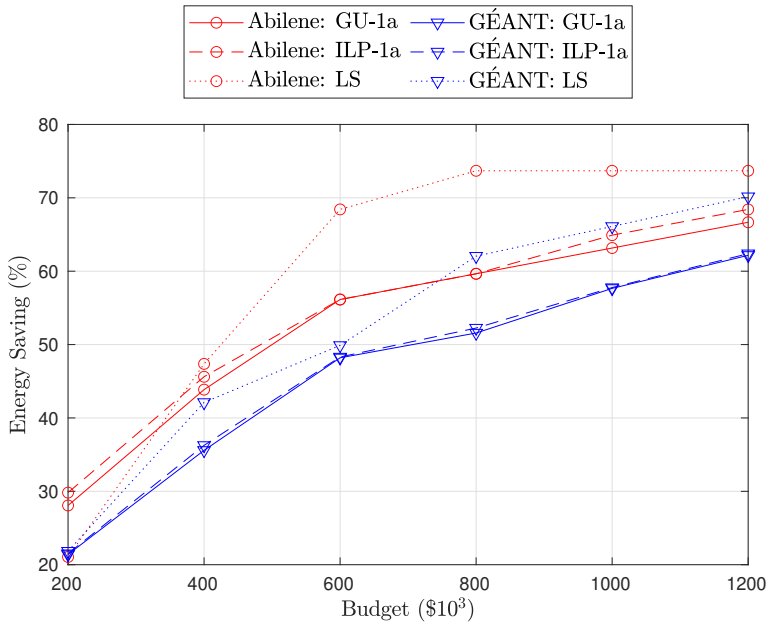
First, let us compare GU-1a, ILP-1a, and LS in terms of their *average* TC and energy saving ε_T for increasing budget $B \in \{\$200K, \$400K, \$600K, \$800K, \$1M, \$1.2M\}$. Each l-switch has an initial upgrade cost of \$100K. The number of stages is set to $T = 3$. Figure 3.12a shows that GU-1a, ILP-1a, and LS produce a higher average TC with a larger budget because they are able to upgrade more switches. When the number of s-switches increases, there are more demands that pass a s-switch, which increases TC. Figure 3.12a also shows that LS produces a higher average TC than GU-1a and ILP-1a. These results are expected because the goal of LS is to maximize the total TC over T stages. For example, for $B = \$200K$, LS has an average TC of 77.86% and 73.33% for Abilene and GÉANT, respectively. The average TC of GU-1a and ILP-1a for Abilene is 50.12% and 43.5%, respectively, while for GÉANT is 51.92% and 36.31%, respectively. However, as the budget increases from \$200K to \$1.2M, the difference between the average TC of GU-1a and LS declines from 35.63%

to 3.83% and 40.68% to 4.69% for Abilene and GÉANT, respectively. More specifically, when budget $B = \$1.2M$ is used for Abilene and GÉANT, the average TC produced by GU-1a (ILP-1a) reaches 96.17% (96.53%) and 95.31% (97.33%), respetively, compared to LS that produces an average TC of 100%. For larger budgets, GU-1a and ILP-1a are able to upgrade more switches at each stage. Thus, the average TC produced by both approaches can reach almost the same rate as LS. Note that LS requires a significantly longer CPU time than GU-1a and ILP-1a. For example, for Abilene (GÉANT) with a maximum budget of $B = \$1.2M$, LS takes 12.43 (210.79) seconds, while GU-1a and ILP-1a require only 0.005 (0.062) and 0.08 (2.01) seconds, respectively. We do not include the analysis for DFN, Deltacom and TATA because LS fails to produce results after running for 24 hours.

Figure 3.12b shows that GU-1a, ILP-1a, and LS produce a higher energy saving ε_T with a larger budget. This is because they are able to upgrade more switches with a larger budget. Figure 3.12b also shows that LS obtains less ε_T than GU-1a and ILP-1a for Abilene with a budget of $B = \$200K$. For GÉANT, LS produces similar energy saving ε_T as GU-1a and ILP-1a. Specifically, for Abilene and GÉANT, LS has an energy saving ε_T of 21.05% and 21.85%, respectively. On the other hand, for both networks, GU-1a produces 28.07% and 21.4%, respectively, and ILP-1a yields 29.82% and 21.51%, respectively. For Abilene and GÉANT with budget $B = \$200K$, LS upgrades 16.67% and 8.7% of l-switches, respectively. In contrast, GU-1a and ILP-1a upgrade twice of those upgraded by LS, i.e., 33.33% and 17.4% for both networks, respectively. For a larger budget $B > \$200K$, LS produces a higher ε_T than GU-1a and ILP-1a for both Abilene and GÉANT. This is because LS uses its entire budget B in stage $t = 1$, while GU-1a and ILP-1a have only a budget of B/T. Thus, in stage $t = 1$, LS could upgrade more switches as compared to GU-1a and ILP-1a, resulting in higher



(a) Traffic Controllability (TC).



(b) Energy Saving (ε_T.)

Figure 3.12. Traffic controllability and energy saving over $T = 3$ stages.

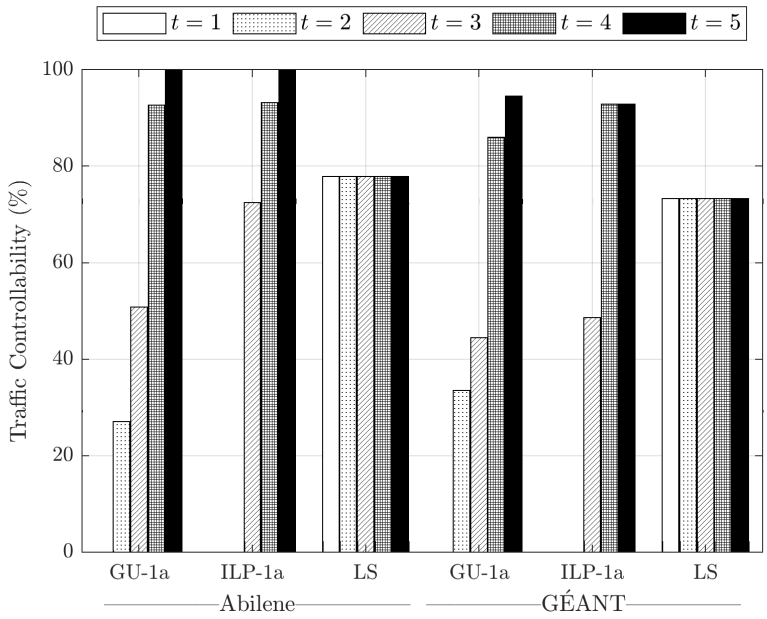
energy savings.

Next, we compare GU-1a, ILP-1a and LS in terms of their TC and energy

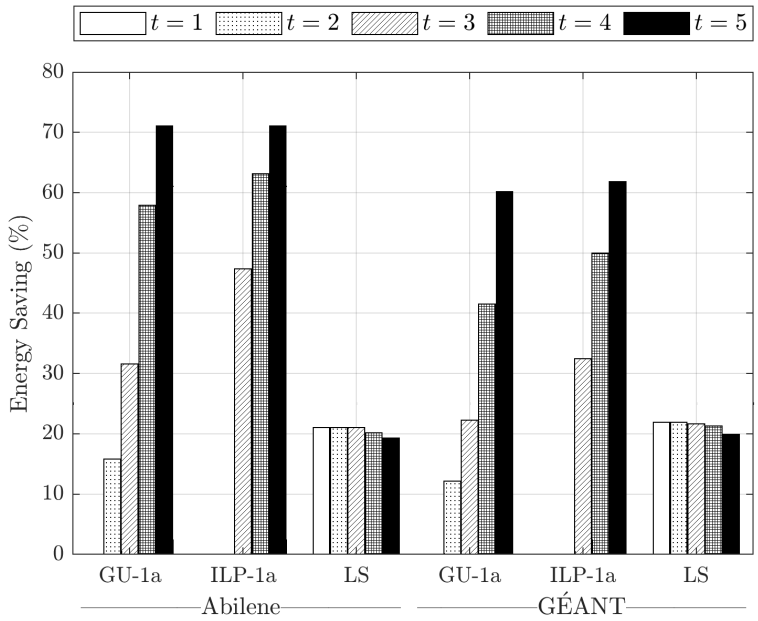
saving ε^t at each stage $t \in \{1, 2, 3, 4, 5\}$. We consider upgrading Abilene and GÉANT with a budget of $B = \$200\text{K}$ over $T = 5$ stages. Figure 3.13a and Figure 3.13b show respectively *per stage* TC and ε^t produced by GU-1a, ILP-1a, and LS for both networks. For both Abilene and GÉANT, Fig 3.13a shows that GU-1a and ILP-1a produce higher TC at later stages, while the TC of LS remains the same over the five stages. This is because LS has spent its entire budget at stage $t = 1$, and thus it has no more budget to upgrade more switches at stages beyond $t > 1$. In contrast, GU-1a and ILP-1a are able to spend a budget of only up to approximately B/T per stage. Here, GU-1a and ILP-1a do not upgrade switches at stage $t = 1$ and $t = \{1, 2\}$, respectively. Thus, both algorithms could upgrade more switches than LS in later stages. In general, GU-1a and ILP-1a are expected to upgrade more switches than LS. This is because the upgrade cost decreases over time.

Figure 3.13b shows the energy saving ε^t of GU-1a, ILP-1a, and LS for Abilene and GÉANT at each stage $t \in \{1, 2, 3, 4, 5\}$. The energy saving at each stage $t+1$ of both GU-1a and ILP-1a is larger than that at stage t; see Figure 3.13b. In contrast, the energy saving ε^t of LS decreases at later stages. This is because both GU-1a and ILP-1a are able to upgrade more switches at the later stages, which result in higher energy savings. In contrast, LS upgrades switches only at stage $t = 1$, and thus results in the same number of s-switches across the tested stages. As traffic size increases by $\mu_d = 22\%$ per stage for all demands, some cables need to be switched on at the later stages. Thus, the energy saving produced by LS decreases over the stages.

We remark that it is advantageous to save energy in later stages due to the rise in electricity cost. For example, reference [29] reports an expected annual increase of 3.29% in average energy expenditure by the commercial sector from 2020 to 2025. Consider the energy saving results in Figure 3.13b. For Abilene, at each



(a) Traffic Controllability (TC) per stage.



(b) Energy Saving ε^t per stage.

Figure 3.13. Traffic Controllability and Energy Saving for each Stage $t \in \{1, \ldots, 5\}$ and $B = \$200\text{K}$.

stage $t \in \{1, 2, 3, 4, 5\}$, GU-1a and LS produce an energy saving of $\varepsilon^t = \{0\%,$ 15.79%, 31.58%, 57.9%, 71.05%$\}$ and $\varepsilon^t = \{21.05\%,$ 21.05%, 21.05%, 20.18%,

19.3%}, respectively. Assuming an annual increase of 3.29% in energy expenditure and an initial cost of \$1 per *on*-cable, GU-1a will be able to save $\$(0\times1+0.1579\times 1.0329+0.3158\times1.0329^2+0.579\times1.0329^3+0.7105\times1.0329^4)=\$1.947$. On the other hand, LS has a cost saving of only $\$(0.2105+0.2174+0.2246+0.2224+0.2167)=\$1.095$. Similarly for GÉANT, GU-1a saves \$1.506, while LS only saves \$1.139.

3.4. Chapter Summary

This chapter addresses the first version of our GMSU problem, i.e., GMSU-1. GMSU-1 aims to upgrade a legacy network that support OSPF to SDN over multiple upgrade stages. The goal is to maximize the number of unused cables adjacent to *s*-switches that can be switch off to save energy. This chapter has presented an ILP called ILP-1 used to formulate GMSU-1 and a heuristic solution called GU-1 to solve GMSU-1. Our experiments on five real networks show that using a larger budget, more upgrade stages, and allowing longer path delays lead to higher energy savings. GU-1 is able to switch-off up to 68.42% of unused cables, and hence save energy, while maintaining the shortest path routing. By incorporating traffic rerouting, GU-1 increases energy savings by up to 9.6%. The simulation results show that, on average, GU-1 produces energy saving within 4.83% from the optimal saving obtained by ILP-1. We find that the average traffic controllability and energy saving produced by GU-1 is less than the existing technique, called LS, that may use all the available budget to upgrade the network in the first stage. However, GU-1 produces significantly higher traffic controllability and energy saving than LS at the later stages. As energy prices increase every year, GU-1 yields higher energy savings at later stages than in earlier stages in order to save more on energy cost. The next Chapter 4 proposes

the second version of GMSU, i.e., GMSU-2, that uses multi-path routing.

Chapter 4

Multi-Stage Upgrade for Green SDN/OSPF-ECMP Networks

This chapter addresses GMSU-2 problem to upgrade a legacy network that supports OSPF-ECMP to SDN over a pre-defined planning stages, and route each traffic demand via a set of multi paths. GMSU-2 aims to maximize the number of *off*-cables adjacent to s-switches to save energy. The goal is constrained by the maximum budget at each stage, maximum delay tolerance, and MLU. Further, GMSU-2 maintains the same routing service to users. More specifically, if a traffic demand in a legacy network is routed via link-disjoint paths, the demand in the upgraded network must be routed via at least two link-disjoint paths. Otherwise, the demand can be routed via multi-paths that can share common link(s), called *non*-link-disjoint paths, or even a single path. In addition, GMSU-2 requires each l-switch complies with OSPF-ECMP. More specifically, each traffic exiting from an l-switch must be equally split and sent via a set of $m \geq 1$ shortest paths. In this case, GMSU-2 includes OSPF-based link-cost setting [38], in addition to switch upgrade and multi-path traffic routing for both l-switches and s-switches.

The layout of this chapter is as follows. Section 4.1 presents GMSU-2 problem

formulation, Section 4.2 describes our proposed heuristic solution, Section 4.3 outlines the evaluation results, and Section 4.4 concludes the chapter and provides future research directions. Note that the work in this chapter has been published in [89] and [90].

4.1. Problem Formulation

The outline of this section is as follows. Section 4.1.1 presents the multi-path routing model and some notations used in GMSU-2. Table 4.1 summarizes the notations used in GMSU-2. Note that Section 2.5 provides the other models and notations used in this chapter. Section 4.1.2 formally defines the problem and introduces the two multi-path routing scenarios used in GMSU-2. Section 4.1.3 illustrates GMSU-2 and Section 4.1.4 mathematically models the problem. Finally, Section 4.1.5 shows that GMSU-2 is an NP-hard problem.

4.1.1. Multi-Path Routing Model

GMSU-2 uses $m \geq 1$ paths to route each traffic demand d. For each demand $d \in \{1, 2, \ldots, |D^0|\}$, let $P_d^x = \{P_{d,i}^x \mid \forall x \in V, x \neq \tau_d, \forall i \in \{1, 2, \ldots, |P_d^x|\}\}$ be a set of paths from node x to node τ_d. The delay of each path $P_{d,i}^x$, denoted by $\delta_{d,i}^x$ (in seconds), is computed as the sum of propagation delays over all links in the path, i.e., $\delta_{d,i}^x = \sum_{(u,v)\in P_{d,i}^x} \pi_{uv}$. GMSU-2 allows users to use paths that are up to $(\sigma - 1) \times 100\%$ longer than their original delays, i.e., $\delta_{\max,d} = \lceil \sigma \times \delta_{\min,d}^{s_d} \rceil$, for a delay multiplier $\sigma \in [1.0, 2.0]$. Let $\mathcal{P}_d^{s_d} \subset P_d^{s_d}$ denote a set of paths in $P_d^{s_d}$ that satisfy delay constraint $\delta_{\max,d}$. Also, let $y_{d,uv,i}^t$ be a binary variable that is set to 1 (0) if link (u, v) is included (not included) in any path $P_{d,i}^{s_d}$. Thus, each path $P_{d,i}^{s_d} \in P_d^{s_d}$ is represented as $P_{d,i}^{s_d} = \{(u, v) \mid y_{d,uv,i}^t = 1, \forall (u, v) \in E\}$. Further, let $R_d^{s_d,t} \subseteq \mathcal{P}_d^{s_d}$ represent a set of $|R_d^{s_d,t}| \geq 1$ paths that are used to route demand d.

Table 4.1. Specific Notations for GMSU-2.

Notation	Definition
P_d^x	A set of paths from node x to node τ_d.
$\delta_{\min,d}^{s_d}$	The minimum delay among all paths in $P_d^{s_d}$.
$\mathcal{P}_d^{s_d}$	A set of paths in $P_d^{s_d}$ with delay within $\delta_{\max,d}$.
$f_{d,i}^t$	The traffic size carried by $\mathcal{P}_{d,i}^{s_d} \in \mathcal{P}_d^{s_d}$ at stage t.
$f_{d,uv}^t$	The flow size of demand d along link (u,v) at stage t.
ψ_{uv}^t	The cost of link (u,v) at stage t.
ψ^t	The set of link costs at stage t.
ψ^0	The set of initial link costs.
$\Psi_{d,i}^{x,t}$	The cost of each path $P_{d,i}^x \in P_d^x$ at stage t.
$\Psi_{\min,d}^{x,t}$	The minimum cost of all paths in P_d^x at stage t.
$R_d^{s_d,0}$	A set of shortest paths in $\mathcal{P}_d^{s_d}$.
$R_d^{s_d,t}$	A set of paths to route demand d at stage t.
$o_{u,a}^t$	The traffic size carried by a shortest path from l-switch u to destination a.
$z_{a,uv}^t$	An indicator of whether link (u,v) is on any shortest path to destination a at stage t.
$h_{v,a}^t$	The path cost from switch v to switch a at stage t.
I	A non-negative integer number not exceeding $2^{16}-1$.
$y_{d,uv,i}^t$	An indicator set to 1 if link (u,v) is included in path $\mathcal{P}_{d,i}^{s_d} \in \mathcal{P}_d^{s_d}$.
$y_{d,i}^t$	An indicator set to 1 if path $\mathcal{P}_{d,i}^{s_d} \in \mathcal{P}_d^{s_d}$ is selected to route demand d at stage t.
l_d	An indicator set to 1 if the shortest paths in $R_d^{s_d,0}$ are link disjoint.

GMSU-2 splits each demand d into $m \geq 1$ traffic flows. Let $f_{d,i}^t \leq \omega_d^t$ denote the flow size or volume of demand d carried by path $R_{d,i}^{s_d,t} \in R_d^{s_d,t}$ for every stage $t \in \{1,2,\ldots,T\}$. Further, GMSU-2 uses $f_{d,uv}^t = \sum_{(u,v)\in R_{d,i}^{s_d,t}, i\in\{1,2,\ldots,|R_d^{s_d,t}|\}} f_{d,i}^t$ to denote the flow size of demand d along link (u,v) at stage t. If demand d flows from nodes u to v, the flow $f_{d,uv}^t > 0$ and $f_{d,vu}^t = 0$. On the other hand, $f_{d,uv}^t = 0$ and $f_{d,vu}^t > 0$ if demand d flows from nodes v to u. Thus, the total traffic volume f_{uv}^t is the total volume of all traffic flows that are carried by link (u,v) at stage t, i.e., $f_{uv}^t = \sum_{d\in\{1,2,\ldots,|D^t|\}} f_{d,uv}^t$.

In OSPF-ECMP, each link is assigned with an OSPF cost. Let $\mathrm{I} = 2^{16-1}$ represent the maximum OSPF cost that can be assigned to each link (u,v) [38]. Here,

each link (u, v) in stage t has cost $\psi_{uv}^t \in \{1, 2, \ldots, \mathrm{I}\}$. Let $\psi^t = \{\psi_{uv}^t \mid \forall (u, v) \in E, \forall t \in \{1, \ldots, T\}\}$ denote a set of link costs for stage t. Thus, ψ^0 denotes the set of initial link costs. The cost of each path $P_{d,i}^x \in P_d^x$ in stage t, denoted by $\Psi_{d,i}^{x,t}$, is computed as the sum of link cost in ψ^t over all links on the path from node x to destination τ_d, i.e., $\Psi_{d,i}^{x,t} = \sum_{(u,v) \in P_{d,i}^x} \psi_{uv}^t$. Let $\Psi_{\min,d}^{x,t}$ denote the minimum cost of all paths in P_d^x at stage t. A path $P_{d,i}^x \in P_d^x$ is called the *shortest path* if its cost is equal to the minimum cost, i.e., $\Psi_{d,i}^{x,t} = \Psi_{\min,d}^{x,t}$. Let $R_d^{s_d,0} \in R^0$ be a set of shortest paths of demand d. Note that the shortest paths in $G^0(V, E)$ have the shortest delay, i.e., $\psi_{uv}^0 = \pi_{uv}$, and thus we have $R_d^{s_d,0} \subseteq \mathcal{P}_d^{s_d}$. Further, any link cost ψ_{uv}^t can be adjusted at stage t and hence, it does not affect the link delay π_{uv}. Finally, we use $R_d^{s_d,t} \in R^t$ to denote a set of paths to route demand d at stage t.

4.1.2. GMSU-2 Problem Definition

Given a legacy network $G^0(V, E)$ that supports OSPF-ECMP, GMSU-2 upgrades a subset l-switches in $G^0(V, E)$ to s-switches V^T over a pre-defined $T \geq 1$ stages. GMSU-2 aims to maximize the total number of unused cables which can be powered off by the upgraded switches per Eq. (2.2). It considers the following three main constraints:

1. **Budget** B: The total cost to upgrade l-switches at each stage t does not exceed the maximum budget B^t. Recall that (i) $\mathrm{B} = \sum_{t=1}^{T} B^t$, and (ii) the upgrade cost of each switch depreciates with a given rate ρ per stage,

2. **Routing**: Each data traffic demand $d \in \{1, \ldots, |D^t|\}$ at each stage t is routed via a set of $m \geq 1$ multi-paths, denoted by $R_d^{s_d,t}$, where each of the paths must satisfy a given path delay constraint $\delta_{\max,d}$. GMSU-2 considers: (i) each s-switch may need to split traffic *unequally* over multi-paths, and

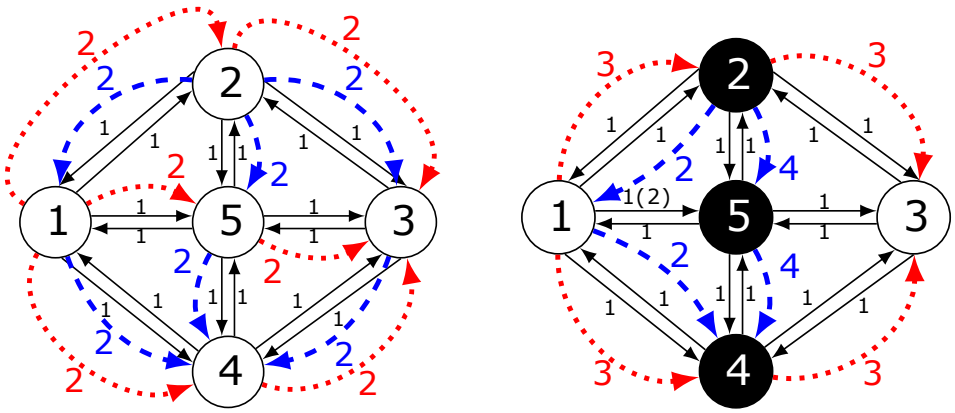
(ii) link costs may need to be adjusted to ensure each l-switch complies with OSPF-ECMP.

3. **Link capacity**: The total traffic volume on each link $(u, v) \in V$ at each stage t is within the maximum capacity of the link $c_{uv}^t = (n_{uv}^t / b_{uv}) \times c_{uv} \times U_{\max}$. Here, GMSU-2 considers the traffic volume of each demand d at each stage t, i.e., ω_d^t, increases with a rate of μ_d per stage.

4.1.3. GMSU-2 Illustration

To illustrate GMSU-2, consider a legacy network $G^0(V, E)$ in Figure 4.1a. Each link (u, v) has the indicated cost and two cables, with each cable has a capacity of five units of data and an MLU of $U_{\max} = 100\%$. Assume the traffic demand from switch 1 to switch 3 is six units, which we denote as $(s_1 = 1, \tau_1 = 3, \omega_1^0 = 6)$. There is also another traffic demand $(s_2 = 2, \tau_2 = 4, \omega_2^0 = 6)$. Thus, we have $D^0 = \{(1, 3, 6), (2, 4, 6)\}$. As shown in Figure 4.1a, switch 1 splits the first demand *equally* into three equal-cost paths $R_1^{1,0} = \{(1, 2, 3), (1, 4, 3), (1, 5, 3)\}$, each with a flow of size two and path cost of two; see the dotted lines. The second demand is also split in a similar manner; see the dashed lines. Assume that an unused cable can be switched off only if at least one of its end nodes is an s-switch. In this case, there is no energy saving.

Now consider a scenario where the upgrade is carried out over one stage with a total budget of $B = \$45$ and the cost to upgrade each l-switch is $p_u^1 = p_u^0 = \$15$. First, consider upgrading l-switches in the set $V^1 = \{1, 2, 3\}$ and the same traffic split and routing as in Figure 4.1a. This allows us to turn off 19 unused cables: one cable in links $\{(1, 2), (2, 1), (2, 3), (2, 5), (1, 5), (5, 3), (1, 4), (3, 4), (4, 3)\}$, and two cables in links $\{(3, 2), (5, 2), (5, 1), (3, 5), (4, 1)\}$. This leads to an energy saving of $\varepsilon^1 = 19/32 \times 100\% = 59.38\%$. As another example, consider upgrading



(a) OSPF-ECMP. **(b)** SDN/OSPF-ECMP.

Figure 4.1. An illustration (a) with an equal distribution of traffic flow over equal-cost multiple paths (OSPF-ECMP), and (b) with a link cost adjustment and each source s-switch can split an unequal amount of traffic. Nodes ○ and ● represent an l-switch and s-switch, respectively. The number next to each link denotes its cost. Lines ····· and - - - denote the paths for demand $(1 \rightarrow 3, 6)$ and $(2 \rightarrow 4, 6)$, with the number next to each line indicates the traffic volume.

l-switches in $V^1 = \{2, 4, 5\}$ (see Figure 4.1b). We see l-switch 1 splitting demand $(s_1 = 1, \tau_d = 3, \omega_1^1 = 6)$ *equally* onto shortest paths $R_1^{1,1} = \{(1, 2, 3),\ (1, 4, 3)\}$; each with a flow of size three. Here, we adjust the cost of link $(1, 5)$ from $\psi_{(1,5)}^1 = 1$ to $\psi_{(1,5)}^1 = 2$ so that path $(1, 5, 3)$ is no longer the shortest path for demand $(1, 3, 6)$; see the link cost in a bracket. On the other hand, s-switch 2 splits demand $(s_2 = 2, \tau_2 = 4, \omega_2^1 = 6)$ onto paths $R_2^{2,1} = \{(2, 1, 4),\ (2, 5, 4)\}$ with *unequal* flow sizes of two and four, respectively. Note that the shortest path $(2, 3, 4)$ is not used so that one cable of link $(3, 4)$ can be switched off. Thus, s-switch 4 and s-switch 5 can now turn off six more cables, i.e., one additional cable on links $(1, 5)$, $(5, 3)$, $(5, 4)$, $(3, 4)$ and two more cables on link $(4, 5)$, which yield a higher energy saving of $\varepsilon^1 = (6 + 19)/32 \times 100\% = 78.12\%$. After the legacy network in Figure 4.1a is upgraded to a *hybrid* SDN in Figure 4.1b, both demands are routed via link-disjoint paths.

4.1.4. GMSU-2 Mathematical Model

GMSU-2 considers two routing scenarios: a) non-link-disjoint paths and b) link-disjoint paths. The traffic of a demand d in scenario-a is split onto multi-paths and these paths can have common link(s). In contrast, the selected paths to route a traffic demand d in scenario-b must not have any common link. We mathematically model GMSU-2 as an MIP, namely MIP-2a (4.1) for scenario-a, and MIP-2b (4.14) for scenario-b. Section 4.1.4.1 outlines MIP-2a, followed by MIP-2b in Section 4.1.4.2. Appendix C provides the code implementation of MIP-2a as well as MIP-2b using Gurobi API for C++ [83]. The appendix also encloses the result of solving MIP-2b using the small example in Figure 4.1. Note that three constraints in MIP-2a, i.e., (4.8), (4.10) and (4.11), and one constraint in MIP-2b, which is (4.17), contain multiplication of two decision variables. In the implementation, the constraints are handled by Gurobi solver using method GRBModel.AddQConstr() [91].

4.1.4.1. Scenario-a: Non-Link-Disjoint-Path Routing

MIP-2a, see (4.1), consists of eight decision variables: (i) the number of *on*-cables n^t_{uv} for each link (u, v), (ii) the binary variable x^t_u as an indicator of whether l-switch u is upgraded at stage t, (iii) the binary variable $y^t_{d,uv,i}$ to indicate if link (u, v) is included in path $P^{s_d}_{d,i} \in P^{s_d}_d$, (iv) the flow size $f^t_{d,i}$ of demand d carried by path $R^{s_d,t}_{d,i}$ at stage t, (v) the binary variable $z^t_{a,uv}$ to indicate whether link (u, v) at stage t is on the shortest path from l-switch u to destination a, (vi) the traffic volume $o^t_{u,a}$ carried via each shortest path from l-switch u to destination a, (vii) the binary variable $h^t_{u,a}$ to denote the path cost from l-switch u to destination a, and (viii) the cost ψ^t_{uv} of link (u, v) at stage t. It aims to minimize the number of *on*-cables over T stages, which is equivalent to maximizing energy saving in

Eq. (2.2). The constraints of MIP-2a can be grouped into three main categories, i.e., constraints that are related to (i) switch upgrade, (ii) traffic routing for s-switches, (iii) traffic routing for l-switches, and (iv) domain of variables.

$$\begin{aligned} \min_{n^t_{uv}, x^t_u, y^t_{d,uv,i}, f^t_{d,i}, z^t_{a,uv}, o^t_{u,a}, h^t_{u,a}, \psi^t_{uv}} \quad & \sum_{t=1}^{T} \sum_{(u,v)\in E} n^t_{uv} \\ \text{s.t.} \quad & (4.2) - (4.13). \end{aligned} \tag{4.1}$$

(i) Switch upgrade: In constraint (4.2), each switch must be upgraded only once. Constraint (4.3) ensures that the total upgrade cost at each stage is less than or equal to $B^t = B/T + \Delta B^{t-1}$, while constraint (4.4) enforces all cables of l-links are powered on. Note that only cables in c-links can be turned off. The formulation of constraint (4.4) is a simple form of its real implementation that uses the set of linear equations (B.1) in Appendix B.

$$\sum_{t=1}^{T} x^t_u \leq 1, \tag{4.2}$$

$$\sum_{u\in V} (p^t_u \times x^t_u) \leq \sum_{k=1}^{t} B^k - \sum_{k=1}^{t-1} \sum_{u\in V} (p^k_u \times x^k_u), \tag{4.3}$$

$$n^t_{uv} = \max\left(n^t_{uv}, b_{uv} \times \left(1 - \sum_{k=1}^{t} x^k_u - \sum_{k=1}^{t} x^k_v\right)\right). \tag{4.4}$$

(ii) Traffic routing for s-switches: Constraint (4.5) conserves flows and ensures there is at least one path connecting source s_d to destination τ_d. Constraint (4.6) ensures that the traffic volume $f^t_{d,i}$ of each selected path $i \in \{1, 2, \ldots, |\mathcal{P}^{s_d}_d|\}$ of demand d sums to ω^t_d. Constraints (4.7) and (4.8) enforce each selected path i that routes demand d to meet the delay tolerance $\delta_{\max,d}$ and link capacity c^t_{uv} of each link (u, v) on the path, respectively. Constraint (4.9) limits

the number of *on*-cables to the bundle size of each link.

$$\sum_{(u,v)\in E} y^t_{d,uv,i} - \sum_{(v,u)\in E} y^t_{d,vu,i} = \begin{cases} 1, & u = s_d \\ -1, & u = \tau_d \\ 0, & u \neq s_d, \tau_d \end{cases}, \tag{4.5}$$

$$\sum_{i=1}^{|\mathcal{P}_d^{s_d}|} f^t_{d,i} = \omega^t_d, \tag{4.6}$$

$$\sum_{(u,v)\in E} (y^t_{d,uv,i} \times \pi_{uv}) \leq \delta_{\max,d}, \tag{4.7}$$

$$\sum_{d=1}^{|D^t|} \sum_{i=1}^{|\mathcal{P}_d^{s_d}|} (y^t_{d,uv,i} \times f^t_{d,i}) \leq (n^t_{uv}/b_{uv}) \times U_{\max} \times c_{uv}, \tag{4.8}$$

$$0 \leq n^t_{uv} \leq b_{uv}. \tag{4.9}$$

(iii) Traffic routing for *l*-switches: Constraints (4.10)–(4.12) ensure that the traffic volume from l-switch u to destination a is split into equal sized segments; each of which has volume $o^t_{u,a}$ and is routed via each shortest path from u to a. Thus, the cost $h^t_{u,a}$ is minimum and ψ^t_{uv} is in the range $\{1, 2, \ldots, \mathrm{I}\}$.

$$\sum_{d=1,\tau_d=a}^{|D^t|} \sum_{i=1}^{|\mathcal{P}_d^{s_d}|} y^t_{d,uv,i} \times f^t_{d,i} \leq z^t_{a,uv} \times \sum_{d=1,\tau_d=a}^{|D^t|} \omega^t_d, \tag{4.10}$$

$$0 \leq o^t_{u,a} - \sum_{d=1,\tau_d=a}^{|D^t|} \sum_{i=1}^{|\mathcal{P}_d^{s_d}|} y^t_{d,uv,i} \times f^t_{d,i} \leq (1 - z^t_{a,uv}) \times \sum_{d=1,\tau_d=a}^{|D^t|} \omega^t_d, \tag{4.11}$$

$$(1 - z^t_{a,uv}) \leq h^t_{v,a} + \psi^t_{uv} - h^t_{u,a} \leq (1 - z^t_{a,uv}) \times \mathrm{I}, \tag{4.12}$$

(iv) Domain of variables: Finally, constraint (4.13) defines the domain of

all decision variables.

$$y^t_{d,uv,i}, x^t_u, z^t_{a,uv} \in \{0,1\}, \quad f^t_{d,i}, o^t_{u,a}, h^t_{u,a} \geq 0. \tag{4.13}$$

Except (4.2), all constraints in MIP-2a (4.1) are for each stage $t \in \{1,2,\ldots,T\}$. Constraint (4.8), (4.9) and (4.4) exist for all links $(u,v) \in E$, while constraint (4.2) applies to each $u \in V$. Constraint (4.5) is for each node $u \in V$, traffic demand $d \in \{1,2,\ldots,|D^t|\}$, and path $i \in \{1,2,\ldots,|\mathcal{P}^{s_d}_d|\}$. While constraint (4.6) applies to each demand $d \in \{1,2,\ldots,|D^t|\}$, constraint (4.7) considers all demands and $|\mathcal{P}^{s_d}_d|$ paths for each demand. Finally, constraints (4.10)–(4.12) are evaluated for every destination $a \in V$ and each link $(u,v) \in E$, with a starting node $u \in V$ is an l-switch, i.e., $x^t_u = 0$.

4.1.4.2. Scenario-b: Link-Disjoint Path Routing

MIP-2a (4.1) is extended to MIP-2b (4.14) to support link-disjoint path routing. More specifically, if traffic demand d in legacy network $G^0(V,E)$ is routed via two or more link-disjoint shortest paths, i.e., $|R^{s_d,0}_d| > 1$, MIP-2b must route the demand via at least two link-disjoint paths in $\mathcal{P}^{s_d}_d$. Otherwise, MIP-2b can route the demand via any one or more paths in $\mathcal{P}^{s_d}_d$. Recall that $\mathcal{P}^{s_d}_d$ is a set of paths from s_d to τ_d with each path has a delay within $\delta_{\max,d}$.

$$\begin{aligned} n^t_{uv}, x^t_u, y^t_{d,uv,i}, f^t_{d,i}, z^t_{a,uv}, o^t_{u,a}, h^t_{u,a}, \psi^t_{uv}, y^t_{d,i} \quad & \min \quad \sum_{t=1}^{T} \sum_{(u,v)\in E} n^t_{uv} \\ & \text{s.t.} \\ (4.2) - (4.12) \text{ and } (4.15) - (4.18). & \end{aligned} \tag{4.14}$$

MIP-2b uses all constraints of MIP-2a (4.1). In addition, MIP-2b includes four additional constraints to ensure a traffic demand that is originally routed

via link-disjoint shortest paths remains using at least two of such paths. Let $y^t_{d,i}$ be another decision variable included in MIP-2b to indicate whether the path $\mathcal{P}^{s_d}_{d,i} \in \mathcal{P}^{s_d}_d$ is selected to route demand d at stage t. Let l_d be another indicator which is set to 1 if the shortest paths in $R^{s_d,0}_d$ are link-disjoint and $|R^{s_d,0}_d| > 1$. On the other hand, if $R^{s_d,0}_d$ contains either one path or non link-disjoint multi-paths, l_d is set to zero. Note that the value of l_d is pre-defined. Constraint (4.15) sets $y^t_{d,i} = 1$ if path $\mathcal{P}^{s_d}_{d,i}$ is able to carry the traffic of demand d, i.e., it has $f^t_{d,i} > 0$. Otherwise, both constraints set $y^t_{d,i} = f^t_{d,i} = 0$, which indicates that path $\mathcal{P}^{s_d}_{d,i}$ does not carry traffic. For every $l_d = 1$, constraint (4.16) guarantees at least two paths in $\mathcal{P}^{s_d}_d$ are selected to route demand d. Then, constraint (4.17) ensures that every pair of selected paths are link-disjoint. In this case, constraint (4.17) evaluates every link $(u, v) \in E$ to ensure that link (u, v) is not simultaneously used by both paths $\mathcal{P}^{s_d}_{d,i}$ and $\mathcal{P}^{s_d}_{d,j}$, i.e., both $y^t_{d,uv,i}$ and $y^t_{d,uv,j}$ cannot be equal to one. For each $l_d = 0$, constraints (4.16) and (4.17) ensure that there is at least one selected path to route demand d. In this case, if there is more than one selected path, they are not necessarily link-disjoint. Note that constraints (4.15) to (4.17) apply to every demand $d \in \{1, 2, \ldots, |D^t|\}$ at each stage $t \in \{1, 2, \ldots, T\}$. Finally, constraint (4.18) defines the domain all variables including the additional decision variable $y^t_{d,i}$.

$$y^t_{d,i} \leq f^t_{d,i} \leq y^t_{d,i} \times \omega^t_d, \tag{4.15}$$

$$\sum_{i=1}^{|\mathcal{P}^{s_d}_d|} y^t_{d,i} \geq l_d + 1, \tag{4.16}$$

$$y^t_{d,i} \times y^t_{d,uv,i} + y^t_{d,j} \times y^t_{d,uv,j} \leq (1 - l_d) + 1, \tag{4.17}$$

$$y^t_{d,uv,i}, x^t_u, z^t_{a,uv}, y^t_{d,i} \in \{0, 1\} \; f^t_{d,i}, o^t_{u,a}, h^t_{u,a} \geq 0. \tag{4.18}$$

4.1.5. GMSU-2 Complexity Analysis

GMSU-2 is related to two NP-hard problems: (i) OSPF cost setting problem [38]: given a network $G(V, E)$, maximum link utilization for each link $(u, v) \in E$, and a set of traffic demands, assign an integer cost for each link to optimize a given network performance metric, e.g., network delay; and (ii) 0-1 Multiple Knapsack Problem (MKP) [87]: given m items, each of which has a profit and weight, and T knapsacks, each of which has a maximum weight capacity, select T-disjoint subsets of items that maximize the total profit, subject to each subset having a total weight no more than its knapsack's capacity.

With respect to problem (i), the network performance of interest is the minimum number of *on*-cables that have sufficient capacity to carry traffic demands. The cost assigned to each link is used to calculate the shortest path from each l-switch to any destination τ_d in D^t. These shortest paths define the total traffic volume on each link, which then determine the number of *on*-cables. Thus, GMSU-2 is at least as hard as the problem in (i).

GMSU-2 can be reduced to MKP when (a) there is no depreciation in switch upgrade cost, and (b) the number of *on*-cables per link $(u, v) \in E$ is known, i.e., the traffic splits and shortest paths used to carry traffic flows of demand $d \in \{1, 2, \ldots, |D^t|\}$ are fixed at each stage t. Note that the profit and weight of each item in MKP are respectively equivalent to the number of *off*-cables for each switch $u \in V$ and the switch upgrade cost p_u^t. Further, the maximum budget at each stage B^t is the same as a knapsack's capacity in MKP. Moreover, GMSU-2 aims to upgrade T disjoint subsets of l-switches that minimize the total number of *on*-cables over multiple stages T, i.e., maximize the total number of *off*-cables instead of the total profit in the MKP. Thus, GMSU-2 is also as hard as MKP. The following section describes a heuristic solution for GMSU-2.

4.2. Heuristic Solutions

This section presents two heuristic solutions for GMSU-2 called GU-2a and GU-2b. Section 4.2.1 first describes GU-2a which routes each traffic demand via non-link-disjoint paths. Then, Section 4.2.2 presents GU-2b which uses GU-2a but adopts link-disjoint paths to route traffic demands. Section 4.2.3 gives an example. Section 4.2.4 analyzes the correctness of GU-2a and GU-2b as well as their time complexity.

4.2.1. Details of GU-2a

As per Algorithm 4.1, GU-2a consists of three parts: (1) initial traffic routing, (2) switch upgrade, and (3) traffic rerouting and link cost setting. Part 1 is used only in stage $t = 1$, while Part 2 is used at the beginning of each stage (in years). On the other hand, traffic rerouting and link cost setting in Part 3 can be used at the beginning of each stage and whenever a significant change occurs in network traffic within the stage, e.g., every week. The following three sections outlines the aforementioned three parts.

4.2.1.1. Part 1 – Initial Traffic Routing

Part 1 initially routes each traffic demand according to OSPF-ECMP. It generates the initial traffic routes R^0 for all demands in D^T and total unused cables w_u for each node $u \in V$. This set of information will be used in Part 2 and Part 3 to upgrade a set of l-switches and reroute traffic demand to maximize energy saving ε_T. More specifically, for each link $(u, v) \in E$, *Line 1* of Algorithm 4.1 sets the initial link cost, denoted as ψ^0_{uv}, to the link delay π_{uv}. For each demand $d \in \{1, 2, \ldots, |D^T|\}$, *Line 3* uses Yen's algorithm [88] to generate set $\mathcal{P}_d^{s_d}$, which

Algorithm 4.1. : **GU-2a – heuristic solution for GMSU-2**

Input: $G^0(V, E)$, T, B, D^0, p_u^0, $U_{\max}$, μ, ρ
Output: R^t, V^t, ε^t, ψ^t

▷ *Part 1: initial traffic routing*
1: Set $\psi_{uv}^0 = \pi_{uv}$ for each link $(u, v) \in E$
2: **for** $(d \in [1, |D^T|])$ **do**
3: Generate set $\mathcal{P}_d^{s_d}$
4: Put each path $\mathcal{P}_{d,i}^{s_d} \in \mathcal{P}_d^{s_d}$ with the shortest delay in $R_d^{s_d,0}$
5: Route flow of size $\omega_d^T / |R_d^{s_d,0}|$ via each path $R_{d,i}^{s_d,0} \in R_d^{s_d,0}$
6: **for** (each $R_{d,i}^{s_d,0} \in R_d^{s_d,0}$ and $(u, v) \in R_{d,i}^{s_d,0}$) **do**
7: $f_{uv}^T = f_{uv}^T + \omega_d^T / |R_d^{s_d,0}|$
8: **end for**
9: **end for**
10: $n_{uv}^T = \lceil f_{uv}^T / (\gamma \times U_{\max}) \rceil$ for each $(u, v) \in E$
11: Compute w_u for each $u \in V$ using (3.2)
12: **for** $(t \in \{1, 2, \ldots, T\})$ **do**
▷ *Part 2: switch upgrade*
13: $\{V^t, \Delta B^t, L\} =$ **Selection**(V^{t-1}, B^t)
14: $B^{t+1} = B^{t+1} + \Delta B^t$
▷ *Part 3: traffic rerouting and link cost setting*
15: $\{R^t, \psi^t\} =$ **MGTE**(R^{t-1}, L, V^t)
16: Compute ε^t using (2.1)
17: $\varepsilon_T = \varepsilon_T + \varepsilon^t / T$
18: **end for**

contains all paths from s_d to τ_d in order of increasing delay and within $\delta_{max,d}$. *Lines 4–8* distribute the traffic volume ω_d^T equally over all shortest paths in $R_d^{s_d,0}$ and compute the total volume f_{uv}^T over each link (u, v). *Line 10* calculates the number of *on*-cables n_{uv}^T for each link (u, v). *Line 11* concludes Part 1 by calculating the total number of unused cables incident at node u using Eq. 3.2.

4.2.1.2. Part 2 – Switch Upgrades

Part 2 greedily selects l-switches starting from the switch with the largest ratio w_u / p_u^t at the earliest possible stage. It uses function **Selection**() detailed in Algorithm 3.2, which is also used by heuristic solution GU-1 in Section 3.2.1.2. Briefly, **Selection**() in *Line 13* of Algorithm 4.1 is used to upgrade a set of l-switches, each of which satisfies Definition 3.1. It returns a set $V^t \subset V$ of *upgraded* l-switches up to stage t, the remaining budget ΔB^t, and set L that stores each

c-link (u, v) with non-zero traffic flow. *Line 14* then adds the remaining budget ΔB^t to the budget for stage $t + 1$.

4.2.1.3. Part 3 – Traffic Rerouting and Link Cost Setting

The upgraded switches V^t, generated in Part 2, are used in Part 3 to increase the number of *off*-cables on every c-link, when possible. Part 3 uses function **MGTE**() or Algorithm 4.2 in *Line 15*. The function adapts the greedy approach proposed in [12] and [14]. Specifically, **MGTE**() switches off as many c-link's cables as possible and reroutes traffic flows over these cables to other paths. It starts from the cable that has the smallest used capacity. The rationale for this greedy approach is that such a cable has the smallest amount of traffic to be rerouted, and thus more likely to be switched off. However, switching off a cable is feasible only if each traffic flow of demand d that passes through the cable can be rerouted via a set $R_d^{s_d,t}$ of *routable* paths defined as follows.

Definition 4.1. A set of paths $R_d^{s_d,t} \subseteq \mathcal{P}_d^{s_d}$ at stage t from source node s_d to destination node τ_d is *routable* if (i) each link $(u, v) \in R_d^{s_d,t}$ has sufficient capacity to carry the flow of demand d, and (ii) each l-switch $x \in R_d^{s_d,t}$ equally distributes each incoming traffic flow d over $m \geq 1$ shortest paths from x to destination τ_d.

Note that Definition 4.1 considers the largest traffic volume, i.e., the flow size ω_d^T of demand d to ensure each *routable* path can carry traffic at any stage $t \leq T$. All paths in the set $R_d^{s_d,0}$ are routable because each path is the shortest path and can carry $\omega_d^T / |R_d^{s_d,0}|$ amount of traffic. Further, the set $R_d^{s_d,t}$ can contain paths with different delays. However, the cost of all selected paths for each demand d from any l-switch in the paths must be equal. In this case, Part 3 adjusts the cost of all links to satisfy OSPF-ECMP for each l-switch.

Let $R^t = \{R_d^{s_d,t} \mid \forall d \in [1, |D^t|], \forall t \in \{1, \ldots, T\}\}$ contain all routable paths for

Algorithm 4.2. : **MGTE()**

Input: R^{t-1}, L, V^t
Output: R^t, ψ^t

1: $R^t = R^{t-1}$, $\psi^t = \psi^{t-1}$ and $\mathcal{L} = L$
2: Generate $R_d^{x,t}$ and $\tilde{P}_d^{x,t} = \{P_d^x - R_d^{x,t}\}$
3: **while** ($\mathcal{L} \neq \{\}$) **do**
4: Find $(u, v) \in \mathcal{L}$ with the smallest r_{uv}
5: Put all paths that pass (u, v) in Q_{uv}
6: $n_{uv}^T = n_{uv}^T - 1$
7: **for** (each path $R_{d,i}^{s_d,t} \in Q_{uv}$ and $r_{uv} > 0$) **do**
8: **if** (**Reroute**($R_{d,i}^{s_d,t}$) == true) **then** // or **RerouteDP**(.)
9: $r_{uv} = r_{uv} - f_{d,i}^T$
10: Update $R_d^{s_d,t}$ and $R_d^{x,t}$
11: **end if**
12: **end for**
13: **if** ($r_{uv} > 0$) **then** *success* = false
14: **else**
15: $\{\psi^t, success\}$= **LinkCost**(R^t, X)
16: **end if**
17: **if** (*success* == false) **then**
18: Revert back each changed set $R_d^{s_d,t}$ to its previous paths
19: $n_{uv}^T = n_{uv}^T + 1$
20: $\mathcal{L} = \mathcal{L} - (u, v)$
21: **else if** ($n_{uv}^T == 0$) **then**
22: $\mathcal{L} = \mathcal{L} - (u, v)$
23: $L = L - (u, v)$
24: **end if**
25: **end while**
26: Compute w_u for each $u \in V - V^t$ using Eq. (3.2)

all demands in D^t at each stage t. Further, let $R_{d,i}^{s_d,t}$ denote the i^{th} *routable path* in $R_d^{s_d,t}$. *Line 1* of **MGTE**() initializes set R^t (ψ^t) with paths (link costs) from the previous stage $t-1$ and set $\mathcal{L}$ with all c-links in set L. We use $R_{d,i}^{x,t} \subseteq R_{d,i}^{s_d,t}$ to denote a *routable subpath* from an l-switch $x \in R_{d,i}^{s_d,t}$ to node τ_d, where $x \neq \tau_d$ is the closest l-switch to source s_d. Let $R_d^{x,t}$ be a set of routable subpaths from l-switch x to destination τ_d, i.e., $R_d^{x,t} = \{R_{d,i}^{x,t} \subseteq R_{d,i}^{s_d,t} \mid x \in R_{d,i}^{s_d,t}, i \in [1, |R_d^{s_d,t}|]\}$. We have $|R_d^{x,t}| \leq |R_d^{s_d,t}|$ because two routable paths for a demand d, e.g., $R_{d,i}^{s_d,t}$ and $R_{d,j}^{s_d,t}$, can have the same subpath, i.e., $R_{d,i}^{x,t} = R_{d,j}^{x,t}$. For example, Figure 4.2 shows six paths from source $s_d = 1$ to destination $\tau_d = 11$. Assume only the following five paths are routable: $(1, 2, 5, 8, 11)$, $(1, 2, 5, 9, 11)$, $(1, 3, 7, 9, 11)$, $(1, 3, 6, 9, 11)$,

and $(1, 4, 6, 9, 11)$. Nodes 5, 3, 4 are the closest l-switches to node 1. Nodes 5 and 3 have two routable subpaths, i.e., $R_d^{3,t} = \{(3, 7, 9, 11), (3, 6, 9, 11)\}$ and $R_d^{5,t} = \{(5, 8, 11), (5, 9, 11)\}$, while node 4 has only one subpath, i.e., $R_d^{4,t} = \{(4, 6, 9, 11)\}$. *Line 2* enumerates each set $R_d^{x,t}$ for every set $R_d^{s_d,t} \in R^t$. Let $\tilde{P}_d^{s_d,t} = \{\mathcal{P}_d^{s_d} - R_d^{s_d,t}\}$ denote a set of paths in $P_d^{s_d}$ that are *not selected* at stage t to route demand d. For each set $\tilde{P}_d^{s_d,t}$, *Line 2* also enumerates a set of subpaths in $\mathcal{P}_d^x$ that are *not selected* at stage t, denoted by $\tilde{P}_d^{x,t} = \{\mathcal{P}_d^x - R_d^{x,t}\}$. For example, traffic from node 1 to 11 in Figure 4.2 has one path $\tilde{P}_d^{1,t} = \{(1, 4, 6, 10, 11)\}$ that is not used to route the traffic. Thus, we have one non-selected subpath, $\tilde{P}_d^{4,t} = \{(4, 6, 10, 11)\}$.

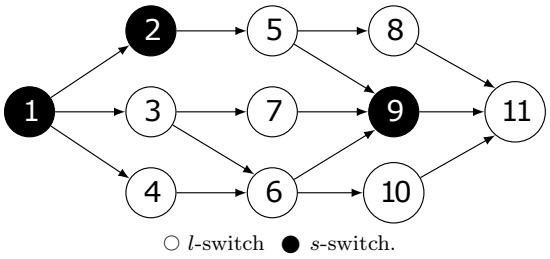


Figure 4.2. An example to generate each set of routable subpaths and a set of non-selected subpaths from source $s_d = 1$ to destination $\tau_d = 11$. Assume path $(1, 4, 6, 10, 11)$ is not selected to route traffic from nodes 1 to 11. Nodes 3, 4 and 5 are the closest l-switches to node 1. There are three sets of routable subpaths $R_d^{3,t} = (3, 7, 9, 11), (3, 6, 9, 11)\}$, $R_d^{4,t} = \{(4, 6, 9, 11)\}$, and $R_d^{5,t} = \{(5, 8, 11), (5, 9, 11)\}$, and one set of non-selected paths $\tilde{P}_d^{4,t} = \{(4, 6, 10, 11)\}$.

Lines 4–6 select a c-link $(u, v) \in \mathcal{L}$ that contains a cable with the least used capacity $r_{uv} = (f_{uv}^T - \gamma \times U_{\max} \times \lfloor f_{uv}^T / \gamma \times U_{\max} \rfloor)$, record each path in every set $R_d^{s_d,t} \in R^t$ that passes link (u, v) in a set Q_{uv}, and turn off one cable in link (u, v). *Line 8* uses the function **Reroute**() to reroute traffic carried by path $R_{d,i}^{s_d,t} \in Q_{uv}$ via $m \geq 1$ alternative paths in set $\mathcal{P}_d^{s_d}$. Recall that criterion (ii) of Definition 4.1 requires each subpath $R_{d,i}^{x,t} \in R_d^{x,t}$ to carry the same traffic size. To satisfy criterion (ii), each of the m paths must carry an additional $f_{d,i}^T/m$ amount of traffic. **Reroute**() considers the following two possible cases in order to find m

paths: 1) the set $R_d^{s_d,t}$ contains only $R_{d,i}^{s_d,t}$ and 2) the set $R_d^{s_d,t}$ contains $R_{d,i}^{s_d,t}$ and $m \geq 1$ paths. For case 1), **Reroute**() finds $m \geq 1$ paths in the set of non-selected paths $\tilde{P}_d^{s_d,t}$; each of which can carry an additional traffic volume of $f_{d,i}^T/m$. For case 2), the function carries out the following three steps:

(a) Use all m paths if each of the m paths is able to carry an additional traffic of size $f_{d,i}^T/m$.

(b) If step (a) fails and s_d is an s-switch, find one of the m paths which has no common node with any of the other $m-1$ paths, i.e., a node-disjoint path that can carry traffic volume $f_{d,i}^T$. If such a path does not exist, find $k \geq 1$ path(s) in the set $\tilde{P}_d^{s_d,t}$. Here, each path must be able to carry an additional traffic of size $f_{d,i}^T/k$.

(c) If step (a) fails and s_d is an l-switch, find one path in the set $\tilde{P}_d^{s_d,t}$ to carry an additional traffic volume of $f_{d,i}^T$.

Step (b) uses a node-disjoint path to ensure that its subpath from any l-switch x to destination τ_d is the only path that carries the additional traffic of size $f_{d,i}^T$. In step (c), path $R_{d,i}^{s_d,t}$ must be rerouted to only one non-selected path $R_{d,j}^{s_d,t} \in \tilde{P}_d^{s_d,t}$ to ensure that each path in $R_d^{s_d,t} - R_{d,i}^{s_d,t}$ and path $R_{d,j}^{s_d,t}$ carry the same volume of traffic demand d. The function **Reroute**() returns false when one of the two cases fails to find m paths.

If *Line 8* is able to reroute path $R_{d,i}^{s_d,t}$, i.e., **Reroute**() returns true, *Line 9* reduces the used capacity r_{uv} by $f_{d,i}^T$. Further, *Line 10* removes path $R_{d,i}^{s_d,t}$ and subpath $R_{d,i}^{x,t}$. It then includes the found m paths and each subpath $R_{d,j}^{x,t}$ of these m paths into the set $R_d^{s_d,t}$ and $R_d^{x,t}$, respectively. Note that updating $R_d^{x,t}$ includes adding or removing subpaths in $\tilde{P}_d^{x,t}$. If *Lines 8–11* fail to reroute all paths in Q_{uv}, i.e., $r_{uv} > 0$, *Line 13* sets *success* to false. Otherwise, *Line 15* calls the function **LinkCost**().

LinkCost() solves the Linear Program (LP) in (4.19) to adjust the link costs in ψ^t such that all subpaths in $R_d^{x,t}$ become the only shortest subpaths from node x to τ_d. It is based on the LP in [92], which aims to minimize the difference in path cost or *excess cost* for every pair of shortest subpaths in each set $R_d^{x,t}$. In this way, the total number of the shortest subpaths in every $R_d^{x,t}$ can be maximized. Briefly, the approach in [92] allocated cost $\psi_{uv}^t > 0$ to each link $(u, v) \in E$ by considering two constraints: (i) all routable subpaths in $R_d^{x,t}$ must have the same minimum cost, i.e., $\Psi_{d,i}^{x,t} = \Psi_{d,j}^{x,t}$ for all $R_{d,i}^{x,t}, R_{d,j}^{x,t} \in R_d^{x,t}$, and (ii) the cost of each routable subpath $R_{d,i}^{x,t} \in R_d^{x,t}$ is less than the cost of each non-selected subpath $\tilde{P}_{d,j}^{x,t} \in \tilde{P}_d^{x,t}$, i.e., $\Psi_{d,i}^{x,t} < \Psi_{d,j}^{x,t}$. The LP in [92] used variable $e_{d,i}^{x,t}$ to denote the *excess cost* for each routable subpath $R_{d,i}^{x,t} \in R_d^{x,t}$ to approximate the optimal link cost. Here the equality in constraint (i) becomes $\Psi_{d,i}^{x,t} - e_{d,i}^{x,t} = \Psi_{d,j}^{x,t} - e_{d,j}^{x,t}$, whilst the inequality in constraint (ii) is converted to $\Psi_{d,i}^{x,t} - e_{d,i}^{x,t} \leq \Psi_{d,j}^{x,t}$.

The LP in [92], however, cannot be applied directly to our problem for two reasons. First, constraint (ii) may have $\Psi_{d,i}^{x,t} = \Psi_{d,j}^{x,t}$ and produce $e_{d,i}^{x,t} = 0$, which makes a non-selected path $\tilde{P}_{d,j}^{x,t}$ become a shortest subpath. In contrast, our link cost setting ensures that any non-selected subpath in $\tilde{P}_d^{x,t}$ cannot be a shortest subpath. Thus, we modify constraint (ii) in [92] to $\Psi_{d,j}^{x,t} - (\Psi_{d,i}^{x,t} - e_{d,i}^{x,t}) \geq 1$ such that we have $e_{d,i}^{x,t} > 0$ when the link cost setting produces $\Psi_{d,i}^{x,t} = \Psi_{d,j}^{x,t}$. Second, the LP in [92] does not consider the maximum delay constraint for each shortest subpath. On the other hand, our proposed LP (4.19) requires each shortest subpath to have delay within a given maximum delay.

To this end, **LinkCost**() is formally defined as:

$$\min \quad \sum_{d=1}^{|D^t|} \sum_{x \in X} \sum_{R_{d,i}^{x,t} \in R_d^{x,t}} e_{d,i}^{x,t} \tag{4.19a}$$

s.t.

$$\sum_{(u,v)\in R^{x,t}_{d,i}} \psi^t_{uv} - e^{x,t}_{d,i} = \sum_{(u,v)\in R^{x,t}_{d,i+1}} \psi^t_{uv} - e^{x,t}_{d,i+1}, \tag{4.19b}$$

$$\sum_{(u,v)\in \tilde{P}^{x}_{d,j}} \psi^t_{uv} - \sum_{(u,v)\in R^{x,t}_{d,1}} \psi^t_{uv} - e^{x,t}_{d,1} \geq 1, \tag{4.19c}$$

$$\psi^0_{uv} \leq \psi^t_{uv} \leq \mathrm{I}, \tag{4.19d}$$

$$\sum_{(u,v)\in R^{x,t}_{d,1}} \psi^t_{uv} \leq \Psi^{x,0}_{\max,d}, \tag{4.19e}$$

$$e^{x,t}_{d,i} \geq 0. \tag{4.19f}$$

Objective (4.19a) minimizes the total excess cost to maximize the number of paths $R^{x,t}_{d,i} \in R^{x,t}_d$ that have an excess cost $e^{x,t}_{d,i}$ of zero. For each set $R^{x,t}_d$, constraint (4.19b) requires each consecutive pair of subpaths, i.e., $R^{x,t}_{d,i}, R^{x,t}_{d,i+1} \in R^{x,t}_d$, to have the same minimum cost. As (4.19b) enforces all subpaths in $R^{x,t}_d$ to have the same cost, (4.19c) only needs one subpath in the set $R^{x,t}_d$, i.e., $R^{x,t}_{d,1}$, to ensure each non-selected subpath $\tilde{P}^{x,t}_{d,j}$ in $\tilde{P}^{x,t}_d$ has a larger cost than every subpath in $R^{x,t}_d$. Constraint (4.19d) ensures that the cost ψ^t_{uv} of each link $(u,v) \in E$ within $[\psi^0_{uv}, \mathrm{I} = 2^{16-1}]$. Let $\Psi^{x,0}_{\max,d}$ be the maximum cost among all subpaths in P^x_d at stage zero. Recall that in Part 1, the cost of each link (u,v) is initialized as the link's delay π_{uv}. Thus, we have $\delta^x_{\max,d} \leq \Psi^{x,0}_{\max,d}$, and each subpath in P^x_d with cost no higher than $\Psi^{x,0}_{\max,d}$ has delay no longer than delay constraint $\delta^x_{\max,d}$. Constraint (4.19e) uses $R^{x,t}_{d,1} \in R^{x,t}_d$ to ensure that the cost of each subpath in $R^{x,t}_d$ is not higher than the maximum cost $\Psi^{x,0}_{\max,d}$. Thus, each subpath in $R^{x,t}_d$ satisfies the maximum delay constraint $\delta^x_{\max,d}$. Both (4.19d) and (4.19e) guarantee that any non-selected subpath $P^x_{d,k} \notin P^x_d$ does not have the minimum cost. The last constraint (4.19f) requires each $e^{x,t}_{d,i}$ to be positive.

If the total excess cost, i.e., (4.19a), is not zero, **LinkCost**() sets *success* to false and returns ψ^t without updating link costs. Further, *Lines 18–20* of

MGTE() revert the routable paths R^t to their previous paths, set the cable(s) in link (u, v) back to *on*, and remove link (u, v) from the set $\mathcal{L}$. If **LinkCost**() successfully updates set ψ^t with new link costs, it sets *success* to true. If link (u, v) has no *on*-cable, i.e., $n_{uv}^T = 0$, *Line 22* and *Line 23* remove the link from sets $\mathcal{L}$ and L, respectively. This allows all cables in the c-link (u, v) to remain *off* in subsequent stages. *Line 26* then updates the unused cables w_u of each l-switch $u \in V - V^t$ because the new routing produced by *Lines 3–25* is able to increase the number of *off*-cables. Finally, *Line 16* and *Line 17* of GU-2a computes ε^t and ε_T, respectively. Overall, Part 3 produces a set R^t that contains all routable paths for all demands in D^t at each stage $t \in \{1, \ldots, T\}$ and a set of link costs ψ^t.

4.2.2. Details of GU-2b

This section presents our approach to enable link-disjoint path routing in GU-2a; we call this approach GU-2b. In GU-2b, we replace the function **Reroute**() in *Line 8* of function **MGTE**() with the function **RerouteDP**(). As in **Reroute**(), the function **RerouteDP**() aims to reroute traffic carried by path $R_{d,i}^{s_d,t} \in Q_{uv}$ via $m \geq 1$ alternative paths in set $\mathcal{P}_d^{s_d}$. For each demand d, the function considers two possible cases: 1) the demand is initially routed via non link-disjoint paths, or 2) the demand is initially routed via link-disjoint paths. For case 1), function **RerouteDP**() uses the function **Reroute**() to reroute demand d using not necessarily link-disjoint paths. For case 2), the function **RerouteDP**() aims to reroute demand d via at least two link-disjoint paths to route demand d. The function carries out the following three steps:

(a) If set $R_d^{s_d,t}$ contains $R_{d,i}^{s_d,t}$ and other $m \geq 2$ paths, where each path can carry an additional traffic of size $f_{d,i}^t/m$, use all of the m paths.

(b) If step (a) fails and s_d is an s-switch, find a node-disjoint path among the $m \geq 2$ paths that can carry an additional traffic of size $f_{d,i}^t/m$. If such path does not exist, find $k \geq 1$ link-disjoint paths in the set $\tilde{P}_d^{s_d,t}$. Here, each path must be able to carry an additional traffic of size $f_{d,i}^T/k$ and are link-disjoint with the m paths.

(c) If step (a) fails and s_d is an l-switch, find one path in the set $\tilde{P}_d^{s_d,t}$ that can carry an additional traffic of size $f_{d,i}^T$ and is link-disjoint with the m paths.

The function **RerouteDP**() returns false when it fails to find the m paths from either of the two cases.

4.2.3. An Example

This example illustrates how to use the three parts of GU-2a and GU-2b to upgrade the legacy network $G^0(V, E)$ in Figure 4.1a, where each link has a delay of one second. The plan is to upgrade the network in $T = 2$ stages using a total budget $B = \$45$, i.e., $B^1 = B^2 = \$22.5$. Each switch u has an initial upgrade cost of $p_u^1 = p_u^0 = \$15$, which is reduced to $p_u^2 = \$12$ at the second stage. The first demand $d = 1$ which is $(s_1 = 1, \tau_1 = 3, \omega_1^0 = 5)$, and the second demand $d = 2$, i.e., $(s_2 = 2, \tau_2 = 4, \omega_2^0 = 5)$, have their traffic size increases by $\mu_d = 0.2$ per stage, i.e., from $\omega_1^1 = \omega_2^1 = 5$ to $\omega_1^2 = \omega_2^2 = 6$. This example considers a delay tolerance $\sigma = 1.1$, e.g., demand $d = 1$ that is routed via path $(1, 2, 3)$ has $\delta_{\min,1}^1 = 2$ and maximum path delay of $\delta_{\max,1} = \lceil 1.1 \times 2 \rceil = 3$ seconds.

In Part 1, GU-2a equally distributes demands $d = 1$ and $d = 2$ via paths with shortest delays in set $\mathcal{P}_1^1$ and $\mathcal{P}_2^2$, respectively. For example, initially demand $d = 1$ has routable paths $R_1^{1,0} = \mathcal{P}_1^1 = \{(1, 2, 3), (1, 5, 3), (1, 4, 3)\}$. Traffic volume $\omega_1^2 = 6$ is then split equally into $\omega_1^2/|R_1^{1,0}| = 2$ units each. Figure 4.1a shows the traffic distribution for both demands. From the distribution, we get the total

traffic volume on each link, e.g., $f^2_{(2,3)} = 2 + 2 = 4$, and the required number of *on*-cables on each link, e.g., $n^2_{(2,3)} = \lceil f^2_{(2,3)}/\gamma \times U_{\max} \rceil = \lceil 2/5 \times 0.8 \rceil = 1$ *on*-cable; thus there are $(b_{(2,3)} - n^2_{(2,3)}) = (2 - 1) = 1$ unused cables for link $(2, 3)$. Thus, there are $w_1 = 8$, $w_2 = w_3 = w_4 = 8$ and $w_5 = 12$ *off*-cables for the respective l-switches $V = \{1, 2, 3, 4, 5\}$.

In Part 2 and stage $t = 1$, **Selection**() upgrades only l-switch $u = 5$ that has the highest ratio $w_5/p^1_5 = 12/15 = 0.8$. Thus, the function returns $V^1 = \{5\}$, remaining budget $\Delta B^1 = \$7.5$, values of $w_1 = 8 - 3 = 6$ and $w_2 = w_4 = w_3 = 6$, and four c-links with traffic flows, i.e., $L = \{(1, 5), (5, 3), (2, 5), (5, 4)\}$. Function **MGTE**() initializes $R^1 = \{R^{1,0}_1 = \{(1, 2, 3),\ (1, 5, 3),\ (1, 4, 3)\},\ R^{2,0}_2 = \{(2, 1, 4),\ (2, 5, 4),\ (2, 3, 4)\}\}$, $\mathcal{L} = L$, $\psi^1_{uv} = \psi^0_{uv} = 1$ for each link $(u, v) \in E$. The function enumerates two sets of routable subpaths $R^{1,1}_1$ and $R^{2,1}_2$ which are the same as their respective routable paths in R^1 because the source of both demands are legacy. Thus, we have $\tilde{P}^{1,1}_1 = \tilde{P}^{2,1}_2 = \{\}$. *Line 6* of **MGTE**() only turns off one cable in c-link $(1, 5)$. Both **Reroute**() or **RerouteDP**(), in *Line 8*, use their second case with step (a) to reroute path $(1, 5, 3) \in Q_{(1,5)}$ to paths $\{(1, 2, 3), (1, 4, 3)\}$. Each of the two paths is able to carry an additional traffic of size $2/2 = 1$ unit; see Figure 4.1b. **LinkCost**() solves the LP in (4.19) and returns a zero excess cost for each selected path in sets $R^{1,1}_1$ and $R^{2,1}_2$. It increases the cost of link $(1, 5)$ by one such that the non-selected path $(1, 5, 3)$ has cost $\psi^1_{(1,5)} + \psi^1_{(5,3)} = 2 + 1 = 3$, which is higher than the selected paths $R^{1,1}_1 = \{(1, 2, 3), (1, 2, 3)\}$, each with a cost of two; see Figure 4.1b. By using f^1_{uv} for each link, e.g., $f^1_{(2,5)} = f^2_{(2,5)}/(1 + 0.2) = 2/1.2 = 1.67$, there are 14 unused cables: one cable each on link $(2, 5)$ and $(5, 4)$ and two cables each on link $(1, 5)$, $(5, 1)$, $(5, 2)$, $(5, 3)$, $(3, 5)$ and $(4, 5)$. Thus, we can save $\varepsilon^1 = 14/32 = 43.75\%$ of energy.

In stage $t = 2$, with budget $B^2 = B^2 + \Delta B^1 = 22.5 + 7.5 = \$30$, **Selection**() returns $V^2 = \{5, 2, 4\}$, $\Delta B^2 = 6$, and $L = \{(2, 5),\ (5, 4),\ (1, 2),\ (2, 1),\ (1, 4),$

$(2,3)$, $(3,4)$, $(4,3)\}$. Further, **MGTE**() reroutes only path $(2,3,4)$, which passes c-link $(3,4)$ to path $(2,5,4)$ to turn off the only cable in the link. As shown in Figure 4.1b, two routable paths carry different traffic volume of demand $d = 2$. This is allowable as the source of the path, i.e., $u = 2$, is now an s-switch. For this last stage, another 11 unused cables can be off, i.e., one cable each on links $(1,2)$, $(2,1)$, $(2,3)$, $(1,4)$, and $(4,3)$, and two cables each on links $(3,2)$, $(3,4)$ and $(4,1)$. Thus, GU-2a obtains energy saving $\varepsilon^2 = (11+14)/32 = 78.12\%$ and average energy saving over $T = 2$ of $\varepsilon_2 = (43.75 + 78.12)/2 = 60.94\%$.

4.2.4. Algorithm Analysis

The following two propositions analyze GU-2a and GU-2b in terms of the algorithms' compliance to all constraints in MIP-2a (4.1) and MIP-2b (4.14) and time complexity, respectively.

Proposition 4.1. *Given a legacy network $G^0(V,E)$, at each stage $t \in \{1,\ldots,T\}$, GU-2a produces (a) a set of s-switches V^t with the total upgrade cost within the maximum available budget B^t, and (b) a routing set R^t and a set of link costs ψ^t that satisfy the following constraints: (i) the maximum link utilization $U_{max} \times c_{uv}$, (ii) the maximum path delay $\delta_{max,d}$ for each demand $d \in [1, D^t]$, and (iii) OSPF-ECMP for each l-switch $x \notin V^t$.*

Proof. For result (a), at each stage $t \in \{1,\ldots,T\}$, Part 2 via function **Selection**() ensures that a switch can be upgraded only if its cost is no more than the remaining budget; see *Line 1* of function **Selection**(). For result $b(i)$, GU-2a and GU-2b respectively use function **Reroute**() and **RerouteDP**() in *Line 8* of function **MGTE**() to find $m \geq 1$ paths, each of which can carry an equal extra traffic volume of size $f^t_{d,i}/m$. This indicates that the total traffic volume f^t_{uv} for each link $(u,v) \in E$ that belongs to every of m paths is within the maximum link

utilization, i.e., $f_{uv}^t \leq U_{\max} \times c_{uv}$. Further, to address requirement $b(ii)$, each of the m paths is found from either $R_d^{s_d,t} \in \mathcal{P}_d^{s_d}$ or $\tilde{P}_d^{s_d,t} \in \mathcal{P}_d^{s_d}$ that has delay no longer than $\delta_{\max,d}$. This proves that *Line 8* of function **MGTE**() produces a set of paths R^t that satisfy the maximum delay constraint. Function **MGTE**() uses either function **Reroute**() or **RerouteDP**() and **LinkCost**() to satisfy $b(iii)$. Functions **Reroute** and **RerouteDP**() always *equally* distribute extra traffic volume $f_{d,i}^t$ to $m \geq 1$ paths. Therefore, each subpath from l-switch x to destination τ_d, i.e., $R_{d,i}^{x,t}$ that resides in any of the m paths, carries the same size of traffic demand d. Afterwards, function **LinkCost**() in *Line 15* of function **MGTE**() ensures each subpath $R_{d,i}^{x,t}$ is a shortest path. It solves LP (4.19) and *only* updates the link costs in ψ^t if each subpath $R_{d,i}^{x,t}$ in every set $R_d^{x,t}$ has a minimum cost. If all calls to functions **Reroute**() (or **RerouteDP**()) and **LinkCost**() return false at every stage $t \in \{1, \ldots, T\}$, GU-2a uses the initial link costs $\psi^t = \psi^0$ and routing $R^t = R^0$. Recall that set R^0 contains all shortest paths within delay $\delta_{\max,d}$. Each path in every $R_d^{s_d,t} \in R^0$ carries an equal traffic size of demand d; see *Line 4* and *Line 5* of GU-2a. Consequently, each path from l-switch x to τ_d carries an equal size of traffic demand d. □

Proposition 4.2. *The time complexity of GU-2a and GU-2b is* $O(|V|^2|E|^2 + \alpha|E|)$.

Proof. Let us first compute the time complexity of function **MGTE()** before analyzing the complexity for GU-2a and GU-2b. Let K be the maximum among the number of paths $|\mathcal{P}_d^{s_d}|$ for each demand d. The time complexity of **MGTE**() is computed as follows. Note that $|D| = |D^t|$ for every $t \in \{1, \ldots, T\}$. *Line 1* takes $O(K|D|+2|E|)$. *Line 2* of Algorithm 4.2 has the worst case of time complexity of $O(K|D||V|)$. *Lines 3–25* are repeated $O(|E|)$ times because, in the worst case, the number of c-links in $\mathcal{L}$ is the as same the number of links in E. For each repetition,

Line 4 and *Line 5* respectively need $O(|E|)$ and $O(|D||E|)$, while *Line 6* takes a constant time. *Line 8* uses either function **Reroute**() or function **RerouteDP**(). Function **Reroute**() falls in either case 1) that takes $O(K|E|)$, or case 2) that consists of three steps. Specifically, Steps (a) and (c) require $O(K|E|)$, while step (b) takes $O(K^2|E|)$. Note that each step must check the residual capacity of each link in each of the K paths. Thus, the worst case of time complexity of **Reroute**() is $O(K^2|E|)$. Function **Reroute**() is used by **RerouteDP**() for its first case. On the other hand, the second case of **RerouteDP**() executes steps (a), (b), and (c). The worst case is in step (b) that also takes $O(K^2|E|)$. Thus, **RerouteDP**() also requires $O(K^2|E|)$. *Line 9* takes a constant time, while *Line 10* needs up to $O(K)$. *Lines 7–12* are repeated in the worst case $O(|D|)$ times, and thus they take $O(K^2|D||E|)$ time. Function **LinkCost**(), called in *Line 15*, solves LP (4.19) in $O(\alpha)$, where α is the worst case run-time to solve the LP. *Line 18* can revert up to $K|D|$ paths and update traffic volume on each link. Thus, its complexity is $O(K|D||E|)$. *Lines 19–20* and *Lines 22–23* take $O(1)$, while *Line 26* takes $O(|E|)$. Thus, the time complexity of **MGTE**() is $O(|D| + |E| + K|D||V| + |E|(|E| + |D||E| + K^2|D||E| + \alpha + K|D||E|) + |E|) = O(|E|(K^2|D||E| + \alpha))$.

We are now ready to show the time complexity of GU-2a. *Line 1* of Algorithm 4.1 needs $O(E)$. Yen's algorithm [88], used in *Line 3*, takes $O(K|D||V|(|E|+ |V|log|V|))$ to generate up to K alternative paths for each demand in D. *Line 4* needs $O(K|D|)$ in worst case and *Lines 5–8* take $O(K|D||E|)$. Note that traffic volume ω_d^T for each demand $d \in D$ can be computed in $O(1)$. *Lines 10–11* require $O(|E|)$. Thus, Part 1 takes in total $O(K|D||V|(|E| + |V|log|V|) + \alpha + K|D| + K|D||E| + K|D||E| + |E|) = O(K|D||V|(|E| + |V|log|V|))$. As discussed in Section 3.2.1.2, function **Selection**() of Algorithm 3.2 in *Line 13* takes $O(|V||E|)$. *Line 14* takes $O(1)$. Function **MGTE**() in *Line 15* takes

$O(|E|(K^2|D||E|+\alpha))$. Line 16 and Line 17 respectively require $O(E)$ and $O(1)$. As *Lines 12–18* are repeated T times, *Lines 13–17* have a time complexity of $O(T(|V||E|+1+|E|(K^2|D||E|+\alpha)+E+1)) = O(T|E|(K^2|D||E|+\alpha))$. Thus, the time complexity of GU-2a is $O(K|D||V|(|E|+|V|log|V|)+T|E|(K^2|D||E|+\alpha)) = O(T|E|(K^2|D||E|+\alpha))$. Since in general we have $|E| \leq |V|^2$, $|D| \leq |V|^2$, and $T = 5$ and $K \leq 20$ are constants, the time complexity of GU-2a is $O(|V|^2|E|^2+\alpha|E|)$. The time complexity of GU-2b is the same as GU-2a because their only difference is on the use of respectively **Reroute**() and **RerouteDP**(), which have the same time complexity. □

4.3. Performance Evaluation

We have implemented GU-2a and GU-2b in C++ and Gurobi [83] to solve MIP-2a and MIP-2b. All experiments are conducted on the platform described in Section 2.7.3. Further, each experiment uses five actual network topologies outlined in 2.3 and a given value for all parameters listed in Table 2.4. Their details are provided in Section 2.7. In addition, for each demand d, Yen's algorithm [88] is used to generate up to 20 paths in increasing order of delays. Here, each path has a delay no longer than $\delta_{\max,d}$.

This section is organized as follows. First, Section 4.3.1 evaluates the scalability of MIP-2a, MIP-2b, GU-2a, and GU-2b in terms of their running time in CPU seconds. Then, Sections 4.3.2 and 4.3.3 analyze the effect of increasing budgets and stages on energy savings, respectively. Next, Section 4.3.4 and Section 4.3.5 study the effect of using single path routing and link-disjoint multi-path routing, respectively, on energy saving. Further, Section 4.3.6 reports the energy saving performance of MIP-2a and GU-2a against prior techniques in [1] and [2]. Finally, Section 4.3.7 provides additional findings.

4.3.1. Running Time

We set the budget to B = \$1.2M and consider $T = 3$ stages to compare the run-time performance (in CPU seconds) of MIP-2a, MIP-2b, GU-2a, and GU-2b. From Table 4.2, we see that the run time of all solutions increases with network size and traffic demands. The table shows that the run time of GU-2a is far less than that of MIP, e.g., 1.57 versus 71942.81 seconds for GÉANT. Similarly, GU-2b runs significantly faster than MIP-2b, e.g., 1.24 versus 95599.2 seconds for the network, i.e., GÉANT. Further, MIP and MIP-2b failed to produce results for DFN, Deltacom and TATA because the optimizer ran out of memory. Thus, for the remaining simulations, we compare the performance of GU-2a against MIP-2a and MIP-2b versus GU-2b using only Abilene and GÉANT.

Table 4.2. Running time.

Name	$\|V\|$	$\|E\|$	$\|D\|$	**Running Time (in CPU seconds)**			
				GU-2a	**MIP-2a**	**GU-2b**	**MIP-2b**
Abilene	12	30	132	0.13	3.08	0.22	2.24
GÉANT	23	74	466	1.57	71942.81	1.24	95599.2
DFN	58	174	3306	24.72	N/A	20.47	N/A
Deltacom	113	322	12656	313.42	N/A	277.27	N/A
TATA	145	372	20880	498.92	N/A	434.65	N/A

4.3.2. Effect of Increasing Budget

Here, we consider B = {\$200K, \$400K, \$600K, \$800K, \$1M, \$1.2M} and $T = 3$. Referring to Figure 4.3, GU-2a and MIP-2a have a higher energy saving ε_T when the budget is large. For Abilene with budget B = \$200K, GU-2a and MIP-2a produce $\varepsilon_T = 35.67\%$ and $\varepsilon_T = 38.01\%$, respectively. Increasing the budget to B = \$1.2M, GU-2a and MIP-2a achieve a higher saving of $\varepsilon_T = 71.64\%$ and

ε_T = 71.93%, respectively. For GÉANT, MIP-2a fails to compute the energy saving ε_T for $B = \{\$200K, \$400K\}$ after running for one week. Running GU-2a on Abilene and GÉANT results in energy saving that is on average only 1.32% and 3.57%, respectively, off from the optimal ε_T value obtained from solving MIP-2a. GU-2a produces the energy saving ε_T of only up to 32.22% and 23.97% for Deltacom and TATA, respectively. The reason is because Deltacom and TATA have a larger number of l-switches to upgrade than the other three networks. It means an allocated budget can only upgrade a significantly smaller percentage of l-switches. As energy saving ε_T is the result of turning off the unused cables in c-links, more s-switches can potentially lead to more switched off cables.

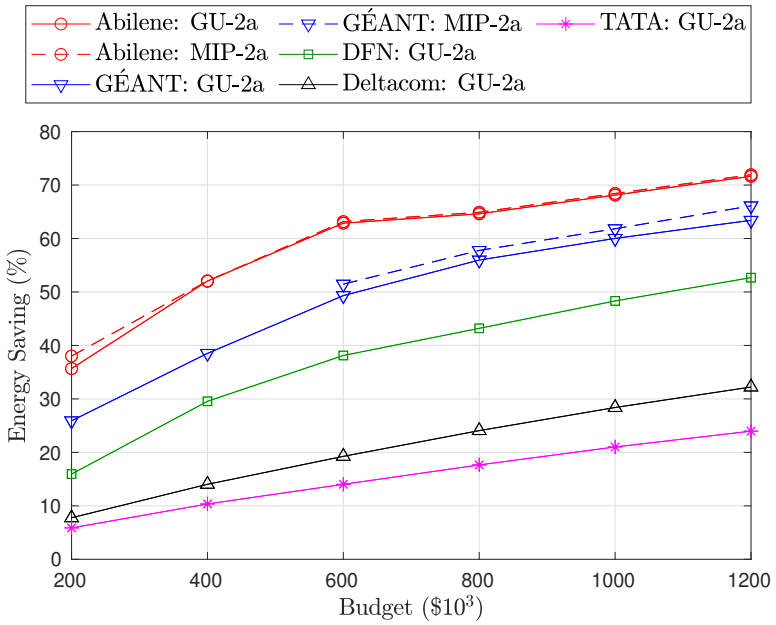


Figure 4.3. Energy saving ε_T of GU-2a and MIP-2a for various budget B.

Note that in the last upgrade stage T, both MIP-2a and GU-2a route the majority of traffic demands via single paths. For example, when the budget B is \$1.2M and $T = 3$ stages, MIP-2a routes only 1.26% and 6.44% of traffic demands via multi-paths for Abilene and GÉANT, respectively. Similarly, GU-2a routes

37.77%, 17.24% and 3.19% of traffic demands via multi-paths for GÉANT, DFN, and Deltacom, respectively. For Abilene, GU-2a routes each of its traffic demands via a single path. Similarly, GU-2a routes only two demands of TATA via multi-paths. Note that there are 18.18%, 76.61%, 61.83%, 67.35% and 73.9% of traffic demands with multi-paths within delay tolerance for Abilene, GÉANT, DFN, Deltacom and TATA, respectively.

4.3.3. Effect of Increasing Number of Stages

Next, we investigate how the number of stages, namely $T = \{1, 2, 3, 4, 5\}$ impact energy saving ε_T. The budget B is \$1.2M. As shown in Figure 4.4, the energy saving ε_T for Abilene, GÉANT, and DFN decreases as T increases. For example, the energy saving ε_T for GU-2a when it runs over Abilene and GÉANT decreases from 74.56% to 66.67% and 75% to 61.15%, respectively, when T is increased from one to five. Notice that for Abilene and GÉANT, GU-2a produces the energy saving ε_T that is on average only 0.94% and 4%, respectively, off from the optimal energy saving, which is produced by MIP-2a. In contrast, the energy saving ε_T for Deltacom and TATA increases from 34.47% to 37.61% and 25.4% to 29.52%, respectively, when T is increased from one to five. For these two larger networks, there are more switches to upgrade in later stages which results in larger energy saving ε_T. In contrast, for smaller networks such as Abilene, a budget of $B =$ \$1.2M can be used to upgrade a larger percentage of switches in earlier stages. As a result, it reduces the number of switches to be upgraded in later stages, and thus fewer unused cables can be turned off. In addition, as the later stages have a higher traffic volume, it is unlikely that these remaining switches have idle or off cables. In other words, upgrading these switches does not significantly increase the energy saving ε_T.

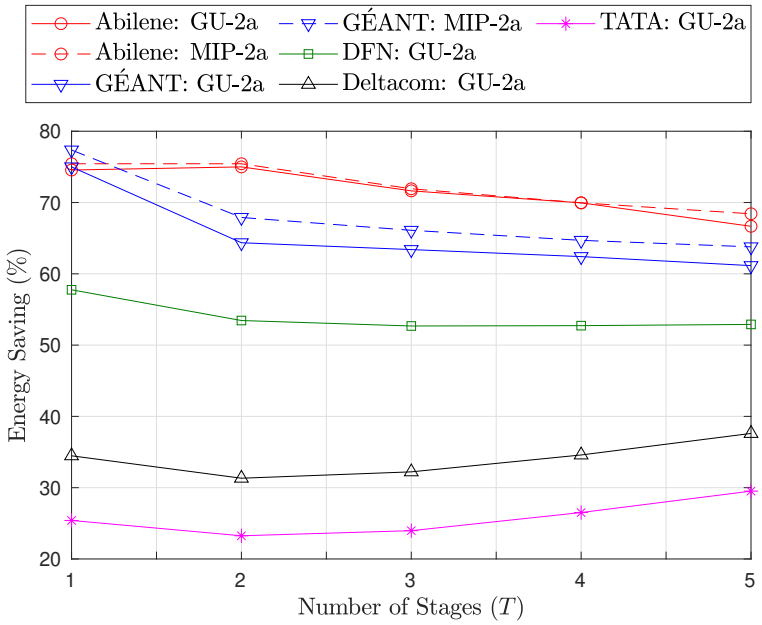


Figure 4.4. Energy saving ε_T of GU-2a and MIP-2a for various T stages.

4.3.4. Multi-Path Versus Single Path Routing

In this section, we aim to compare the energy saving ε_T calculated by MIP-2a and GU-2a against that computed by ILP-1b and GU-1b, respectively. Recall that ILP-1b and GU-1b use a single path that satisfies the same delay tolerance as MIP-2a and GU-2a to route each traffic demand. ILP-1b is the optimal approach that provides the optimal energy saving ε_T, while GU-1b is the heuristic approach that produces a sub-optimal ε_T value. Further, similar to MIP-2a and GU-2a, ILP-1b and GU-1b perform rerouting at each upgrade stage. Here, we consider budget $B = \{\$200K, \$400K, \$600K, \$800K, \$1M, \$1.2M\}$ and $T = 3$ upgrade stages.

As shown in Figure 4.5, the energy saving of MIP-2a is very close to that of ILP-1b for each budget. Similarly, Figure 4.6 shows that GU-2a and GU-1b result in similar energy saving. For Abilene, MIP-2a and ILP-1b produce the same saving. On average, for GÉANT, MIP-2a produces 0.91% lower energy saving ε_T

than ILP-1b. Similarly, GU-2a produces 0.29% and 1.62% less energy saving than GU-1b for Abilene and GÉANT, respectively. Further, the energy saving ε_T of GU-2a is 1.77%, 0.72%, and 0.06% lower than that of GU-1b for DFN, Deltacom and TATA, respectively. GU-1b is more likely to have successful traffic rerouting because GU-2a requires each l-switch x to distribute traffic over $k \geq 1$ shortest paths from x to the flow's destination. Note that traffic rerouting in GU-1b is subjected only to path delay tolerance and MLU threshold. In addition, ILP-1b and GU-1b are computationally faster than MIP-2a and GU-2a, respectively. For example, ILP-1b respectively runs in 0.06 and 30.31 seconds for Abilene and GÉANT, while GU-1b requires less than 2 seconds for each network. The reason is because both ILP-1b and GU-1b do not include link-cost setting.

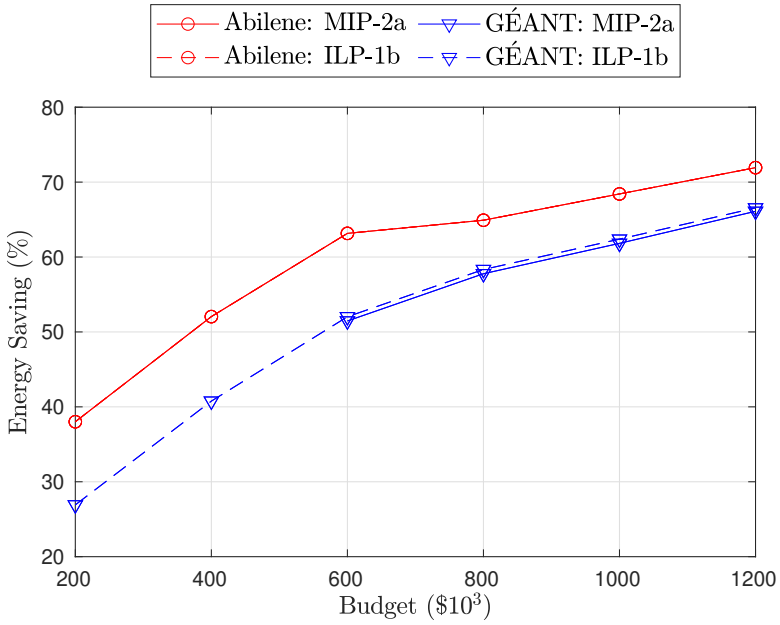


Figure 4.5. Energy saving ε_T of MIP-2a and ILP-1b.

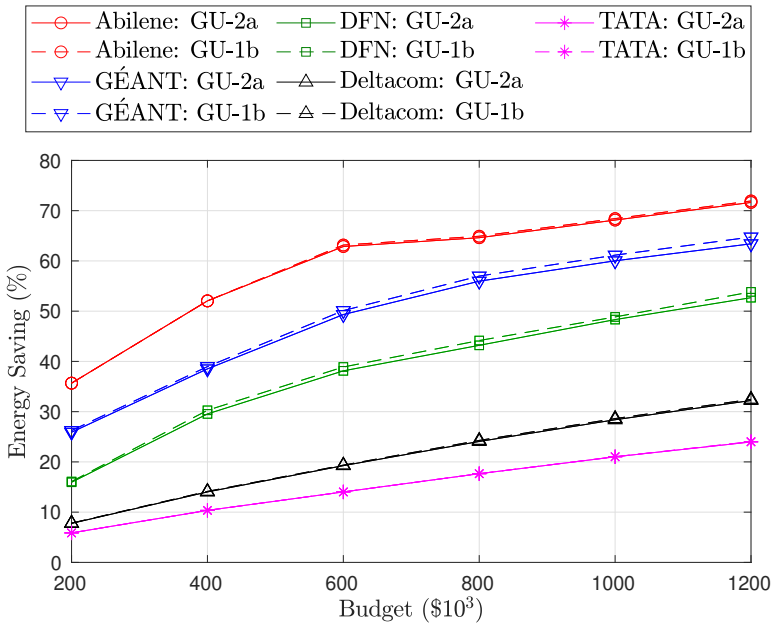


Figure 4.6. Energy saving ε_T of GU-2a and GU-1b.

4.3.5. Effect of Routing via Link-disjoint Paths

This simulation aims to show the impact of routing traffic demands via link-disjoint paths. It uses $B = \{\$200K, \$400K, \$600K, \$800K, \$1M, \$1.2M\}$ and $T = 3$ stages. As shown in Figure 4.7, the energy saving of GU-2b is only 1.32% and 3.64% less than the savings of MIP-2b for Abilene and GÉANT, respectively. As an example, for budget $B = \$1.2M$, MIP-2b produces energy saving ε_T of 71.93% and 65.77% for Abilene and GÉANT, respectively. For the two respective networks, GU-2b yields 71.64% and 63.29% of energy saving. For GÉANT, MIP-2b fails to produce results for $B = \{\$200K, \$400K\}$ after running for one week. Similarly, MIP-2b fails to obtain results for DFN, Deltacom and TATA.

Overall, as shown in Figure 4.3 and Figure 4.7, MIP-2b and GU-2b produce energy savings that are close to those of MIP-2a and GU-2a, respectively. For Abilene, all solutions, i.e., MIP-2b, GU-2b, MIP-2a and GU-2a, produce the same ε_T. For GÉANT, the average saving ε_T of MIP-2b is only 0.32% less than

that of MIP-2a. Similarly, the saving of GU-2b is only 0.63% less than that of GU-2a. Similarly, for DFN and Deltacom, the energy saving obtained by GU-2b is only 0.36% and 0.06% off, respectively, from the saving of GU-2a. Moreover, GU-2b and GU-2a produce the same saving for TATA. The reason is because MIP-2b and GU-2b route the majority of traffic demands via single paths. More specifically, for Abilene with budget B = \$1.2M, MIP-2b cannot route any traffic demand via link-disjoint paths. It routes only 2.27% of demands via non link-disjoint multi-paths. For GÉANT and budget B = \$1.2M, MIP-2b routes 10.3% and 7.58% of traffic demands via link-disjoint and non-link-disjoint paths, respectively. Similarly, GU-2b routes all demands of Abilene via single path routing, while for GEANT, it uses link-disjoint and non-link-disjoint paths to route only 10.3% and 29.4% of traffic demands, respectively. Note that the percentage of traffic demands routed over link-disjoint paths that also satisfies a given delay tolerance for Abilene, GÉANT, DFN, Deltacom, and TATA is 6.06%, 55.15%, 9.83%, 2.96%, and 3.86%, respectively.

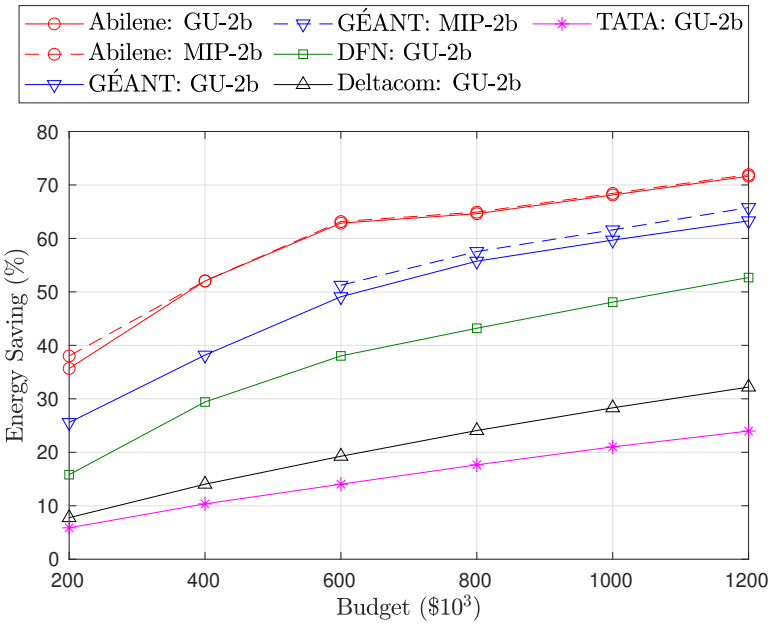


Figure 4.7. Energy saving ε_T of MIP-2b and GU-2b.

4.3.6. GU-2a versus Two Existing Solutions

In this section, we compare the performance of GU-2a against two existing solutions, which are Local Search (LS) [1] and Energy-Efficient Genetic Algorithm for Hybrid-SDNs (EEGAH-MNL) [2], in terms of traffic controllability and energy saving. For brevity, in this chapter we call EEGAH-MNL as GA. Briefly, LS aims to upgrade l-switches over multi-stages subject to a given total budget B. However, the goal is to maximize the total traffic controllability over $T \geq 1$ stages, denoted by TC. Moreover, LS is allowed to use its entire budget in one stage. On the other hand, GA aims to minimize the power consumption of links that are adjacent to an s-switch (c-links) and s-switches in a single upgrade stage, i.e., $T = 1$. Both LS and GA consider single path routing. Note that GA generates each shortest path using only the powered-on links, and thus producing paths with long delays. Both LS and GA consider non-bundled links where they only have one cable.

We compare the performance of GU-2a, LS, and GA using the following scenarios: (i) single path routing with 10% delay tolerance, (ii) initial upgrade cost of $p_u^0 = \$100\text{K}$ for each switch u with decrease rate of $\rho = 40\%$; all switches have the same upgrade cost, (iii) a set of budget $B = \{\$200\text{K}, \$400\text{K}, \$600\text{K}, \$800\text{K}, \$1\text{M}, \$1.2\text{M}\}$, (iv) $T = 3$ upgrade stages, (v) MLU threshold of $U_{\max} = 80\%$, (vi) traffic size of each demand d increases with rate $\mu_d = 22\%$, (vii) each link contains $b_{uv} = 4$ cables, and (viii) only s-switches can turn off unused cables.

Next, we provide additional settings for our simulations:

1. We consider the following link models: (i) each link contains only one cable, i.e., $b_{uv} = 1$, and (ii) each link contains $b_{uv} = 4$ cables. For model (ii), we calculate the energy saving for LS and GA from the traffic volume on each link. Specifically, each link with traffic volume ω uses an equivalent of

$\lceil \omega/\gamma \times U_{\max} \rceil$ cables, where γ is the capacity of each cable.

2. To simulate multi-stage upgrades for GA, we run the algorithm $T = 3$ times. At each stage $t \in \{1, 2, 3\}$, the number of l-switches that can be replaced by s-switches in GA is equal to the number of upgraded l-switches in GU-2a. Note that we assume all switches have the same upgrade cost so that GA can upgrade the same number of l-switches as GU-2a in decreasing order of the number of l-links.

3. The fitness function of GA is changed to the sum of the total number of powered-on links, assuming that the power rate of all links is the same.

Note that LS fails to produce results for DFN, Deltacom and TATA after running for three days. Thus, we use only Abilene and GÉANT to compare the TC and the energy saving ε_T performance of GU-2a, LS and GA. The following Sections 4.3.6.1 and 4.3.6.2 discuss the TC and the energy saving results.

4.3.6.1. Performance on TC

The TC results shown in Figure 4.8 apply to non-bundled and bundled link models because the results are exactly the same. As shown in Figure 4.8, LS consistently produces, on average, a higher TC than GU-2a and GA for Abilene and GÉANT. The results are expected as the goal of LS is to maximize TC. As an example, for budget $B = \$200\text{K}$, LS produces 38.35% and 49.75% higher TC than GU-2a, and 43.1% and 48% higher TC than GA for Abilene and GÉANT, respectively. However, as the budget increases to $B = \$1.2\text{M}$, the difference between TC of LS and GU-2a (LS and GA) reduces to only 5.95% (9.01%) and 4.69% (5.7%) for the two respective networks. The reason is because the budget at each stage becomes larger with increasing budget B. Thus, GU-2a and GA upgrade most of the l-switches at earlier stages and hence, produces TC with values closer to

LS. As shown in Figure 4.8, GU-2a and GA produce comparable TC values. At maximum, GU-2a results in 3.17% and 9.49% lower TC than GA for Abilene and GÉANT respectively. The reason is because GA upgrades l-switches with the highest total number of l-links or *node degree*. On the other hand, GU-2a selects l-switches which do not necessarily have the highest total number of node degrees. Note that switches with the highest node degree are likely to be traversed by more end-to-end paths [40]. To further analyze TC performance, we show the value of TC at each stage in Figure 4.9. Note that GU-2a and GA produce a similar trend, and thus the figure only compares the results of GU-2a and LS.

Figure 4.9 shows the TC produced by GU-2a and LS at stage 1 to 3 using a budget of $B = \$200\text{K}$. GU-2a consistently produces higher TC at the last stage, whilst TC of LS remains the same over the three stages. For Abilene, the TC produced by GU-2a increases drastically from 18.58% to 94.97%, while LS yields the same TC of 77.86% from stage $t = 1$ to $t = 3$. Similarly for GÉANT, the TC achieved by GU-2a escalates from 11.07% to 74.17%, whilst LS produces the same TC of 69.04% for each stage t. The reason is because LS spends its entire budget upgrading l-switches in the first stage. In contrast, GU-2a has a maximum budget to spend at each stage. Further, GU-2a aims to maximize the energy saving ε_T, while LS aims to maximize TC. Thus, on average, LS results in a higher TC.

4.3.6.2. Energy saving performance

This section first evaluates the energy saving ε_T produced by GU-2a, LS and GA for non-bundled and bundled link models. Then, it analyzes the saving ε^t of GU-2a and LS at each stage $t \in \{1, 2, 3\}$. Lastly, it shows the advantage of saving more energy at later stages on energy cost.

Non-bundled links model: Figure 4.10 shows the energy saving ε_T for

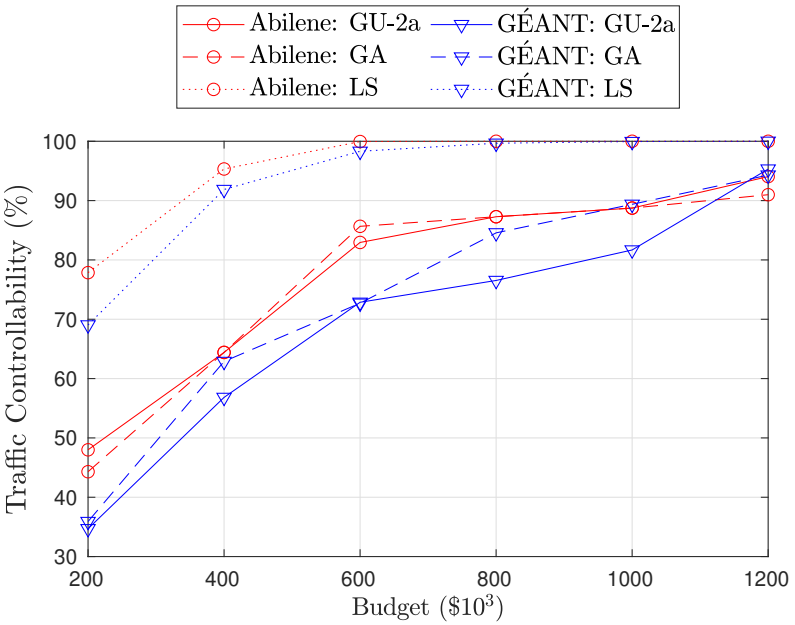


Figure 4.8. Traffic Controllability of GU-2a, LS [1], and GA [2].

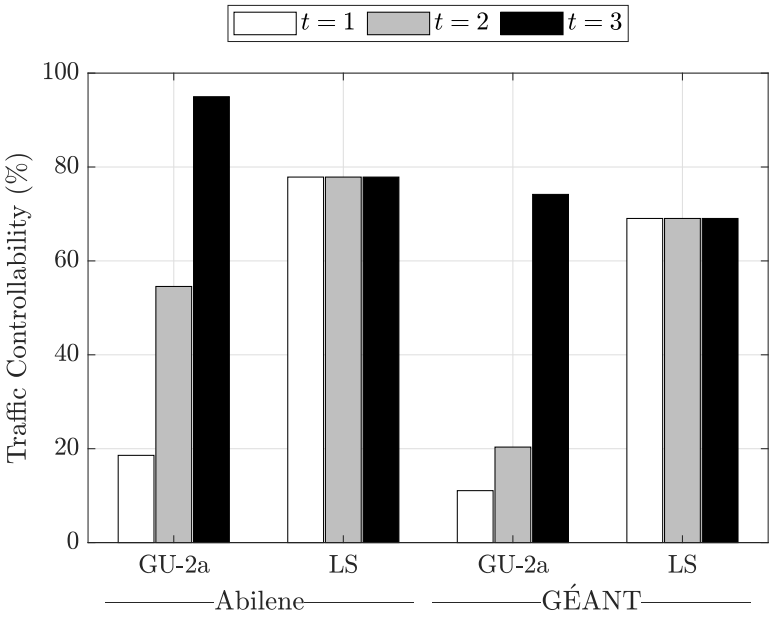


Figure 4.9. Traffic Controllability for $T = 3$ and $B = \$200\text{K}$.

the non-bundled link model. We see that LS uses all links to route a set of end-to-end traffic demands via shortest paths, and hence no energy saving. In contrast, GU-2a and GA can save energy because both solutions turn off as many links as possible and route traffic demands using the remaining active links. For Abilene, both solutions produce the same energy saving. As an example, for budget $B = \$200K$, GU-2a and GA produce the same $\varepsilon_T = 4.44\%$ which increases to $\varepsilon_T = 6.67\%$ for larger budget $B = \$1.2M$. On the other hand, for GÉANT and budget $B = \$200K$, GA results in 12.5% less ε_T than GU-2a. As the budget increases to $B = \$1.2M$, GU-2a significantly overcomes GA with 26.32% higher saving. Note that the energy savings of GU-2a outperforms those of GA for the other networks, i.e., DFN, Deltacom and TATA.

Bundled link model: We evaluate the energy saving performance of LS and GA for the bundled-links model. Figure 4.11 shows that LS can save energy. It produces less energy saving ε_T than GU-2a and GA with budget up to $B = \$200K$ for Abilene and GÉANT. For budget $B = \$200K$, LS yields energy saving ε_T of 21.05% and 21.96% for Abilene and GÉANT, respectively. On the other hand, GU-2a respectively produces higher energy saving ε_T of 29.24% and 22.3% for Abilene and GÉANT. Similarly for Abilene, GA produces the same saving $\varepsilon_T = 29.24\%$ which is higher than LS. However, GA produces energy saving ε_T of 21.28% which is slightly less than LS for GÉANT. Note that GU-2a produces 4.19%, 7.45%, and 1.94% higher energy saving than GA for DFN, Deltacom and TATA. However, as the budget increases, LS produces higher energy saving than GU-2a and GA. For Abilene and GÉANT with budget $B = \$800K$, LS obtains respectively 16.67% and 14% higher energy saving than GU-2a. Similarly, LS results in 16.67% and 16% higher saving than GA for the two respective networks. We use Figure 4.12 to explain the reasons for the higher ε_T values that are produced by LS when the budget increases. We consider only GU-2a in

Figure 4.12 to analyze the energy saving performance against LS at each stage. The reason is because the energy savings produced by GA for all budgets, as shown in Figure 4.11, are the same for Abilene and only 2.68% off from the savings resulted by GU-2a for GÉANT.

Energy Saving per Stage: Figure 4.12 shows a comparison of the ε_T produced by GU-2a and LS at each stage $t \in \{1, 2, 3\}$ for Abilene and GÉANT using a budget of $B = \$200\text{K}$ and bundled link model. As shown in Figure 4.12, ε_T increases at each stage for GU-2a, while ε_T of LS decreases slightly at later stages, especially for GÉANT. The reason is because LS uses its entire budget at stage $t = 1$ and thus the number of s-switches upgraded by LS remains the same from stage 1 to 3. Recall that the traffic size increases at a rate of $\mu = 22\%$ per stage, and thus some cables need to be switched on, which decrease the energy saving of LS over $T = 3$ stages. Moreover, budget $B = \$400\text{K}$ and $B = \$200\text{K}$ are not sufficiently large for LS to upgrade all l-switches of Abilene and GÉANT in only one stage, respectively. On the other hand, GU-2a constrains the maximum budget that can be spent at each stage. Thus, GU-2a is able to upgrade more switches at the later stages, which increases energy savings. It is important to note that, in general, GU-2a would upgrade a larger number of switches than LS since the upgrade cost decreases over stages.

Benefit of More Energy Saving at Later Stage: The following case study shows the benefit of saving more energy in later stages. Note that, in general, electricity cost increases in later years. For example, in the United States, reference [29] projects an annual increase in energy prices of 3.29% from 2020 to 2025. Assume Abilene and GÉANT carries out an upgrade every two-year using a total budget of $B = \$200\text{K}$ for $T = 3$ upgrade stages. For Abilene, GU-2a and LS produce $\{15.79\%, 26.32\%, 64.91\%\}$ and $\{21.05\%, 21.05\%, 21.05\%\}$ of energy saving, respectively, at each stage. Assuming an initial energy cost of \$1 per *on-*

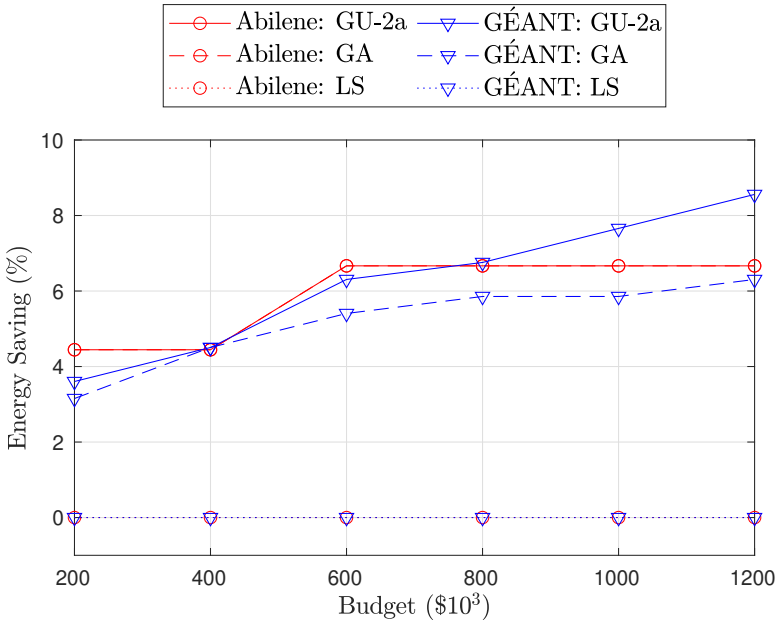


Figure 4.10. Energy saving ε_T of GU-2a, LS [1], and GA [2] for link model with single cable.

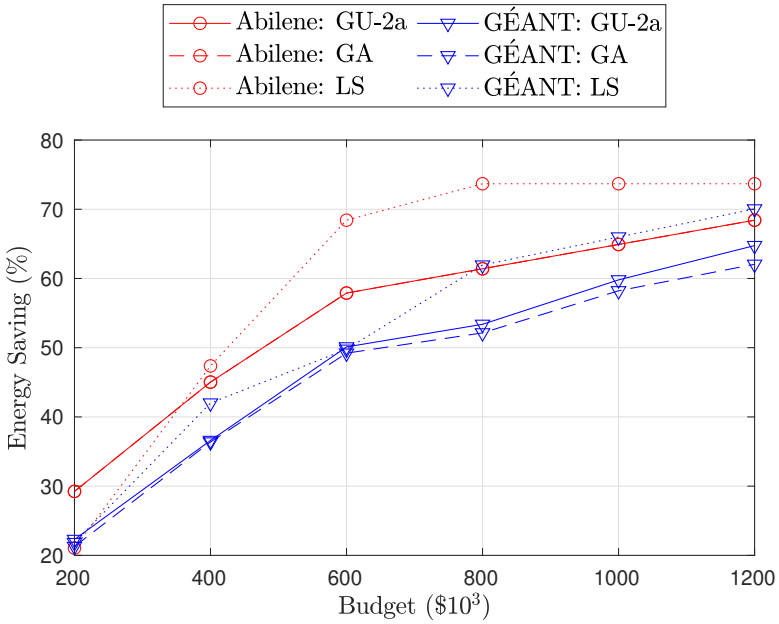


Figure 4.11. Energy saving ε_T of GU-2a, LS [1], and GA [2] for bundled link model.

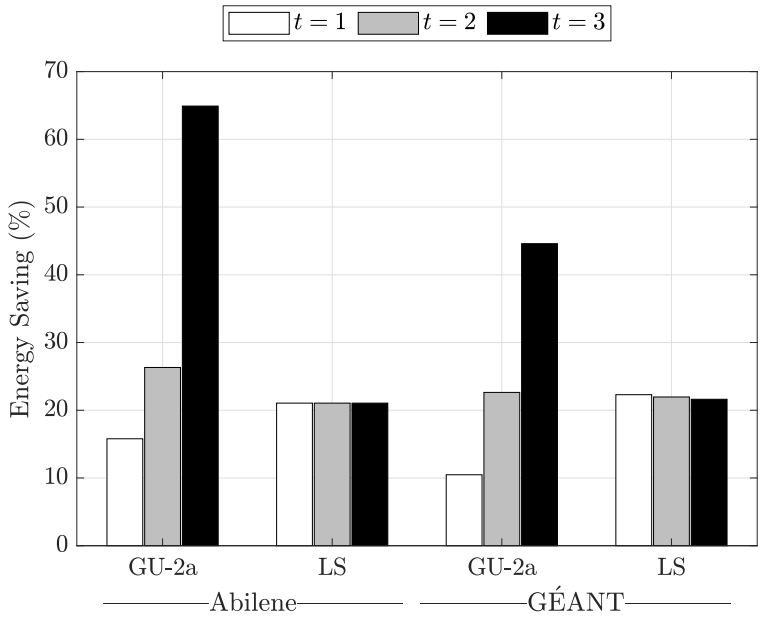


Figure 4.12. Energy saving ε_T for $T = 3$ and $B = \$200\text{K}$.

cable, GU-2a will be able to save $\$(0.1579 + 0.2632 \times 1.0329^2 + 0.6491 \times 1.0329^4) = \$1.775$, while LS saves only \$0.6747. For GÉANT, GU-2a saves \$0.8538 which is slightly higher than LS, which only saves \$0.7033.

4.3.7. Additional Results

This section reports additional findings in terms of increased path delays and link utilization when using our approach. Further, it analyzes the benefit of using *l*-switches that can turn off unused cables, e.g., those that support the IEEE 802.3az standard. Let us call this *green* *l*-switch as *gl*-switch. In addition, it discusses the energy saving performance of our approach against existing techniques in non-SDNs and pure SDNs. Sections 4.3.7.1, 4.3.7.2, and 4.3.7.3 use the total budget of $B = \$1.2\text{M}$ and the total number of upgrade stages is $T = 3$. On the other hand, Sections 4.3.7.4 and 4.3.7.5 use a different total budget B over the same total number of upgrade stages, i.e., $T = 3$.

4.3.7.1. Path Delay

Figure 4.13 shows a small increase in path delays for all networks when $B = \$1.2M$ is used. More specifically, path delays produced by MIP-2a (GU-2a) for Abilene and GÉANT, on average, are increased by 0.43% (1%) and 2.84% (0.01%), respectively. Further, only 4.3% (12.12%) and 28.39% (0.25%) of the paths in the respective networks have 10% longer delays. Note that all simulations allow 10% delay tolerance and each path originally uses the shortest path. Thus, each path cannot have a lower delay or more than 10% increase in delay. For DFN, Deltacom and TATA, there is no increase in path delay for a budget of $B = \$1.2M$. This is because the said budget can only upgrade 37.93%, 25.66%, and 19.31% of switches in the respective networks. However, when we increase the budget such that GU-2a can upgrade more switches and all links are c-links, GU-2a is able to route some demands via longer paths to maximize energy saving; see the results in Figure 4.13 for $B \gg \$1.2M$. For example, there are respectively 22.39% and 16.51% of traffic demands that use longer paths for Deltacom and TATA. In this case, the average path delay of the two networks is increased by 2.24% and 1.65%, respectively.

4.3.7.2. Link Utilization

To see the effect of our MIP-2a and GU-2a on link utilization, we first measure the initial utilization of all links of each network, i.e., before upgrading the network. Recall that the initial routing of each demand follows the OSPF-ECMP protocol. As shown in Figure 4.14, we find that the maximum link utilization in the five networks ranges between 18% and 36% when all cables are turned on. More specifically, for Abilene and GÉANT, the maximum link utilization is 18.16% and 35.05%, respectively. Then, we measure link utilization of each network after

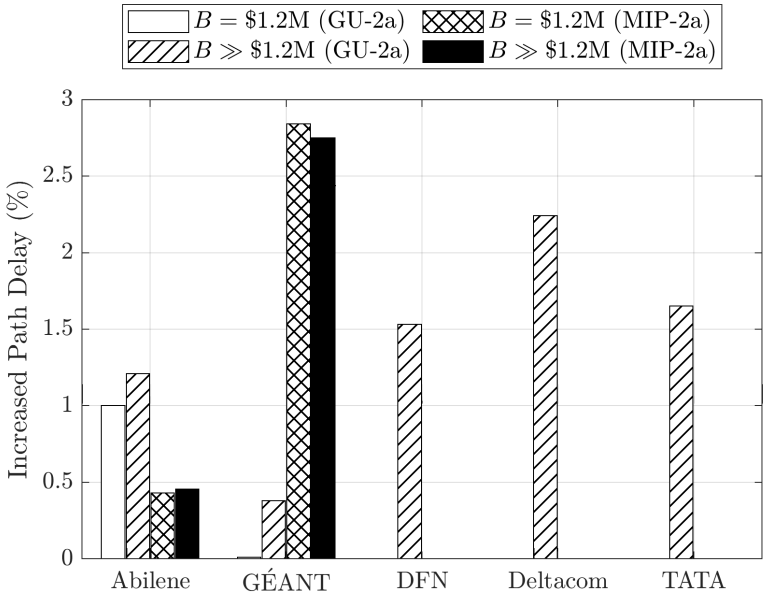


Figure 4.13. Increase in path delay produced by GU-2a and MIP-2a.

upgrading the network using MIP-2a or GU-2a. Using MIP-2a, the maximum link utilization in Abilene and GÉANT decreases to 17.81% and 23.18%, respectively. On the other hand, GU-2a does not change the maximum link utilization for all networks. The reason is because GU-2a limits the number of *on*-cables on each link at each stage according to the number of *on*-cables used at the last upgrade stage T when performing traffic rerouting. Moreover, GU-2a reroutes any traffic demand at each stage by using its largest volume at stage T. Thus, the maximum link utilization is less likely to increase significantly.

4.3.7.3. Energy Savings in Networks with Green Legacy-Switches

This section examines the effect of using *gl*-switches, i.e., *l*-switches that support energy efficient technology, e.g., IEEE 802.3az, to turn-off unused cables in each *l*-link. Recall that the reported energy savings in all previous sections consider non *gl*-switches, and thus unused cables in each *l*-link are still *on*. For this examination, we modify Eq. (2.1) to include unused cables in both *c*-links and

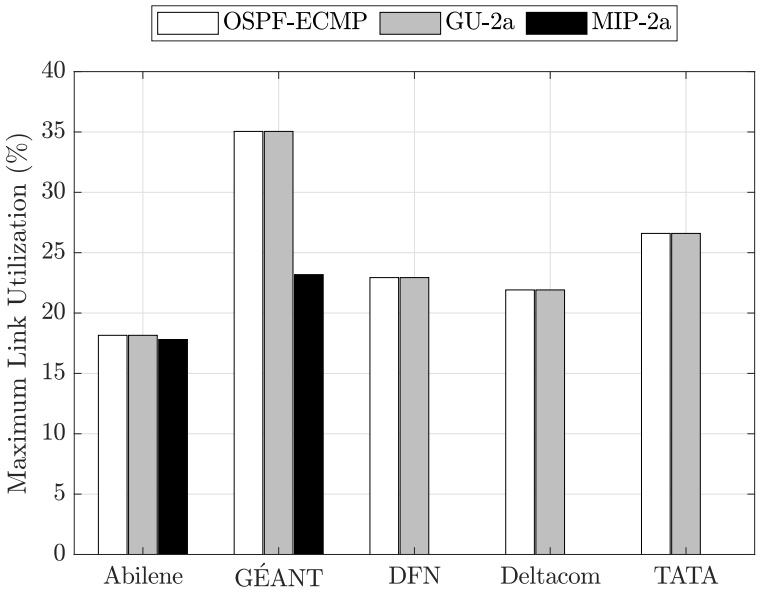


Figure 4.14. Maximum link utilization produced by OSPF-ECMP, GU-2a, and MIP-2a.

l-links. Figure 4.15 shows that MIP-2a increases the energy saving of Abilene and GÉANT from 71.93% to 75.44% and 66.1% to 77.7%, respectively. Similarly, GU-2a improves the energy saving of Abilene and GÉANT from 71.64% to 75.15% and 63.4% to 74.78%, respectively. Further, For DFN, Deltacom and TATA, GU-2a increases their saving from 52.68% to 74.81%, 32.22% to 77.43%, and 23.97% to 74.55%, respectively. The additional saving accumulates because the unused cables in each *l*-link can now be powered off by *gl*-switches, and hence saving more energy.

4.3.7.4. Energy Savings in non-SDNs

This section evaluates the energy savings in legacy networks or *non-SDNs*. More specifically, we use the greedy-based heuristic solution, called MSPF-LF, and its simulation results in [14] to represent the energy saving in non-SDNs. Similar to GU-2a, the MSPF-LF approach in [14] considered multi-path routing, delay

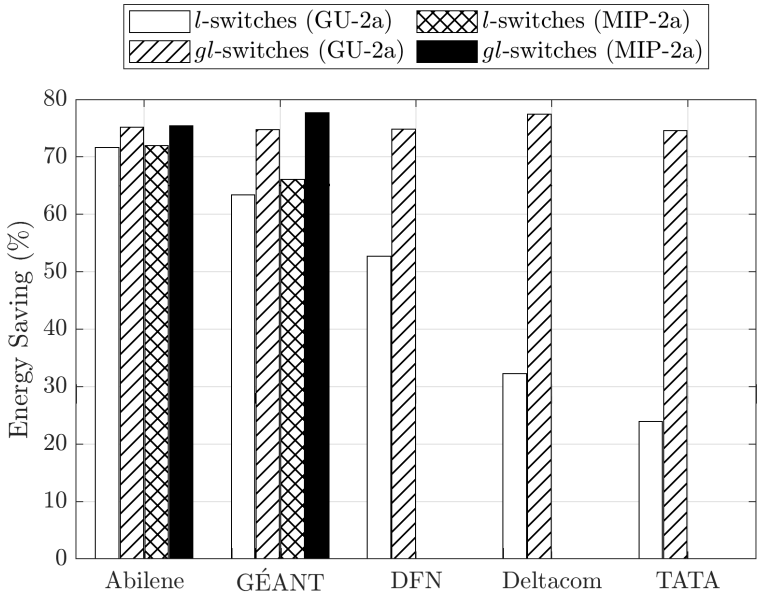


Figure 4.15. Energy saving performance with *l*-switches and *gl*-switches.

constraint, maximum link utilization threshold, and bundled links. Further, the results reported in [14] use the same network topologies as ours, i.e., Abilene and GÉANT. In addition, MSPF-LF used *gl*-switches that can also perform traffic rerouting. Thus, for GU-2a, we use a total budget B that is sufficiently large to upgrade *l*-switches that can control all traffic flows and turn off all unused cables at the beginning of each upgrade stage. As reported in [14], MSPF-LF produced 73% and 74% energy saving for Abilene and GÉANT respectively; see Figure 4.16. On the other hand, the energy saving produced by GU-2a is 75.14% and 76.46% for the respective networks. Thus, our results are better than those reported for MSPF-LF.

4.3.7.5. Energy Savings in Pure SDNs

This section presents the energy saving in *pure* SDNs. To represent the energy savings, we use GA [2] and LS [1]. In this case, except for the total budget B, we use the same scenarios and settings as in Section 4.3.6 for GU-2a, LS

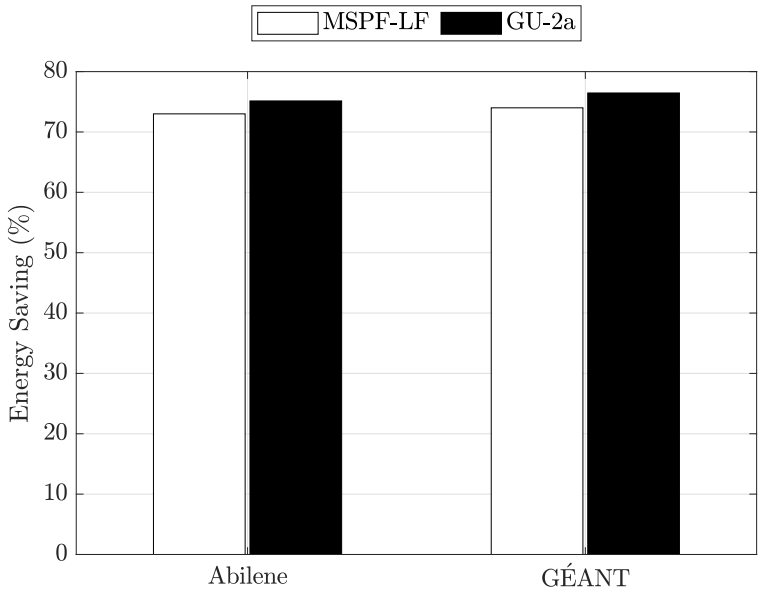


Figure 4.16. Energy saving performance in non-SDN.

and GA. We use a sufficiently large budget to upgrade all l-switches at the first stage and calculate energy saving ε_T over $T = 3$ stages. Figure 4.17 shows that MIP-2a obtains the optimal energy saving of 75.44% and 77.7% for Abilene and GÉANT, respectively. GU-2a and GA produce the same energy saving of 75.44% for Abilene and 77.03% and 75.34%, respectively, for GÉANT. For LS, the energy saving for both networks is 73.68% and 74.44%, respectively. The results show that our solutions, i.e., MIP-2a and GU-2a, outperform both GA and LS for *pure* SDNs. Further, we observe that the energy saving achieved in *pure* SDNs is higher than those in non-SDNs and hybrid SDNs; viz. Section 4.3.6.

4.4. Chapter Summary

This chapter considers the problem of upgrading a legacy network that supports OSPF-ECMP into an SDN over multiple stages. The aim is that an upgraded network must maximize energy saving. To do so, we consider the maximum

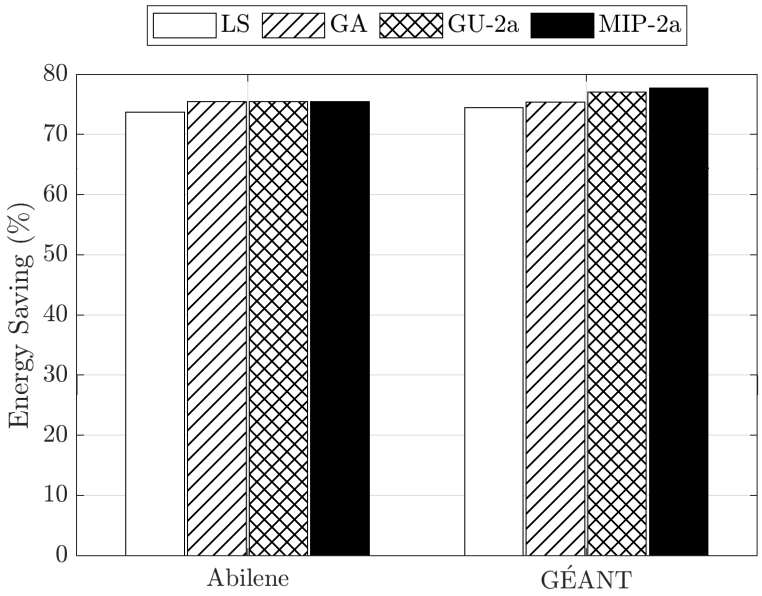


Figure 4.17. Energy saving performance in non-SDN and pure SDN.

available budget at each stage, MLU, maximum path delay, and each l-switch must comply with OSPF-ECMP. This chapter considers two routing scenarios: 1) multi-path and 2) link-disjoint. We have formulated an MIP-2a for scenario-1 and its extension, called MIP-2b, for scenario-2. In addition, we have proposed two heuristic solutions: GU-2a for scenario-1 and GU-2b for scenario-2. Our simulations have shown that GU-2a and GU-2b require significantly less CPU time than MIP-2a and MIP-2b, respectively. Further, GU-2a and GU-2b obtain energy saving that is only up to 4% off from the optimal saving obtained by MIP-2a and MIP-2b, respectively. The energy saving of MIP-2b and GU-2b when considering link-disjoint paths is only 0.63% off from the saving attained by MIP-2a and GU-2a. Moreover, GU-2a produces up to 1.77% less energy saving than GU-1b, which uses single path routing. We find that increasing budget and number of stages result in larger energy savings. Further, GU-2a produces higher energy saving at later stages than an existing technique, called LS, that tends to spend its entire budget at the first stage. As the energy price (in $) is expected

to increase every year, GU-2a is expected to perform better than LS in terms of reducing the OPEX of networks. The next Chapter 5 addresses controller placement problem in GMSU.

Chapter 5

Multi-Stage Switch Upgrade and Controllers Placement for Green SDN/OSPF Networks

This chapter addresses GMSU-3 problem to upgrade a legacy network that supports bundled links and OSPF to SDN over T stages. The upgrade considers replacing a subset of l-switches with a set of s-switches and deploys a set of controllers to manage the s-switches. The goal is to maximize the number of *off* cables adjacent to s-switches to save energy. It uses a single path within delay tolerance and MLU to route both data and control traffic. Specifically, control traffic has a *backup* path that is link-disjoint with the *active* path that is used to route the traffic. The backup paths must also satisfy a given delay requirement and MLU. GMSU-3 uses the maximum budget at each stage to upgrade switches and deploy controllers. In addition, it considers three additional constraints with respect to the controller location, the association between s-switches and controllers, and the controller capacity.

The layout of this chapter is organized as follows. Section 5.1 provides the

problem formulation of GMSU-3, Section 5.2 proposes a heuristic solution, Section 5.3 evaluates the performance of MIP and heuristic solutions, and Section 5.4 concludes the chapter. Note that the work in this chapter has been submitted to *IEEE Access* and is currently under review.

5.1. Problem Formulation

Section 5.1.1 provides the system model used in GMSU-3. Note that Section 2.5 provides other models and notations used in this chapter. Section 5.1.2 formally defines GMSU-3 and Section 5.1.3 provides its illustration. Section 5.1.4 presents the mathematical model of GMSU-3. Finally, Section 5.1.5 shows the NP-hardness of GMSU-3.

5.1.1. System Model for GMSU-3

This section describes a set of models used in GMSU-3, i.e., for network, switch-controller association, traffic flow, routing, and budget. Note that the models for network, traffic flow, routing, and budget in GMSU-3 extend those models discussed in Section 2.5. Table 5.1 provides a list of notations that are used in GMSU-3. The following subsections outline the models and the notations in details.

5.1.1.1. Network Model

For each stage t, an operator replaces a set of l-switches to s-switches and deploys one or more controllers to manage s-switches. Let C^t represent a set of controllers that have been deployed up to stage t. An s-switch u is a switch located at node $u \in V$, denoted by $\mathcal{S}(u)$, while a controller at node u is represented by $\mathcal{C}(u)$. For each controller $\mathcal{C}(u)$, node u must contain an s-switch, i.e., each controller must

Table 5.1. Specific notations for GMSU-3

Notation	Description
$\hat{p}^t$	The deployment cost of each controller at stage t.
$\hat{\rho}$	The decrease cost rate for each controller at each stage.
C^t	The set of controllers at stage t.
$\mathcal{S}(u)$	A switch at node u.
$\mathcal{C}(u)$	A controller at node u.
$q^t_{u,a}$	An indicator set to 1 if s-switch u is associated to controller a.
A^t_a	The set of s-switches being associated to controller a at stage t.
$\hat{\omega}^t_u$	The number of *cpps* generated by s-switch u at stage t.
$\hat{\mu}_u$	The increase rate of the number of *cpps* for switch u per stage.
θ	The capacity of each controller (in *cpps*).
$\hat{R}^t_d$	The set of paths to route each demand d in D^t at stage t.
$\hat{x}^t_u$	An indicator set to 1 if controller u is deployed at stage t.
$\mathcal{I}_d$	An indicator set to 1 if demand d is an s-s or s-c demand.
$y^t_{d,uv,i}$	An indicator set to 1 if link (u,v) is included in path $\hat{R}^t_{d,i} \in \hat{R}^t_d$.
$y^t_{d,i}$	An indicator set to 1 if path $\hat{R}^t_{d,i} \in \hat{R}^t_d$ is an active path.

be *co-located* with an s-switch in order to reduce traffic delay and ensure high communication reliability [71]. An s-switch that is co-located with a controller is called a *co*-switch. Hence, for each *co*-switch u, node u represents both a controller $\mathcal{C}(u) \in C^t$ and an s-switch $\mathcal{S}(u) \in V^t$. For brevity, unless it is necessary to differentiate between a node u and an s-switch or a controller at node u, this chapter uses u and $\mathcal{S}(u)$ as well as u and $\mathcal{C}(u)$ interchangeably.

5.1.1.2. Switch and Controller Association Models

Each s-switch is associated to a controller, where in each stage t, an s-switch $u \in V^t$ is associated to a controller $a \in C^t$. Let $q^t_{u,a}$ be a binary indicator that is set to one when switch u is associated to a controller a at stage t. Let $A^t_a \subseteq V^t$ be a set of all s-switches that are associated to controller a at stage t. Each

co-switch is permanently associated to its controller. However, the association of other *s*-switches may change in subsequent stages. Define the set A^t to record all switch-to-controller associations at stage t; formally, set $A^t = \{A^t_a \mid a \in C^t\}$. Let $\hat{\omega}^t_u$ denote the number of control-packet-per-second or *cpps* that originates from *s*-switch u at stage t. The value of $\hat{\omega}^t_u$ increases at a rate of $\hat{\mu}_u \geq 0$ per stage, i.e., $\hat{\omega}^{t+1}_u = \lceil \hat{\omega}^t_u \times (1 + \hat{\mu}_u) \rceil$, for $t < T$. GMSU-3 assumes each controller generates the same *cpps* to each of its associated *s*-switch at every stage. In addition, each controller has a limited capacity. As per [63], [68], and [72], GMSU-3 assumes all controllers have the same capacity θ (in *cpps*). Consequently, at each stage t, the total *cpps* from all its associated *s*-switches must be less than θ, i.e., $\sum_{u \in A^t_a} \hat{\omega}^t_u \leq \theta$.

5.1.1.3. Traffic Flow Model

There are three types of traffic in SDN: (i) *switch-switch* or *s-s*, (ii) *switch-controller* or *s-c*, and (iii) *controller-controller* or *c-c*. Traffic *s-s* contains data packets between *s*-switches, a pair of *l*-switches, or an *s*-switch and an *l*-switch. Traffic *s-s* can originate from an *s*-switch or an *l*-switch. Note that *l*-switches rely on OSPF to determine the least cost path between switches. Traffic type *s-c* corresponds to control packets between an *s*-switch and its controller [61]. For example, an *s*-switch generates *s-c* traffic when requesting forwarding rules for a new *s-s* traffic flow or when it sends its status. On the other hand, a controller generates *s-c* traffic when it sends forwarding rules and maintenance information. Both *s-s* traffic and *s-c* traffic share the same communication link or *in-band channel*. Lastly, traffic *c-c* represents control packets between controllers. GMSU-3 ignores *c-c* traffic because each *c-c* traffic is typically routed via a dedicated control network to ensure fast and reliable communication [62], [63].

GMSU-3 uses set D^t to represent a set of *s-s* and *s-c* traffic demands at stage

t. Let $D^t_{\text{ss}} \subset D^t$ and $D^t_{\text{sc}} \subset D^t$ denote the sets of *s-s* and *s-c* demands at stage t, respectively. Here, sets $D^t_{\text{ss}} \cup D^t_{\text{sc}} = D^t$ and $D^t_{\text{ss}} \cap D^t_{\text{sc}} = \{\}$, with demand $d > |D^t_{\text{ss}}|$ refers to an *s-c* demand. Note that $D^0_{\text{ss}} = D^1_{\text{ss}}$ is a given set of *s-s* traffic demands in $G^0(V, E)$, and ω^0_d is the initial traffic volume of demand d. GMSU-3 assumes the total number of *s-s* demands remains the same over T stages, i.e., $|D^t_{\text{ss}}|$ is constant. Each *s-c* demand in D^t_{sc} exists only if there is an association between *s*-switch $s_d \in V^t$ and controller $\tau_d \in C^t$ at stage t, i.e., $q^t_{s_d,\tau_d} = 1$ and $q^t_{\tau_d,s_d} = 1$. For each association, there are two possible *s-c* demands in D^t_{sc}: (i) from a switch to controller, i.e., $(s_d, \tau_d, \omega^t_d)$, and (ii) from a controller to a switch, i.e., $(s_{d+1}, \tau_{d+1}, \omega^t_{d+1})$, where $s_d = \tau_{d+1} \in V^t$ and $\tau_d = s_{d+1} \in C^t$. Let β denote the average size of a control packet (in Byte). Thus, the traffic volume of each *s-c* demand is computed as $\omega^t_d = \hat{\omega}^t_{s_d} \times \beta$, and $\omega^t_{d+1} = \omega^t_d$.

5.1.1.4. Routing Model

Let $\hat{R}^t_d \subseteq \mathcal{P}_d$ as a set of paths that are used to route demand $d \in \{1, \ldots, |D^t|\}$ at stage t. Each *s-s* demand d is routed via a single path; thus we have $|\hat{R}^t_d| = 1$. On the other hand, each *s-c* demand d uses up to two (s_d, τ_d) link-disjoint paths; thus, we have $|\hat{R}^t_d| = 1$ or $|\hat{R}^t_d| = 2$. For $|\hat{R}^t_d| = 2$, one path, called the *active* path is used to route demand d, while the other serves as *backup*. Note that there is no path for each *s-c* demand d between a *co*-switch and its controller, i.e., for this case we have $\hat{R}^t_d = \{\}$. The reason is because they are located at the same node and can communicate directly without delay. GMSU-3 uses $R^t = \{\hat{R}^t_d \mid d \in \{1, \ldots, |D^t|\}\}$ to denote the set of all paths to route all *s-s* and *s-c* demands at stage t.

5.1.1.5. Budget Model

Recall that each switch u has an upgrade cost p_u^t. Let $\hat{p}^t$ denote the deployment cost of each controller at stage t. Both costs include the purchase price and installation cost. The upgrade cost of a switch corresponds to its processing capability, i.e., whether it is an edge or core switch [1]. Thus, we assume a switch with a larger *cpps* is more expensive than a switch with a lower *cpps*. All controllers have the same deployment cost and capacity θ. Denote $\hat{\rho}$ as the per-stage depreciation rate of controller deployment cost; here rate $\hat{\rho}$ satisfies $0 \leq \hat{\rho} < 1$. Thus, the cost to deploy each controller at stage t is $\hat{p}^t = \hat{p}^0 \times (1-\hat{\rho})^{t-1}$. Note that $\hat{p}^0$ is the initial cost of deploying a controller.

5.1.2. GMSU-3 Problem Definition

Given a legacy network $G^0(V, E)$ that support OSPF, GMSU-3 upgrades a subset of l-switches with a set of s-switches V^T and deploys a set of controllers C^T over $T \geq 1$ stages. The aim is to maximize energy saving by shutting down a maximum number of unused cables per Eq. (2.2). GMSU-3 considers the following four main constraints:

1. **Budget B:** The total cost to upgrade l-switches and deploy a set of controllers at each stage t does not exceed the maximum budget B^t. Recall that (i) $\mathrm{B} = \sum_{t=1}^{T} B^t$, and (ii) the deployment cost of each switch and a controller depreciates with a given rate ρ and $\hat{\rho}$ per stage, respectively

2. **Routing:** Each path in set $\hat{R}_d^t$ must satisfy a given path delay constraint $\delta_{\max,d}$. GMSU-3 considers (i) a single path to route s-s demand d in D_{ss}^t, (ii) up to two link-disjoint paths for s-c demand d in D_{sc}^t; one path is used to route traffic, while the other serves as backup, and (iii) in-band channel

for *s-s* and *s-c* traffics, i.e., their routing paths use the same set of links.

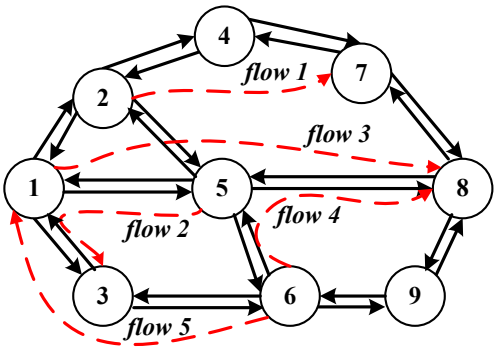
3. **Link capacity**: The total traffic volume on each link $(u, v) \in V$ at each stage t is within the maximum capacity of the link $c^t_{uv} = (n^t_{uv}/b_{uv}) \times c_{uv} \times U_{\max}$. Here, GMSU-3 considers the traffic volume ω^t_d of each demand d in (i) set D^t_{ss} increases with a rate of μ_d, and (ii) set D^t_{sc}, where $\omega^t_d = \hat{\omega}^t_{s_d} \times \beta$, increases with rate $\hat{\mu}_{s_d}$. In this context, if there are two link-disjoint paths within maximum delay $\delta_{\max,d}$ for *s-c* demand d, each link in the paths must have sufficient capacity to carry the same amount of control packets ω^t_d.

4. **Controller placement**: the placement of each controller must ensure: (i) each controller is co-located with an *s*-switch, (ii) each *s*-switch is associated to exactly one controller, and (iii) a set of *s*-switches associated to a controller have a total number of *cpps* within the capacity θ of the controller.

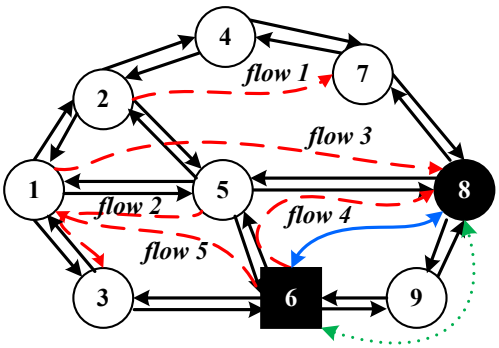
5.1.3. GMSU-3 Illustration

Figure 5.1 illustrates GMSU-3. The example legacy network shown in Figure 5.1a has $|V| = 9$ nodes and $|E| = 24$ directed links. Each link (u, v) has $b_{uv} = 2$ cables, meaning the network has in total 48 cables. Each cable has a capacity of $\gamma = 5$ units, MLU of $U_{\max} = 80\%$, and propagation delay of $\pi_{u,v} = 1$ second. Assume the network upgrade is planned for $T = 2$ stages with a total budget $B = \$80$; the budget at each stage is $B^1 = B^2 = \$40$. The upgrade cost for each switch u is $p^1_u = p^0_u = \$12$ and the deployment cost for each controller is $\hat{p}^1 = \hat{p}^0 = \$10$. Both costs are depreciated by rate $\rho = \hat{\rho} = 20\%$. Thus, the costs are decreased to $p^2_u = 9.6$ and $\hat{p}^2 = 8$, respectively. Each *s*-switch generates $\hat{\omega}^1_u = 10$ *cpps*, which is increased by $\hat{\mu_u} = 20\%$ at stage $t = 2$; the size is increased to $\hat{\omega}^2_u = 12$ *cpps*. Each controller can handle at maximum $\theta = 36$ *cpps*. Thus, one controller can process packets from a maximum of three *s*-switches. Further, this example considers

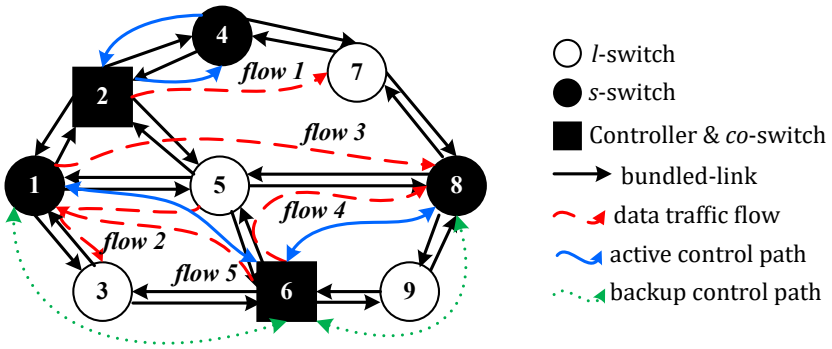
a uniform control packet size of $\beta = 0.01$ and assumes a delay tolerance of no longer than 10% of the original shortest path in $G^0(V, E)$, i.e., $\sigma = 1.1$. There are five data traffic demands each of which is routed via its shortest path; see dashed-red lines in Figure 5.1a. Here, the set of initial traffic contains only data demands, i.e., $D^0 = D^1_{\text{ss}} = D^0_{\text{ss}} = \{(s_1 = 2, \tau_1 = 7, \omega^0_1 = 5), (s_2 = 5, \tau_2 = 3, \omega^0_2 = 1), (s_3 = 1, \tau_3 = 8, \omega^0_3 = 5), (s_4 = 6, \tau_4 = 8, \omega^0_4 = 1), (s_5 = 6, \tau_5 = 1, \omega^0_5 = 1)\}$. The volume of each s-s demand increases with rate $\mu_d = 20\%$ at stage $t = 2$, i.e., $D^2_{\text{ss}} = \{(s_1 = 2, \tau_1 = 7, \omega^2_1 = 6), (5, 3, 1.2), (1, 8, 6), (6, 8, 1.2), (6, 1, 1.2)\}$.



(a) A legacy network.



(b) A *hybrid* SDN at stage 1.



(c) A *hybrid* SDN at stage 2.

Figure 5.1. An illustration of GMSU-3 for two stages. At stage 1, l-switch 6 and 8 are upgraded and associated to a controller 6 co-located with s-switch 6. At stage 2, new s-switch 1 is associated to controller 6, while new s-switches 2 and 4 are associated to new controller 2 that is co-located with s-switch 2.

In the *first* stage, i.e., $t = 1$, budget $B^1 = \$40$ is sufficient to upgrade two l-

switches, e.g., switches 6 and 8 and deploy one controller which is co-located with switch 6; see Figure 5.1b. Both switches are associated with the controller. Thus, we have $V^1 = \{6, 8\}$, $C^1 = \{6\}$. Both switches are associated with the controller, i.e., $A^1 = \{A^1_6 = \{6, 8\}\}$. The association set A^1_6 creates two s-c demands from both switches to the controller and other two demands from the controller to the switches, i.e., $D^1_{sc} = \{(s_6 = 6, \tau_6 = 6, \omega^1_6 = \hat{\omega}^1_6 \times \beta = 10 \times 0.01 = 0.1)$, $(s_7 = 6, \tau_7 = 6, \omega^1_7 = 0.1)$, $(s_8 = 8, \tau_8 = 6, \omega^1_8 = 0.1)$, $(s_9 = 6, \tau_9 = 8, \omega^1_9 = 0.1)\}$. Thus, we have $D^1 = D^0 \cup D^1_{sc}$. The first two s-c demands have $\hat{R}^1_6 = \hat{R}^1_7 = \{\}$ because controller $\mathcal{C}(6)$ is co-located with switch $\mathcal{S}(6)$. On the other hand, the other two demands have two link-disjoint paths $\hat{R}^1_8 = \{(8, 5, 6), (8, 9, 6)\}$ and $\hat{R}^1_9 = \{(6, 5, 8), (6, 9, 8)\}$. Paths $(8, 5, 6)$ and $(6, 5, 8)$, shown as solid-blue lines in Figure 5.1b, route control traffic from switch 8 to controller 6 and from 6 to 8. Paths $(8, 9, 6)$ and $(6, 9, 8)$, shown in dotted-green lines, serve as backups. Figure 5.1b also shows that demand $(s_5 = 6, \tau_5 = 1, \omega^1_5 = 1)$ is rerouted from the original shortest path $\mathcal{P}_{5,1} = (6, 3, 1)$ to $\hat{R}^1_5 = (6, 5, 1)$. The total volume on each link defines the required number of used cables n^t_{uv}. For example, link $(5, 6)$ has a total volume of $f^1_{(5,6)} = 1 + 1 + 0.1 + 0.1 = 2.2$ units and it requires only one cable to carry $f^1_{(5,6)}$. While link $(5, 8)$ needs both cables to carry traffic, the other two links $(6, 5)$ and $(8, 5)$ need only one cable. The four links are adjacent to s-switches 6 and 8. Thus, one unused cable of links $(5, 6)$, $(6, 5)$, and $(8, 5)$ can be turned off, whilst link $(5, 8)$ has no unused cable to power off. In contrast, there are eight links that are adjacent to both switches, i.e., $(3, 6)$, $(6, 3)$, $(6, 9)$, $(9, 6)$, $(7, 8)$, $(8, 7)$, $(8, 9)$, and $(9, 8)$, do not carry any traffic. Therefore, all the cables of these links can be switched-off. Thus, Figure 5.1b contains $3 + 2 \times 8 = 19$ unused cables that can be switched-off. Assuming each unused cable does not consume energy, while each other cable consumes the same amount of energy, the energy saving in Figure 5.1b is $\varepsilon^1 = 19/48 \times 100\% = 39.58\%$.

In the *second* stage, i.e., $t = 2$, adding the remaining budget from stage $t = 1$, i.e., $\Delta B = \$6$, to budget B^2, we have $B^2 = 40 + 6 = 46$. This budget is sufficient to upgrade three more l-switches, namely switches 1, 2 and 4, and deploy a new controller. Assume that the controller is co-located with s-switch 2 and manages switches 2 and 4, while switch 1 is managed by controller 6; see Figure 5.1c. As also shown in the figure, s-c traffic between switch 4 and controller 2 has no backup paths. This is because each path that is link-disjoint with paths $(2, 4)$ and $(4, 2)$ has delay exceeding the delay tolerance. The resulting hybrid SDN as shown in Figure 5.1c has 34 unused cables: two cables each in links $\{(1, 2)$, $(2, 1)$, $(3, 1)$, $(2, 5)$, $(5, 2)$, $(3, 6)$, $(6, 3)$, $(7, 4)$, $(6, 9)$, $(9, 6)$, $(7, 8)$, $(8, 7)$, $(8, 9)$, $(9, 8)\}$ and one cable each in links $\{(1, 3)$, $(5, 1)$, $(4, 2)$, $(5, 6)$, $(6, 5)$, $(8, 5)\}$. This equates to $\varepsilon^2 = 34/48 \times 100\% = 70.83\%$ in energy saving. The average energy saving over the two stages is $\varepsilon_2 = (39.58\% + 70.83\%)/2 = 55.2\%$.

5.1.4. GMSU-3 Mathematical Model

We formulate GMSU-3 as an MIP called MIP-3. It comprises of eight decision variables: (i) the number of *on*-cables n^t_{uv} for each link (u, v), (ii) the binary variable x^t_u which is an indicator of whether l-switch u is upgraded at stage t, (iii) the binary variable $\hat{x}^t_u$ set to one when a controller $\mathcal{C}(u)$ is deployed at node u at stage t, (iv) the binary variable $q^t_{u,a}$ to indicate whether s-switch u is associated to controller a at stage t, (v) the binary variable $\mathcal{I}_d$ set to one if $d \in \{1, \ldots, |D^t|\}$ is an s-s demand or an s-c demand, (vi) the binary variable $y^t_{d,uv,i}$ set to one if link (u, v) is included in path $\hat{R}^t_{d,i} \in \hat{R}^t_d$, (vii) the binary variable $y^t_{d,i}$ which is an indicator set to one if path $\hat{R}^t_{d,i} \in \hat{R}^t_d$ is an active path, and (viii) the total number of *on*-cables $n^t_{uv,\text{all}}$ of each link (u, v) when all selected paths for each s-c demand d carry the same amount of control packets ω^t_d. MIP-3's objective,

see (5.1), is to minimize the number of *on*-cables over T stages. This objective is equivalent to maximizing the energy saving as quantified by Eq. (2.2). Formally,

$$\begin{aligned} n^t_{uv}, x^t_u, \hat{x}^t_u, q^t_{v,a}, \mathcal{I}_d, y^t_{d,uv,i}, y^t_{d,i}, n^t_{uv,all} \quad & \min \quad \sum_{t=1}^{T} \sum_{(u,v)\in E} n^t_{uv} \\ \text{s.t.} \quad & (5.2) - (5.20). \end{aligned} \tag{5.1}$$

We have provided the code implementation of MIP-3 using Gurobi API for C++ [83] in Appendix C. The appendix also includes the result of solving MIP-3 using the small example in Figure 5.1. Note that constraints (5.7) and (5.18) contain multiplication of two decision variables. In the implementation, the constraints are handled by Gurobi solver using method GRBModel.AddQConstr() [91]. Further, constraints (5.2)–(5.20) are grouped into budget, controller placement, traffic routing, and domain of variables. The following sections provide their details.

5.1.4.1. Budget Constraints

Constraint (5.2) guarantees that the total cost to upgrade l-switches and to deploy controllers at each stage t does not exceed the maximum budget B^t. Constraints (5.3) and (5.4) ensure that each switch and each controller are deployed only once. Constraint (5.5) uses the function max(), which is the simplified form of its implementation using a set of linear equations (B.1) in Appendix B. The constraint enforces all cables in l-links to be turned on. Note that only unused cables of c-links can be turned off.

$$\sum_{u\in V}(p^t_u \times x^t_u + \hat{p}^t \times \hat{x}^t_u) \leq \sum_{k=1}^{t} B^k - \sum_{k=1}^{t-1}\sum_{u\in V}(p^k_u \times x^k_u + \hat{p}^k \times \hat{x}^k_u), \tag{5.2}$$

$$\sum_{t=1}^{T} x_u^t \leq 1, \tag{5.3}$$

$$\sum_{t=1}^{T} \hat{x}_u^t \leq 1, \tag{5.4}$$

$$n_{uv}^t = \max\left(n_{uv}^t, b_{uv} \times \left(1 - \sum_{k=1}^{t} x_u^k - \sum_{k=1}^{t} x_v^k\right)\right). \tag{5.5}$$

5.1.4.2. Controller Placement Constraints

Next, constraint (5.6) guarantees that each controller is co-located with an *s*-switch. Constraint (5.7) ensures that each controller is associated to its *co*-switch. In constraint (5.8), each *s*-switch must be associated to exactly one controller. As per constraints (5.9) and (5.10), each controller must be associated to at least one *s*-switch and the total generated control packets is no larger than the capacity of the controller. Constraint (5.11) ensures that an *s*-*c* association exists, i.e., $q_{u,a}^t = 1$, if and only if *l*-switch u is upgraded to an *s*-switch and a controller is deployed at node a.

$$\sum_{k=1}^{t} \hat{x}_u^k \leq \sum_{k=1}^{t} x_u^k, \tag{5.6}$$

$$\sum_{k=1}^{t} x_u^k \times \sum_{k=1}^{t} \hat{x}_u^k \leq q_{u,u}^t, \tag{5.7}$$

$$\sum_{a \in V} q_{u,a}^t = \sum_{k=1}^{t} x_u^k, \tag{5.8}$$

$$\sum_{u \in V} q_{u,a}^t \geq \sum_{k=1}^{t} \hat{x}_a^k, \tag{5.9}$$

$$\sum_{u \in V} (\hat{\omega}_u^t \times q_{u,a}^t) \leq \theta, \tag{5.10}$$

$$q_{u,a}^t \leq \min\left(\sum_{k=1}^{t} x_u^k, \sum_{k=1}^{t} \hat{x}_a^k\right). \tag{5.11}$$

Note that we use the function min() in (5.11) to simplify its real implementation that uses the set of two linear equations (B.2) discussed in Appendix B.

5.1.4.3. Traffic Routing constraints

Constraint (5.12) sets indicator $\mathcal{I}_d$ to one if $d \in \{1, \ldots, |D^t|\}$ is an *s-s* demand or an *s-c* demand; the latter applies only when *s*-switch s_d is associated to controller τ_d, for $s_d \neq \tau_d$. Otherwise, the indicator is set to zero. Recall that d is an *s-s* or *s-c* demand if $d \leq |D^t|$ or $d > |D^t|$, respectively. Constraints (5.13) to (5.18) are binding only if we have $\mathcal{I}_d = 1$. Let K_d be the number of paths used to route demand d. We set $K_d = 1$ for each *s-s* demand. For an *s-c* demand, we set $K_d = 2$ if set $\mathcal{P}_d$ contains two link-disjoint paths; otherwise, it is $K_d = 1$. Constraint (5.13) ensures flow conservation for both *s-s* and *s-c* traffic. For each *s-c* demand d with $K_d = 2$ paths, constraint (5.14) ensures the two paths are link-disjoint. Constraints (5.15) and (5.16) enforce every selected path i to meet its maximum delay and capacity $c_{uv} \times U_{\max}$ of each link (u, v) on the path, respectively. We use $n^t_{uv,\text{all}}$ to denote the required number of *on*-cables per link when all selected paths for each *s-c* demand d carry the same amount of control packets ω^t_d. Constraints (5.17) and (5.18) guarantee only one path, indicated by $y^t_{d,i} = 1$, that routes either a *s-s* or *s-c* demand. Constraint (5.19) limits the number of *on*-cables n^t_{uv} to the maximum *on*-cables $n^t_{uv,\text{all}}$, which is not larger than the bundle size of each link $(u, v) \in E$.

$$\mathcal{I}_d = \begin{cases} 1, & d \in \{1, \ldots, |D^t_{\text{ss}}|\} \\ q^t_{s_d,\tau_d} + q^t_{\tau_d,s_d}, & d > |D^t_{\text{ss}}| \end{cases}, \tag{5.12}$$

$$\sum_{i=1}^{K_d} \sum_{(u,v)\in E} y_{d,uv,i}^t - \sum_{i=1}^{K_d} \sum_{(v,u)\in E} y_{d,vu,i}^t = \begin{cases} \mathcal{I}_d, & u = s_d \\ -\mathcal{I}_d, & u = \tau_d \\ 0, & u \neq s_d, \tau_d \end{cases}, \tag{5.13}$$

$$\sum_{i=1}^{K_d} y_{d,uv,i}^t \leq \mathcal{I}_d, \tag{5.14}$$

$$\sum_{(u,v)\in E} (y_{d,uv,i}^t \times \pi_{uv}) \leq \delta_{\max,d} \times \mathcal{I}_d, \tag{5.15}$$

$$\sum_{d=1,\mathcal{I}_d=1}^{|D^t|} \sum_{i=1}^{K_d} (y_{d,uv,i}^t \times \omega_d^t) \leq (n_{uv,\text{all}}^t / b_{uv}) \times U_{\max} \times c_{uv}, \tag{5.16}$$

$$\sum_{i=1}^{K_d} y_{d,i}^t = \mathcal{I}_d, \tag{5.17}$$

$$\sum_{d=1,\mathcal{I}_d=1}^{|D^t|} \sum_{i=1}^{K_d} (y_{d,i}^t \times y_{d,uv,i}^t \times \omega_d^t) \leq (n_{uv}^t / b_{uv}) \times U_{\max} \times c_{uv}, \tag{5.18}$$

$$0 \leq n_{uv}^t \leq n_{uv,\text{all}}^t \leq b_{uv}, \tag{5.19}$$

5.1.4.4. Domain of Variables constraint

Finally, constraint (5.20) defines decision variables x_u^t, $\hat{x}_u^t$, $q_{u,a}^t$, $\mathcal{I}_d$, $y_{d,uv,i}^t$, and $y_{d,i}^t$ to be binary.

$$x_u^t, \hat{x}_u^t, q_{v,a}^t, \mathcal{I}_d, y_{d,uv,i}^t, y_{d,i}^t \in \{0,1\}. \tag{5.20}$$

Note that constraint (5.2) - (5.20), except (5.3) and (5.4), apply to each stage $t \in \{1,\ldots,T\}$. Constraint (5.3), (5.4), and (5.6) - (5.10) apply to all nodes in V. Constraint (5.5), (5.16), (5.18), and (5.19) exist for each link $(u,v) \in E$. Constraint (5.11) is for each node $u \in V$ and every node $a \in V$. Constraint (5.12) and (5.17) apply to each demand $d \in \{1,\ldots,|D^t|\}$. Finally, constraint (5.13)–(5.15) consider all demands. Further, constraint (5.13), (5.14), and (5.15) are

also evaluated for every node $u \in V$, each link $(u, v) \in E$, and each of K_d paths, respectively.

5.1.5. GMSU-3 Complexity Analysis

GMSU-3 is related to an NP-hard Multiple Knapsack Problem (MKP) [87]. Briefly, MKP considers T knapsacks and m items. Each knapsack has a maximum weight capacity, and each item has a profit and weight. The objective of MKP is to select T-disjoint subsets of items that maximize the total profit such that the total weight of each subset is no larger than its knapsack's capacity. Notice that the following parameters used in GMSU-3 and MKP, respectively, are equivalent: (i) T upgrade stages and T knapsacks, (ii) $|V|$ l-switches and m items, (iii) maximum budget B^t at each stage t and capacity of each knapsack, (iv) upgrade cost p_u^t and weight of each item, and (v) *off*-cables adjacent to the switch and profit of each item. Further, the objective of GMSU-3, i.e., selecting T-disjoint subsets of s-switches that maximize the number of *off*-cables, is equivalent to selecting T-disjoint subsets of items that maximize the total profit. However, GMSU-3 considers the following additional parameters that do not exist in MKP: (i) depreciation rate of each switch upgrade cost, (ii) controller deployment cost, its location, and its association to s-switches, and (iii) paths to route s-s and s-c demands. Thus, GMSU-3 is at least as hard as MKP. The next section describes the heuristic solution for GMSU-3.

5.2. Heuristic Solution

This section outlines a heuristic solution called **GU-3**. Specifically, Section 5.2.1 contains the details of GU-3, followed by Section 5.2.3 that presents the time complexity analysis of GU-3.

5.2.1. Details of GU-3

GU-3 consists of three parts: 1) routing initialization, 2) network upgrade, and 3) traffic rerouting; see Algorithm 5.1. Part 1 generates the initial routing for all *s*-*s* traffic demands in set D^0_{ss}. It uses the said routing to obtain the total number of *off*-cables that are adjacent to each *l*-switch $u \in V$, denoted by w_u. Each switch's *off*-cables, i.e., w_u, is the input to Part 2. Specifically, Part 2 determines the deployment of *s*-switches and controllers that maximize the number of *off*-cables over T stages. Part 3 increases, if possible, the number of *off*-cables obtained in Part 2. The detail of Part 1, Part 2, and Part 3 are given in Section 5.2.1.1, Section 5.2.1.2, and Section 5.2.1.3, respectively.

Algorithm 5.1. : **GU-3 – heuristic solution for GMSU-3**

Input: $G^0(V, E)$, T, B, D^0_{ss}, p^0_u, $U_{\max}$, μ_d, ρ, σ
Output: V^t, C^t, A^t, D^t, R^t, n^t_{uv}, ε^t

▷ *Part 1: routing initialization*
for $(d \in \{1, \ldots, |D^T_{\text{ss}}|\})$ **do**
  Generate set $\mathcal{P}_d$
  $R^0 = R^0 \cup \mathcal{P}_{d,1}$
  $f^T_{uv} = f^T_{uv} + \omega^T_d$ for each $(u, v) \in R^0_d \in R^0$
end for
$n^T_{uv} = \lceil f^T_{uv}/(\gamma \times U_{\max}) \rceil$ for each $(u, v) \in E$
Compute w_u for each $u \in V$ using Eq. (3.2)
▷ *Part 2: network upgrade*
for $(t \in \{1, \ldots, T\})$ **do**
  $\{V^t, C^t, A^t, D^t, R^t, \Delta B^t, L\} = \textbf{Deployment}(V^{t-1}, C^{t-1}, A^{t-1}, D^{t-1}, R^{t-1}, B^t)$
  $B^{t+1} = B^{t+1} + \Delta B^t$
▷ *Part 3: traffic rerouting*
  $R^t = \textbf{GTE-SC}(V^t, R^t, L)$
  Compute ε^t using Eq. (2.1)
  $\varepsilon_T = \varepsilon_T + \varepsilon^t/T$
end for

5.2.1.1. Part 1 – Routing Initialization

Part 1 of Algorithm 5.1 initially routes each *s*-*s* demand as per OSPF. It then uses the initial route of all *s*-*s* demands to calculate each switch's total unused cables.

More specifically, *Line 2* of GU-3 uses Yen's algorithm [88] to generate the first k shortest paths $\mathcal{P}_d$ for each s-s demand $d \in \{1, \ldots, |D_{\text{ss}}^T|\}$ in increasing order of their delay, i.e., $\mathcal{P}_d = (\mathcal{P}_{d,1}, \mathcal{P}_{d,2}, \ldots, \mathcal{P}_{d,k})$, where path $\mathcal{P}_{d,1}$ has the shortest delay. *Line 3* adds the shortest path $\mathcal{P}_{d,1}$ of each s-s demand d into set R^0. *Line 4* computes the largest required capacity, denoted by f_{uv}^T for each link $(u, v) \in E$, when each demand d is routed by the OSPF protocol. Recall that the traffic at the last stage, i.e., ω_d^T, has the maximum volume. *Line 6* then calculates the number of *on*-cables n_{uv}^T of link (u, v) based on f_{uv}^T. *Line 7* computes the total number of *off*-cables w_u that are adjacent to each switch $u \in V$ using Eq. (3.2). The motivation for using $b_{uv} - n_{uv}^T$ in Eq. (3.2) is following Observation 3.1.

5.2.1.2. Part 2 – Network Upgrade

For each stage t, *Line 9* of Algorithm 5.1 calls **Deployment**() to upgrade a set of l-switches into s-switches that can turn off the maximum possible number of unused cables, and associate each of them to a controller. The function **Deployment**() contains three main steps: (i) find a *feasible* controller, see Definition 5.1 below, from set C^t for each candidate l-switch upgrade, (ii) deploy a new controller if Step (i) fails to find a *feasible* controller, (iii) upgrade the switch and associate it with the *feasible* controller found in Step (i) or (ii). Note that Step (ii) assumes there is sufficient budget to upgrade one l-switch and deploy one controller. Further, in Step (ii), if the deployed controller is co-located with an s-switch v, the function disassociates v from its previous controller. Finally, if Step (iii) fails, the function considers the next candidate l-switch.

Definition 5.1. A controller a is a *feasible* controller for an s-switch u if (i) it has sufficient capacity to manage s-switch u, i.e., $\sum_{v \in A_a^t} \hat{\omega}_v^T + \hat{\omega}_u^T \leq \theta$, and (ii) there is at least a *feasible* path, see Definition 5.2, from u to a and from a to u.

Definition 5.2. A path $\mathcal{P}_{d,i} \in \mathcal{P}_d$ for s-c traffic demand d between switch u and controller a is *feasible* if adding traffic volume ω_d^T to the current total volume f_{uv}^T of each link $(u,v) \in \mathcal{P}_{d,i}$ does not exceed its MLU, i.e., $f_{uv}^T + \omega_d^T \leq c_{uv} \times U_{\max}$.

Note that Definition 5.1 uses the $\hat{\omega}_u^T$ value of each switch u to ensure that the association between s-switch u and controller a is feasible at each stage $t \leq T$. A controller is always feasible for its co-located switch. Further, we use f_{uv}^T and ω_d^T in Definition 5.2 to ensure that each *feasible* path can carry traffic demands at any stage $t \leq T$. Finally, if there are two *feasible* paths $\mathcal{P}_{d,i}, \mathcal{P}_{d,j} \in \mathcal{P}_d$ that are link-disjoint, they must be used to route the demand. Thus, set $\hat{R}_d^t$ may contain only one *feasible* path or two link-disjoint *feasible* paths.

Algorithm 5.2. : **Deployment()**

Input: V^{t-1}, C^{t-1}, A^{t-1}, D^{t-1}, R^{t-1}, B^t
Output: V^t, C^t, A^t, D^t, R^t, ΔB^t

1: $V^t = V^{t-1}$, $C^t = C^{t-1}$, and $A^t = A^{t-1}$
2: $D^t = D^{t-1}$, $R^t = R^{t-1}$, and $X = V - V^t$
3: **for** (each switch $u \in X$ that has $p_u^t \leq B^t$, $w_u > 0$, and $\max_{u \in X}\{w_u/p_u^t\}$) **do**
4: **if** (**SelectC**($\mathcal{S}(u)$) finds a *feasible* controller $a \in C^t$) **then**
5: $\{V^t, X, \Delta B^t\} =$ **UpgradeS**($\mathcal{S}(u)$)
6: $\{A^t, D^t, R^t\} =$ **Associate**($\mathcal{S}(u), \mathcal{C}(a)$)
7: **else if** (**LocateC**(B^t, u) returns a 2-tuple (a, v)) **then**
8: $\{C^t, \Delta B^t\} =$ **DeployC**($\mathcal{C}(a)$).
9: **if** (node a contains an s-switch) **then**
10: Find $x \in C^t$ for $a \in A_x^t$
11: $\{A^t, D^t, R^t\} =$ **Disassociate**($\mathcal{S}(a), \mathcal{C}(x)$)
12: $\{A^t, D^t, R^t\} =$ **Associate**($\mathcal{S}(a), \mathcal{C}(a)$)
13: **end if**
14: $\{V^t, X, \Delta B^t\} =$ **UpgradeS**($\mathcal{S}(v)$)
15: $\{A^t, D^t, R^t\} =$ **Associate**($\mathcal{S}(v), \mathcal{C}(a)$)
16: **else** $X = X - u$
17: **end if**
18: **end for**
19: $n_{uv}^T = \lceil f_{uv}^T/(\gamma \times U_{\max}) \rceil$ for each $(u,v) \in E$

Algorithm 5.2 shows the details of the function **Deployment**(). First, *Line 1* of Algorithm 5.2 initializes V^t, C^t, and A^t with the set of s-switches, controllers, and associations from stage $t-1$, respectively. Note that V^0, C^0, A^0, and D_{sc}^0 are

empty sets. *Line 2* initializes set D^t with all traffic demands and sets R^t with the paths used to route the demands at stage $t-1$. Recall that D^0 contains only *s-s* traffic demands. Further, *Line 2* uses set X to store all *l*-switches yet to be upgraded.

Next, *Line 3* of Algorithm 5.2 selects one *candidate* *l*-switch u to be upgraded at stage t. The switch must satisfy Definition 3.1 in which its upgrade cost is within budget, i.e., $p_u^t \leq B^t$, and it has the total unused cables $w_u > 0$. Further, it must have the largest w_u/p_u^t. For each selected candidate *l*-switch u, *Line 4* uses **SelectC**($\mathcal{S}(u)$) to find a *feasible* controller $a \in C^t$ that can be associated with switch u. Specifically, **SelectC**($\mathcal{S}(u)$) prioritizes a *feasible* controller a with two link-disjoint *feasible* paths between switch u and controller a. Function **Deployment**() considers whether **SelectC**() finds a *feasible* controller.

If **SelectC**() finds a *feasible* controller $a \in C^t$ for switch u, *Line 5* of Algorithm 5.2 first uses **UpgradeS**($\mathcal{S}(u)$) to upgrade switch u. It performs the following five steps: (i) add u to set V^t, (ii) remove u from set X, (iii) reduce budget B^t by the upgrade cost of switch u, i.e., $B^t = B^t - p_u^t$, (iv) reduce the number of *off*-cables w_v by the number of *off*-cables that are adjacent to u and v, i.e., $w_v = w_v - (b_{uv} - n_{uv}^T)$, for each *l*-switch neighbor $v \in V - V^t$ of u, and (v) put every *c*-link (u, v) with non-zero traffic flow in a set L. *Line 6* then uses **Associate**($\mathcal{S}(u), \mathcal{C}(a)$) to associate the upgraded switch u to its *feasible* controller $a \in C^t$. This function performs the following five tasks: (i) add *s*-switch u into set A_a^t, (ii) generate two *s-c* demands d and $d+1$, i.e., (u, a, ω_d^t) and (a, u, ω_{d+1}^t), respectively, and add them into set D^t, (iii) add their respective *feasible* paths $\hat{R}_d^t$ and $\hat{R}_{d+1}^t$ to set R^t, (iv) determine *active* path $\hat{R}_{d,i}^t \in \hat{R}_d^t$ and $\hat{R}_{d+1,i}^t \in \hat{R}_{d+1}^t$, and (v) increase traffic volume f_{uv}^T and $f_{u'v'}^T$ of each link (u, v) in active path $\hat{R}_{d,i}^t \in \hat{R}_d^t$ and each link (u', v') in active path $\hat{R}_{d+1,i}^t \in \hat{R}_{d+1}^t$ by ω_d^T and ω_{d+1}^T, i.e., $f_{uv}^T = f_{uv}^T + \omega_d^T$ and $f_{u'v'}^T = f_{u'v'}^T + \omega_{d+1}^T$, respectively. In

task (iv), each *active* path is the path that uses the smallest additional *on*-cables, i.e., $\min_{\hat{R}^t_{j,i} \in \hat{R}^t_j} \{\sum_{(u,v) \in \hat{R}^t_{j,i}} (\lceil f^T_{uv}/(\gamma \times U_{\max}) \rceil - \lceil (f^T_{uv} - \omega^T_j)/(\gamma \times U_{\max}) \rceil)\}$ for $j \in \{d, d+1\}$ and $|\hat{R}^t_j| = 2$. Further, task (v) uses ω^T_d and f^T_{uv} to ensure that each *active* path can carry *s-c* traffic demands at any stage $t \leq T$. Note that tasks (iv) and (v) are performed only if switch u and controller a are co-located at different nodes, i.e., $u \neq a$, $\hat{R}^t_d \neq \{\}$, and $\hat{R}^t_{d+1} \neq \{\}$.

If **SelectC**() fails to find a feasible controller, *Line 7* of Algorithm 5.2 uses **LocateC**(B^t, u) to select one node in set $\{V - C^t\}$ to deploy a new controller. More specifically, **LocateC**() returns a 2-tuple (a, v), where a is the node at which the new controller is deployed and v is the node at which an l-switch is to be upgraded. To determine whether to place a controller at node a, it uses *one* of the following options, which it iterates through in order: (a) at node $a \neq u$ that has an s-switch, (b) at node $a = u$, or (c) at node $a \neq u$ that has an l-switch.

LocateC() returns (a, u), (u, u) or (a, a) for option (a), (b) or (c), respectively. Thus, for option (a) and (b), the available budget B^t must be sufficient to deploy one controller and an s-switch at node u, i.e., $B^t \geq \hat{p}^t + p^t_u$. For option (c), budget B^t is used to deploy an s-switch and a controller at node a, and thus, we have $B^t \geq \hat{p}^t + p^t_a$. It means that for option (c), the l-switch u is not upgraded. Recall that each controller must be co-located with an s-switch. Except for option (b), **LocateC**() greedily selects a controller a that has the largest ratio w_a/p^t_a and the largest number of two link-disjoint paths with all l-switches in $V - V^t$. In addition, for option (a), the function also selects a controller a that has two link-disjoint paths with l-switch u. It selects a controller randomly for any remaining tie. **LocateC**() returns False when none of the three options (a)–(c) are feasible. In this case, *Line 16* of **Deployment**() removes u from set X so that *Line 3* considers the next candidate l-switch in X.

Next, if **LocateC**() successfully returns (a, u), (u, u) or (a, a) for options (a),

(b), or (c), respectively, *Lines 8–15* aim to complete the following tasks:

- **A 2-tuple** (a, u): (i) use function **DeployC**$(\mathcal{C}(a))$ in *Line 8* to deploy controller a, (ii) use *Line 10* to find s-switch a's previous *feasible* controller $x \in C^t$ where $a \in A_x^t$, (iii) use function **Disassociate**$(\mathcal{S}(a), \mathcal{C}(x))$ in *Line 11* to dis-associate s-switch a from controller x, (iv) associate s-switch $\mathcal{S}(a)$ with controller $\mathcal{C}(a)$ using **Associate**$(\mathcal{S}(a), \mathcal{C}(a))$ in *Line 12*, (v) upgrade l-switch u using **UpgradeS**$(\mathcal{S}(u))$ in *Line 14*, and (vi) associate s-switch u with controller a using **Associate**$(\mathcal{S}(u), \mathcal{C}(a))$ in *Line 15*.

- **A 2-tuple** (u, u): (i) use function **DeployC**$(\mathcal{C}(u))$ in *Line 8* to deploy controller u, (ii) use function **UpgradeS**$(\mathcal{S}(u))$ in *Line 14* to upgrade l-switch u, and (iii) use function **Associate**$(\mathcal{S}(u), \mathcal{C}(u))$ in *Line 15* to associate s-switch $\mathcal{S}(u)$ with controller $\mathcal{C}(u)$.

- **A 2-tuple** (a, a): (i) use function **DeployC**$(\mathcal{C}(a))$ in *Line 8* to deploy controller a, (ii) use function **UpgradeS**$(\mathcal{S}(a))$ in *Line 14* to upgrade l-switch a, and (iii) use function **Associate**$(\mathcal{S}(a), \mathcal{C}(a))$ in *Line 15* to associate s-switch $\mathcal{S}(a)$ with controller $\mathcal{C}(a)$.

The details of **DeployC**() and **Disassociate**() are as follows. **DeployC**$(\mathcal{C}(a))$ performs the following two steps: (i) add controller a into set C^t, and (ii) reduce budget B^t by deployment cost $\hat{p}^t$. **Disassociate**$(\mathcal{S}(a), \mathcal{C}(x))$ carries out the following three tasks: (i) remove any s-c demand d generated for s-switch a's previous association with controller x from set D^t, (ii) decrease the total volume f_{uv}^T of every link (u, v) in active path $\hat{R}_{d,i}^t \in \hat{R}_d^t$ by the demand's volume ω_d^T, and (iii) remove switch a from set A_x^t.

Lines 3–18 are repeated until (i) all switches in X have been upgraded in *Line 5* or *Line 14* or removed in *Line 16*, i.e., $X = \{\}$, or (ii) each remaining

switch $u \in X$ has cost larger than the remaining budget B^t, i.e., $p_u^t > \Delta B^t$, or (iii) there are no unused cables to turn-off, i.e., $w_u = 0$. *Line 19* then computes the number of *on*-cables n_{uv}^T according to the current volume f_{uv}^T for each link (u, v). *Line 10* of GU-3 adds the budget ΔB^t to B^{t+1}.

5.2.1.3. Part 3 –Traffic Rerouting

Line 11 of GU-3 uses function **GTE-SC**(V^t, R^t, L) to increase the number of unused cables that can be powered off by s-switches. The function adopts traffic rerouting in Algorithm 3.3. Recall that Algorithm 3.3 performs four main steps:

(a) Find a c-link $(u, v) \in L$ with a cable that has the smallest used capacity $r_{uv} = (f_{uv}^T - \gamma \times U_{\max} \times \lfloor f_{uv}^T / \gamma \times U_{\max} \rfloor)$.

(b) Store each path $\hat{R}_{d,i}^t \in \hat{R}_d^t$ that passes link (u, v) into set Q_{uv} and switch off one of the link's *on*-cables.

(c) Find *routable path* $\mathcal{P}_{d,j}$ in set $\{\mathcal{P}_d - R_{d,i}^t\}$. Briefly, path $\mathcal{P}_{d,j}$ is called *routable* if each link (u, v) in the path has sufficient capacity to carry an additional traffic volume of ω_d^T, i.e., $(f_{uv}^T + \omega_d^T) \leq (\gamma \times n_{uv}^T \times U_{\max})$.

(d) Recalculate the *off*-cables w_u for each l-switch u in set $V - V^t$.

Note that Step (b) and (c) are repeated for each c-link $(u, v) \in L$. Further, Step (c) is repeated for each path $\hat{R}_{d,i}^t \in Q_{uv}$ or until all paths that use the powered-off cable in c-link (u, v) are rerouted onto their alternative path, i.e., $r_{uv} = 0$.

Function **GTE-SC**() extends Step (c) to maintain the existing two link-disjoint paths for any s-c traffic demand d, i.e., $|\hat{R}_d^t| = 2$ paths. Here, Step (c) considers the following three cases when finding the *routable* path $\mathcal{P}_{d,j}$: (i) path $\mathcal{P}_{d,j}$ is the backup path in set $\hat{R}_d^t$, (ii) path $\mathcal{P}_{d,j}$ is link-disjoint with the two paths in set $\hat{R}_d^t$, i.e., $\mathcal{P}_{d,j} \notin \hat{R}_d^t$, or (iii) path $\mathcal{P}_{d,j}$ is not in set $\hat{R}_d^t$ and $|\hat{R}_d^t| = 1$

path. For case (i), the extended Step (c) sets path $\hat{R}^t_{d,i}$ as backup while path $\mathcal{P}_{d,j}$ as *active*. On the other hand, for cases (ii) and (iii), it replaces path $\hat{R}^t_{d,i}$ with path $\mathcal{P}_{d,j}$. For any case, the updated Step (c) decreases and increases the total volume f^T_{uv} for each link (u, v) in path $\hat{R}^t_{d,i}$ and $\mathcal{P}_{d,j}$, respectively, by ω^T_d. *Line 12* of Algorithm 5.1 then computes the energy saving ε^t using Eq. (2.1). Finally, *Line 13* computes the average energy saving over T stages.

5.2.2. Example

This example considers a legacy network in Figure 5.1a and their input parameters described in Section 5.1.3. In Part 1 of GU-3, the routing initialization for *s-s* demands in D^2_{ss}, shown in Figure 5.1a, produces the highest total traffic volume f^2_{uv} for each link. For example, link $(5, 8)$ carries traffic *flow 3* and *flow 4* with maximum size of 6 and 1.2 units, respectively. Thus, the total volume $f^2_{(5,8)} = \omega^T_3 + \omega^T_4 = 6 + 1.2 = 7.2$ units. The link requires $n^2_{(5,8)} = 2$ *on*-cables out of its two cables. The number of *on*-cables of each link is then used to calculate the total number of *off*-cables w_u for each switch: $w_1 = (b_{(1,2)} - n^2_{(1,2)}) + (b_{(2,1)} - n^2_{(2,1)}) + (b_{(1,5)} - n^2_{(1,5)}) + (b_{(5,1)} - n^2_{(5,1)}) + (b_{(1,3)} - n^2_{(1,3)}) + (b_{(3,1)} - n^2_{(3,1)}) = 2+2+0+1+2+1 = 8$ cables, $w_2 = 10$, $w_3 = 7$, $w_4 = 4$, $w_5 = 9$, $w_6 = 11$, $w_7 = 4$, $w_8 = 10$, and $w_9 = 8$.

For the first stage, initially, $V^0 = \{\}$, $C^0 = \{\}$, $A^0 = \{\}$, $D^1 = D^0$, $R^1 = R^0 = \{\{(2, 4, 7)\}, \{(5, 1, 3)\}, \{(1, 5, 8)\}, \{(6, 5, 8)\}, \{(6, 3, 1)\}$, and $X = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$. The first iteration of *Lines 3–18* of Part 2 selects a switch at node 6 or $\mathcal{S}(6)$ because it has the highest ratio $w_6/p^1_6 = 11/10 = 1.1$. It then deploys the first controller at node 6 or $\mathcal{C}(6)$, upgrades switch $\mathcal{S}(6)$, and associates switch $\mathcal{S}(6)$ with controller $\mathcal{C}(6)$. The total number of unused cables of switch 6's neighbors, i.e., switches 3, 5, and 9, becomes $w_3 = 7 - 4 = 3$, $w_5 = 9 - 2 = 7$, and $w_9 = 8 - 4 = 4$ cables. The next iteration of Part 2 upgrades switch $\mathcal{S}(8)$ and

associates $\mathcal{S}(8)$ with controller $\mathcal{C}(6)$. At stage $t = 1$, Part 2 of GU-3 produces the same results, for sets V^t, C^t, A^t, D^t, and R^t, as described in Section 5.1.3. Part 3 of GU-3 increases the number of unused cables in link $(6, 3)$ by rerouting *s-s* traffic *flow 5* from path $(6, 3, 1)$ to $(6, 5, 1)$. Thus, the upgraded network in Figure 5.1b saves $\varepsilon^1 = 39.58\%$ of energy.

The total number of unused cables of the remaining l-switches are $w_1 = 8$, $w_2 = 10$, $w_3 = 3$, $w_4 = 4$, $w_5 = 7$, $w_7 = 2$, and $w_9 = 0$. Thus, for stage 2, the first upgraded switch is $\mathcal{S}(2)$ and it is associated to controller $\mathcal{C}(6)$. The next upgrade is for switch $\mathcal{S}(1)$ and a new controller is required because controller $\mathcal{C}(6)$ has no remaining capacity. Switches $\mathcal{S}(2)$ and $\mathcal{S}(8)$ have the same total unused cables $w_2 = w_8$ but switch $\mathcal{S}(2)$ has two link-disjoint paths to all l-switches except switch 4; switch $\mathcal{S}(8)$ has no two link-disjoint paths with l-switches 1, 5, 7, and 9. Thus, the second controller $\mathcal{C}(2)$ is deployed at node 2 and switch $\mathcal{S}(2)$ is disassociated from controller $\mathcal{C}(6)$ and associated with controller $\mathcal{C}(2)$. Switch $\mathcal{S}(1)$ is then associated to controller $\mathcal{C}(6)$. The last upgraded switch is $\mathcal{S}(4)$ and it is associated to $\mathcal{C}(2)$ which is the only existing controller with sufficient capacity. Figure 5.1c shows all active and backup paths selected to route all *s-c* traffic demands. At this final stage $t = 2$, GU-3 also produces the same output described in Section 5.1.3. It saves $\varepsilon^2 = 70.83\%$ of energy with an overall energy saving over two stages of $\varepsilon_2 = 55.2\%$.

5.2.3. Algorithm Analysis

We conclude this Section with the run-time complexity of GU-3.

Proposition 5.1. *The time complexity of GU-3 is* $O(|V|^2|E|^2)$.

Proof. The routing initialization in *Lines 1–7* of Algorithm 3.1 has the worst case time complexity of $O(k|D^0_{ss}||V|(|E| + |V|log|V|))$. Here, *Line 2* of Algo-

rithm 5.1 uses Yen's algorithm [88] that takes $O(k|D^0_{ss}||V|(|E| + |V|log|V|))$. *Lines 3–4* require $O(1)$ and $O(|D^0_{ss}||E|)$, respectively. Both Lines 6 and 7 have the worst case time complexity of $O(|E|)$. The function **Deployment**() in *Line 9* of Algorithm 5.1 requires $O(k|V|^2|E|)$ because (i) the initialization of all sets in *Lines 1–2* of Algorithm 5.2 needs $O(|D^T|)$, (ii) functions **SelectC**(), **UpgradeS**(), **Associate**(), **LocateC**(), and **DeployC**() require $O(k|V||E|)$, $O(|V||E|)$, $O(k|V||E|)$, $O(k|V|^2|E|)$, and $O(1)$, respectively, (iii) *Line 10* of Algorithm 5.2 takes $O(|V|)$, (iv) function **Disassociate**() needs $O(|D^T_{sc}| + |E|)$, and (v) *Line 19* takes $O(|E|)$. *Line 10* of Algorithm 5.1 requires $O(1)$. The function **GTE-SC**() in *Line 11* of Algorithm 5.1 takes $O(k|D^T||E|^2)$. More specifically, Step (a) of **GTE-SC**() needs $O(E)$ to find a c-link with the smallest used capacity. Step (b) requires $O(|D^T||E|)$ to check every path in set R^t that uses the c-link. Step (c) takes $O(k|E|)$ to find a *routable* path among the generated k shortest path for each demand. Step (b) is repeated $O(|E|)$ times and thus, it requires $O(|D^T||E|^2)$. On the other hand, Step (c) is repeated $O(|E|)$ times, each of which is repeated $O(|D^T|)$ times. Thus, Step (c) takes $O(k|D^T||E|^2)$. Step (d) requires $O(|V|)$. *Line 12* of Algorithm 5.1 needs $O(|E|)$ while *Line 13* requires $O(1)$. Repeating *Lines 8–14* of Algorithm 5.1 in T times, *Lines 9–13* require $O(Tk|V|^2|E| + Tk|D^T||E|^2)$. Thus, GU-3 has the worst case run-time of $O(k|D^0_{ss}||V|(|E| + |V|log|V|) + Tk|V|^2|E| + Tk|D^T||E|^2) = O(Tk|D^T||E|^2) = O(|V|^2|E|^2)$. Note that we consider $|D^0_{ss}| \leq |V|^2$, $|D^T_{sc}| \leq |V|^2$, $|D^T| \leq 2|V|^2$, $|E| \leq |V|^2$, and parameters k and T are constants. □

5.3. Performance Evaluation

We have implemented GU-3 in C++ and used Gurobi [83] to solve MIP-3. All experiments are conducted on the platform described in Section 2.7.3. Further,

each experiment uses the five network topologies shown in Table 2.3 of Section 2.7.

In addition to parameter values outlined in Table 2.4 of Section 2.7, every experiment in this section uses the following additional parameter values. For each demand d, Yen's algorithm [88] is used to generate up to $k = 10$ paths within maximum delay $\delta_{\max,d}$. For each switch, its initial upgrade *cost* and *cpps*, is randomly assigned from one of the following (*cost*, *cpps*) values: ($50K, 100K), ($100K, 200K), and ($150K, 300K). More specifically, for each switch, we draw a random number from the Normal distribution $\mathcal{N}(2, 0.5)$ and round the number to the nearest integer. A switch u that draws a value of one, two or three is assigned ($p_u^0 =$ $50K, $\hat{\omega}_u^1 =$ 100K *cpps*), (100K$, 200K) or ($150K, 300K), respectively. Each switch's *cpps* increases at a rate of $\hat{\mu}_u = 22\%$ per stage. Control packets have size $\beta = 160$ bytes [68]. For each controller, we set its capacity θ to 7800K *cpps* and use an initial deployment cost of $\hat{p}^0 =$ $25K. Following [68], the capacity $\theta =$ 7800K *cpps* is calculated from the controller access bandwidth of 10 Gbps and control packet size $\beta = 160$ bytes, i.e., $\theta =$ 10 Gbps/(160 bytes $\times$ 8 bits). Similar to the work in [69], the cost to deploy a controller is cheaper than the upgrade cost of an s-switch. Lastly, a depreciation cost rate of $\hat{\rho} = 40\%$ is used for each controller.

The following sections are organized as follows. First, Section 4.3.1 evaluates the scalability of MIP-3 and GU-3 solutions in terms of their running time in CPU seconds. Section 4.3.2 and Section 4.3.3 then analyze the effect of increasing budgets on energy saving and the total number of deployed s-switches and controllers over the T stages. Next, Section 5.3.4 studies the impact of controller placement on network performances, e.g., energy saving and path delay. Finally, Section 5.3.5 compares the performances of GU-3 against an existing technique, i.e., the Improved Genetic Controller Placement Algorithm (IGCPA) [19].

5.3.1. Running Time

This experiment studies the run time performance of MIP-3 and GU-3. It uses a budget of $B = 1.2$M and $T = 3$ stages. As shown in Table 5.2, the run time of MIP-3 is significantly longer than that of GU-3 for the tested five networks. MIP-3 requires 14.46, 27542.88, 73670.16, and 111293.31 seconds, while GU-3 takes only 0.17, 1.28, 21.05, and 256.34 seconds to produce the results for Abilene, GÉANT, DFN, and Deltacom, respectively. These results show that on average 80.44% of GU-3's running time is used for generating k alternative paths using Yen's algorithm [88]. Further, MIP-3 failed to produce results for TATA because the optimizer ran out of memory. Thus, the performance comparisons between MIP-3 and GU-3 in the following sections use only Abilene, GÉANT, DFN, and Deltacom.

Table 5.2. Running time of MIP-3 and GU-3.

Name	$\|V\|$	$\|E\|$	$\|D^0_{ss}\|$	**Running Time (in CPU seconds)**	
				MIP-3	**GU-3**
Abilene	12	30	132	14.46	0.17
GÉANT	23	74	466	27542.88	1.28
DFN	58	174	3306	73670.16	21.047
Deltacom	113	322	12656	111293.31	256.34
TATA	145	372	20880	N/A	413.27

5.3.2. Impact of Increasing Budget

This experiment studies the impact of budget on energy saving, the number of upgraded switches, and the numbers of controllers. The following Sections 5.3.2.1, 5.3.2.2, and 5.3.2.3 provide the details. Here, we use five budget values, i.e., $B \in \{\$400\text{K}, \$600\text{K}, \$800\text{K}, \$1\text{M}, \$1.2\text{M}\}$, and $T = 3$ stages.

5.3.2.1. Energy Saving

Referring to Figure 5.2, we see that MIP-3 and GU-3 produce more energy saving ε_T for larger budget values. The energy saving of GU-3 for Abilene, GÉANT, DFN, and Deltacom is only up to 1.67%, 2.62%, 4.71%, and 3.91% off from the optimal energy saving computed by MIP-3. For example, using a budget of $B =$ \$400K, MIP-3 and GU-3 respectively produce energy saving ε_T of 52.63% and 51.75% for Abilene. Their energy saving increases to 68.42% and 67.54% when using a larger budget of $B =$ \$1.2M. The increasing energy saving in Figure 5.2 is reasonable because a large budget allows MIP-3 and GU-3 to upgrade more l-switches.

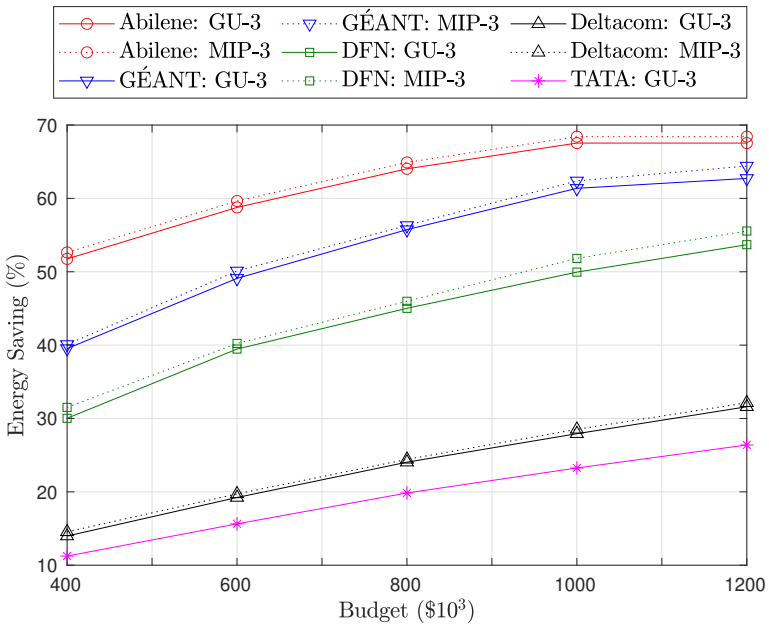


Figure 5.2. Energy saving ε_T of MIP-3 and GU-3 for various budget B values.

5.3.2.2. Number of Upgraded Switches

Figure 5.3 shows that larger budget B value allows MIP-3 and GU-3 to upgrade more switches. As an example, for Abilene, GÉANT, DFN, and Deltacom, MIP-3 increases the percentage of upgraded l-switches $\mathcal{V}^T$ from 58% to 91.67%, 34.78% to 73.91%, 13.79% to 39.66%, and 9.74% to 24.78%, respectively when budget B is increased from \$400K to \$1.2M. However, for Abilene, when budget B is increased from \$600K to \$800K, MIP-3 decreases the percentage $\mathcal{V}^T$ from 75% to 50%. Recall that MIP-3 and GU-3 aim to maximize energy saving ε_T, and thus larger budget B value *does not always* increase the number of upgraded switches. One possible reason is because maximizing ε_T may require the algorithms to upgrade more l-switches at earlier stages. In this case, there is less remaining budget to upgrade switches in the later stages. Thus, in total, there are fewer number of switches that can be upgraded. Recall that the switch upgrade cost at later stages is cheaper than the cost at earlier stages.

As shown in Figure 5.3, MIP-3 upgrades more switches than GU-3 for Abilene for all budget values except for B = \$800K and GÉANT from budget value B = \$1M. The reason is because MIP-3 deploys s-switches and co-locates controllers to the s-switches as many as possible so that the number of cables used to route s-c demands can be minimized. On the other hand, GU-3 upgrades similar number of switches as compared to MIP-3 for DFN and Deltacom. Further, GU-3 does not upgrade more switches for Abilene and GÉANT even when it has remaining budget. The percentage $\mathcal{V}^T$ for both networks remains constant at 50% and 56.52% when budget B is increased from \$400K and \$800K to \$1.2M, respectively. The reason is because GU-3 will upgrade a switch only if it can turn off unused cables. For both networks, the respective budget B value \$400K and \$800K is sufficient to upgrade all l-switches that have unused cables.

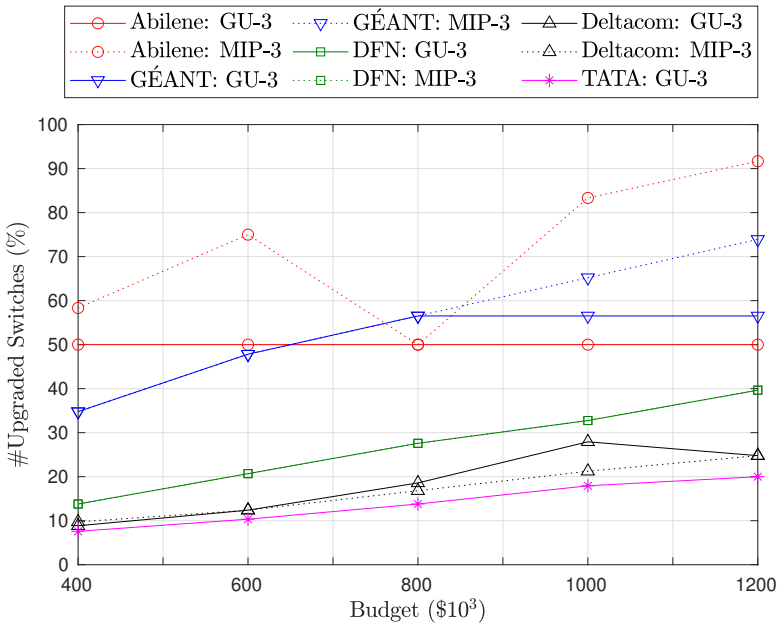


Figure 5.3. The percentage of upgraded l-switches $\mathcal{V}^T$ for MIP-3 and GU-3 given various budget B values.

5.3.2.3. Number of Deployed Controllers

Figure 5.4 shows that larger budget allows MIP-3 to deploy more controllers. As an example, for Abilene and GÉANT with budget B = \$1.2M, MIP-3 deploys up to $|C^T| = 11$ controllers for 11 s-switches and $|C^T| = 13$ for 17 s-switches, respectively. Thus, 100% and 56.52% of the s-switches in Abilene and GÉANT are co-located with a controller. This helps minimize the number of cables used to route s-c traffic demands. Moreover, MIP-3 is able to deploy more controllers for both networks because controller deployment cost is low. To show the case when controller deployment cost is expensive, we rerun MIP-3 for Abilene and GÉANT using B = \$1.2M and set the controller deployment cost to $\hat{p}^0$ = \$300K. We find that the number of controllers $|C^T|$ produced by MIP-3 decreases from $|C^T| = 11$ to $|C^T| = 5$ controllers and from $|C^T| = 13$ to $|C^T| = 2$ controllers for Abilene and GÉANT, respectively. The respective energy saving of Abilene

and GÉANT also reduces from 68.42% to 59.65% and 64.41% to 53.94%. The decreasing energy saving for Abilene and GÉANT is inline with the decrease in their respective number of upgraded switches. Specifically, the percentage of upgraded switches $\mathcal{V}^T$ of Abilene and GÉANT decreases from 91.67% to 50% and 73.91% to 60.87%, respectively. These results are reasonable because MIP-3 has to reduce the number of upgraded switches in order to deploy five and two controllers for Abilene and GÉANT, respectively. For DFN and Deltacom, a budget of $B = \$1.2\text{M}$ is sufficient for MIP-3 to deploy only up to three and two controllers, respectively. This is because there are a number of l-switches with unused cables yet to be upgraded by MIP-3 for both networks.

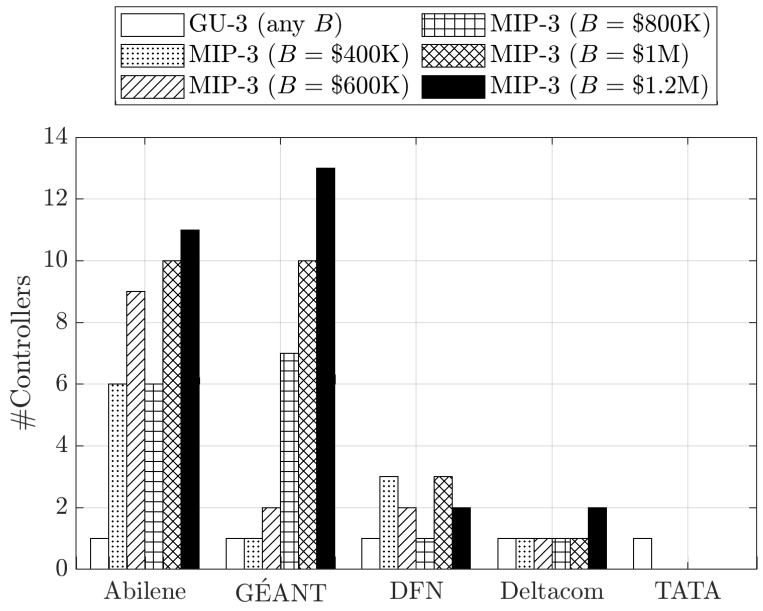


Figure 5.4. Total number of deployed controllers $|C^T|$ for MIP-3 and GU-3 for various budget B values.

On the other hand, Figure 5.4 shows that GU-3 consistently deploys one controller when the budget is no larger than $B = \$1.2\text{M}$. The reason is because the controller has sufficient capacity to manage all s-switches. As an example,

budget $B = \$1.2\text{M}$ can be used by GU-3 to upgrade $\mathcal{V}^T = 50\%$ and $\mathcal{V}^T = 20\%$ of l-switches for Abilene and TATA, respectively. Our experiment shows that 20.99% and 82.05% of the capacity $\theta = 7800\text{K}$ is used to handle the traffic load (in *cpps*) from s-switches in Abilene and TATA, respectively. Recall that GU-3 deploys a controller only if there is no *feasible* controller as per Definition 5.1 for any of the upgraded l-switches. To show the effect of smaller capacity $\theta = 1100\text{K}$ on the number of deployed controllers $|C^T|$, we rerun GU-3 for Abilene and TATA using $B = \$1.2\text{M}$. We find that GU-3 deploys more controllers, which are $|C^T| = 2$ and $|C^T| = 6$ controllers for Abilene and TATA, respectively. The energy saving and the number of upgraded switches for Abilene remains the same. The reason is that budget value of $B = \$1.2\text{M}$ is sufficient to upgrade 50% of Abilene's l-switches as well as deploying its two controllers. However, the energy saving ε_T for TATA decreases slightly from 26.39% to 25.49%, and the percentage of upgraded l-switches also decreases from 20% to 18.62%. The results for TATA are reasonable because for the given budget, GU-3 must reduce the number of upgraded l-switches in order to deploy its six controllers.

5.3.3. Effect of Increasing Number of Stages

This experiment investigates the impact of using a longer planning horizon T on the energy saving and the number of deployed switches and controllers; Sections 5.3.3.1, 5.3.3.2, and 5.3.3.3 provide the details. We use a budget value of $B = \$1.2\text{M}$ and vary T from one to five. Note that MIP-3 fails to obtain a result for Deltacom when there are $T = 5$ stages.

5.3.3.1. Energy Saving

As shown in Figure 5.5 the energy saving ε_T for Abilene, GÉANT, and DFN decreases for larger T values. For these three networks, a budget of $B = \$1.2\text{M}$ can be used to upgrade a larger percentage of l-switches at earlier stages. Further, as the later stages have a higher traffic volume, the smaller number of remaining l-switches in later stages have fewer idle cables that can be turned off. This means that upgrading these switches does not significantly increase ε_T. For example, the saving ε_T produced by MIP-3 and GU-3 for Abilene decreases from 75.44% to 68.42% and 74.56% to 66.32, respectively. For Deltacom and TATA, however, there are more s-switches to upgrade in later stages. The energy saving ε_T produced by MIP-3 and GU-3 for Deltacom increases from $\varepsilon_T = 33.54\%$ to $\varepsilon_T = 34.43\%$ and $\varepsilon_T = 32.92\%$ to $\varepsilon_T = 33.56\%$, respectively, when T increases from one to four. Similarly, the saving produced by GU-3 for TATA increases from $\varepsilon_T = 27.96\%$ to $\varepsilon_T = 31.55\%$ for value $T = 1$ to $T = 5$. Figure 5.5 also shows that for Abilene, GÉANT, DFN, and Deltacom, GU-3 produces a saving ε_T that is up to 3.08%, 4.07%, 3.36%, and 2.38%, respectively, off from the result produced by MIP-3.

5.3.3.2. Number of Upgraded Switches

Figure 5.6 shows the percentage $\mathcal{V}^T$ of upgraded l-switches for different number of stages T. As shown in the figure, MIP-3 does not always upgrade more switches for larger T values. Recall that the goal of MIP-3 is not to maximize the percentage $\mathcal{V}^T$ but to maximize energy saving ε_T. In contrast, the percentage $\mathcal{V}^T$ computed by GU-3 never decreases with the increasing stages T. As an example, for Abilene, GU-3 upgrades the same number of l-switches, i.e., $\mathcal{V}^T = 50\%$ of switches, when the value of T increases from one to five. The reason is because

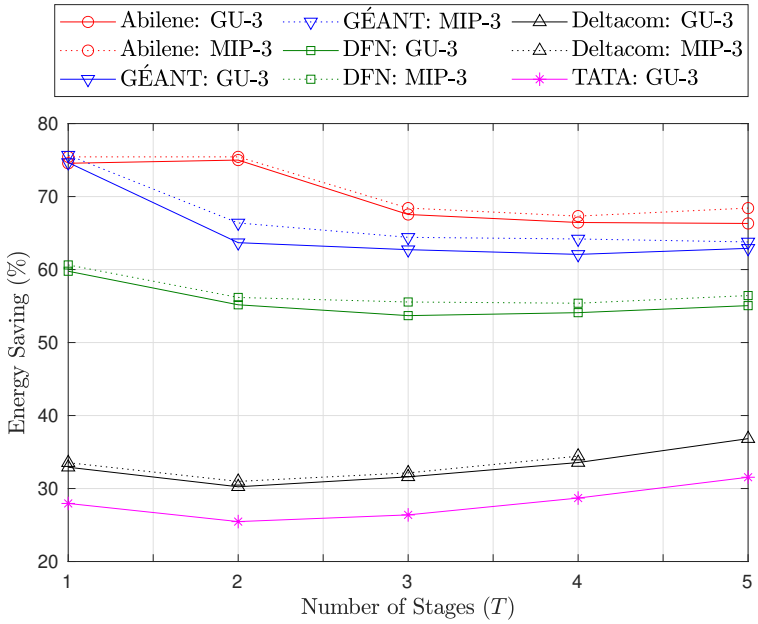


Figure 5.5. Energy saving ε_T of MIP-3 and GU-3 for various stage T values.

budget B = \$1.2M is sufficient for GU-3 to upgrade all l-switches that are adjacent to each unused cable in only one stage. For the same budget, however, MIP-3 and GU-3 require more than one stage to upgrade all l-switches with unused cables for GÉANT, DFN, and Deltacom. Thus, percentage $\mathcal{V}^T$ increases for larger values of T. For example, MIP-3 and GU-3 increase the percentage $\mathcal{V}^T$ of Deltacom from 13.27% to 31.86% and 13.27% to 30.97%, respectively, when T increases from one to four.

As also shown in Figure 5.6, MIP-3 upgrades more switches than GU-3. The reason is because GU-3 upgrades switches that have unused cables to turn off and deploy a new controller only if there is no existing *feasible* controllers. On the other hand, MIP-3 deploys s-switches and co-locates controllers to the s-switches as many as possible so that the number of cables used to route s-c demands can be minimized. As the results, MIP-3 upgrades more switches, hence deploys more controllers (see Section 5.3.3.3), than GU-3.

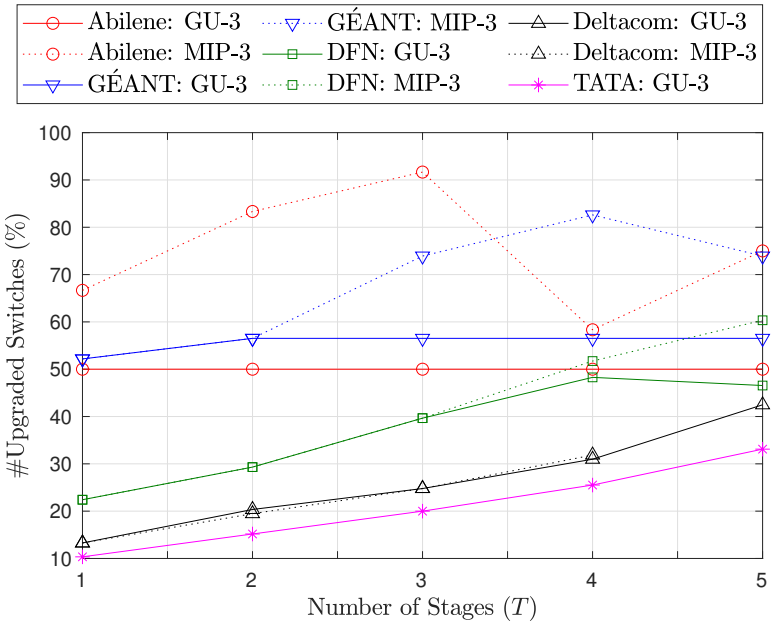


Figure 5.6. The percentage of upgraded l-switches $\mathcal{V}^T$ for MIP-3 and GU-3 for various stage T values.

5.3.3.3. Number of Deployed Controllers

In terms of the number of controllers $|C^T|$, larger number of s-switches makes MIP-3 and GU-3 deploy more controllers. As an example, for DFN, MIP-3 deploys $|C^T| = 2$ controllers, which increases significantly to $|C^T| = 26$ controllers when T increases from one to five. The reason is because the cost to deploy a controller is cheaper at later stages. Moreover, a budget of $B = \$1.2\text{M}$ is sufficient to upgrade all l-switches, which helps turn off idle cables at earlier stages. The number of controllers deployed by GU-3 for DFN, however, increases marginally from $|C^T| = 1$ controller to only $|C^T| = 2$ controllers. This is because GU-3 deploys a new controller only if no existing controllers satisfy Definition 5.1. For Deltacom and TATA, however, there are more l-switches to upgrade in later stages. MIP-3 and GU-3 produce the same number of controllers $|C^T| = 2$ for Deltacom using value $T = 4$. For value $T = 5$, GU-3 produces $|C^T| = 3$ controllers. For TATA,

GU-3 deploys $|C^T| = 1$ to $|C^T| = 3$ controllers for an increasing number of stages from value $T = 1$ to $T = 5$.

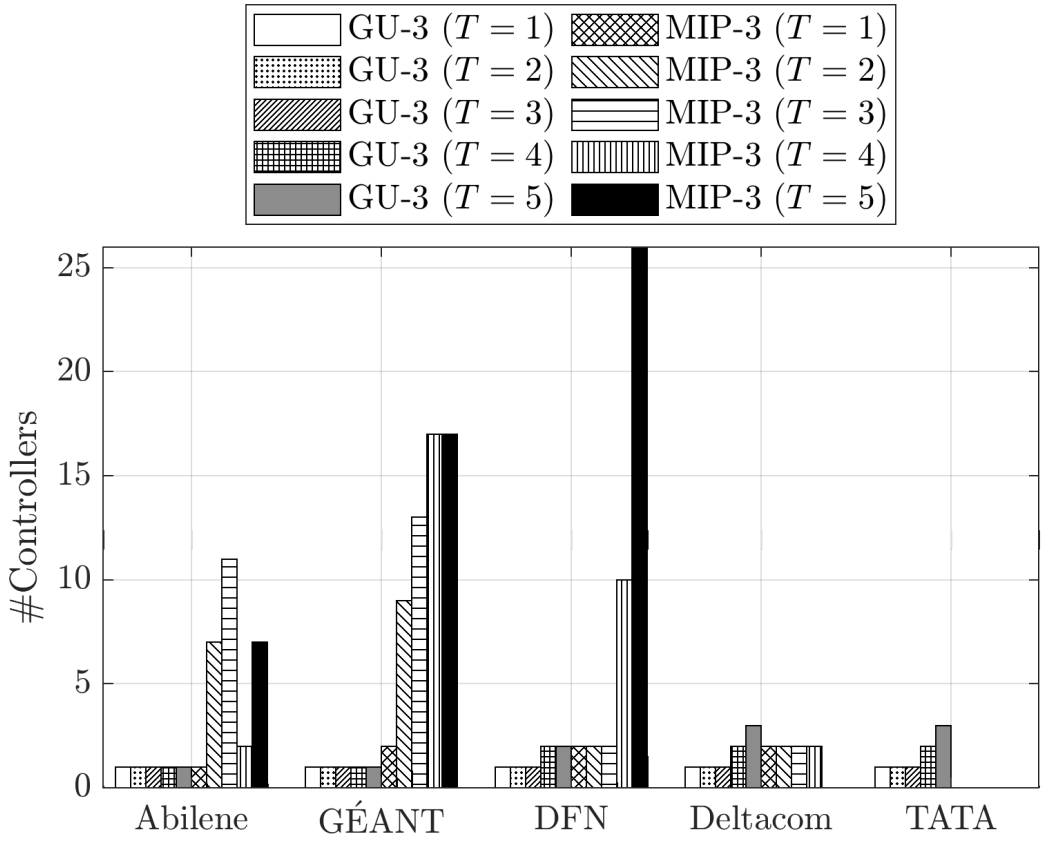


Figure 5.7. Total deployed controllers $|C^T|$ for MIP-3 and GU-3 given various stage T values.

5.3.4. Effect of Controller Placement

This section aims to evaluate the impact of using our *selective* controller placement on energy saving, number of *s*-*c* traffic demands with two link-disjoint paths, path delay of *s*-*s* and *s*-*c* traffic, and number of deployed *s*-switches and controllers.

For the evaluation, we compare GU-3 against GU-3^a - a version of GU-3 that

uses *arbitrary* controller placement, i.e., deploy controllers and associates each *s*-switch to a controller arbitrarily. More specifically, we modify Algorithm 5.2 for GU-3^a such that (i) function **SelectC()** arbitrarily selects any existing *feasible* controller, and (ii) function **LocateC()** deploys a new controller at any arbitrary node. We use budget values $B \in \{\$400K, \$1.2M\}$, planning horizon $T = 3$ stages, and controller capacity $\theta = 1100K$ *cpps*. For each network and budget, the results for GU-3^a are averaged over ten runs with a maximum standard deviation of 3.64. The following Sections 5.3.4.1, 5.3.4.2, 5.3.4.3, and 5.3.4.4 discuss the simulation results, which are then summarized in Section 5.3.4.5.

5.3.4.1. Energy saving

Figure 5.8 shows that using selective controller placement produces higher energy saving than using arbitrary controller placement. More specifically, GU-3 produces 8.7% and 0.78% higher energy saving than GU-3^a for Abilene with budget $B = \$400K$ and $B = \$1.2M$, i.e., $\varepsilon_T = 51.75\%$ and $\varepsilon_T = 67.54\%$ for GU-3 versus $\varepsilon_T = 47.25\%$ and $\varepsilon_T = 67.02\%$ for GU-3^a, respectively. Similarly, for budget $B = \$400K$ ($B = \$1.2M$), GU-3 produces 14.63% (3.56%), 16.23% (6.73%), 11.83% (10.81%), and 20.02% (10.57%) higher saving than GU-3^a for GÉANT, DFN, Deltacom and TATA, respectively. The reason is because the selective controller placement of GU-3 aims to optimize the location of each deployed controller and the switch-controller association to maximize energy saving.

5.3.4.2. Number of *s*-*c* traffic that uses two link-disjoint paths

Figure 5.9 shows that GU-3 is able to route more *s*-*c* traffics via two link-disjoint paths than GU-3^a. For example, for DFN and Deltacom with budget $B = \$1.2M$, 19.05% (7.81%) and 7.14% (0.77%) of *s*-*c* traffics for GU-3 (GU-3^a) are routed

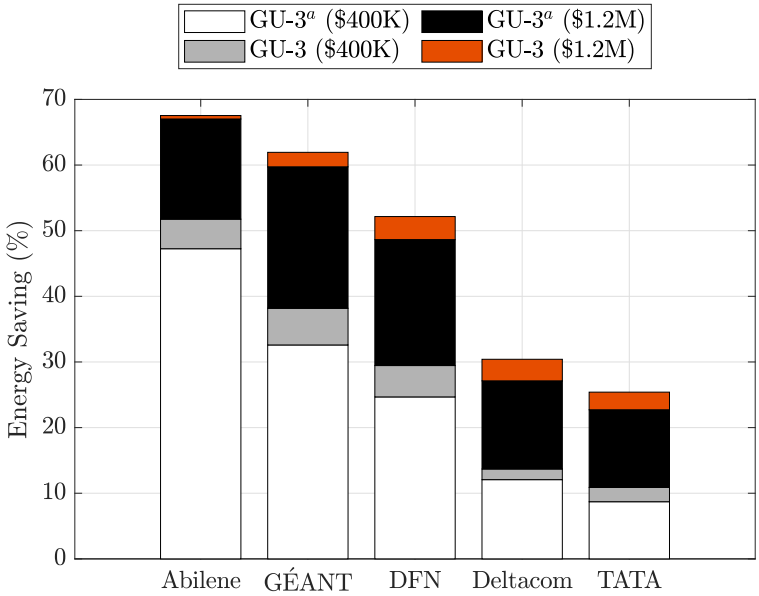


Figure 5.8. Energy saving of GU-3 and GU-3^a.

via two link-disjoint paths, respectively. The reason is because GU-3 always prioritizes an association that has two link-disjoint paths between an s-switch and a controller. Furthermore, GU-3 selects a node that not only has the largest number of unused cables per upgrade cost unit, but also the largest number of two link-disjoint paths to l-switches to co-locate a new controller. Note that for Abilene, each s-c traffic demand generated by both GU-3 and GU-3^a has no two-link-disjoint paths that are within the maximum delay $\delta_{\max,d}$.

5.3.4.3. Path delay of *s-s* and *s-c* traffic

Figure 5.10 shows that GU-3 and GU-3^a produce similar increase in path delay. For example, on average, GU-3 (GU-3^a) increases the path delay for GÉANT and Deltacom by 1.3% (1.11%) and 0.11% (0.13%), respectively, when using budget value B = \$400K. We find similar results for budget B = \$1.2M.

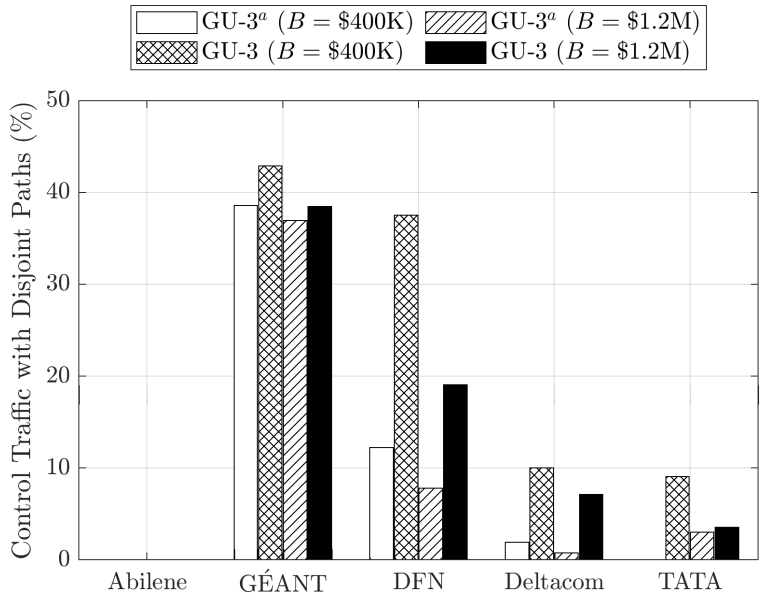


Figure 5.9. The percentage of *s-c* traffic demands with two link-disjoint paths for GU-3 and GU-3^a.

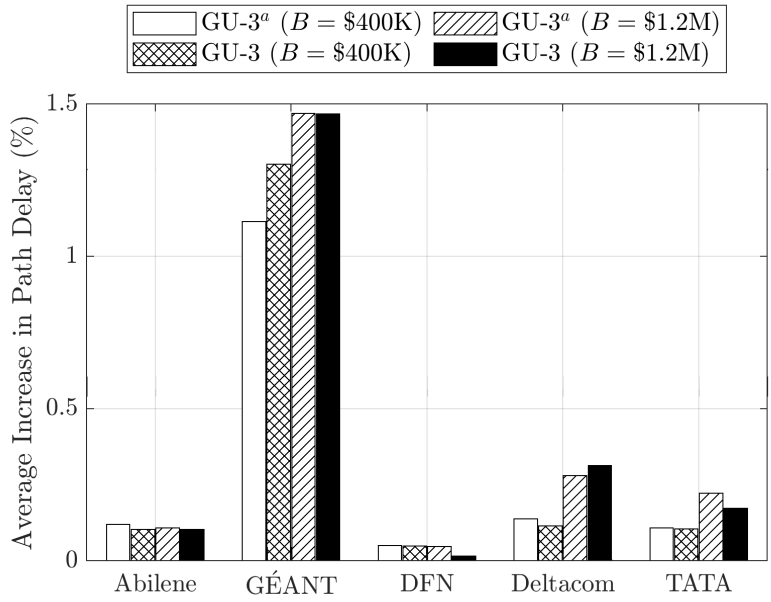


Figure 5.10. Increasing in path delay for GU-3 and GU-3^a.

5.3.4.4. Number of deployed *s*-switches and controllers

Figure 5.11 and Figure 5.12 show that GU-3 and GU-3^a deploy similar number of s-switches and controllers. More specifically, for budget value B = \$400K, except for TATA where GU-3 upgrades one switch more than GU-3^a, both solutions upgrade the same number of switches for each other network. Further, they deploy the same number of controllers $|C^T| = 2$ for all networks. For budget value B = \$1.2M, GU-3 upgrades one switch more than GU-3^a for Deltacom and TATA, and one switch less for each other network. In terms of controllers, GU-3^a deploys one controller more than GU-3 for GÉANT. Both solutions deploy the same number of controllers for each other network. The simulation results are reasonable because the aim of our selective controller placement is not to minimize the number of deployed s-switches nor controllers.

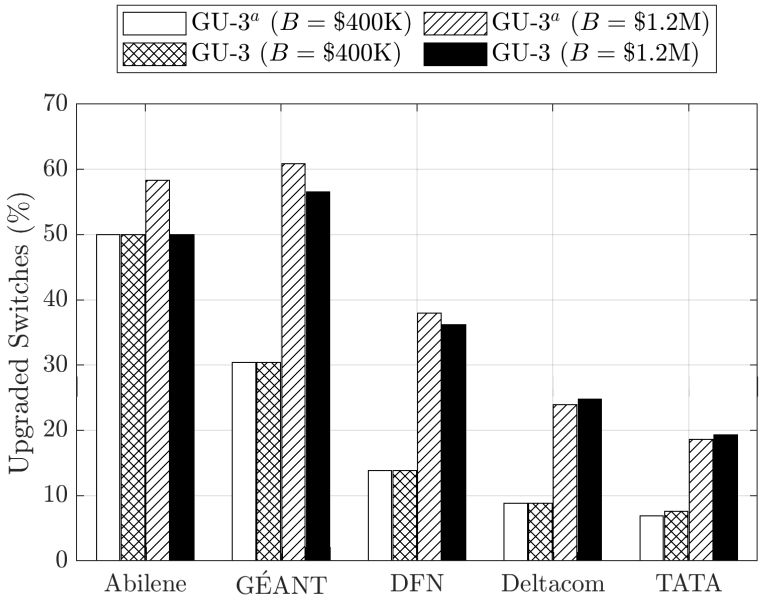


Figure 5.11. The percentage of upgraded l-switches for GU-3 and GU-3^a.

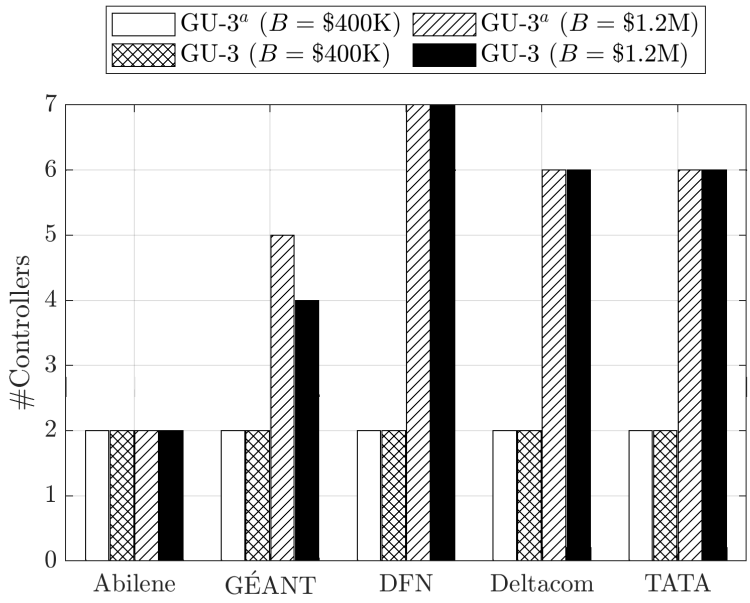


Figure 5.12. The total number of controllers for GU-3 and GU-3^a.

5.3.4.5. Result Summary

In summary, our simulation shows that our selective controller placement (i) produces higher energy saving and route more number of control traffic demands via link-disjoint paths than arbitrary placement, and (ii) marginally affects the number of deployed switches and controllers, and path delay. It is worth noting that GU-3 spends up to 7.24% smaller budget than GU-3^a. Further, on average, a controller in GU-3 has up to 12% smaller load (in cpps) than GU-3^a. Finally, both solutions have about the same running time. These notes further show the merits of using our selective controller placement.

5.3.5. GU-3 Versus IGCPA

This section presents the performance comparisons of GU-3 against an existing solution called IGCPA [19]. Briefly, IGCPA aims to strategically deploy a set of controllers of a *pure* SDN in $T = 1$ planning stage. Its goal is to maximize

energy saving. IGCPA uses GreCo [20] to associate each s-switch to one of the deployed controllers, and to route network traffic. GreCo routes each s-c traffic and s-s traffic via a single path. Further, GreCo considers only maximum path delay constraint for s-c traffic routing, and each link contains only one cable.

For fair comparisons, we use the following setup in our simulation: (i) $T = 1$ stage, MLU threshold $U_{\max} = 100\%$, controller capacity $\theta = 1100$K *cpps*, and bundle size $b_{uv} = 4$ cables for each link (u, v), (ii) Budget B in GU-3 for each network is set sufficiently large to upgrade *all* switches and deploy controllers to manage them, and (iii) for each network, we use GU-3 to find the number of controllers needed to manage *all* s-switches. Then, we deploy the same number of controllers for IGCPA, i.e., three, five, 11, 21, and 28 controllers for Abilene, GÉANT, DFN, Deltacom, and TATA, respectively.

Figure 5.13 shows that GU-3 produces smaller energy saving than IGCPA. More specifically, GU-3 produces energy saving ε_T of 74.56%, 76.35%, 76%, 76.86%, and 74.66% for Abilene, GÉANT, DFN, Deltacom, and TATA respectively. IGCPA, on the other hand, saves 5.55%, 6.61%, 4.34%, 5.08%, and 2.97% higher energy than GU-3 for the respective five networks. These results are reasonable because each s-s traffic d in GU-3 must be routed via a path with delay no longer than $\delta_{\max,d}$. In contrast, GreCo [20] that is used in IGCPA has no maximum delay constraint for s-s traffics. As a result of using IGCPA, 29.17%, 54.06%, 57.13%, 30.2%, and 46.46% of s-s traffic demands for Abilene, GÉANT, DFN, Deltacom, and TATA, respectively, use paths with delay longer than $\delta_{\max,d}$. Note that, for these five networks, these paths have delay up to 98.28% longer than the delay of their original shortest paths. In contrast, for the five networks, GU-3 generates respectively *only* 6.06%, 9.23%, 10.5%, 6.95%, and 18.83% paths that are longer than their corresponding shortest path. More importantly, each path in GU-3 is within its maximum delay requirement $\delta_{\max,d}$. Our simulation

shows that IGCPA requires a significantly higher CPU time to produce results than GU-3. For example, IGCPA takes 32312.91 and 121092.48 seconds to produce results for Deltacom and TATA, respectively. In contrast, GU-3 requires only 254.36 and 394.26 seconds for the two respective networks.

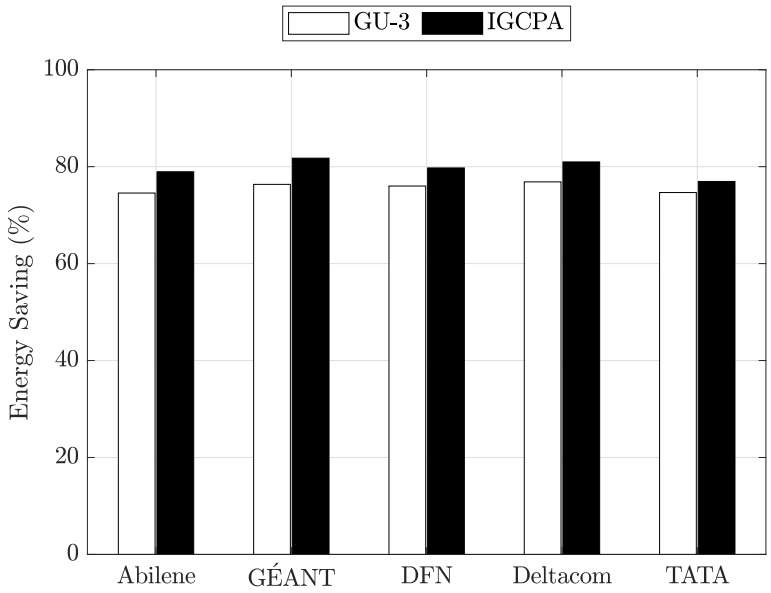


Figure 5.13. Energy saving of GU-3 and IGCPA.

5.4. Chapter Summary

This chapter studies network planning solutions, namely MIP-3 and GU-3, which are used to plan the upgrade of a legacy network into an SDN over multiple stages. The computed solution aims to minimize the energy cost of an upgraded network by strategically upgrading l-switches and installing SDN controllers. The results show that (i) increasing an operator's budget and planning stages result in more upgraded l-switches leading to higher energy savings up to 68.42%, (ii) GU-3 produces energy saving that is within 4.71% from optimal, (iii) MIP-3 de-

ploys more controllers than GU-3 in order to maximize energy saving, (iv) GU-3 that uses a selective controller placement results in a higher energy saving than an arbitrary controller placement, and (v) GU-3 runs significantly faster than an existing solution while producing competitive results in energy saving and ensuring each traffic is routed via a path within a given delay constraint. Note that this section does not include the evaluation of MIP-3 and GU-3 performances in terms of traffic controllability as in Chapters 3 and 4. Similar to ILP-1 and GU-1 discussed in Chapter 3 as well as MIP-2a, MIP-2b, GU-2a, and GU-2b presented in Chapter 4, MIP-3 and GU-3 would upgrade more switches and produce more traffic controllability at the later stages than the prior solution called Local Search [1]. The reason is because MIP-3 and GU-3 also consider budget constraint and depreciation of upgrade cost at each stage.

Chapter 6

Conclusion

6.1. Summary

This thesis proposes three versions of a novel problem, called Green Multi-Stage Upgrade (GMSU), to upgrade a legacy network to a *green* SDN in multiple planning stages. The goal of the three versions, i.e., GMSU-1, GMSU-2, and GMSU-3, is to maximize the number of unused cables that can be turned off by s-switches to save energy. The similarities and differences between the three versions, e.g., in terms of their constraints, are summarized in Table 2.2 of Section 2.6. Further, this thesis shows the NP-hardness of the three versions. In Chapter 3, we present an ILP called ILP-1 and a heuristic called GU-1 to solve GMSU-1. Chapter 4 outlines two MIPs, i.e., MIP-2a and MIP-2b, and two heuristic solutions, i.e., GU-2a and GU-2b, to solve GMSU-2. In Chapter 5, we proposed an MIP, called, MIP-3, and a heuristic solution, called GU-3, to solve GMSU-3. Each of the three chapters presents the time complexity of the heuristic solutions as well as their performance evaluation and comparisons with some prior works.

This thesis uses five real network topologies, i.e., Abilene, GÉANT, DFN, Deltacom and TATA, listed in Table 2.3, to analyse the performances of the

solutions for GMSU-1, GMSU-2, and GMSU-3. The evaluated performances include running time, energy saving, number of upgraded switches, and number of deployed controllers. The experimental results are summarized as follows:

1. For all versions, i.e., GMSU-1, GMSU-2, and GMSU-3, strategically replacing l-switches with s-switches is able to maximize the number of unused cables that can be turned off to save energy. For example, ILP-1 and GU-1 for GMSU-1 save 73.98% and 68.42% of Abilene's energy consumption, respectively, when using budget \$1.2M and three planning stages.

2. For all versions, increasing the budget and planning stages results in larger energy saving. As an example, for Deltacom, GU-2a of GMSU-2 increases energy saving over three planning stages from 7.76% to 32.22% when budget-value increases from \$200K to \$1.2M. Similarly, using budget value of \$1.2M, GU-2a increases energy saving for the network from 34.47% to 37.61% when the planning stage is increased from one to five.

3. All heuristic solutions for all versions run significantly faster than their corresponding ILP/MIP while producing energy savings that are off only by up to 5%. As an example, for GMSU-3, GU-3, which uses three planning stages, produces energy saving that is up to 2.62% off from the optimal saving produced by MIP-3 for GÉANT. Further, ILP/MIP cannot generate results for large-size networks, e.g., TATA, due to the hardness of each of the three problems.

4. Routing traffic via multi-paths can save similar energy as routing a traffic via a single path. For example, the energy saving of GU-2a is only 1.77% lower than that of GU-1 for DFN. Moreover, the increase in path delay for multi-path routing and single path routing is not significantly different.

For example, while GU-2a does not increase the average path delay of DFN, GU-1 increases the delay by only 0.08%.

5. Using a selective controller placement results in higher energy saving than using an arbitrary controller placement. As an example, GU-3 produces up to 20.02% higher energy saving for TATA than GU-3 that uses an arbitrary controller placement.

6. This thesis uses Local Search (LS) [1], Energy-Efficient Genetic Algorithm for hybrid SDNs (EEGAH-MNL) [2], and Improved Genetic Controller Placement Algorithm (IGCPA) [19] as benchmark solutions. LS maximizes traffic controllability over multiple planning stages, while EEGAH-MNL and IGCPA consider saving energy in a single stage. Further, the three prior solutions assume each link contains only a single cable. In this case, some adjustments, such as the number of stages are made to the solutions of GMSU-1, GMSU-2, and GMSU-3 in order to make fair comparisons against the three prior solutions. The benchmark results show that (i) GU-1 and GU-2a yield higher energy saving at later stages that saves more on energy cost than LS [1] that produces more saving in earlier stages, (ii) GU-2a saves more energy than EEGAH-MNL [2], and (iii) GU-3 has competitive results in energy saving and runs significantly faster than IGCPA [19].

6.2. Future Work

One can consider the following three possible extensions of GMSU-1, GMSU-2, and GMSU-3 as future works.

Firstly, one can extend GMSU-1, GMSU-2, and GMSU-3 problems for a scenario when traffic demands are *dynamic*, i.e., the traffic changes over time. Recall

that the three problems require a given set of traffic demands, i.e., *static* traffic demands at each given time. For the extension, one needs to modify the proposed heuristic solutions for the three problems so that they upgrade switches and generate traffic routing according to anticipated changes in traffic demands.

Secondly, the three problems in this thesis can be extended in a scenario where there are multiple types of energy sources, e.g., *non-renewable* generated from fossil fuels and *renewable* energy, such as solar and wind [93]. For this case, the problem considers different energy sources can have different cost per unit energy. The goal of this problem is to upgrade a legacy network to SDN over multiple planning stages so that the overall energy cost is minimized. However, the goal must satisfy the set of constraints of each problem, e.g., maximum budget at each stage and performance requirements such as delay and link capacity.

Thirdly, one may design an alternative solution that uses machine learning approach for each of the three problems, i.e., GMSU-1, GMSU-2, and GMSU-3. It is important to note that machine learning has been successfully used to solve, among others, network optimization problems, such as network traffic routing, traffic prediction, and controller placement in SDN [94]. One may expect that machine learning based solutions would be suitable not only to solve the three problems but also their two suggested extensions, i.e., GMSU for dynamic traffic demands and multiple types of energy sources.

In addition, our future work will formulate theoretical bound for each (i) heuristic solution proposed in the thesis, and (ii) machine learning solution that will be developed to solve each of the three versions of GMSU and their possible extension.

Appendices

Appendix A

Copyright Information

The copyright agreements for published papers allow authors to re-use their material in derivative works. Thus, copyright permission is not required. The following copyright information was obtained from IEEE conferences and journals in which the author has published.

IEEE COPYRIGHT AND CONSENT FORM

To ensure uniformity of treatment among all contributors, other forms may not be substituted for this form, nor may any wording of the form be changed. This form is intended for original material submitted to the IEEE and must accompany any such material in order to be published by the IEEE. Please read the form carefully and keep a copy for your files.

Green Multi-Stage Upgrade for Bundled-Link SDNs with Budget Constraint
Mrs. Lely Hiryanto, Dr. Sieteng Soh, Prof. Kwan-Wu Chin and Dr. Mihai M Lazarescu
2019 29th International Telecommunication Networks and Applications Conference (ITNAC)

COPYRIGHT TRANSFER

The undersigned hereby assigns to The Institute of Electrical and Electronics Engineers, Incorporated (the "IEEE") all rights under copyright that may exist in and to: (a) the Work, including any revised or expanded derivative works submitted to the IEEE by the undersigned based on the Work; and (b) any associated written or multimedia components or other enhancements accompanying the Work.

GENERAL TERMS

1. The undersigned represents that he/she has the power and authority to make and execute this form.
2. The undersigned agrees to indemnify and hold harmless the IEEE from any damage or expense that may arise in the event of a breach of any of the warranties set forth above.
3. The undersigned agrees that publication with IEEE is subject to the policies and procedures of the IEEE PSPB Operations Manual.
4. In the event the above work is not accepted and published by the IEEE or is withdrawn by the author(s) before acceptance by the IEEE, the foregoing copyright transfer shall be null and void. In this case, IEEE will retain a copy of the manuscript for internal administrative/record-keeping purposes.
5. For jointly authored Works, all joint authors should sign, or one of the authors should sign as authorized agent for the others.
6. The author hereby warrants that the Work and Presentation (collectively, the "Materials") are original and that he/she is the author of the Materials. To the extent the Materials incorporate text passages, figures, data or other material from the works of others, the author has obtained any necessary permissions. Where necessary, the author has obtained all third party permissions and consents to grant the license above and has provided copies of such permissions and consents to IEEE

You have indicated that you DO wish to have video/audio recordings made of your conference presentation under terms and conditions set forth in "Consent and Release."

CONSENT AND RELEASE

1. In the event the author makes a presentation based upon the Work at a conference hosted or sponsored in whole or in part by the IEEE, the author, in consideration for his/her participation in the conference, hereby grants the IEEE the unlimited, worldwide, irrevocable permission to use, distribute, publish, license, exhibit, record, digitize, broadcast, reproduce and archive, in any format or medium, whether now known or hereafter developed: (a) his/her presentation and comments at the conference; (b) any written materials or multimedia files used in connection with his/her presentation; and (c) any recorded interviews of him/her (collectively, the "Presentation"). The permission granted includes the transcription and reproduction of the Presentation for inclusion in products sold or distributed by IEEE and live or recorded broadcast of the Presentation during or after the conference.
2. In connection with the permission granted in Section 1, the author hereby grants IEEE the unlimited, worldwide, irrevocable right to use his/her name, picture, likeness, voice and biographical information as part of the advertisement, distribution and sale of products incorporating the Work or Presentation, and releases IEEE from any claim based on right of privacy or publicity.

BY TYPING IN YOUR FULL NAME BELOW AND CLICKING THE SUBMIT BUTTON, YOU CERTIFY THAT SUCH ACTION CONSTITUTES YOUR ELECTRONIC SIGNATURE TO THIS FORM IN ACCORDANCE WITH UNITED STATES LAW, WHICH AUTHORIZES ELECTRONIC SIGNATURE BY AUTHENTICATED REQUEST FROM A USER OVER THE INTERNET AS A VALID SUBSTITUTE FOR A WRITTEN SIGNATURE.

Lely Hiryanto	03-10-2019
Signature	**Date (dd-mm-yyyy)**

Information for Authors

AUTHOR RESPONSIBILITIES

The IEEE distributes its technical publications throughout the world and wants to ensure that the material submitted to its publications is properly available to the readership of those publications. Authors must ensure that their Work meets the requirements as stated in section 8.2.1 of the IEEE PSPB Operations Manual, including provisions covering originality, authorship, author responsibilities and author misconduct. More information on IEEE's publishing policies may be found at http://www.ieee.org/publications_standards/publications/rights/authorrightsresponsibilities.html Authors are advised especially of IEEE PSPB Operations Manual section 8.2.1.B12: "It is the responsibility of the authors, not the IEEE, to determine whether disclosure of their material requires the prior consent of other parties and, if so, to obtain it." Authors are also advised of IEEE PSPB Operations Manual section 8.1.1B: "Statements and opinions given in work published by the IEEE are the expression of the authors."

RETAINED RIGHTS/TERMS AND CONDITIONS

- Authors/employers retain all proprietary rights in any process, procedure, or article of manufacture described in the Work.
- Authors/employers may reproduce or authorize others to reproduce the Work, material extracted verbatim from the Work, or derivative works for the author's personal use or for company use, provided that the source and the IEEE copyright notice are indicated, the copies are not used in any way that implies IEEE endorsement of a product or service of any employer, and the copies themselves are not offered for sale.
- Although authors are permitted to re-use all or portions of the Work in other works, this does not include granting third-party requests for reprinting, republishing, or other types of re-use.The IEEE Intellectual Property Rights office must handle all such third-party requests.
- Authors whose work was performed under a grant from a government funding agency are free to fulfill any deposit mandates from that funding agency.

AUTHOR ONLINE USE

- **Personal Servers**. Authors and/or their employers shall have the right to post the accepted version of IEEE-copyrighted articles on their own personal servers or the servers of their institutions or employers without permission from IEEE, provided that the posted version includes a prominently displayed IEEE copyright notice and, when published, a full citation to the original IEEE publication, including a link to the article abstract in IEEE Xplore. Authors shall not post the final, published versions of their papers.
- **Classroom or Internal Training Use.** An author is expressly permitted to post any portion of the accepted version of his/her own IEEE-copyrighted articles on the author's personal web site or the servers of the author's institution or company in connection with the author's teaching, training, or work responsibilities, provided that the appropriate copyright, credit, and reuse notices appear prominently with the posted material. Examples of permitted uses are lecture materials, course packs, e-reserves, conference presentations, or in-house training courses.
- **Electronic Preprints.** Before submitting an article to an IEEE publication, authors frequently post their manuscripts to their own web site, their employer's site, or to another server that invites constructive comment from colleagues. Upon submission of an article to IEEE, an author is required to transfer copyright in the article to IEEE, and the author must update any previously posted version of the article with a prominently displayed IEEE copyright notice. Upon publication of an article by the IEEE, the author must replace any previously posted electronic versions of the article with either (1) the full citation to the

IEEE COPYRIGHT FORM

To ensure uniformity of treatment among all contributors, other forms may not be substituted for this form, nor may any wording of the form be changed. This form is intended for original material submitted to the IEEE and must accompany any such material in order to be published by the IEEE. Please read the form carefully and keep a copy for your files.

Green Multi-Stage Upgrade for Bundled-Link SDNs with Budget and Delay Constraints
Hiryanto, Lely; Soh, Sieteng; Chin, Kwan-Wu; Lazarescu, Mihai
IEEE Transactions on Green Communications and Networking

COPYRIGHT TRANSFER

The undersigned hereby assigns to The Institute of Electrical and Electronics Engineers, Incorporated (the "IEEE") all rights under copyright that may exist in and to: (a) the Work, including any revised or expanded derivative works submitted to the IEEE by the undersigned based on the Work; and (b) any associated written or multimedia components or other enhancements accompanying the Work.

GENERAL TERMS

1. The undersigned represents that he/she has the power and authority to make and execute this form.
2. The undersigned agrees to indemnify and hold harmless the IEEE from any damage or expense that may arise in the event of a breach of any of the warranties set forth above.
3. The undersigned agrees that publication with IEEE is subject to the policies and procedures of the IEEE PSPB Operations Manual.
4. In the event the above work is not accepted and published by the IEEE or is withdrawn by the author(s) before acceptance by the IEEE, the foregoing copyright transfer shall be null and void. In this case, IEEE will retain a copy of the manuscript for internal administrative/record-keeping purposes.
5. For jointly authored Works, all joint authors should sign, or one of the authors should sign as authorized agent for the others.
6. The author hereby warrants that the Work and Presentation (collectively, the "Materials") are original and that he/she is the author of the Materials. To the extent the Materials incorporate text passages, figures, data or other material from the works of others, the author has obtained any necessary permissions. Where necessary, the author has obtained all third party permissions and consents to grant the license above and has provided copies of such permissions and consents to IEEE

BY TYPING IN YOUR FULL NAME BELOW AND CLICKING THE SUBMIT BUTTON, YOU CERTIFY THAT SUCH ACTION CONSTITUTES YOUR ELECTRONIC SIGNATURE TO THIS FORM IN ACCORDANCE WITH UNITED STATES LAW, WHICH AUTHORIZES ELECTRONIC SIGNATURE BY AUTHENTICATED REQUEST FROM A USER OVER THE INTERNET AS A VALID SUBSTITUTE FOR A WRITTEN SIGNATURE.

Lely Hiryanto	19-05-2021
Signature	**Date (dd-mm-yyyy)**

Information for Authors

AUTHOR RESPONSIBILITIES

The IEEE distributes its technical publications throughout the world and wants to ensure that the material submitted to its publications is properly available to the readership of those publications. Authors must ensure that their Work meets the requirements as stated in section 8.2.1 of the IEEE PSPB Operations Manual, including provisions covering originality,

IEEE COPYRIGHT AND CONSENT FORM

To ensure uniformity of treatment among all contributors, other forms may not be substituted for this form, nor may any wording of the form be changed. This form is intended for original material submitted to the IEEE and must accompany any such material in order to be published by the IEEE. Please read the form carefully and keep a copy for your files.

Green Multi-Stage Upgrade for Bundled-Links SDN/OSPF-ECMP Networks
Lely Hiryanto,Sieteng Soh,Kwan-Wu Chin,Duc-Son Pham,Mihai Lazarescu
ICC 2021 - IEEE International Conference on Communications

COPYRIGHT TRANSFER

The undersigned hereby assigns to The Institute of Electrical and Electronics Engineers, Incorporated (the "IEEE") all rights under copyright that may exist in and to: (a) the Work, including any revised or expanded derivative works submitted to the IEEE by the undersigned based on the Work; and (b) any associated written or multimedia components or other enhancements accompanying the Work.

GENERAL TERMS

1. The undersigned represents that he/she has the power and authority to make and execute this form.
2. The undersigned agrees to indemnify and hold harmless the IEEE from any damage or expense that may arise in the event of a breach of any of the warranties set forth above.
3. The undersigned agrees that publication with IEEE is subject to the policies and procedures of the IEEE PSPB Operations Manual.
4. In the event the above work is not accepted and published by the IEEE or is withdrawn by the author(s) before acceptance by the IEEE, the foregoing copyright transfer shall be null and void. In this case, IEEE will retain a copy of the manuscript for internal administrative/record-keeping purposes.
5. For jointly authored Works, all joint authors should sign, or one of the authors should sign as authorized agent for the others.
6. The author hereby warrants that the Work and Presentation (collectively, the "Materials") are original and that he/she is the author of the Materials. To the extent the Materials incorporate text passages, figures, data or other material from the works of others, the author has obtained any necessary permissions. Where necessary, the author has obtained all third party permissions and consents to grant the license above and has provided copies of such permissions and consents to IEEE

You have indicated that you DO wish to have video/audio recordings made of your conference presentation under terms and conditions set forth in "Consent and Release."

CONSENT AND RELEASE

1. In the event the author makes a presentation based upon the Work at a conference hosted or sponsored in whole or in part by the IEEE, the author, in consideration for his/her participation in the conference, hereby grants the IEEE the unlimited, worldwide, irrevocable permission to use, distribute, publish, license, exhibit, record, digitize, broadcast, reproduce and archive, in any format or medium, whether now known or hereafter developed: (a) his/her presentation and comments at the conference; (b) any written materials or multimedia files used in connection with his/her presentation; and (c) any recorded interviews of him/her (collectively, the "Presentation"). The permission granted includes the transcription and reproduction of the Presentation for inclusion in products sold or distributed by IEEE and live or recorded broadcast of the Presentation during or after the conference.
2. In connection with the permission granted in Section 1, the author hereby grants IEEE the unlimited, worldwide, irrevocable right to use his/her name, picture, likeness, voice and biographical information as part of the advertisement, distribution and sale of products incorporating the Work or Presentation, and releases IEEE from any claim based on right of privacy or publicity.

BY TYPING IN YOUR FULL NAME BELOW AND CLICKING THE SUBMIT BUTTON, YOU CERTIFY THAT SUCH ACTION CONSTITUTES YOUR ELECTRONIC SIGNATURE TO THIS FORM IN ACCORDANCE WITH UNITED STATES LAW, WHICH AUTHORIZES ELECTRONIC SIGNATURE BY AUTHENTICATED REQUEST FROM A USER OVER THE INTERNET AS A VALID SUBSTITUTE FOR A WRITTEN SIGNATURE.

Lely Hiryanto	04-02-2021
Signature	**Date (dd-mm-yyyy)**

Information for Authors

AUTHOR RESPONSIBILITIES

The IEEE distributes its technical publications throughout the world and wants to ensure that the material submitted to its publications is properly available to the readership of those publications. Authors must ensure that their Work meets the requirements as stated in section 8.2.1 of the IEEE PSPB Operations Manual, including provisions covering originality, authorship, author responsibilities and author misconduct. More information on IEEE's publishing policies may be found at http://www.ieee.org/publications_standards/publications/rights/authorrightsresponsibilities.html Authors are advised especially of IEEE PSPB Operations Manual section 8.2.1.B12: "It is the responsibility of the authors, not the IEEE, to determine whether disclosure of their material requires the prior consent of other parties and, if so, to obtain it." Authors are also advised of IEEE PSPB Operations Manual section 8.1.1B: "Statements and opinions given in work published by the IEEE are the expression of the authors."

RETAINED RIGHTS/TERMS AND CONDITIONS

- Authors/employers retain all proprietary rights in any process, procedure, or article of manufacture described in the Work.
- Authors/employers may reproduce or authorize others to reproduce the Work, material extracted verbatim from the Work, or derivative works for the author's personal use or for company use, provided that the source and the IEEE copyright notice are indicated, the copies are not used in any way that implies IEEE endorsement of a product or service of any employer, and the copies themselves are not offered for sale.
- Although authors are permitted to re-use all or portions of the Work in other works, this does not include granting third-party requests for reprinting, republishing, or other types of re-use.The IEEE Intellectual Property Rights office must handle all such third-party requests.
- Authors whose work was performed under a grant from a government funding agency are free to fulfill any deposit mandates from that funding agency.

AUTHOR ONLINE USE

- **Personal Servers**. Authors and/or their employers shall have the right to post the accepted version of IEEE-copyrighted articles on their own personal servers or the servers of their institutions or employers without permission from IEEE, provided that the posted version includes a prominently displayed IEEE copyright notice and, when published, a full citation to the original IEEE publication, including a link to the article abstract in IEEE Xplore. Authors shall not post the final, published versions of their papers.
- **Classroom or Internal Training Use.** An author is expressly permitted to post any portion of the accepted version of his/her own IEEE-copyrighted articles on the author's personal web site or the servers of the author's institution or company in connection with the author's teaching, training, or work responsibilities, provided that the appropriate copyright, credit, and reuse notices appear prominently with the posted material. Examples of permitted uses are lecture materials, course packs, e-reserves, conference presentations, or in-house training courses.
- **Electronic Preprints.** Before submitting an article to an IEEE publication, authors frequently post their manuscripts to their own web site, their employer's site, or to another server that invites constructive comment from colleagues. Upon submission of an article to IEEE, an author is required to transfer copyright in the article to IEEE, and the author must update any previously posted version of the article with a prominently displayed IEEE copyright notice. Upon publication of an article by the IEEE, the author must replace any previously posted electronic versions of the article with either (1) the full citation to the

Creative Commons Attribution License (CCBY)

Multi-Path Routing in Green Multi-Stage Upgrade for Bundled-Links SDN/OSPF-ECMP Networks
Lely, Hiryanto; Soh, Sieteng; Chin, Kwan-Wu ; Pham, Duc-Son; Lazarescu, Mihai
IEEE Access

By clicking the checkbox at the bottom of this page you, as the author or representative of the author, confirm that your work is licensed to IEEE under the Creative Commons Attribution 4.0(CCBY 4.0). As explained by the Creative Commons web site, this license states that IEEE is free to share, copy, distribute and transmit your work under the following conditions:

- Attribution - Users must attribute the work in the manner specified by the author or licensor (but not in any way that suggests that they endorse the users or their use of the work).

With the understanding that:

- **Waiver** - Any of the above conditions can be waived if users get permission from the copyright holder.
- **Public Domain** - Where the work or any of its elements is in the public domain under applicable law, that status is in no way affected by the license.
- **Other Rights** - In no way are any of the following rights affected by the license:
  - A user's fair dealing or fair use rights, or other applicable copyright exceptions and limitations;
  - The author's moral rights;
  - Rights other persons may have either in the work itself or in how the work is used, such as publicity or privacy rights.

For any reuse or distribution, users must make clear to others the license terms of this work.

Upon clicking on the checkbox below, you will not only confirm that your submission is under the CCBY license but you will also be taken to IEEE's Terms of Use, which will require your signature.

[X] I confirm the submitted work is licensed to IEEE under the Creative Commons Attribution 4.0 United States (CCBY 4.0)

TERMS AND CONDITIONS OF AN AUTHOR'S USE OF THE CREATIVE COMMONS ATTRIBUTION LICENSE (CCBY)

1. Creative Commons Licensing

To grow the commons of free knowledge and free culture, all users are required to grant broad permissions to the general public to re-distribute and re-use their contributions freely. Therefore, for any text, figures, or other work in any medium you hold the copyright to, by submitting it, you agree to license it under the Creative Commons Attribution 4.0 Unported License.

2. Attribution

As an author, you agree to be attributed in any of the following fashions: a) through a hyperlink (where possible) or URL to the article or articles you contributed to, b) through a hyperlink (where possible) or URL to an alternative, stable online copy which is freely accessible, which conforms with the license, and which provides credit to the authors in a manner equivalent to the credit given on this website, or c) through a list of all authors.

3. Terms of Publication

A. By submitting your work to IEEE, you agree to comply with the IEEE Publication Services and Products Board Operations

Manual (the "Operations Manual"), including, but not limited to, the specific provisions referenced herein(except to the extent any provision of the Operations Manual requires assignment of copyright in your work to IEEE).

B. Submission to this IEEE journal does not guarantee publication. By submitting your work to this journal you, as author, recognize that your work may be rejected for any reason. All submissions shall be reviewed by the Editor in accordance with section 8.2.2 of the Operations Manual.

C. Should your paper be rejected IEEE will not exercise any of the rights granted to it under the Creative Commons Attribution 4.0 Unported License.

D. IEEE takes intellectual property protection seriously and is opposed to plagiarism in any fashion. Accordingly, you consent to having your work submitted to a plagiarism detection tool and to be bound by IEEE policies concerning plagiarism and author misconduct.

E. IEEE distributes its technical publications throughout the world and wants to ensure that the material submitted to its publications is properly available to the readership of those publications. You must ensure that your work meets the requirements as stated in section 8.2.1 of the Operations Manual, including provisions covering originality, authorship, author responsibilities and author misconduct. More information on IEEE's publishing policies may be found at https://www.ieee.org/publications/rights/author-rights-responsibilities.html.

F. You warrant that your work, including and any accompanying materials, is original and that you are the author of the work. To the extent your work incorporates text passages, figures, data or other material from the works of others, you represent and warrant that you have obtained all third party permissions and consents to grant the rights herein and have provided copies of such permissions and consents to IEEE. As stated in section 8.2.1B12 of the Operations Manual: "It is the responsibility of the authors, not the IEEE, to determine whether disclosure of their material requires the prior consent of other parties and, if so, to obtain it."

G. You are advised of Operations Manual section 8.1.1B: "Statements and opinions given in work published by the IEEE are the expression of the authors."

H. You agree that publication of a notice of violation as a corrective action for a confirmed case of plagiarism, as described in Section 8.2.4 of the IEEE PSPB Publications Operations Manual, does not violate any of your moral rights.

I. You agree to indemnify and hold IEEE and its parents, subsidiaries, affiliates, officers, employees, agents, partners and licensors harmless from any claim or demand, including reasonable attorneys' fees, due to or arising out of: (1) content you submit, post, transmit or otherwise make available through IEEE's publishing program; (2) your use of this IEEE journal; (3) your violation of these Terms of Use; or (4) your violation of any rights of another party.

BY TYPING IN YOUR FULL NAME BELOW AND CLICKING THE SUBMIT BUTTON, YOU CERTIFY THAT SUCH ACTION CONSTITUTES YOUR ELECTRONIC SIGNATURE TO THIS FORM IN ACCORDANCE WITH UNITED STATES LAW, WHICH AUTHORIZES ELECTRONIC SIGNATURE BY AUTHENTICATED REQUEST FROM A USER OVER THE INTERNET AS A VALID SUBSTITUTE FOR A WRITTEN SIGNATURE.

Lely Hiryanto
Signature

28-06-2021
Date

Questions about the submission of the form or manuscript must be sent to the publication's editor. Please direct all questions about IEEE copyright policy to:
IEEE Intellectual Property Rights Office, copyrights@ieee.org, +1-732-562-3966

Appendix B

Linearization of max() and min()

The function max() in Eq. (3.1h), Eq. (4.4), and Eq. (5.5) is a simplified form of its real implementation using Gurobi API for C++. The implementation uses the following set of linear formulations:

$$\begin{aligned} n_{uv}^t &\geq n_{uv}^t \\ n_{uv}^t &\geq b_{uv} \times (1 - \sum_{k=1}^{t} x_u^k - \sum_{k=1}^{t} x_v^k) \\ n_{uv}^t &\leq n_{uv}^t + M \times (1 - g_{uv}) \\ n_{uv}^t &\leq b_{uv} \times (1 - \sum_{k=1}^{t} x_u^k - \sum_{k=1}^{t} x_v^k) + M \times g_{uv} \end{aligned} \tag{B.1}$$

The above set of equations (B.1), which applies to all links $(u, v) \in E$, uses a decision binary variable g_{uv} and a constant $M = \max_{(u,v)\in E} b_{uv}$.

The function min() in Eq. (5.11) is used to simplify its real implementation that uses the following set of two linear equations.

$$\begin{aligned} q_{u,a}^{t} &\leq \sum_{k=1}^{t} x_{u}^{k} \\ q_{u,a}^{t} &\leq \sum_{k=1}^{t} \hat{x}_{a}^{k} \end{aligned} \tag{B.2}$$

Each equation in (B.2) is evaluated for each node $u \in V$ and every node $a \in V$.

Appendix C

Code and Solution

C.1. ILP-1

C.1.1. Code Implementation

```
...
//Decision variables
for(t = 0; t < T; ++t) {
      //on-cables
      n[t] = model.addVars(nE, GRB_INTEGER);
      for (l = 0; l < nE; ++l) n[t][l].set(GRB_DoubleAttr_Obj, 1);
      //switch status: legacy or sdn
      x[t] = model.addVars(nV, GRB_BINARY);
      //all upgraded switch up to stage t
      s[t] = model.addVars(nV, GRB_BINARY);
      //c-link or l-link
      y[t] = model.addVars(nE, GRB_BINARY);
      //selected shortest path for each demand
      z[t] = new GRBVar *[nD];
      for (d = 0; d < nD; ++d) z[t][d] = model.addVars(nE, GRB_BINARY);
}
//Constraints
//one time switch upgrade
for (u = 0; u < nV; ++u) {
      GRBLinExpr totalx = 0;
      for(t = 0; t < T; ++t) totalx += x[t][u];
      model.addConstr(totalx <= 1);
}
for(t = 0; t < T; ++t) {
      //Switch Upgrade - maximum Budget per stage
      GRBLinExpr totalprice = 0;
      GRBLinExpr totalbudget = 0;
      GRBLinExpr usedbudget = 0;
      for (u = 0; u < nV; ++u) totalprice += (x[t][u] * price[t][u]);
```

```
    for (k = 0; k <= t; ++k) totalbudget += budget[k];
    for (k = 0; k < t; ++k) for (u = 0; u < nV; ++u) usedbudget += (x[k][u] * price[k][u]);
    model.addConstr(totalprice <= totalbudget - usedbudget);
    //only s-switch can turn off unused cables
    for (u = 0; u < nV; ++u) {
        GRBLinExpr totalx = 0;
        for (k = 0; k <= t; ++k) totalx += x[k][u];
        model.addConstr(s[t][u] == totalx);
    }
    for (l = 0; l < nE; ++l) {
        model.addConstr(n[t][l] >= n[t][l]);
        model.addConstr(n[t][l] >= G.E[l].b * (1 - s[t][G.E[l].u-1] - s[t][G.E[l].v-1]));
        model.addConstr(n[t][l] <= n[t][l] + M * (1 - y[t][l]));
        model.addConstr(n[t][l] <= G.E[l].b * (1 - s[t][G.E[l].u-1] - s[t][G.E[l].v-1]) + M * y[t][l]);
    }
    //on-cable lower and upper bound for (l = 0; l < nE; ++l) {
        model.addConstr(n[t][l] <= G.E[l].b);
        model.addConstr(n[t][l] >= 0);
    }
    //Single Path Traffic Routing
    //flow conservation
    for (d = 0; d < nD; ++d) {
        for (u = 0; u < nV; ++u) {
            GRBLinExpr yuv = 0;
            for (l = 0; l < nE; ++l) if (G.E[l].u-1 == u) yuv += z[t][d][l];
            GRBLinExpr yvu = 0;
            for (l = 0; l < nE; ++l) if (G.E[l].v-1 == u) yvu += z[t][d][l];
            model.addConstr(yuv - yvu == NodeType[u][d]);
        }
    }
    //Delay
    for (d = 0; d < nD; ++d) {
        GRBLinExpr delay = 0;
        for (l = 0; l < nE; ++l) delay += z[t][d][l] * G.E[l].w;
        model.addConstr(delay <= ceil(D.D[d].delta * SIGMA));
    }
    //MLU
    for (l = 0; l < nE; ++l) {
        GRBLinExpr fuv = 0;
        for (d = 0; d < nD; ++d) fuv += z[t][d][l] * D.D[d].omega * mu[t];
        model.addConstr(fuv <= G.E[l].c / G.E[l].b * MLU * n[t][l]);
    }
}
//The objective
model.set(GRB_IntAttr_ModelSense, GRB_MINIMIZE);
// Solve
model.optimize();
...
```

C.1.2. Solution for Small Example

```
Academic license - for non-commercial use only - expires 2023-02-03
Gurobi Optimizer version 9.1.2 build v9.1.2rc0 (mac64[arm])
Thread count: 8 physical cores, 8 logical processors, using up to 8 threads
Optimize a model with 465 rows, 372 columns and 1578 nonzeros
Model fingerprint: 0xd55fc0da
Variable types: 0 continuous, 372 integer (324 binary)
Coefficient statistics:
  Matrix range     [6e-01, 2e+02]
  Objective range  [1e+00, 1e+00]
  Bounds range     [1e+00, 1e+00]
  RHS range        [1e+00, 4e+02]
Presolve removed 298 rows and 151 columns
Presolve time: 0.01s
Presolved: 167 rows, 221 columns, 822 nonzeros
Variable types: 0 continuous, 221 integer (177 binary)
Found heuristic solution: objective 82.0000000
Found heuristic solution: objective 58.0000000

Root relaxation: objective 4.858333e+01, 185 iterations, 0.00 seconds

    Nodes    |    Current Node    |     Objective Bounds      |     Work
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time

     0     0   48.58333    0   34   58.00000   48.58333  16.2%     -    0s
H    0     0                       57.0000000   48.58333  14.8%     -    0s
H    0     0                       55.0000000   48.58333  11.7%     -    0s
     0     0   51.83333    0   52   55.00000   51.83333  5.76%     -    0s
H    0     0                       53.0000000   51.83333  2.20%     -    0s
     0     0   51.83333    0   52   53.00000   51.83333  2.20%     -    0s

Cutting planes:
  Gomory: 4
  Implied bound: 1
  MIR: 7
  RLT: 1

Explored 1 nodes (271 simplex iterations) in 0.02 seconds
Thread count was 8 (of 8 available processors)

Solution count 5: 53 55 57 ... 82

Optimal solution found (tolerance 1.00e-04)
Best objective 5.300000000000e+01, best bound 5.300000000000e+01, gap 0.0000%
```

Figure C.1. Solution for Example in Figure 3.1.

```
Stage-1
  upgraded switches = { 6 }: 1 l-switches
  #on-cables = 37
  #off-cables = 11
  ES = 22.9167%
  TC = 100%
  Path Delay = 0%
  Traffic Demands & their single path :
        [s=2, t=7: vol=2.4] => < (2,4) (4,7) >
        [s=1, t=8: vol=4.8] => < (1,5) (5,8) >
        [s=6, t=8: vol=1.2] => < (6,5) (5,8) >
        [s=5, t=3: vol=1.2] => < (5,1) (1,3) >
        [s=6, t=1: vol=0.6] => < (6,5) (5,1) >
  Links and on-cables :
        link(1,2):2 cable(s)    link(2,1):2 cable(s)
        link(1,3):2 cable(s)    link(3,1):2 cable(s)
        link(1,5):2 cable(s)    link(5,1):2 cable(s)
        link(2,4):2 cable(s)    link(4,2):2 cable(s)
        link(2,5):2 cable(s)    link(5,2):2 cable(s)
        link(3,6):0 cable(s)    link(6,3):0 cable(s)
        link(4,7):2 cable(s)    link(7,4):2 cable(s)
        link(5,6):0 cable(s)    link(6,5):1 cable(s)
        link(5,8):2 cable(s)    link(8,5):2 cable(s)
        link(6,9):0 cable(s)    link(9,6):0 cable(s)
        link(7,8):2 cable(s)    link(8,7):2 cable(s)
        link(8,9):2 cable(s)    link(9,8):2 cable(s)
```

Figure C.2. Solution for Example in Figure 3.1 (cont.).

```
Stage-2
  upgraded switches = { 2 8 }: 2 l-switches
  #on-cables = 16
  #off-cables = 32
  ES = 66.6667%
  TC = 100%
  Path Delay = 0%
  Traffic Demands & their single path :
        [s=2, t=7: vol=2.4] => < (2,4) (4,7) >
        [s=1, t=8: vol=4.8] => < (1,5) (5,8) >
        [s=6, t=8: vol=1.2] => < (6,5) (5,8) >
        [s=5, t=3: vol=1.2] => < (5,1) (1,3) >
        [s=6, t=1: vol=0.6] => < (6,5) (5,1) >
  Links and on-cables :
        link(1,2):0 cable(s)    link(2,1):0 cable(s)
        link(1,3):2 cable(s)    link(3,1):2 cable(s)
        link(1,5):2 cable(s)    link(5,1):2 cable(s)
        link(2,4):1 cable(s)    link(4,2):0 cable(s)
        link(2,5):0 cable(s)    link(5,2):0 cable(s)
        link(3,6):0 cable(s)    link(6,3):0 cable(s)
        link(4,7):2 cable(s)    link(7,4):2 cable(s)
        link(5,6):0 cable(s)    link(6,5):1 cable(s)
        link(5,8):2 cable(s)    link(8,5):0 cable(s)
        link(6,9):0 cable(s)    link(9,6):0 cable(s)
        link(7,8):0 cable(s)    link(8,7):0 cable(s)
        link(8,9):0 cable(s)    link(9,8):0 cable(s)
Total Upgrades = 33.3333%
Average Energy Saving = 44.7917%
Average TC = 100%
Average Path Delay = 0%
```

Figure C.3. Solution for Example in Figure 3.1 (cont.).

C.2. MIP-2a and MIP-2b

C.2.1. Code Implementation

```
...
//Decision variables
for(t = 0; t < T; ++t) {
    see Code 1
    //selected shortest path for each demand
    y[t] = new GRBVar **[nD];
    for (d = 0; d < nD; ++d) {
        y[t][d] = new GRBVar *[D.AP[d].size()];
        for (k = 0; k < D.AP[d].size(); ++k) y[t][d][k] = model.addVars(nE, GRB_BINARY);
    }
    //traffic flow per link on each path
    f[t] = new GRBVar *[nV];
    for (u = 0; u < nV; ++u) f[t][u] = model.addVars(nE);
    //traffic flow per path
    fp[t] = new GRBVar *[nD];
    for (d = 0; d < nD; ++d) fp[t][d] = model.addVars(D.AP[d].size());
    //traffic flow per path out[t] = model.addVars(nD);
    //selected path per demand
    pp[t] = new GRBVar *[nD];
    for (d = 0; d < nD; ++d) pp[t][d] = model.addVars(D.AP[d].size(), GRB_BINARY);
    //selected shortest path to any destination
    z[t] = new GRBVar *[nV];
    for (u = 0; u < nV; ++u) z[t][u] = model.addVars(nE, GRB_BINARY);
    //out traffic for legacy switch
    yout[t] = new GRBVar *[nV];       for (u = 0; u < nV; ++u) yout[t][u] = model.addVars(nV);
    //distance between node u and v
    h[t] = new GRBVar *[nV];
    for (u = 0; u < nV; ++u) h[t][u] = model.addVars(nV);
    //the link weight
    psi[t] = model.addVars(nE, GRB_INTEGER);
}
//Constraints
//One time switch upgrade (see Code 1)
for(t = 0; t < T; ++t) {
    //Switch Upgrade - maximum Budget per stage (see Code 1)
    //Multi-path traffic routing
    //flow conservation
    for (d = 0; d < nD; ++d) {
        for (k = 0; k < D.AP[d].size(); ++k) {
            for (u = 0; u < nV; ++u) {
                yuv = 0;
                for (l = 0; l < nE; ++l) if (G.E[l].u - 1 == u) yuv += y[t][d][k][l];
                yvu = 0;
                for (l = 0; l < nE; ++l) if (G.E[l].v - 1 == u) yvu += y[t][d][k][l];
                model.addConstr(yuv - yvu == NodeType[u][d]);
            }
        }
    }
    //Delay
    for (d = 0; d < nD; ++d) {
        for (k = 0; k < D.AP[d].size(); ++k) {
            delay = 0;
            for (l = 0; l < nE; ++l) delay += y[t][d][k][l] * G.E[l].w;
```

```
                model.addConstr(delay <= ceil(D.D[d].delta * SIGMA));
            }
        }
        //for link-disjoint paths only
        for (d = 0; d < nD; ++d) {
            if (isDP[d] == true) {
                for (l = 0; l < nE; ++l) {
                    for (k = 0; k < D.AP[d].size() - 1; ++k) {
                        for (m = k + 1; m < D.AP[d].size(); ++m) model.addQConstr(pp[t][d][k] *
y[t][d][k][l] + pp[t][d][m] * y[t][d][m][l] <= 1);
                    }
                }
                np = 0;
                for (k = 0; k < D.AP[d].size(); ++k) np += pp[t][d][k];
                model.addConstr(np >= 2);                    {
            }
        }
        //Path flow
        for (d = 0; d < nD; ++d) {
            fd = 0;
            for (k = 0; k < nK; ++k) {
                model.addConstr(pp[t][d][k] <= fp[t][d][k]);
                model.addConstr(fp[t][d][k] <= pp[t][d][k] * D.D[d].omega * mu[t]);
                model.addConstr(fp[t][d][k] >= 0);
                fd += fp[t][d][k];
            }
            model.addConstr(fd == D.D[d].omega * mu[t]);
        }
        //MLU
        for (l = 0; l < nE; ++l) {
            GRBQuadExpr fuv = 0;
            for (d = 0; d < nD; ++d) {
                for (k = 0; k < D.AP[d].size(); ++k) fuv += y[t][d][k][l] * fp[t][d][k];
            }
            model.addQConstr(fuv <= G.E[l].c / G.E[l].b * MLU * n[t][l]);
        }
    }
    //The objective (see Code 1)
    ...
```

C.2.2. Solution for Small Example

```
Academic license - for non-commercial use only - expires 2023-02-03
Gurobi Optimizer version 9.1.2 build v9.1.2rc0 (mac64[arm])
Thread count: 8 physical cores, 8 logical processors, using up to 8 threads
Optimize a model with 197 rows, 378 columns and 542 nonzeros
Model fingerprint: 0x648f19b4
Model has 192 quadratic constraints
Model has 400 general constraints
Variable types: 138 continuous, 240 integer (208 binary)
Coefficient statistics:
  Matrix range     [1e+00, 2e+02]
  QMatrix range    [1e+00, 1e+00]
  QLMatrix range   [1e+00, 5e+00]
  Objective range  [1e+00, 1e+00]
  Bounds range     [1e+00, 1e+00]
  RHS range        [1e+00, 7e+04]
  QRHS range       [1e+00, 1e+00]
Presolve added 128 rows and 74 columns
Presolve time: 0.01s
Presolved: 853 rows, 644 columns, 2889 nonzeros
Presolved model has 160 SOS constraint(s)
Variable types: 329 continuous, 315 integer (283 binary)

Root relaxation: objective 0.000000e+00, 93 iterations, 0.00 seconds

    Nodes    |    Current Node    |     Objective Bounds      |     Work
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time

     0     0    0.00000    0   37          -    0.00000      -     -    0s
H    0     0                       10.0000000    0.00000   100%     -    0s
     0     0    4.80000    0   71   10.00000    4.80000  52.0%     -    0s
H    0     0                        8.0000000    4.80000  40.0%     -    0s
     0     0    4.80000    0   51    8.00000    4.80000  40.0%     -    0s
H    0     0                        7.0000000    4.80000  31.4%     -    0s
     0     0    4.80000    0   55    7.00000    4.80000  31.4%     -    0s
     0     0    4.80000    0   57    7.00000    4.80000  31.4%     -    0s
     0     0    4.80000    0   12    7.00000    4.80000  31.4%     -    0s
     0     0    4.80000    0   11    7.00000    4.80000  31.4%     -    0s
     0     0    4.80000    0   14    7.00000    4.80000  31.4%     -    0s
     0     0    4.80000    0   38    7.00000    4.80000  31.4%     -    0s
     0     0    4.80000    0    9    7.00000    4.80000  31.4%     -    0s
     0     0    4.80000    0   45    7.00000    4.80000  31.4%     -    0s
     0     0    4.80000    0    8    7.00000    4.80000  31.4%     -    0s
     0     0    4.80000    0   28    7.00000    4.80000  31.4%     -    0s
     0     0    4.80000    0   14    7.00000    4.80000  31.4%     -    0s
     0     0    4.80000    0   26    7.00000    4.80000  31.4%     -    0s
     0     0    4.80000    0   14    7.00000    4.80000  31.4%     -    0s
     0     0    4.80000    0   14    7.00000    4.80000  31.4%     -    0s
```

Figure C.4. Solution for Example in Figure 4.1.

```
     0     0    4.80000    0   36    7.00000    4.80000  31.4%     -    0s
     0     0    6.00000    0   45    7.00000    6.00000  14.3%     -    0s
     0     0    6.00000    0   17    7.00000    6.00000  14.3%     -    0s
     0     0     cutoff    0         7.00000    7.00000  0.00%     -    0s

Cutting planes:
  Gomory: 5
  Cover: 7
  Implied bound: 31
  Clique: 8
  MIR: 16
  Flow cover: 4
  RLT: 32
  Relax-and-lift: 1

Explored 1 nodes (2990 simplex iterations) in 0.16 seconds
Thread count was 8 (of 8 available processors)

Solution count 4: 7 7 8 10

Optimal solution found (tolerance 1.00e-04)
Best objective 7.000000000000e+00, best bound 7.000000000000e+00, gap 0.0000%
Stage-1
        upgraded switches = { 2 4 5 } : 3 l-switches
        #on-cables = 7  #off-cables = 25  ES = 78.125%
        Demands & Paths:
           [s=1, t=3; vol=6]
                (1,4) (4,3) => vol = 3
                (1,2) (2,3) => vol = 3
           [s=2, t=4; vol=6]
                (2,1) (1,4) => vol = 5
                (2,5) (5,4) => vol = 1
        Multiple Paths = 0 over 2 demands => 0 %
        Disjoint Paths = 2 over 2 demands => 100 %
        (1,2):1{1}      (2,1):0{1}
        (1,4):0{2}      (4,1):0{1}
        (1,5):1{1}      (5,1):0{1}
        (2,5):1{1}      (5,2):0{1}
        (2,3):1{1}      (3,2):0{2}
        (3,4):1{1}      (4,3):0{1}
        (3,5):0{2}      (5,3):1{1}
        (4,5):0{1}      (5,4):1{1}

Total Upgrades = 60%
Average Energy Saving = 78.125%
```

Figure C.5. Solution for Example in Figure 4.1 (cont.).

C.3. MIP-3

C.3.1. Code Implementation

```
...
//Decision variables
for(t = 0; t < T; ++t) {
    see Code 1
    //all sc paths carry the same copy of control packets
    m[t] = model.addVars(nE, GRB_INTEGER);
    //controller status: deployed at switch or not
    c[t] = model.addVars(nV, GRB_BINARY);
    //all upgraded switch up to stage t
    sc[t] = model.addVars(nV, GRB_BINARY);
    //s-c association
    a[t] = new GRBVar *[nV];
    for (u = 0; u < nV; ++u) a[t][u] = model.addVars(nV, GRB_BINARY);
    //selected shortest path for each s-s demand
    z[t] = new GRBVar *[nDss];
    d = 0; j = 0;
    do {
        if (D.D[j].type == ss) {
            z[t][d] = model.addVars(nE, GRB_BINARY);
            Dss.push_back(j);
            ++d;
        }
        ++j;
    } while (d < nDss);
    //selected shortest path for each s-c demand
    zd[t] = new GRBVar **[nD];
    for (d = 0; d < nD; ++d) {
        zd[t][d] = new GRBVar *[K[d]];
        for (k = 0; k < K[d]; ++k) zd[t][d][k] = model.addVars(nE, GRB_BINARY);
    }
    //path indicator for active path
    PP[t] = new GRBVar *[nD];
    for (d = 0; d < nD; ++d) PP[t][d] = model.addVars(K[d], GRB_BINARY);
    //s-c traffic volume per active path
    SCvol[t] = new GRBVar *[nD];
    for (d = 0; d < nD; ++d) SCvol[t][d] = model.addVars(K[d]);
    //s-c traffic volume per TLDP
    SCv[t] = new GRBVar *[nD];
    for (d = 0; d < nD; ++d) SCv[t][d] = model.addVars(K[d]);
}
//Constraints
//One time switch upgrade (see Code 1)
for(t = 0; t < T; ++t) {
    //Switch Upgrade - maximum Budget per stage (see Code 1)
    //Controller Placement
    //each s-switch is assigned to exactly one controller
    for (u = 0; u < nV; ++u) {
        asc = 0;
        for (v = 0; v < nV; ++v) asc += a[t][u][v];
        model.addConstr(asc == s[t][u]);
    }
    //a controller manages only an s-switch if it is deployed
    for (u = 0; u < nV; ++u) {
```

```
        for (v = 0; v < nV; ++v) {
            // there is an association if controller v is deployed
            model.addConstr(a[t][u][v] <= sc[t][v]);
            // there is an association if node u is an s-switch
            model.addConstr(a[t][u][v] <= s[t][u]);
        }
    }
    //each controller must manage at least one s-switch
    for (u = 0; u < nV; ++u) {
        asc = 0;
        for (v = 0; v < nV; ++v) asc += a[t][v][u];
        model.addConstr(asc >= sc[t][u]);
    }
    //capacity of each controller
    for (u = 0; u < nV; ++u) {
        asc = 0;
        for (v = 0; v < nV; ++v) asc += a[t][v][u] * ceil(G.V[v].Omega * mu[t]);
        model.addConstr(asc <= G.V[u].Theta);
    }
    for (u = 0; u < nV; ++u) {
        //an s-switch attached to controller automatically be mapped to the controller
        model.addQConstr(s[t][u] * sc[t][u] <= a[t][u][u]);
        //a controller must be linked/co-located with an s-switch
        model.addConstr(sc[t][u] <= s[t][u]);
    }
    //Active/Backup Traffic Routing
    //flow conservation for s-s traffic demand (see Code 1)
    //delay for s-s traffic demand (see Code 1)
    //flow conservation - from an s-switch to exactly one controller
    for (d = 0; d < nD; ++d) {
        for (k = 0; k < K[d]; ++k) {
            for (u = 0; u < nV; ++u) {
                yuv = 0;
                for (l = 0; l < nE; ++l) if (G.E[l].u - 1 == u) yuv += zd[t][d][k][l];
                yvu = 0;
                for (l = 0; l < nE; ++l) if (G.E[l].v - 1 == u) yvu += zd[t][d][k][l];
                model.addGenConstrIndicator(a[t][D.D[d].s - 1][D.D[d].t - 1], true, yuv - yvu ==
NodeType[u][d]);
                model.addGenConstrIndicator(a[t][D.D[d].t - 1][D.D[d].s - 1], true, yuv - yvu ==
NodeType[u][d]);
            }
        }
    }
    //link-disjoint paths for s-c demands
    for (d = 0; d < nD; ++d) {
        if (K[d] > 1) {
            for (l = 0; l < nE; ++l) {
                for (k = 0; k < K[d] - 1; ++k) {
                    for (j = k + 1; j < K[d]; ++j) {
                        model.addGenConstrIndicator(a[t][D.D[d].s - 1][D.D[d].t - 1], true, zd[t][d][k][l]
+ zd[t][d][j][l] <= 1);
                        model.addGenConstrIndicator(a[t][D.D[d].t - 1][D.D[d].s - 1], true, zd[t][d][k][l]
+ zd[t][d][j][l] <= 1);
                    }
                }
            }
        }
    }
    //Delay for s-c demands
    for (d = 0; d < nD; ++d) {
        for (k = 0; k < K[d]; ++k) {
            delay = 0;
```

```
                for (l = 0; l < nE; ++l) delay += zd[t][d][k][l] * G.E[l].w;
                model.addGenConstrIndicator(a[t][D.D[d].s - 1][D.D[d].t - 1], true, delay <=
ceil(D.D[d].delta * SIGMA));
                model.addGenConstrIndicator(a[t][D.D[d].t - 1][D.D[d].s - 1], true, delay <=
ceil(D.D[d].delta * SIGMA));
            }
        }
        //active path
        for (d = 0; d < nD; ++d) {
            if (K[d] > 0) {
                yuv = 0;
                for (k = 0; k < K[d]; ++k) yuv += PP[t][d][k];
                model.addGenConstrIndicator(a[t][D.D[d].s - 1][D.D[d].t - 1], true, yuv == 1);
                model.addGenConstrIndicator(a[t][D.D[d].t - 1][D.D[d].s - 1], true, yuv == 1);
            }
        }
        //s-c traffic volume
        for (d = 0; d < nD; ++d) {
            for (k = 0; k < K[d]; ++k) {
                model.addGenConstrIndicator(a[t][D.D[d].s - 1][D.D[d].t - 1], true, SCvol[t][d][k] ==
PP[t][d][k] * ceil(G.V[D.D[d].s - 1].Omega * mu[t]) * BETA);
                model.addGenConstrIndicator(a[t][D.D[d].t - 1][D.D[d].s - 1], true, SCvol[t][d][k] ==
PP[t][d][k] * ceil(G.V[D.D[d].t - 1].Omega * mu[t]) * BETA);
                model.addGenConstrIndicator(a[t][D.D[d].s - 1][D.D[d].t - 1], true, SCv[t][d][k] ==
ceil(G.V[D.D[d].s - 1].Omega * mu[t]) * BETA);
                model.addGenConstrIndicator(a[t][D.D[d].t - 1][D.D[d].s - 1], true, SCv[t][d][k] ==
ceil(G.V[D.D[d].t - 1].Omega * mu[t]) * BETA);
            }
        }
        //MLU (S-S + S-C)
        for (l = 0; l < nE; ++l) {
            fall = 0; //all paths are used to carry the same traffic
            fone = 0; //only the active path carry the traffic
            for (d = 0; d < nDss; ++d) {
                fall += z[t][d][l] * D.D[Dss[d]].omega * mu[t];
                fone += z[t][d][l] * D.D[Dss[d]].omega * mu[t];
            }
            for (d = 0; d < nD; ++d) {
                for (k = 0; k < K[d]; ++k) {
                    fall += zd[t][d][k][l] * SCv[t][d][k];
                    fone += zd[t][d][k][l] * SCvol[t][d][k];
                }
            }
            model.addQConstr(fall <= G.E[l].c / G.E[l].b * MLU * m[t][l]);
            model.addQConstr(fone <= G.E[l].c / G.E[l].b * MLU * n[t][l]);
        }
        //on-cable lower and upper bound
        for (l = 0; l < nE; ++l) {
            model.addConstr(m[t][l] <= G.E[l].b);
            model.addConstr(m[t][l] >= 0);
        }
    }
    //The objective (see Code 1)
    ...
```

C.3.2. Solution for Small Example

```
Academic license - for non-commercial use only - expires 2023-02-03
Gurobi Optimizer version 9.1.2 build v9.1.2rc0 (mac64[arm])
Thread count: 8 physical cores, 8 logical processors, using up to 8 threads
Optimize a model with 894 rows, 1470 columns and 3134 nonzeros
Model fingerprint: 0xbc13e931
Model has 66 quadratic constraints
Model has 1816 general constraints
Variable types: 356 continuous, 1114 integer (1018 binary)
Coefficient statistics:
  Matrix range     [1e+00, 1e+02]
  QMatrix range    [1e+00, 1e+00]
  QLMatrix range   [1e+00, 6e+00]
  Objective range  [1e+00, 1e+00]
  Bounds range     [1e+00, 1e+00]
  RHS range        [1e+00, 8e+02]
Presolve added 123 rows and 0 columns
Presolve removed 0 rows and 461 columns
Presolve time: 0.05s
Presolved: 1355 rows, 1290 columns, 5822 nonzeros
Variable types: 274 continuous, 1016 integer (534 binary)
Found heuristic solution: objective 72.0000000

Root relaxation: objective 2.963084e+01, 315 iterations, 0.00 seconds

    Nodes    |    Current Node    |     Objective Bounds      |     Work
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time

     0     0   29.63084    0  111   72.00000   29.63084  58.8%     -    0s
H    0     0                       46.0000000   29.63084  35.6%     -    0s
     0     0   33.39306    0  101   46.00000   33.39306  27.4%     -    0s
H    0     0                       45.0000000   33.39306  25.8%     -    0s
     0     0   33.53161    0  105   45.00000   33.53161  25.5%     -    0s
     0     0   33.53161    0  105   45.00000   33.53161  25.5%     -    0s
H    0     0                       44.0000000   33.53161  23.8%     -    0s
H    0     0                       43.0000000   34.75804  19.2%     -    0s
     0     0   35.21488    0   97   43.00000   35.21488  18.1%     -    0s
     0     0   35.29734    0  105   43.00000   35.29734  17.9%     -    0s
H    0     0                       42.0000000   36.61960  12.8%     -    0s
H    0     0                       41.0000000   36.97427  9.82%     -    0s
     0     0   36.97427    0  103   41.00000   36.97427  9.82%     -    0s
     0     0   37.44170    0  111   41.00000   37.44170  8.68%     -    0s
     0     0   37.48312    0  103   41.00000   37.48312  8.58%     -    0s
     0     0   38.24787    0  100   41.00000   38.24787  6.71%     -    0s
     0     0   38.24787    0   83   41.00000   38.24787  6.71%     -    0s
```

Figure C.6. Solution for Example in Figure 5.1.

```
Stage t = 1:
        #s-switches 2 (22.2222 %)
        s-switches (#2) = {6(120) 8(120) }
        #controllers 1 (11.1111 %)
        Controllers (#1) = {6(240) }
        Controller-6: {6 8 }:55.5556 % (2 s-switches)
        Data Routing:
                (s=1, t=8: vol=5 ): <(1,2) (2,5) (5,8) >
                (s=2, t=7: vol=5 ): <(2,4) (4,7) >
                (s=5, t=3: vol=1 ): <(5,1) (1,3) >
                (s=6, t=1: vol=1 ): <(6,5) (5,1) >
                (s=6, t=8: vol=1 ): <(6,5) (5,8) >
        Control Routing:
                (s=6, t=6: vol=0.01 ) is via out-band!
                (s=6, t=8) => a:<(6,5) (5,8) >[ vol=0.01 ] b:<(6,9) (9,8) >[]
                (s=8, t=6) => a:<(8,5) (5,6) >[ vol=0.01 ] b:<(8,9) (9,6) >[]
        On-Cables:
                (1,2:5) = 2 cable(s)    (2,1:0) = 2 cable(s)
                (1,3:1) = 2 cable(s)    (3,1:0) = 2 cable(s)
                (1,5:0) = 2 cable(s)    (5,1:2) = 2 cable(s)
                (2,4:5) = 2 cable(s)    (4,2:0) = 2 cable(s)
                (2,5:5) = 2 cable(s)    (5,2:0) = 2 cable(s)
                (3,6:0) = 0 cable(s)    (6,3:0) = 0 cable(s)
                (4,7:5) = 2 cable(s)    (7,4:0) = 2 cable(s)
                (5,6:0.01) = 1 cable(s) (6,5:2.01) = 1 cable(s)
                (5,8:6.01) = 2 cable(s) (8,5:0.01) = 1 cable(s)
                (6,9:0) = 0 cable(s)    (9,6:0) = 0 cable(s)
                (7,8:0) = 0 cable(s)    (8,7:0) = 0 cable(s)
                (8,9:0) = 0 cable(s)    (9,8:0) = 0 cable(s)

        Energy Saving (on = 29, off = 19, total = 48) = 39.5833 %
        #off-links = 8; #on-links = 16
        MLU = 60.1 % with average of 2.50417 %
        TLDP = 2 out of 4 s-c demands (50 %)
        TLDPwcolocated = 2 out of 2 s-c demands (100 %)
         min load = 55.5556 %
         max load = 55.5556 %
         ave load = 55.5556 %
         std load = 0 %
        Data : Control => 99.6933 : 0.306748 (in %)

Stage t = 2:
        #s-switches 5 (55.5556 %)
        s-switches (#3) = {1(96) 4(96) 5(96) }
        #controllers 3 (33.3333 %)
        Controllers (#2) = {1(192) 4(192) }
        Controller-1: {1 5 8 }:100 % (3 s-switches)
```

Figure C.7. Solution for Example in Figure 5.1 (cont.).

```
        Controller-4: {4 }:33.3333 % (1 s-switches)
        Controller-6: {6 }:33.3333 % (1 s-switches)
        Data Routing:
                (s=1, t=8: vol=6 ): <(1,5) (5,8) >
                (s=2, t=7: vol=6 ): <(2,4) (4,7) >
                (s=5, t=3: vol=1.2 ): <(5,1) (1,3) >
                (s=6, t=1: vol=1.2 ): <(6,5) (5,1) >
                (s=6, t=8: vol=1.2 ): <(6,5) (5,8) >
        Control Routing:
                (s=1, t=1: vol=0.012 ) is via out-band!
                (s=1, t=5) => a:<(1,5) >[ vol=0.012 ] b:<(1,2) (2,5) >[]
                (s=1, t=8) => a:<(1,5) (5,8) >[ vol=0.012 ]
                (s=4, t=4: vol=0.012 ) is via out-band!
                (s=5, t=1) => a:<(5,1) >[ vol=0.012 ] b:<(5,2) (2,1) >[]
                (s=6, t=6: vol=0.012 ) is via out-band!
                (s=8, t=1) => a:<(8,5) (5,1) >[ vol=0.012 ]
        On-Cables:
                (1,2:0) = 0 cable(s)    (2,1:0) = 0 cable(s)
                (1,3:1.2) = 1 cable(s)  (3,1:0) = 0 cable(s)
                (1,5:6.024) = 2 cable(s)        (5,1:2.424) = 1 cable(s)
                (2,4:6) = 2 cable(s)    (4,2:0) = 0 cable(s)
                (2,5:0) = 0 cable(s)    (5,2:0) = 0 cable(s)
                (3,6:0) = 0 cable(s)    (6,3:0) = 0 cable(s)
                (4,7:6) = 2 cable(s)    (7,4:0) = 0 cable(s)
                (5,6:0) = 0 cable(s)    (6,5:2.4) = 1 cable(s)
                (5,8:7.212) = 2 cable(s)        (8,5:0.012) = 1 cable(s)
                (6,9:0) = 0 cable(s)    (9,6:0) = 0 cable(s)
                (7,8:0) = 0 cable(s)    (8,7:0) = 0 cable(s)
                (8,9:0) = 0 cable(s)    (9,8:0) = 0 cable(s)

        Energy Saving (on = 12, off = 36, total = 48) = 75 %
        #off-links = 16; #on-links = 8
        MLU = 72.12 % with average of 3.005 %
        TLDP = 2 out of 10 s-c demands (20 %)
        TLDPwcolocated = 2 out of 4 s-c demands (50 %)
         min load = 33.3333 %
         max load = 100 %
         ave load = 55.5556 %
         std load = 31.427 %
        Data : Control => 99.2366 : 0.763359 (in %)

Total s-switches = 55.5556 %
Total Controllers = 3 controllers
Average Energy Saving = 57.2917 %
Average MLU = 66.11 %
Average TLDP = 35 %
```

Figure C.8. Solution for Example in Figure 5.1 (cont.).

Bibliography

[1] K. Poularakis, G. Iosifidis, G. Smaragdakis, and L. Tassiulas, "One step at a time: Optimizing SDN upgrades in ISP networks," in *Proc. IEEE INFOCOM*, Atlanta, GA, USA, May 2017, pp. 1–9.

[2] J. Galán-Jiménez, "Legacy IP-upgraded SDN nodes tradeoff in energy-efficient hybrid IP/SDN networks," *Comp. Commun.*, vol. 114, pp. 106–123, Dec. 2017.

[3] Sandhya, Y. Sinha, and K. Haribabu, "A survey: Hybrid SDN," *J. Netw. Comput. Appl.*, vol. 100, pp. 35–55, Dec. 2017.

[4] R. Amin, M. Reisslein, and N. Shah, "Hybrid SDN networks: A survey of existing approaches," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 4, pp. 3259–3306, 2018.

[5] S. Khorsandroo, A. G. Sanchez, A. S. Tosun, J. M. A. Rodríguez, and R. Doriguzzi-Corin, "Hybrid SDN evolution: A comprehensive survey of the state-of-the-art," *Comput. Netw.*, p. 107981, 2021.

[6] S. Jain *et al.*, "B4: experience with a globally-deployed software defined WAN," in *ACM SIGCOMM*, Hong Kong, China, Aug. 2013, pp. 3 – 14.

[7] C.-Y. Hong *et al.*, "Achieving high utilization with software-driven WAN," in *Proc. ACM SIGCOMM*, Hong Kong, China, Aug 2013, pp. 15–26.

[8] S. Natarajan, A. Ramaiah, and M. Mathen, "A software defined cloud-gateway automation system using openflow," in *2013 IEEE 2nd International Conference on Cloud Networking (CloudNet)*. IEEE, 2013, pp. 219–226.

[9] S. Cash, V. Jain, L. Jiang, A. Karve, J. Kidambi, M. Lyons, T. Mathews, S. Mullen, M. Mulsow, and N. Patel, "Managed infrastructure with ibm cloud openstack services," *IBM Journal of Research and Development*, vol. 60, no. 2-3, pp. 6–1, 2016.

[10] Z. Latif, K. Sharif, F. Li, M. M. Karim, S. Biswas, and Y. Wang, "A comprehensive survey of interface protocols for software defined networks," *Journal of Network and Computer Applications*, vol. 156, p. 102563, 2020.

[11] R. Maaloul, L. Chaari, and B. Cousin, "Energy saving in carrier-grade networks: A survey," *Computer Standards & Interfaces*, vol. 55, pp. 8–26, Jan. 2018.

[12] W. Fisher, M. Suchara, and J. Rexford, "Greening backbone networks: reducing energy consumption by shutting off cables in bundled links," in *Proc. ACM SIGCOMM*, New Delhi, India, Aug. 2010, pp. 29–34.

[13] G. Lin, S. Soh, K.-W. Chin, and M. Lazarescu, "Efficient heuristics for energy-aware routing in networks with bundled links," *Comput. Netw.*, vol. 57, no. 8, pp. 1774–1788, Jun. 2013.

[14] G. Lin, S. Soh, M. Lazarescu, and K.-W. Chin, "Power-aware routing in networks with delay and link utilization constraints," in *IEEE LCN*, Florida, USA, 2012, pp. 272–275.

[15] G. Lin, S. Soh, K.-W. Chin, and M. Lazarescu, "Energy aware two disjoint paths routing," *Journal of Network and Computer Applications*, vol. 43, pp. 27–41, 2014.

[16] J. Moulierac and T. K. Phan, "Optimizing IGP link weights for energy-efficiency in multi-period traffic matrices," *Comp. Commun.*, vol. 61, pp. 79–89, 2015.

[17] Y. Wei, X. Zhang, L. Xie, and S. Leng, "Energy-aware traffic engineering in hybrid SDN/IP backbone networks," *J. Commun. Netw.*, vol. 18, no. 4, pp. 559–566, Aug. 2016.

[18] N. Huin, M. Rifai, F. Giroire, D. L. Pacheco, G. Urvoy-Keller, and J. Moulierac, "Bringing energy aware routing closer to reality with SDN hybrid networks," *IEEE Trans. Green Commun. Netw.*, vol. 2, no. 4, pp. 1128 – 1139, Dec. 2018.

[19] Y. Hu, T. Luo, N. C. Beaulieu, and C. Deng, "The energy-aware controller placement problem in software defined networks," *IEEE Commun. Lett*, vol. 21, no. 4, pp. 741–744, 2016.

[20] A. Ruiz-Rivera, K.-W. Chin, and S. Soh, "GreCo: An energy aware controller association algorithm for software defined networks," *IEEE Commun. Lett.*, vol. 19, no. 4, pp. 541–544, Apr. 2015.

[21] "802.1AX-2020 – link aggregation," https://1.ieee802.org/tsn/802-1ax-rev, accessed: Sep 2021.

[22] K. Christensen, P. Reviriego, B. Nordman, M. Bennett, M. Mostowfi, and J. Maestro, "IEEE 802.3az: the road to energy efficient ethernet," *IEEE Commun. Mag.*, vol. 48, no. 11, pp. 50–56, Nov. 2010.

[23] M. Rodriguez-Perez, M. Fernandez-Veiga, S. Herreria-Alonso, M. Hmila, and C. Lopez-Garcia, "Optimum traffic allocation in bundled energy-efficient ethernet links," *IEEE Syst. J.*, vol. 12, no. 1, pp. 593–603, Mar. 2018.

[24] J. Moy *et al.*, "Ospf version 2," *STD 54, RFC 2328, April*, 1998.

[25] A. S. Tanenbaum and D. J. Wetherall, "Computer networks. 5th," *Pearson Education*, 2011.

[26] C. Hopps *et al.*, "Analysis of an equal-cost multi-path algorithm," RFC 2992, November, Tech. Rep., 2000.

[27] A. Iselt, A. Kirstadter, A. Pardigon, and T. Schwabe, "Resilient routing using mpls and ecmp," in *2004 Workshop on High Performance Switching and Routing, 2004. HPSR.* IEEE, 2004, pp. 345–349.

[28] J. Elmirghani, "ICT industry tackles growth in CO_2 emissions," 2015. [Online]. Available: http://greenict.ieee.org/blog/201509/ict-industry-tackles-growth-co2-emissions

[29] U.S. Energy Information Administration, "Annual energy outlook 2019," accessed on: Jul. 6, 2020. [Online]. Available: https://www.eia.gov/outlooks/aeo/data/browser/#/?id=3-AEO2020&cases=ref2020&sourcekey=0

[30] E. Ghazisaeedi and C. Huang, "Off-peak energy optimization for links in virtualized network environment," *IEEE Trans. Cloud Comput.*, vol. 5, no. 2, pp. 155–167, Apr.-Jun. 2017.

[31] M. Zhang, C. Yi, B. Liu, and B. Zhang, "GreenTE: Power-aware traffic engineering," in *Proc. 18th IEEE ICNP*, Kyoto, Japan, Oct. 2010, pp. 21–30.

[32] L. Belkhir and A. Elmeligi, "Assessing ICT global emissions footprint: Trends to 2040 & recommendations," *J. Cleaner Production*, vol. 177, pp. 448–463, Mar. 2018.

[33] M. Gupta and S. Singh, "Greening of the internet," in *Proc. ACM SIGCOMM*, Karlsruhe, Germany, Oct. 2003, pp. 19–26.

[34] M. F. Tuysuz, Z. K. Ankarali, and D. Gozupek, "A survey on energy efficiency in software defined networks," *Comput. Netw.*, vol. 113, pp. 188–204, Feb. 2017.

[35] D. B. Rawat and S. R. Reddy, "Software defined networking architecture, security and energy efficiency: A survey," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 1, pp. 325–346, 2017.

[36] "IEEE 802.1AX-2014 - IEEE standard for local and metropolitan area networks – link aggregation," 2014. [Online]. Available: https://standards.ieee.org/standard/802_1AX-2014.html

[37] J. Galán-Jiménez and A. Gazo-Cervero, "Designing energy-efficient link aggregation groups," *Ad Hoc Networks*, vol. 25, pp. 595–605, 2015.

[38] B. Fortz, J. Rexford, and M. Thorup, "Traffic engineering with traditional ip routing protocols," *IEEE Commun. Mag.*, vol. 40, no. 10, pp. 118–124, Oct. 2002.

[39] I. F. Akyildiz, A. Lee, P. Wang, M. Luo, and W. Chou, "A roadmap for traffic engineering in SDN-OpenFlow networks," *Comput. Netw.*, vol. 71, pp. 1–30, Oct. 2014.

[40] D. K. Hong, Y. Ma, S. Banerjee, and Z. M. Mao, "Incremental deployment of SDN in hybrid enterprise and ISP networks," in *Proc. Symp. on SDN Research*, Santa Clara, CA, USA, Dec. 2016, pp. 1–7.

[41] Y. Guo, Z. Wang, Z. Liu, X. Yin, X. Shi, J. Wu, Y. Xu, and H. J. Chao, "SOTE: Traffic engineering in hybrid software defined networks," *Comput. Netw.*, vol. 154, pp. 60–72, 2019.

[42] G. Wang, Y. Zhao, J. Huang, and Y. Wu, "An effective approach to controller placement in software defined wide area networks," *IEEE Trans. Netw. Service Manag.*, vol. 15, no. 1, pp. 344–355, 2017.

[43] G. Wang, Y. Zhao, J. Huang, and W. Wang, "The controller placement problem in software defined networking: A survey," *IEEE Network*, vol. 31, no. 5, pp. 21–27, Sep./Oct. 2017.

[44] Y. Guo, F. Kuipers, and P. Van Mieghem, "Link-disjoint paths for reliable QoS routing," *International Journal of Communication Systems*, vol. 16, no. 9, pp. 779–798, 2003.

[45] "OSPF standardization report," https://www.ietf.org/rfc/rfc2329.txt, accessed: Sep 2021.

[46] F. Hu, Q. Hao, and K. Bao, "A survey on software-defined network and OpenFlow: From concept to implementation," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 4, pp. 2181–2206, 2014.

[47] "OpenFLow," https://opennetworking.org/sdn-resources/customer-case-studies/openflow, accessed: Sep 2021.

[48] P. Goransson, C. Black, and T. Culver, *Software defined networks: a comprehensive approach*. Morgan Kaufmann, 2016.

[49] B. Pfaff, J. Pettit, T. Koponen, E. Jackson, A. Zhou, J. Rajahalme, J. Gross, A. Wang, J. Stringer, P. Shelar *et al.*, "The design and implementation of open vswitch," in *12th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 15)*, 2015, pp. 117–130.

[50] "Understand the openflow on catalyst 9000 series switches," https://www.cisco.com/c/en/us/support/docs/switches/catalyst-9300-series-switches/217210-understand-openflow-on-catalyst-9000-ser.html, accessed: Oct 2021.

[51] ONF, "OpenFlow switch specification: Version 1.3.3 (wire protocol 0x04)," ONF, Tech. Rep., Sep 2013. [Online]. Available: https://www.opennetworking.org/wp-content/uploads/2014/10/openflow-spec-v1.3.3.pdf

[52] H. Xu, X.-Y. Li, L. Huang, H. Deng, H. Huang, and H. Wang, "Incremental deployment and throughput maximization routing for a hybrid SDN," *IEEE/ACM Trans. Netw.*, vol. 25, no. 3, pp. 1861–1875, Jun. 2017.

[53] S. Ahmad and A. H. Mir, "Scalability, consistency, reliability and security in SDN controllers: A survey of diverse SDN controllers," *Journal of Network and Systems Management*, vol. 29, no. 9, pp. 1–59, 2021.

[54] P. Berde, M. Gerola, J. Hart, Y. Higuchi, M. Kobayashi, T. Koide, B. Lantz, B. O'Connor, P. Radoslavov, W. Snow *et al.*, "ONOS: towards an open, distributed SDN OS," in *Proceedings of the third workshop on Hot topics in software defined networking*, 2014, pp. 1–6.

[55] Ryu. [Online]. Available: https://ryu-sdn.org/

[56] J. Medved, R. Varga, A. Tkacik, and K. Gray, "Opendaylight: Towards a model-driven SDN controller architecture," in *Proceeding of IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks 2014*. IEEE, 2014, pp. 1–6.

[57] K.-K. Yap, M. Motiwala, J. Rahe, S. Padgett, M. Holliman, G. Baldus, M. Hines, T. Kim, A. Narayanan, A. Jain *et al.*, "Taking the edge off with espresso: Scale, reliability and programmability for global internet peering," in *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*, Los Angeles, CA, USA, Aug. 2017, pp. 432–445.

[58] P. Lin, J. Bi, and H. Hu, "BTSDN: BGP-based transition for the existing networks to SDN," *Wireless Personal Communications*, vol. 86, no. 4, pp. 1829–1843, 2016.

[59] S. Agarwal, M. Kodialam, and T. V. Lakshman, "Traffic engineering in software defined networks," in *Proc. IEEE INFOCOM*, Turin, Italy, Apr. 2013, pp. 2211 – 2219.

[60] T. Das, V. Sridharan, and M. Gurusamy, "A survey on controller placement in SDN," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 1, pp. 472–503, 2020.

[61] S.-C. Lin, P. Wang, I. F. Akyildiz, and M. Luo, "Towards optimal network planning for software-defined networks," *IEEE Trans. Mob. Comput.*, vol. 17, no. 12, pp. 2953–2967, 2018.

[62] B. Görkemli, S. Tatlıcıoğlu, A. M. Tekalp, S. Civanlar, and E. Lokman, "Dynamic control plane for SDN at scale," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 12, pp. 2688–2701, 2018.

[63] M. T. I. ul Huque, W. Si, G. Jourjon, and V. Gramoli, "Large-scale dynamic controller placement," *IEEE Trans. Netw. Service Manag.*, vol. 14, no. 1, pp. 63–76, 2017.

[64] P. Vizarreta, C. M. Machuca, and W. Kellerer, "Controller placement strategies for a resilient SDN control plane," in *2016 8th International Workshop on Resilient Networks Design and Modeling (RNDM)*. IEEE, 2016, pp. 253–259.

[65] D. Katz, K. Kompella, and D. Yeung, "Traffic engineering (te) extensions to ospf version 2," RFC 3630, September, Tech. Rep., 2003.

[66] K. Ishiguro, V. Manral, A. Davey, and A. Lindem, "Traffic engineering extensions to ospf version 3," *RFC5329, Work in Progress*, 2006.

[67] B. Heller, R. Sherwood, and N. McKeown, "The controller placement problem," *ACM SIGCOMM Computer Communication Review*, vol. 42, no. 4, pp. 473–478, 2012.

[68] G. Yao, J. Bi, Y. Li, and L. Guo, "On the capacitated controller placement problem in software defined networks," *IEEE Commun. Lett*, vol. 18, no. 8, pp. 1339–1342, 2014.

[69] Z. Guo, W. Chen, Y.-F. Liu, Y. Xu, and Z.-L. Zhang, "Joint switch upgrade and controller deployment in hybrid software-defined networks," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 5, pp. 1012–1028, 2019.

[70] T. Das and M. Gurusamy, "Incept: Incremental controller placement in software defined networks," in *2018 27th International Conference on Computer Communication and Networks (ICCCN)*. Hangzhou, China: IEEE, Jul - Aug. 2018, pp. 1–6.

[71] ——, "Resilient controller placement in hybrid SDN/legacy networks," in *Proc. IEEE Globecom*. Abu Dhabi, UAE: IEEE, Dec. 2018, pp. 1–7.

[72] ——, "Multi-objective control plane dimensioning in hybrid SDN/legacy networks," *IEEE Trans. Netw. Service Manag.*, 2021.

[73] H. Wang, Y. Li, D. Jin, P. Hui, and J. Wu, "Saving energy in partially deployed software defined networks," *IEEE Trans. Comput.*, vol. 65, no. 5, pp. 1578–1592, May 2016.

[74] A. Fernandez-Fernandez, C. Cervello-Pastor, and L. Ochoa-Aday, "Achieving energy efficiency: An energy-aware approach in SDN," in *Proc. IEEE Globecom*, Washington DC, USA, Dec. 2016, pp. 1–7.

[75] M. Caria, A. Jukan, and M. Hoffmann, "A performance study of network migration to SDN-enabled traffic engineering," in *Proc. IEEE Globecom*, Atlanta, GA, USA, Dec. 2013, pp. 1391 – 1396.

[76] T. Das, M. Caria, A. Jukan, and M. Hoffmann, "A techno-economic analysis of network migration to software-defined networking," *arXiv preprint arXiv:1310.0216*, 2013.

[77] M. Tanha, D. Sajjadi, R. Ruby, and J. Pan, "Traffic engineering enhancement by progressive migration to SDN," *IEEE Commun. Lett.*, vol. 22, no. 3, pp. 438–441, Mar. 2018.

[78] Publically available code. [Online]. Available: https://goo.gl/EXoZZZ

[79] The Internet Topology Zoo. [Online]. Available: http://www.topology-zoo.org/dataset.html

[80] M. Roughan, "Simplifying the synthesis of internet traffic matrices," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 35, no. 5, pp. 93–96, Oct. 2005.

[81] Yin Zhang's Abilene Traffic Matrix. [Online]. Available: http://www.cs.utexas.edu/~yzhang/research/AbileneTM/

[82] M. Roughan. (2017) Internet Traffic Matrices. [Online]. Available: http://www.maths.adelaide.edu.au/matthew.roughan/\project/traffic_matrix/

[83] "Gurobi optimizer reference manual," http://www.gurobi.com, accessed: Nov 2019.

[84] L. Hiryanto, S. Soh, K.-W. Chin, and M. Lazarescu, "Green multi-stage upgrade for bundled-link SDNs with budget constraint," in *IEEE ITNAC*, Auckland, New Zealand, Nov. 2019, pp. 32 – 38.

[85] ——, "Green multi-stage upgrade for bundled-link SDNs with budget and delay constraints," *IEEE Trans. Green Commun. Netw.*, vol. 5, no. 3, pp. 1410–1425, 2021.

[86] S. Even, A. Itai, and A. Shamir, "On the complexity of time table and multi-commodity flow problems," in *Proc. 16th Annual Symposium on Foundations of Computer Science (SFCS)*, Oct. 1975, pp. 184–193.

[87] P. Toth and S. Martello, *Knapsack problems: Algorithms and computer implementations*. Wiley, 1990.

[88] J. Y. Yen, "Finding the k shortest loopless paths in a network," *Management Science*, vol. 17, no. 11, Jul. 1971.

[89] L. Hiryanto, S. Soh, K.-W. Chin, S. Pham, and M. Lazarescu, "Green multi-stage upgrade for bundled-links SDN/OSPF-ECMP networks," in *IEEE ICC*, Montreal, Canada, Jun. 2021.

[90] L. Hiryanto, S. Soh, K.-W. Chin, D.-S. Pham, and M. M. Lazarescu, "Multi-path routing in green multi-stage upgrade for bundled-links SDN/OSPF-ECMP networks," *IEEE Access*, vol. 9, pp. 99 073–99 091, 2021.

[91] "Constraint," https://www.gurobi.com/documentation/9.5/refman/constraints.html#subsubsection:QuadraticConstraints, accessed: May 2022.

[92] R. Mahajan, N. Spring, D. Wetherall, and T. Anderson, "Inferring link weights using end-to-end measurements," in *2nd ACM SIGCOMM Workshop on Internet measurement*, 2002, pp. 231–236.

[93] Y. Yang, D. Wang, D. Pan, and M. Xu, "Wind blows, traffic flows: Green internet routing under renewable energy," in *Proc. IEEE INFOCOM*, San Francisco, CA, USA, Apr. 2016, pp. 1–9.

[94] J. Xie, F. R. Yu, T. Huang, R. Xie, J. Liu, C. Wang, and Y. Liu, "A survey of machine learning techniques applied to software defined networking (SDN): Research issues and challenges," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 1, pp. 393–430, 2018.