

Verbesserung des Routings in Satellitenkonstellationen durch Software-Defined Networking und Bestärkendes Lernen

Manuel M. H. Roth

Vollständiger Abdruck der von der Fakultät für Elektrotechnik und Informationstechnik der Universität der Bundeswehr München zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs (Dr.-Ing.)

angenommenen Dissertation.

1. Prüfer: Universitätsprofessor Dr.-Ing. Andreas Knopp, MBA
2. Prüfer: Universitätsprofessor Dr.-Ing. Giovanni Giambene

Die Dissertation wurde am 11.09.2025 bei der Universität der Bundeswehr München eingereicht und durch die Fakultät für Elektrotechnik und Informationstechnik am 11.02.2026 angenommen. Die mündliche Prüfung fand am 26.03.2026 statt.

Dissertation

Enhancing Routing in Satellite Constellations with Software-Defined Networking and Reinforcement Learning

Manuel M. H. Roth



Faculty of Electrical Engineering and Information Technology
Institute of Information Technology
Chair of Signal Processing

Supervisor and first reviewer	Professor Dr.-Ing. Andreas Knopp, MBA
Second reviewer	Professor Dr.-Ing. Giovanni Giambene

August 2025

Abstract

The emergence of interconnected satellite mega-constellations has highlighted the need for optimized routing and network management solutions to enable reliable broadband connectivity. Given the intended integration into terrestrial networks in 5G and beyond architectures, the demand for efficient routing to support more users and higher traffic volumes becomes increasingly important. To achieve higher system throughput while maintaining quality of service requirements and ensuring constellation autonomy, tailored routing solutions are required. Dynamic network characteristics as well as on-board limitations, such as the processing power, have to be accommodated in the design. In this thesis, novel routing protocols and traffic engineering algorithms are proposed to address these challenges. A load-balancing in-orbit solution based on distributed software-defined networking is presented to enable flexible and adaptive routing strategies. The performance of the protocol is evaluated in different scenarios using a system-level simulator specifically developed to enable fine-grained analyses of performance indicators. The viability and validity of the approach is underlined by a demonstration in a testbed based on space-qualified hardware. In addition, traffic engineering algorithms can be employed to further improve load-balancing performance. By proactively avoiding potential bottlenecks, efficient routing configurations can be found which enable higher throughput. This optimization, described by the multi-commodity flow problem, is NP-hard. Given the limited on-board processing power, complex rule-based approaches are impractical. Conventional optimization techniques are not well-suited due to their computational complexity and limited adaptability. Therefore, deep reinforcement learning approaches are derived from first principles for the given scenario. Underlying traffic patterns are learned directly from the environment. A novel soft actor-critic method, which operates on sets of candidate paths, is proposed. Its learned traffic engineering capabilities surpass the performance of comparable approaches. Overall, the proposed routing protocol and data-driven load-balancing methods extend the state of the art and provide valuable insights into network optimization in satellite constellations.

Supervisor: Professor Dr.-Ing. Andreas Knopp, MBA

Kurzfassung

Die Entwicklung vernetzter Satelliten-Megakonstellationen hat den Bedarf an optimierten Routing- und Netzwerkmanagementlösungen für zuverlässige Breitbandkonnektivität verdeutlicht. Im Hinblick auf die beabsichtigte Integration dieser Satellitennetze in terrestrische 5G und darüber hinausgehende Architekturen wird die Nachfrage nach effizientem Routing, welches mehr Nutzer und ein höheres Datenvolumen ermöglicht, zunehmend bedeutsam. Um höheren Systemdurchsatz unter Einhaltung von Dienstgüte-Anforderungen zu erzielen, und die Autonomie der Konstellation zu sichern, sind maßgeschneiderte Routingansätze erforderlich. Dynamische Netzwerkmerkmale als auch Einschränkungen an Bord, wie z.B. die Prozessorleistung, müssen in das Design eingezogen werden. Die in dieser Arbeit vorgeschlagenen Routingprotokolle und Algorithmen für Verkehrssteuerung lösen diese Herausforderungen. Es wird ein lastausgleichender bordeigener Ansatz präsentiert, der auf verteiltem Software-Defined Networking beruht und flexible sowie adaptive Routingstrategien ermöglicht. Die Leistungsfähigkeit dieses Protokolls wird in verschiedenen Szenarien evaluiert mittels eines speziell zu diesem Zwecke entwickelten Simulators auf Systemebene. Dieser ermöglicht detaillierte Analysen relevanter Leistungsindikatoren. Die Umsetzbarkeit und Gültigkeit des Ansatzes wird durch eine Demonstration auf einem Versuchsstand aus weltraumtauglicher Hardware belegt. Des Weiteren können Algorithmen zur Verkehrssteuerung verwendet werden, um die Lastverteilung weiter zu verbessern. Indem potenzielle Engpässe proaktiv vermieden werden, können effiziente Routingkonfigurationen gefunden werden, die einen höheren Durchsatz ermöglichen. Diese Optimierung, welcher das Multi-Commodity-Strömungsproblem zugrunde liegt, ist NP-schwer. Aufgrund der limitierten Rechenkapazität an Bord sind komplexe regelbasierte Ansätze nicht anwendbar. Konventionelle Optimierungsmethoden sind nicht gut geeignet wegen ihrer Komplexität und begrenzten Anpassungsfähigkeit. Deshalb werden Ansätze des bestärkenden Lernens aus ersten Prinzipien für das gegebene Szenario abgeleitet. Zugrundeliegende Verkehrsmuster werden so direkt aus der Umgebung gelernt. Eine neuartige Methode basierend auf Soft-Actor-Critic, welche mit einer Auswahl von Kandidatenpfaden operiert, wird vorgeschlagen. Die gelernten Fähigkeiten zur Verkehrssteuerung übertreffen die Leistungen vergleichbarer Ansätze. Insgesamt erweitern das vorgeschlagene Routingprotokoll und die Verkehrssteuerung den aktuellen Stand der Technik und liefern wertvolle Erkenntnisse für die Netzwerkoptimierung in Satellitenkonstellationen.

Betreuer: Universitätsprofessor Dr.-Ing. Andreas Knopp, MBA

Acknowledgments

First and foremost, I would like to express my gratitude to my supervisors professor Andreas Knopp, and Thomas Delamotte for their continuous support and their interest in the topic. Andreas, I appreciate the time you have taken to listen to my plans, and to bring structure and focus to my research. I am also grateful for your trust: you were open to my research ideas and gave me the opportunity to present in front of large audiences. Thomas, thank you for your enthusiasm and pragmatic approach, which relieved a lot of stress on my side. I would also like to thank my fellow doctoral researchers at the chair of signal processing who welcomed me warmly to the group and with whom I survived many conferences.

Moreover, I would like to thank my great colleagues at DLR. I cannot list all of them, so I will focus on direct collaborators of this work. Special thanks to my group leader Hermann Bischl who launched my research in this topic. Hermann provided guidance through this external PhD, and I owe the completion of this work to his efforts to acquire related projects. I also want to thank Sandro Scalise, my department head, for keeping my bureaucratic overhead low, which in itself is an incredible feat, and for believing in the topic. Thank you Hartmut Brandt and Stefan Erl for being safe havens in the treacherous seas of simulator software implementations. I am also grateful to Svilen Dimitrov for inspiring the topic and helping me get started with research, to Anupama Hegde for technical discussions as well as sage advice, and Thomas Jerkovits for his insatiable interest in reinforcement learning, pushing us forward. Additionally, I want to thank my brother in arms, Marcel Grec, with whom I started my DLR journey, and who I still share an office with. I will always remember the smell of kerosene in the morning.

I am very grateful to my amazing family. Thank you very much to my parents Pia and Armin. You are the parents everybody wishes for. All of my work is the result of the happiness you have provided me with. Thank you very much to my brother Simon who always has my back, no matter the distance. I would also like to thank some of my friends outside of work, who followed my progress on this endeavor. Notably, Lukas and Christina, Paul, Jannik, Enno, Felix, Thomas, Simon, as well as my former room mates, and all of the friends I made along the way in Munich. Above all, I have to thank Liliia, the best of us. Thank you for your love and support. Without you this might have taken forever, and I am still waiting for you to update my writing streak. Lastly, I would like to dedicate this dissertation to my grandparents Margareta and Alfons Zöller, as well as Irma and Josef Roth, who were sadly not able to see to its completion. I wish we could have celebrated this milestone together. Gone but never lost.

Manuel Markus Hubert Roth
Munich, August 2025

Contents

Abstract	v
Kurzfassung	vii
Acknowledgments	ix
List of Acronyms	xiii
List of Operators	xv
List of Symbols	xvi
1. Introduction	1
1.1. Background and Motivation	1
1.2. Problem Description	2
1.2.1. Dynamic Network Topology and Traffic Requirements	2
1.2.2. Efficient Routing Configurations	4
1.2.3. Research Questions	5
1.3. Key Concepts and Proposed Approaches	5
1.4. Research Contributions	11
1.5. Thesis Outline	12
2. Routing Fundamentals for Satellite Constellation Networks	15
2.1. Satellite Constellation Network Architecture	15
2.1.1. Space Segment	15
2.1.2. Ground Segment	20
2.1.3. Connectivity	22
2.1.4. Traffic Models	24
2.1.5. Unique Challenges in Satellite Constellation Networks	26
2.2. Routing Protocol Overview	28
2.2.1. State-of-the-art Routing Protocols	28
2.2.2. Standardization Activity	30
2.2.3. Key Performance Indicators	31
2.3. Network Optimization Theory	35
2.3.1. Graph-Theoretic Representations	35
2.3.2. Multi-Commodity Flow Problem	38
3. Distributed Load-balanced Software-Defined Networking	43
3.1. Protocol Design	43
3.1.1. Protocol Architecture Details	46

3.1.2. Geographical Routing Insights	51
3.1.3. Set-based Source-Routing Insights	56
3.2. Routing Schemes	62
3.2.1. Intra-cluster Routing	62
3.2.2. Inter-cluster Routing	65
3.3. Comparative Quantitative Evaluation	67
3.3.1. System-level Constellation Routing Simulator	67
3.3.2. Performance Evaluation	68
3.4. Demonstration of Feasibility	77
3.4.1. Demonstrator Setup	77
3.4.2. Software-Defined Networking Controller	77
3.4.3. Functional Evaluation	78
3.5. Key Findings and Discussion	83
4. Reinforcement Learning-based Traffic Engineering	89
4.1. Machine Learning for Traffic Engineering Enhancements	89
4.1.1. Viability for Satellite Constellation Networks	91
4.1.2. Derivation from First Principles	93
4.2. Deep Reinforcement Learning for Hop-based Routing	98
4.2.1. Centralized Constrained Deep Q-Networks	98
4.2.2. Fully-Decentralized Multi-Agent Deep Q-Networks	101
4.2.3. Limitations and Further Perspectives	106
4.3. Deep Reinforcement Learning for Path-based Routing	107
4.3.1. Soft Actor-Critic Framework	108
4.3.2. Path-based Architecture	110
4.3.3. Performance Evaluation	114
4.4. Implications and Discussion	121
5. Conclusion	125
5.1. Summary	125
5.2. Future Perspectives	126
A. Exemplary On-Board Processing Hardware	129
B. Configuration of Simulator	131
C. Deep Learning Framework	133
List of Figures	135
List of Tables	139
Bibliography	141

List of Acronyms

AI	Artificial Intelligence
BFS	Breadth-First Search
BGP	Border Gateway Protocol
CDRL	Centralized DRL
CL-DC	Centralized Learning - Decentralized Control
CNN	Convolutional Neural Network
Dec-POMDP	Decentralized Partially Observable MDP
DL	Deep Learning
DNN	Deep Neural Network
DP	Dynamic Programming
DQN	Deep Q-Network
DRL	Deep Reinforcement Learning
DUT	Device Under Test
EMP	Equal Multi-Path
ESL	Earth-Satellite Link
F-MCFP	Fractional Multi-Commodity Flow Problem
FD-MADRL	Fully-Distributed Multi-Agent DRL
GEVR	God's Eye View Routing
GNN	Graph Neural Network
GW	Gateway station
I-MCFP	Integer Multi-Commodity Flow Problem
IDLB	Independent Distributed Load-balanced routing
IGMP	Internet Group Management Protocol
IGP	Interior Gateway Protocol
IP	Internet Protocol
IS-IS	Intermediate System to Intermediate System
ISL	Inter-Satellite Link
ITU	International Telecommunication Union
KPI	Key Performance Indicator
LCF	Least Congested Path First
LEO	Low Earth Orbit
LP	Linear Programming
LSTM	Long Short-Term Memory
MAC	Media Access Control
MADRL	Multi-Agent DRL
MCFP	Multi-Commodity Flow Problem
MDP	Markov Decision Process
ML	Machine Learning
MLP	Multilayer Perceptron

MPLS	Multi-Protocol Label Switching
MSE	Mean Squared Error
MTU	Maximum Transmission Unit
NCC	Network Control Center
NHCN	Next-Hop Cluster Node
NTN	Non-Terrestrial Network
O-RAN	Open Radio Access Network
OSPF	Open Shortest Path First
PIM	Protocol Independent Multicast
PIM-SM	PIM Sparse Mode
QoE	Quality of Experience
QoS	Quality of Service
RF	Radio Frequency
RL	Reinforcement Learning
RNN	Recurrent Neural Network
RP	Rendezvous Point
SAC	Soft Actor Critic
SCN	Satellite Constellation Network
SDN	Software-Defined Networking
SDP	Stochastic Dynamic Programming
SPF	Shortest Path First
SPT	Shortest Path Tree
SR	Segment Routing
SR-TE	Segment Routing Traffic Engineering
TE	Traffic Engineering
TN	Terrestrial Network
TRL	Technology Readiness Level
TTL	Time To Live
USN	User Satellite Node
UT	User Terminal

List of Operators

$ \cdot $	Cardinality of a set, tuple, or vector (number of elements)
$x \cdot y$	Element-wise multiplication or dot product
$x \times y$	Cross product or Cartesian product operator
$\exp(\cdot)$	Exponential function e^x
$\log(\cdot)$	Logarithmic function
$\sum(\cdot)$	Summation operator over a sequence of values
$f \circ g$	Function composition operator, apply function f on g
$\min(\cdot)$	Minimum value function, returning smallest element in a set
$\max(\cdot)$	Maximum value function, returning largest element in a set
$P(\cdot)$	Probability distribution or measure over a random variable
$Pr(\cdot)$	Probability of a specific event occurring
$P(x y)$	Conditional probability statement, x given y
$\mathbb{E}[\cdot]$	Expected value of a random variable
$\mathbb{1}[\cdot]$	Indicator function, returns 1 if condition is true and 0 otherwise
∇	Gradient, vector differential operator
Δ	Laplace operator, corresponding to ∇^2
$O(\cdot)$	Big-O notation for complexity
$\mathbb{R}$	Set of real numbers
$\mathbb{Z}$	Set of integers

List of Symbols

A	Area, if circular: A_{circ} , if hexagonal: A_{hex}
$\mathbf{A}_t$	Adjacency matrix of a graph at time step t
B	Batch size
C	Cost function, at time step t : C_t
D	Replay buffer
$\mathbf{F}_t$	Flow matrix at time step t
G	Antenna gain, if transmitter: G_{tx} , if receiver: G_{rx}
G_{Earth}	Gravitational constant of Earth
H	Heuristic function
J_ρ	Jitter, relative delay variance of packet ρ
K	Number of commodities
M_{Earth}	Mass of Earth
$M_{<\cdot>}$	Number of $<\cdot>$
$N_{<\cdot>}$	Number of $<\cdot>$
Q_π	Q-function for given policy π
R_{Earth}	Radius of Earth
R_t	Aggregated reward at time step t
T	Time duration
$\mathbf{U}_t$	Utilization matrix of a graph at time step t
V	Value function, at time step t : V_t
W_p	Weight of path p
$\mathbf{a}$	Output vector of neural network layer after activation function
a_t	Action at time step t

b	Bias vector of neural network layer
$b(i)$	Buffer capacity of vertex i
$c(i, j)$	Capacity on edge between vertices i and j
c	Speed of light in a vacuum
$\text{dist}(i, j)$	Distance between vertices i and j
d_t^k	Demand of flow k at given time step t
$f_t(i, j)$	Flow at time step t on edge between vertices i and j
$h(i, j)$	Value of heuristic function H for edge between vertices i and j
h_o	Altitude of satellite orbit
k	Commodity
$l_p(i, j)$	Propagation delay of edge between vertices i and j
$l_q(i)$	Queuing delay of vertex i
$l_r(i)$	Routing delay of vertex i
L	End-to-end latency
$l_s(i)$	Switching delay of vertex i
l_t	Local delay
o_t	Observation at time step t
p	Path through network, describing a set of edges
r_t	Reward at time step t
σ	Activation function
s_t	State at time step t
t	Time step
$u(i, j)$	Utilization on edge between vertices i and j
u_{th}	Utilization threshold
v_{dst}	Destination node, egress node
v_{src}	Source node, ingress node
w	Weights of neural network layer
$w(i, j)$	Weight of edge between vertices i and j

w	Weighting factor
x	Terminal on ground
$\mathbf{y}$	Target vector
$\hat{\mathbf{y}}$	Estimation vector
y_t	Target of reinforcement learning at time step t
$\mathbf{z}$	Pre-action value vector of neural network layer
$\mathbf{0}$	Zero vector
Λ	Rate of signaling event
Φ	Transition function between states
Ξ	Weighting factor for rewards
α_s	Softmin factor
α	Learning rate
$\alpha_{\mathcal{H}}$	Entropy temperature factor
β	Angle
η	Data rate of link
δ	Spacing angle
ϵ	Elevation angle
η_{eff}	Efficiency of optical transmitter or receiver
γ	Discount factor
ι	Layer of neural network
λ	Wavelength
ω	Stochastic function describing random traffic events
μ	Weighting factor
ν	Weighting factor
ϕ	Policy parameters of neural network (critic), i.e. weights and biases
π_{θ}	Policy of reinforcement learning agent parametrized by θ
ψ	Angle
ρ	Individual packet

τ	Delay function
θ	Policy parameters of of neural network (actor), i.e. weights and biases
ϑ	Nadir angle
ξ	Weighting factor for rewards
ζ_ρ	Size of packet ρ
$\mathcal{A}$	Set of actions, action space
$\mathcal{C}$	Set of clusters
$\mathcal{E}$	Set of edges, if time-dependent: $\mathcal{E}_t$
$\mathcal{G}$	Graph, if time-dependent: $\mathcal{G}_t$
$\mathcal{H}$	Entropy term
$\mathcal{L}$	Set of layers of neural network
$\mathcal{P}$	Set of paths, if time-dependent: $\mathcal{P}_t$
$\mathcal{SC}$	Complexity in space
$\mathcal{S}$	Set of states, state space
$\mathcal{TC}$	Complexity in time
$\mathcal{V}$	Set of vertices, if time-dependent: $\mathcal{V}_t$
$\mathcal{X}$	Set of terminals on ground, if time-dependent: $\mathcal{X}_t$
$\mathcal{J}(\pi_\theta)$	Expected return of policy π parameterized by θ

1. Introduction

1.1. Background and Motivation

Routing in Satellite Constellation Networks (SCNs) has become an increasingly important topic for multiple reasons. Firstly, driven by the NewSpace economy, a re-emergence of commercial Low Earth Orbit (LEO) satellite constellations has been observed [1,2]. Today, there are numerous deployed and planned constellations, including Iridium NEXT, OneWeb, Starlink, Kuiper, Rivada and IRIS² [3–7]. These systems are designed to provide ubiquitous connectivity, enabling users to send and receive data globally, as long as a connection to a satellite can be established. By using Inter-Satellite Links (ISLs), state-of-the-art constellations are interconnected, spanning a space-borne network. As traffic can thus be forwarded along multiple satellites, the number of required Gateway stations (GWs) on ground is significantly decreased.

SCNs enable a variety of connectivity use cases thanks to their unique characteristics. In the context of global connectivity, satellite systems provide services such as Internet access, particularly in areas with limited terrestrial infrastructure, for example in crisis management scenarios. Moreover, transmissions over longer distances using SCNs hold the promise of reducing end-to-end latency, due to the higher speed of light in a vacuum compared to a glass fiber medium [8]. This is of special interest for delay-critical applications, such as high-frequency trading [9].

Furthermore, end-to-end connections between individual users through the space network can be established. Such physically isolated communication pathways are particularly valuable for governmental and business clients requiring privacy and control. The growing importance of LEO SCNs in defense applications has highlighted the need for satellite systems as backbone or core networks [7, 10, 11]. In addition, ongoing standardization efforts are focused on the seamless integration of SCNs with terrestrial networks. Most notable are the forthcoming releases of the 6G standard, which natively integrate Non-Terrestrial Networks (NTNs) [12–15].

It is important to note that broadband Internet access, over-the-air updates, live streaming, or exchange of large Machine Learning (ML) models, are important applications for future constellations [1,16,17]. As a result, next-generation satellite systems are expected to handle large traffic volumes of diverse services. In addition, in current SCNs, such as Starlink, periodic throughput drops have been observed [18], further highlighting the importance of optimizing routing and network management. Improving overall system throughput while maintaining high Quality of Experience (QoE) is thus essential for future constellations serving various governmental, business and private use cases.

Given these trends, highly complex, three-dimensional space networks will require advanced routing solutions to manage traffic on the ISLs efficiently [19]. Therefore, new adaptive

1. Introduction

and intelligent approaches are needed, which surpass state-of-the-art routing and network management solutions, and comply with these developments.

In this investigation, the focus lies on the maximization of system throughput in SCNs. As a packet-based system is assumed, this translates to routing more traffic through the network by forwarding packets between satellites. To this end, the creation of bottlenecks has to be actively avoided, as they result in the loss of packets. By optimizing the flow of data through the network and minimizing congestion, broadband Internet access can be provided to many users.

This objective is a critical aspect of the economic viability of an SCN system, as business models rely heavily on the capacity to support substantial client bases [16]. While the system characteristics dictate the theoretical maximum data rates within a constellation, optimized routing strategies are required to achieve the desired performance. Solutions have to comply with diverse Quality of Service (QoS) requirements of the resulting traffic while making efficient use of the available network resources. Refined strategies are hence required, which can adapt to changing conditions by adjusting their routing policies.

1.2. Problem Description

The principal optimization problem this investigation is concerned with, is the compliance with QoS requirements for high network loads. In other words, how can **throughput be maximized while complying with system and service requirements?**

There are two core aspects characterizing this optimization problem. Firstly, SCNs represent a distinct, highly dynamic network environment. As a result of this varying topology, routing traffic according to its QoS requirements necessitates advanced techniques. Well-suited protocols, which facilitate such methods, have to be designed specifically for the reference scenarios. The second aspect is the need for efficient routing configurations to enable higher network loads. Given the constrained nature of the system scenarios, such as the limited on-board processing power of the satellites, novel approaches are of interest.

1.2.1. Dynamic Network Topology and Traffic Requirements

Satellite constellation networks with ISLs allow for ubiquitous global connectivity and enable many fixed and mobile applications and use cases [20]. The interconnection of satellites via ISLs permits a reduction of ground stations: the satellites are able to forward traffic among them, towards a node which is connected to a destination on ground. There is no need to distribute ground stations according to coverage patterns to guarantee connectivity.

In many modern use cases, interconnected SCNs are considered to serve broadband multimedia, voice and data traffic - typically based on the terrestrial Internet Protocol (IP) packets [19]. For concurrent satellite system designs, User Terminals (UTs) are considered to connect an end-user, private or corporate, to the satellite constellation. Dedicated stations on ground act as GWs toward the terrestrial Internet. In this context, an IP node which is directly connected to a UT is denoted as a User Satellite Node (USN). However, as this investigation is focused on routing in NTN, no assumptions about USNs are made.

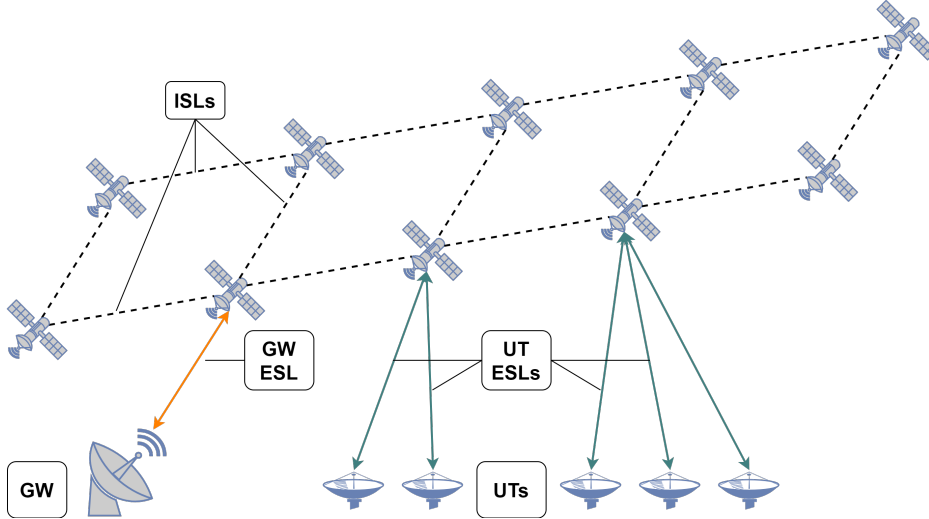


Figure 1.1.: Schematic representation of a sub-section of an SCN. Labels illustrate GWs, UTs, ISLs, and Earth-Satellite Links (ESLs). Distances and entity sizes not to scale. Further ISLs have been omitted.

UTs are considered the final relevant hop. A general overview of this setup is shown in Figure 1.1.

The resulting network of satellites is inherently distributed, constantly in motion, and physically large. To maintain resilient and robust routes, dynamic routing protocols are required [21]. Naturally, depending on the relevant scenario, the principle focus of the constellation and suitable routing approaches may vary.

First and foremost, any LEO SCN represents a dynamic topology. This means that different nodes of the constellation network, i.e. different satellites, will connect to the same UT or GW on the ground within minutes. Hence, the resulting topology of ISLs and ESLs is highly dynamic: frequent handover events are observed.

In addition, the physical size of the spanned satellite network poses significant challenges. Given the potentially considerable distances between endpoints, substantial propagation delays can occur. The resulting latency is thus primarily driven by these delays, which are determined by the speed of light. Therefore, centralized decision-making may not be able to ensure suitable reactivity in the context of routing and network management.

Due to the geographical distribution of GWs and UTs on the ground, non-uniform traffic patterns emerge. An exemplary distribution of these ground segment entities is shown in Figure 1.2. In combination with the dynamic movement of the constellation relative to Earth, they can result in hot spots, which are not topological. Different parts of the constellation network represent potential bottlenecks over time. Therefore, adaptive approaches which accommodate traffic according to the current network state are necessary.

Moreover, multiple service or application requirements have to be fulfilled. In this context, QoS encompasses several key performance indicators such as latency, jitter, and packet loss. To fulfill these requirements of significant network loads, LEO SCNs require specifically

1. Introduction

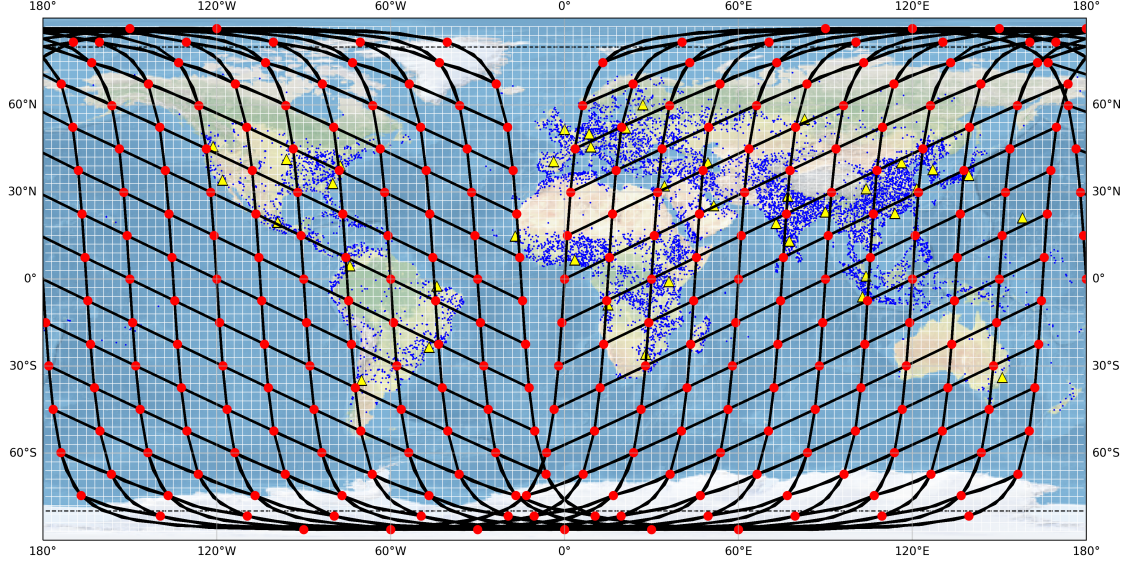


Figure 1.2.: Exemplary scenario of a space segment consisting of a LEO satellite constellation (red dots) inter-connected via ISLs (black lines) and a corresponding ground segment consisting of UTs (blue dots) and GWs (yellow triangles). Map projection skews distances and angles.

tailored routing and network management protocols [22].

1.2.2. Efficient Routing Configurations

To enable high system throughput within SCNs, informed decision-making about which paths are best-suited to support incoming traffic is necessary. Previous path selection choices impact subsequent choices, the sub-problems of finding the best route for a given request overlap. To achieve optimal performance, all of the resulting routes have to be chosen in a way which minimizes congestion.

This optimization can be formulated by the Multi-Commodity Flow Problem (MCFP). More precisely, the MCFP describes the problem of determining an optimal routing configuration for a given set of commodities with different source-destination pairs and an associated demand. In this context, commodities represent the different, individual data transmissions between users or between users and GWs (or vice-versa). Computing this optimal routing configuration is computationally complex; the integer MCFP is NP-hard in general [23]. This means that there is no algorithm which can solve the problem in polynomial time.

For dynamic scenarios, with continuously starting and stopping traffic, the problem is particularly hard to formulate and to solve iteratively. Instead, heuristic rule-based approaches have been proposed to improve performance rather than using exhaustive solutions [24]. However, their Traffic Engineering (TE) capabilities are limited. The schemes do not anticipate concrete developments of the network state. Therefore, their performance varies and may not approximate ideal decision-making. Choosing suitable paths, which result in near-optimal routing configurations in an SCN with relatively low complexity, remains a

challenging optimization problem.

Moreover, the unique characteristics of SCNs necessitate tailored solutions which can operate within the constraints of the network. Due to the described dynamic and physically large topology, a centralized computation of routing configurations is likely not feasible, depending on the specific system considered. As the traffic patterns evolve within the satellite topology moving relative to Earth, configurations have to be adjusted accordingly to maintain performance. For enhanced constellation autonomy and faster reactivity to changes, on-board routing schemes may be preferred. In this context, the complexity of the approach becomes a critical factor. Low-complexity solutions which approximate near-optimal performance are required.

1.2.3. Research Questions

Based on the identified problems for routing and network management in SCNs, corresponding research questions can be formulated. The described characteristics highlight the need for tailored solutions. The optimization problem is two-fold: on the one hand, methods and algorithms are needed which maximize system throughput under constraints presented by the scenario. On the other hand, a protocol has to be designed which enables the actual application of such methods in the dynamic SCN scenario. Therefore, the following research questions are formulated, which will be investigated in the course of this thesis:

- What low-complexity, on-board load-balancing strategies can effectively approximate optimal solutions for the MCFP in SCNs?
- Which frameworks and architectures can enable these techniques, and how should suitable protocols be tailored for application in dynamic SCN scenarios?
- How can data-driven methods be leveraged for network control to formulate and dynamically adjust effective load-balancing strategies, and what are the implications for network performance?

Building on the third question, this investigation explores the potential of data-driven techniques for network control tasks. The algorithmic deficit in maximizing throughput within the constraints of the reference system motivates the application of ML. By combining novel protocol designs with suitable ML-based methods, approaches which improve network performance are provided. The investigation builds on promising key concepts for the design and control of next-generation networks.

1.3. Key Concepts and Proposed Approaches

The proposed solutions are based on two state-of-the-art paradigms: distributed Software-Defined Networking (SDN) and Deep Reinforcement Learning (DRL). These core concepts have been selected due to their potential to address the described challenges for enhancing routing in SCNs. As the emergence of SDN has transformed the field of networking by enabling improved flexibility and programmability, it is considered a suitable candidate to enable adaptive routing approaches. DRL, on the other hand, has emerged as a promising method for network control, given its potential to approximate solutions to high-dimensional

1. Introduction

control problems. Using Deep Learning (DL), effective policies can be learned even in complex environments.

Distributed Software-Defined Networking

SDN is a framework which decouples the network control and the forwarding of data. As a result, a control plane as well as a data plane are introduced. This logical separation allows for more flexible and efficient network management. The network resources are managed and orchestrated using a centralized SDN controller. By using this unified view of the network, specific policies can be defined, traffic prioritized, and resources allocated dynamically. The controller forwards these instructions to the other devices in the network. Thus, granular packet processing and on-board switching rules are installed. Moreover, the control logic is defined in software. This softwarization of the decision-making is advantageous, as it enables updates and adjustments.

In short, the framework offers improved flexibility through software and virtualization, scalability, fast reconfiguration, increased resiliency, and service-aware QoS handling [25,26]. An overview of the general architecture is provided in Figure 1.3. The logical split between data and control plane is shown with typically considered protocols. For completeness, the application plane is included, which is, however, not relevant for the investigation as no specific applications are assumed. OpenFlow is the most commonly used protocol enabling the information exchange between controller and forwarding device. This interface is called "southbound", corresponding to the control hierarchy shown in the figure. Higher-level network management and administration is assumed to be handled by network control centers on ground. This management layer is assumed to operate on a different, slower time scale than the control plane.

All relevant network state information is aggregated on a central control unit. Thus, the controller software can make informed routing decisions and forward these instructions accordingly. For this reason, SDN is well suited to enable advanced load-balancing strategies, which are critical when maximizing throughput. Fine-grained forwarding rules can be established to distribute traffic among the network resources and consider individual QoS requirements.

However, this centralization poses problems in the context of SCNs. Due to significant propagation delays given the size of spanned networks, using a centralized routing scheme is rather impractical. The reactivity of the network controller is impeded, and significant signaling overhead is expected. Moreover, a single point of failure is introduced, meaning that local disruptions can have constellation-wide consequences. Given these scalability problems, a physically distributed control architecture is preferable for SCNs. However, in completely decentralized schemes, enabling efficient load-balancing is challenging given the lack of coordination. So, instead of fully centralizing or decentralizing, a hybrid hierarchical solution has to be considered.

Distributed SDN is a particularly promising approach in this context, as it maintains advantages of centralized SDN while providing scalability and reliability. Instead of a single controller, there are multiple control units; the network is sub-partitioned accordingly. In a distributed architecture with a flat hierarchy, controllers are responsible for individual do-

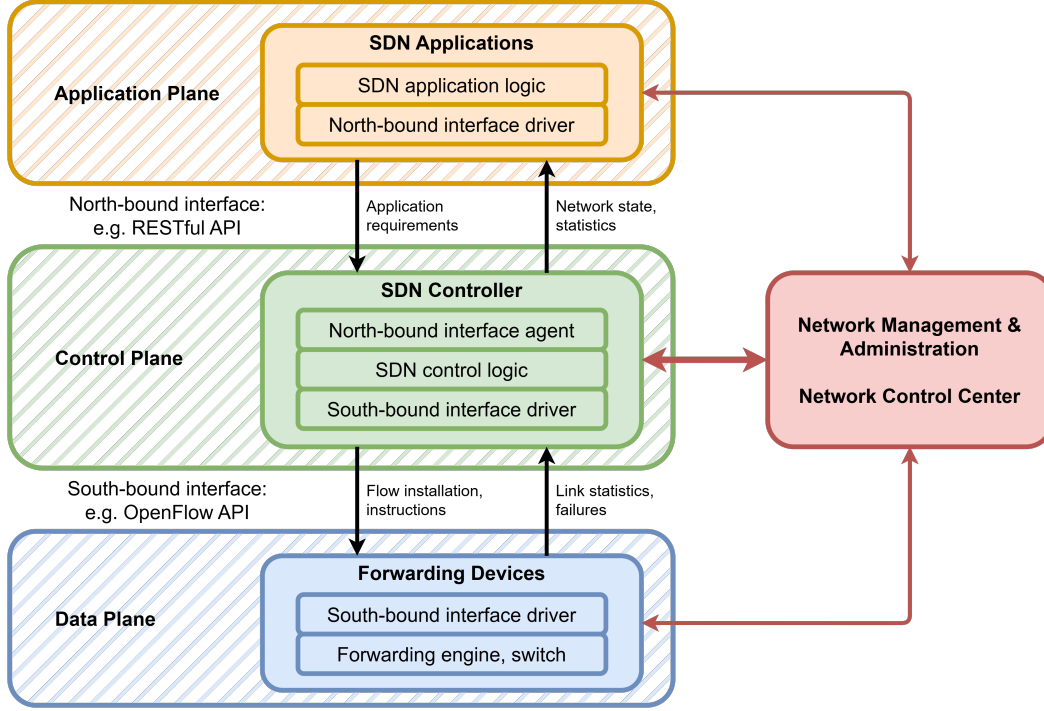


Figure 1.3.: Diagram of logical SDN architecture. The SDN controller interacts with the application plane via a dedicated northbound interface, and with the forwarding devices via a southbound interface.

main, which contain subsets of forwarding devices. This network of sub-networks is highly scalable as new domains can be added when expanding. For the scenario at hand, these domains can be clusters of neighboring satellites. While the corresponding SDN controllers are able to make decisions locally, a logically consistent view of the global network can be maintained by exchanging necessary information between clusters. Intra-domain as well as inter-domain routing strategies are possible.

This decision-making is enabled by the routing logic of the controller software. Thanks to the softwarization, the role of SDN controller can actually be taken over by all nodes with suitable hardware. Thus, in regenerative satellite systems, certain satellites can effectively be designated as SDN cluster controllers. In case of failures, the role can be handed over to neighboring satellites. This enables a space-borne network control with improved reliability and reactivity. However, designing a well-suited architecture to the defined use cases requires tailored algorithms and in-depth analyses of the structural choices.

Deep Reinforcement Learning

DRL is a DL technique focused on optimal control [27]. At its core, it is based on using Deep Neural Networks (DNNs) to approximate even complex functions. In general, DL methods represent a sub-field of ML, which is in itself a sub-field of Artificial Intelligence (AI) [28]. As there is an algorithmic deficit for optimal routing configuration in SCNs, ML approaches can represent viable options: they are trained directly on data to approximate underlying

1. Introduction

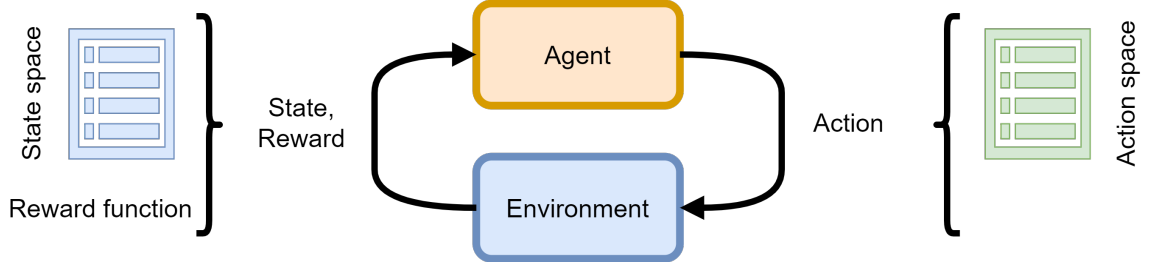


Figure 1.4.: Schematic illustration of a general RL framework, depicting the interaction of an agent with an environment.

patterns [29,30]. In fact, DNNs enable the learning of lower-dimensional manifold representations, i.e. geometric structures, which can efficiently capture complex problems [31]. So, they are able to learn function approximations even for non-convex problems for which no closed-form analytical solutions may be available. This ability to learn such functions is particularly useful for network management. Many key problems in this context are typically considered NP-hard [32]. Once trained, DL-based algorithms can thus provide significant speedups over conventional exhaustive or heuristic approaches.

Furthermore, DL architectures are generally able to handle large amounts of input parameters. They have the ability to converge towards a minimum even in high-dimensional problem spaces [28,33]. Since the efficiency of routing in complex and dynamic network environments is determined by multiple distinct KPIs with many inputs, it represents an intriguing application of data-driven algorithms.

DRL considers agents in environments, which are typically in the form of Markov Decision Processes (MDPs). MDPs provide a formal, mathematically idealized framework for modeling reinforcement learning problems [27]. The general Reinforcement Learning (RL) setup is shown in Figure 1.4. In this framework, an agent interacts at a certain time step with an environment by choosing an action from a set of possible actions. By interacting, the agent receives rewards which it seeks to maximize. Thus, the agent is able to directly learn policies from its environment.

Agents balance exploiting known rewards through greedy selection versus exploring unknown states and actions with potentially better long-term rewards. As cumulative rewards are considered, the impact of sequential decisions is implicitly evaluated. RL agents are typically trained in simulative environments which encompass the state space and provide suitable reward functions. Depending on the use case, specific DRL techniques are more suitable than others. Since various DRL architectures and flavors exist, the specific fit for the considered environment and action spaces needs to be established. As an example, certain techniques are more resilient to overestimation, which can be crucial depending on the environment. Accordingly, trade-offs have to be analyzed when designing suitable DRL architectures.

In the context of routing in SCNs, DRL can be used to optimize decision-making based on the current network conditions. An exemplary setup is shown in Figure 1.5, illustrating the interaction between a DRL agent and a satellite network environment. The former is equipped with a DNN to decide on actions, and a memory component for stability. The agent learns from past interactions by sampling experiences from the corresponding memory

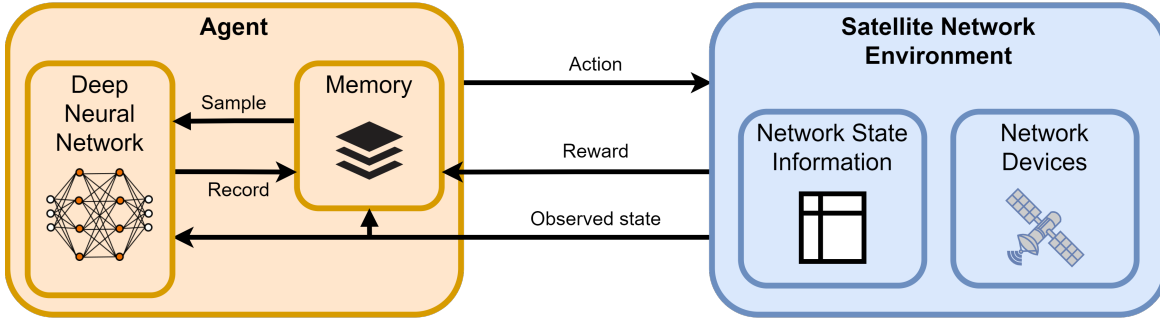


Figure 1.5.: Diagram of a generic DRL agent with a satellite network environment.

buffer. The environment, on the other hand, describes the evolution of the system state over time. In this context, an observed state may consist of information about devices and network indicators, such as link loads. As DRL is able to handle many input parameters, it is a promising approach for routing problems with many Key Performance Indicators (KPIs). Underlying traffic and forwarding patterns are learned by the agent. Conventionally, Dynamic Programming (DP) is considered the only option to solve general stochastic optimal control problems [27]. However, DP approaches require a good knowledge of the environment and have high computational complexity for high-dimensional inputs.

Converging towards optimal solutions can be a significant challenge for DRL agents. Rewards often only partially capture the desired behavior, or may represent it indirectly. As a result, during training, agents fail to converge to a global optimum. To mitigate such issues, a careful design of state and action spaces is required. Most importantly, well-formulated reward functions are essential. This requires intricate knowledge of the underlying optimization problem, the simulated environment, and the resulting data. For routing in SCNs, a black-box methodology is not feasible. A DRL solution without considering the unique challenges and underlying complexities does not converge. A deliberate tailored approach is needed to achieve the desired level of performance.

Proposed Approaches

Following these insights, a novel distributed load-balancing routing protocol based on SDN is proposed. The approach is specifically tailored to SCNs and can be adjusted or extended flexibly depending on the scenario requirements. It represents a promising in-orbit solution for routing and network management in autonomous satellite networks. The protocol is based on a sub-division of the SCN into clusters. These SDN sub-domains are fixed within the topology of the constellation, and are independent of the infrastructure on ground. A satellite near the middle of the cluster is assumed to take on the role of the local SDN controller.

The approach is based on insights from several routing concepts for SCNs focusing on improving underlying trade-offs. So, geographical addressing is used to minimize signaling overhead. A proactive distribution of traffic on valid routes makes use of the inherent path diversity of SCNs. In addition, distributed network control is applied to provide scalability and reactivity. In order to maximize the achievable throughput of the protocol, bottlenecks

1. Introduction

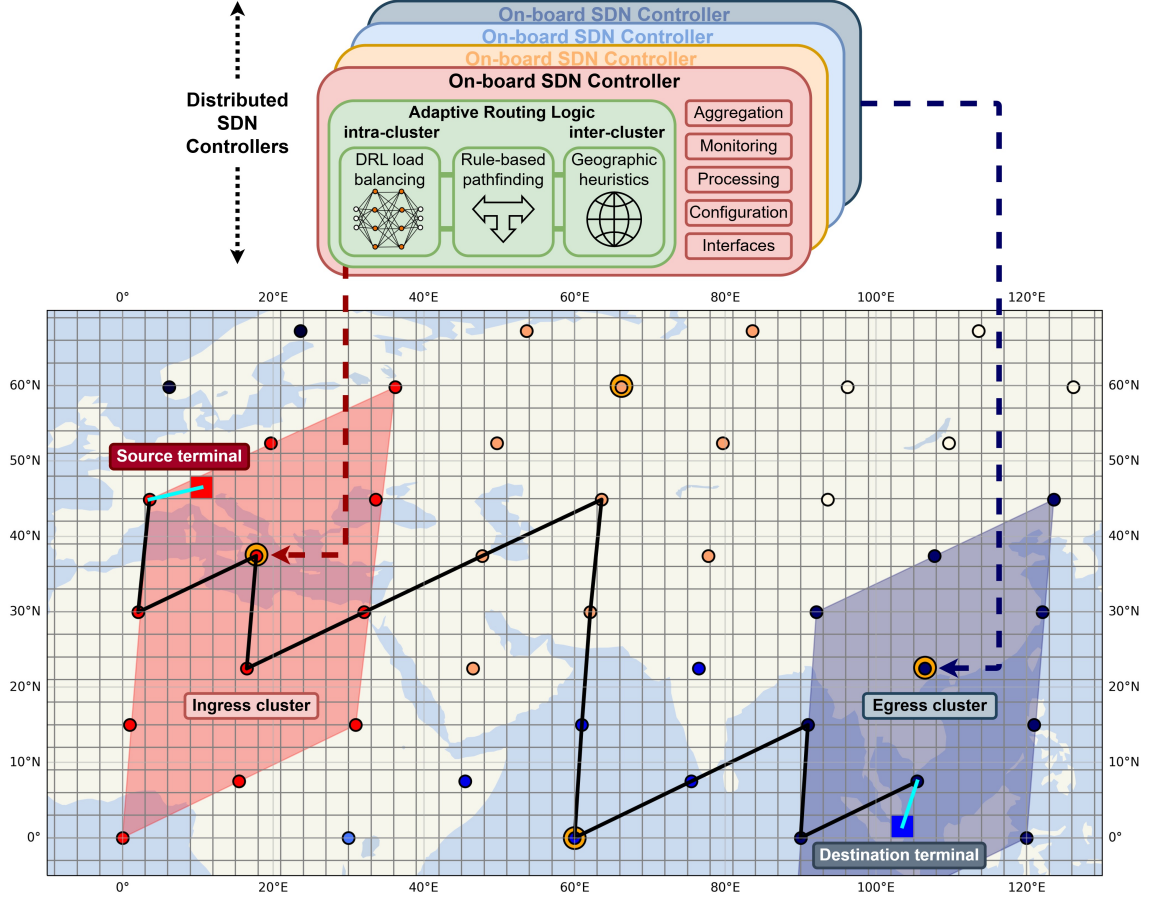


Figure 1.6.: Overview of the proposed approach, including the DRL-enhanced on-board SDN controller architecture, for an exemplary path from Munich to Singapore across several clusters.

and congestion have to be minimized by balancing the incoming traffic. To this end, a dedicated QoS-aware routing algorithm is proposed. The performance is thus improved, meaning that high QoS compliance is maintained for increasing network loads.

In order to approximate solutions to the underlying MCFP with low complexity, DRL methods are considered for adaptive TE. Building on the proposed distributed SDN-based protocol, the idea is to achieve near-optimal flow configurations to enhance intra-cluster routing. Trained models are assumed to be run on-board the satellites, making feasible complexity a key design consideration. Specifically, architectures for hop-based as well as path-based solutions are derived and investigated. Agents are trained and fine-tuned in simulative environments to make effective routing decisions. Ultimately, this combined approach leverages the strengths of both SDN and DRL, providing a robust and adaptive solution for enhanced routing in SCNs.

An overview of the proposed combined solution is provided in Figure 1.6. A transmission across multiple clusters is illustrated, from the source terminal in the ingress geographical cell (red rectangle) to the destination terminal in the egress cell (blue rectangle). The resulting

path consists of multiple ISLs (black lines) as well as ingress and egress ESLs (cyan lines). Orange rings indicate the distributed on-board SDN controllers. The considered SDN control architecture including its routing logic is shown on top. As described, a DRL-based traffic engineering extension is used for intra-cluster load balancing. Here, the promising path-based approach is shown which operates on rule-based pathfinding algorithms to make decisions. For routing between clusters, geographical heuristics are used to reduce signaling.

Overall, the proposed architecture has been designed to fulfill the increasing throughput requirements of satellite networks. It combines a novel, distributed routing protocol with promising DRL enhancements. By mitigating bottleneck formation through informed routing decisions, the proposed architecture enables support for a larger customer base, offering enhanced scalability and potentially increased profitability for next-generation satellite systems.

1.4. Research Contributions

The research presented in this thesis is related to investigations from various projects and activities. The findings and key insights have been disseminated through peer-reviewed publications. The corresponding conference papers, journal articles, and contributions to reports are listed below.

Conferences

- M. M. H. Roth, H. Brandt, and H. Bischl, “Distributed SDN-based Load-balanced Routing for Low Earth Orbit Satellite Constellation Networks,” in *2022 11th Advanced Satellite Multimedia Systems Conference and the 17th Signal Processing for Space Communications Workshop (ASMS/SPSC)*, Sep. 2022, pp. 1–8. [34]
- M. M. H. Roth, “Analyzing Source-Routed Approaches for Low Earth Orbit Satellite Constellation Networks,” in *Proceedings of the 1st ACM Workshop on LEO Networking and Communication, in LEO-NET ’23*. Oct. 2023, pp. 43–48. [35]
- M. M. H. Roth, A. Hegde, T. Delamotte, A. Knopp, “Shaping Rewards, Shaping Routes: On Multi-Agent Deep Q-Networks for Routing in Satellite Constellation Networks,” in *The First Joint European Space Agency / IAA Conference on AI in and for Space (SPAICE2024)*, Sep. 2024, pp. 100-104. [36]
- M. M. H. Roth, T. Jerkovits, A. Hegde, T. Delamotte, and A. Knopp, “Deep Reinforcement Learning for Adaptive Traffic Engineering in Satellite Constellation Networks,” in *2025 12th Advanced Satellite Multimedia Systems Conference and the 18th Signal Processing for Space Communications Workshop (ASMS/SPSC)*, Feb. 2025, pp. 1–8. [37]
- A. Hegde, M. M. H. Roth, and H. Bischl, “Routing in Low Earth Orbit Satellite Networks: Constrained DQN-Based Approach,” in *2025 IEEE Wireless Communications and Networking Conference (WCNC)*, Mar. 2025, pp. 1-6. [38]

Journals

- M. M. H. Roth, H. Brandt, and H. Bischl, “Implementation of a geographical routing scheme for low Earth orbiting satellite constellations using intersatellite links,” in *International Journal of Satellite Communications and Networking*, vol. 39, no. 1, pp. 92–107, 2021. [39]
- M. M. H. Roth, H. Brandt, H. Bischl, D. Fernández Piñas, and G. Acar, “IDLB: An SDN-based Load-balancing Routing Protocol for Autonomous Satellite Constellation Networks,” to be published in *International Journal of Satellite Communications and Networking*. Pre-print available at Authorea, Apr. 2025. [40]
- M. M. H. Roth, T. Jerkovits, A. Hegde, T. Delamotte, and A. Knopp, “Path-based Deep Reinforcement Learning for On-board Routing in Satellite Constellation Networks,” to be submitted in *Invited Special Issue: ASMS*, of *International Journal of Satellite Communications and Networking*. Aug. 2025.

Reports

- S. Kota, G. Giambene, M. Abbak, M. Abdelsadek, M.-S. Alouini, M. Amay, S. Babu, J. Bas, V. Baños González, C. Carballo González, P. Cassarà, S. Chaudhari, D. Dalai, T. Darwish, T. de Cola, T. Delamotte, A. Dutta, A. Dwivedi, H. ElBakoury, M. Enright, M. Giordani, A. Gotta, E. Hammad, T. Khattab, A. Knopp, G. Karabulut Kurt, T. Li, P. G. Madoery, B. S. Manoj, J.-D. Medjo Me Biomo, P. Pillai, P. Rawat, M. M. H. Roth, P. Saxena, P. Scanlan, A. Sharma, M. S. Solaija, R. Sperber, Z. Sun, D. Tarchi, N. Varshney, S. Verma, R. Wang, S. Watts, H. Yanikomeroglu, K. Zhao, and L. Zhao, “Satellite,” to be published in *IEEE International Network Generations Roadmap - 2024 Edition, 2024 IEEE Future Networks World Forum (FNWF)*, IEEE, 2025.
 - Contribution to ML-based routing strategies.

1.5. Thesis Outline

In **Chapter 2**, the fundamentals of routing in SCNs are presented, laying the groundwork for understanding the unique challenges encountered in such systems. This includes the description of relevant reference system aspects. The space segment, the ground segment, the connectivity between networking entities and devices, as well as the considered traffic models are introduced. The differences to terrestrial wireless networks are highlighted as well. For routing protocols, a concise overview over the state of the art is given, including an assessment and a look into standardization developments. To provide a basis for evaluation, the relevant KPIs are formulated as well. Lastly, fundamentals of the underlying network optimization theory are provided. Graph-based models and the MCFP are introduced, which establish the theoretical background for the following optimization approaches.

In **Chapter 3**, a distributed SDN-based routing protocol is proposed and analyzed. As this novel solution utilizes insights from previous investigations of geographical and source routing, the corresponding methods are introduced. Then, the design of the proposed protocol is presented in detail. Extensive simulations highlight the advantages and limitations of the

approach. The evaluation is preceded by a description of the specifically designed network simulation tools. Additionally, the results of a demonstration in a hardware test bed are provided to further validate the approach. The chapter concludes with a discussion of the proposed protocol, which represents a potent framework for ML enhancements.

In **Chapter 4**, RL-based schemes are investigated, representing promising extensions to the distributed protocol. To highlight potential advantages over rule-based approaches, the viability of DRL-based methods for advanced traffic engineering is derived from first principles. Building on these insights, hop-based as well as path-based methodologies are proposed. For hop-based approaches, centralized and fully decentralized multi-agent architectures using Deep Q-Networks (DQNs) are presented and analyzed. Inherent robustness and scalability limitations are discussed. To overcome these shortcomings, a path-based approach using a soft actor-critic design is proposed, which is capable of approximating optimal solutions to the underlying MCFP. A quantitative evaluation with multiple benchmarks highlights the general performance and viability of the solution. The implications of these results and corresponding limitations are also discussed.

To conclude, **Chapter 5** summarizes the contributions of this work and provides recommendations for further investigations. The research questions posed in this work are answered and perspectives on related areas of research are presented.

2. Routing Fundamentals for Satellite Constellation Networks

This chapter introduces the characteristics of SCNs and their distinctions from terrestrial networks. Two core reference scenarios are presented, which provide the practical context of this investigation. The assumed properties of the terminals on ground, including their distribution, are discussed, and the considered traffic models are introduced. Furthermore, a comprehensive review of state-of-the-art routing protocols and algorithms from literature is provided. Subsequently, the considered performance metrics for this investigation are presented. The chapter concludes with a foundation of relevant network optimization techniques, which forms the basis for novel approaches developed in subsequent chapters.

2.1. Satellite Constellation Network Architecture

The design aspects of SCNs and the considered reference constellations are discussed in Section 2.1.1. The terrestrial components and their connectivity to the space-borne network are described in Section 2.1.2, and Section 2.1.3, respectively. The considered traffic models are presented in Section 2.1.4.

If higher traffic loads are expected, suitable and effective routing protocols become a critical aspect of the system design. The key design goals for these protocols are the fulfillment of QoS requirements of different applications and the optimal usage of the available network resources. Due to the special network characteristics and conditions of SCNs, protocols from terrestrial contexts may not be applicable. There are significant differences to terrestrial networks, which are described in 2.1.5.

Similar reference scenarios have been investigated in related publications [34, 35, 40].

2.1.1. Space Segment

LEO satellite constellations are based on specific system topologies, which utilize the inherent regular orbits to provide continuous coverage. Typically, such systems consist of satellites in multiple equal-radius circular orbits with an equal number of satellites in each orbit. Two main approaches for regular orbital patterns have been particularly influential and are most widely used today: Walker star and Walker delta topologies [41].

A Walker star constellation topology is characterized by polar or near-polar orbits. The orbital planes cross at or in proximity to the respective poles. This architectural arrangement results in adjacent co-rotating planes, which are designated as seams. Due to the relative speed of the satellites in co-rotating planes, increased Doppler frequency shifts are observed

2. Routing Fundamentals for Satellite Constellation Networks

for connections across the seams [42]. The orbital planes of ascending nodes are spaced laterally at angular intervals of approximately π/N_{planes} , with N_{planes} denoting their number. Due to the Doppler shifts, there may or may not be ISL connections across the two inherent seams of this design pattern. A well-known constellation following this design pattern is the Iridium constellation [43, 44].

The so-called Walker delta pattern is typified by orbits in equal inclination at the angle delta, spaced equally around a reference plane, most commonly the equator. Ascending nodes are spaced at intervals of $2\pi/N_{\text{planes}}$. Due to the inclined orbits, there is usually no global coverage in contrast to Walker star. Typically, no satellites are covering the relative north and south poles. Compared to equally sized Walker star constellations, more uniform coverage is provided. Based on the resulting orbital propagation of satellites, Walker delta constellations with ISLs facilitate lateral (east-to-west and vice versa) transmissions, because of the reduced number of inter-plane hops [8]. A prominent example of this design pattern is phase one of SpaceX's Starlink [3, 4].

Orbital Characteristics

The time to complete one orbit T_{orbit} around Earth is described by Kepler's third law [42]:

$$T_{\text{orbit}} = 2\pi \sqrt{\frac{(r_{\text{Earth}} + h_o)^3}{G_{\text{grav}} M_{\text{Earth}}}} \quad (2.1)$$

Here, the numerator describes the semi-major axis of an orbit. In the case of circular orbits, it corresponds to the radius, which consists of the altitude of the satellite h_o and the radius of the Earth r_{Earth} . G_{grav} denotes the gravitational constant ($6.67430 \times 10^{-11} \text{ Nm}^2/\text{kg}^2$), and M_{Earth} is the mass of Earth. Based on T_{orbit} , the time of a fly-by of a satellite can be derived.

In real-world systems, the actual position of a satellite varies slightly due to atmospheric and gravitational influences. Satellites are oftentimes equipped with small propulsion systems to correct their position if required. The effects of these small positional variations on the overall transmission characteristics are generally not significant. Consequently, only the theoretical positions of satellites are considered. It should also be noted that intra-plane links are generally easier to maintain, as the distance and angle between the satellites remains constant. Still, the bandwidth and availability of ISLs are generally considered equal in this work.

In general, a shutdown of the inter-plane ISLs is required at maximum latitude (for high inclinations near the poles), because of the switch in relative position of the orbital planes. So, a neighboring satellite which was previously on one side will find itself on the other side of a given satellite. This switch has to be accounted for when reactivating the inter-plane ISLs. Since the relative speed and angle of satellites in co-rotating planes increases at the poles, interruptions due to pointing limitations of the hardware as well as significant Doppler effects need to be avoided. The Doppler frequency f_d is defined by [45]:

$$f_d = f_0 \left(\frac{v_{\text{rel}}}{c} \right) \quad (2.2)$$

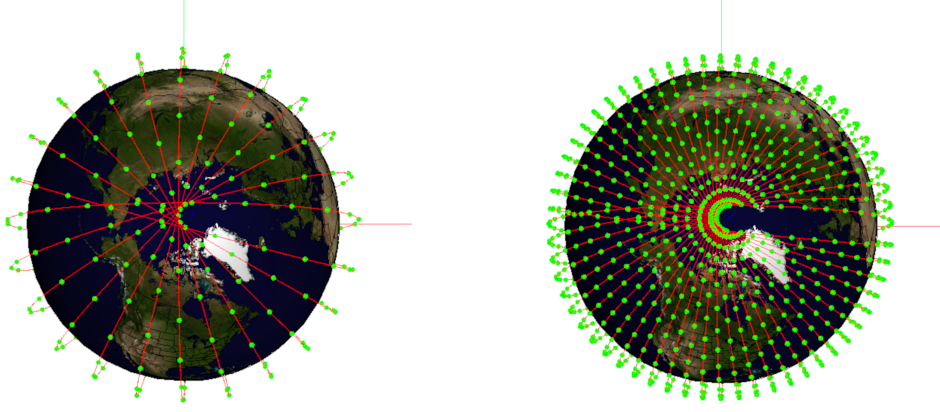


Figure 2.1.: Top-down view of SCN-288 (left) and SCN-1440 (right) satellite constellations [46].

Here, v_{rel} is the relative velocity, c is the speed of light, and f_0 is the transmitted frequency. As the relative speed of counter-rotating planes $v_{rel,counter}$ is almost twice the speed of a satellite v_{sat} , the resulting Doppler is significant:

$$v_{rel,counter} \approx 2 \cdot v_{sat} = 2 \cdot \sqrt{\frac{G_{grav} M_{Earth}}{r_{Earth} + h_o}} \quad (2.3)$$

For a constellation at an altitude of 780 km, this results in a relative velocity of approximately $53\,800 \text{ km h}^{-1}$. Given the linear relation with the Doppler frequency, compensation can thus be difficult. For these reasons, shutdown latitudes are utilized. If the satellite is above or below the respective latitudes, the inter-plane ISLs are deactivated and the antennas are re-adjusted to establish new links.

Considered Reference Constellations

Two single-layer near-polar Walker star constellations are considered as the primary reference systems. Nevertheless, the proposed protocol is generally suited to support multi-layer or multi-shell constellations. The designs have been proposed to be investigated by ESA in the context of the ROPRO project [47], and have also found interest in further activities [48, 49]. They mirror recent trends in constellation sizing and characteristics of intended constellations: they share similarities with public plans of OneWeb and Rivada [4, 6]. The reference systems do not directly represent currently deployed constellations networks, but rather proposed designs for potential next-generation satellite systems.

The two principal reference constellations are called **SCN-288** and **SCN-1440**. Both constellations follow a similar architecture, i.e., in inclination, arrangement, and spacing. As quasi-polar Walker star constellations [41] with an inclination of 86.4° , they share similarities in terms of general design with the Iridium constellation [5, 44]. SCN-288 consists of 288 satellites which are arranged in 12 planes at an altitude of 780 km. This results in 24 satellites per plane, with an angular distance of 15° . The angular phase offset between co-rotating planes is 7.5° . SCN-1440 corresponds to an up-scaled version of SCN-288 at

Table 2.1.: Summary of space segment of SCN-288 and SCN-1440.

Space segment characteristic	SCN-288	SCN-1440
Number of satellites	288	1440
Number of planes	12	30
Number of satellites per plane	24	48
Satellite altitude [km]	780	600
Orbital inclination [°]	86.4	86.4
Cross-seam planes spacing [°]	15	6
Co-rotating planes spacing [°]	15	6
Angular phase offset co-rotating planes [°]	7.5	3.75
Inter-plane ISL shut-down latitude [°]	80	80

an altitude of 600 km. Here, 1440 satellites in 30 planes with 48 satellites per plane are proposed. A top-down view of the two constellations is shown in Figure 2.1.

Each satellite of the constellation possesses four ISLs: two intra-plane ISLs and two inter-plane ISLs. The spacing of the co-rotating and counter-rotating planes follows the standard design pattern for this type of constellation. A summary of their characteristics is provided in Table 2.1.

The shutdown latitudes mainly depend on the pivoting speed of the antennas and the angular velocity between the two satellites involved. State-of-the-art optical laser terminals are able to pivot rapidly, for instance Tesat's ConLCT is able to pivot at 2.5°s^{-1} on both axes (azimuth and elevation) simultaneously and handle Doppler shifts of up to 3.0 GHz [47]. So, reorientation and target tracking works at latitudes of at least $\pm 80^\circ$ for the defined orbit altitudes. Relative speeds between satellites at these orbits are also well within Doppler tolerances of the laser communication terminal.

Coverage and Minimum Elevation Angle

Regarding the minimum elevation angle of the user terminals, a number of aspects of the reference constellations have to be taken into consideration. In the assumed system, the coverage beams of the satellites are broken down into spot beams to use frequency reuse. To guarantee coverage, the footprints of the satellites have to overlap. However, smaller coverage areas correspond to higher minimum elevation angles for the terminals on ground, allowing for improved transmission conditions.

The scan angle of a satellite ϑ_{max} , also called maximum nadir angle, is linked to the minimum elevation angle ϵ_{min} of the terminal on ground. ψ_{max} denotes the angle from the center of the Earth between the sub-satellite point and a terminal at the edge of the coverage area. The scan angle is described by [42]:

$$\vartheta_{max} = \pi - \psi_{max} - \left(\epsilon_{min} + \frac{\pi}{2}\right) \quad (2.4)$$

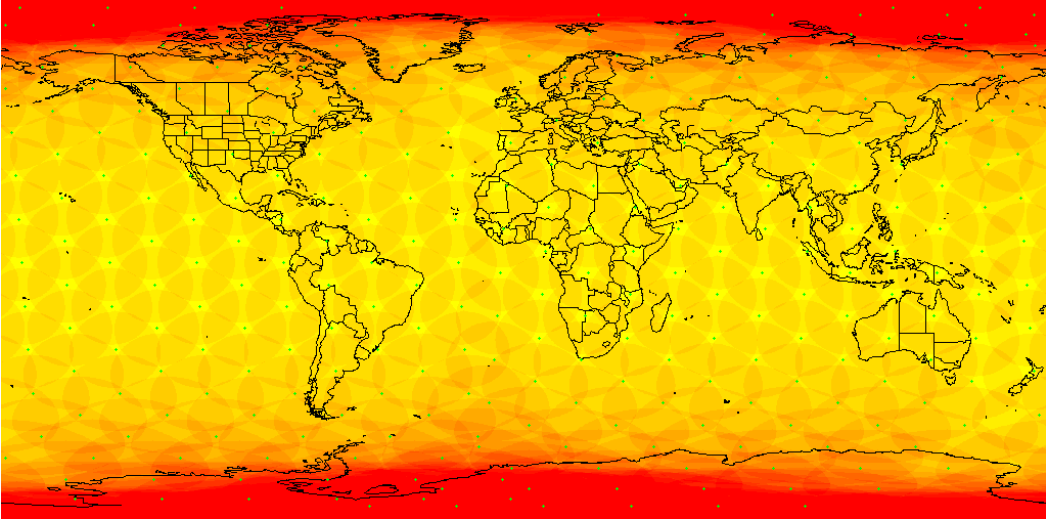


Figure 2.2.: Coverage of SCN-288 with a minimum elevation angle of 20° [46].

ψ_{max} is dependent on the minimum elevation angle:

$$\psi_{max} = \arccos\left(\frac{r_{\text{Earth}}}{(r_{\text{Earth}} + h_o)} \cos(\epsilon_{min})\right) - \epsilon_{min} \quad (2.5)$$

The resulting coverage area is thus:

$$A_{\text{circle}} = \pi (\psi_{max} r_{\text{Earth}})^2 \quad (2.6)$$

As the coverage areas are circular and thus need to overlap, spherical hexagons without overlapping are used to represent the coverage areas instead. The area is calculated by splitting the spherical hexagon into six spherical triangles, with the edge angle β_{edge} :

$$\beta_{\text{edge}} = \arctan\left(\frac{\sqrt{3}}{\cos(\psi_{max})}\right) \quad (2.7)$$

$$A_{\text{hexagon}} = 6r_{\text{Earth}}^2 \left(2\beta_{\text{edge}} - \frac{2\pi}{3}\right) \quad (2.8)$$

Using these formulas, the highest elevation angle possible based on the number of satellites and the inter-plane offset, while guaranteeing global coverage, is approximately 29° for SCN-288. In order to provide flexibility in the case of failures, a minimum elevation angle of 20° is considered for SCN-288, which is shown in Figure 2.2.

Using the same approach, the coverage of SCN-1440 can be evaluated. For SCN-1440, there are more than eight satellites visible in most areas when using a ground terminal elevation angle of at least 20° . Since four concurrently visible satellites are considered sufficiently diverse, a minimum elevation angle of approximately 30° is assumed. The resulting coverage is shown in Figure 2.3. The corresponding maximum visibility of a satellite fly-by is 236 s.

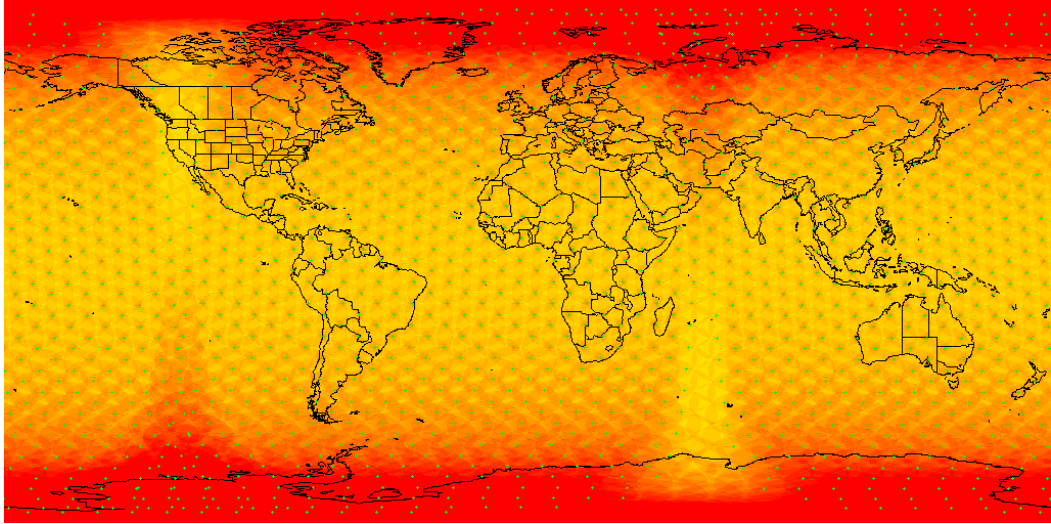


Figure 2.3.: Coverage of SCN-1440 with a minimum elevation angle of 30° [46].

The minimum elevation angle requirements for GWs are different from those of UTs. GWs are generally assumed to be larger, they use dedicated structures with more powerful antennas. Their position is chosen specifically to improve satellite connections. So, for SCN-288, a minimal elevation angle of 10° is assumed for GWs. In this case, at least four satellites are visible for a GW. For SCN-1440, a minimal elevation angle of 20° is assumed for GWs, so at least six satellites are visible.

2.1.2. Ground Segment

The ground segment is assumed to consist of UTs and GWs. They represent the sources and destinations of all of the data traffic which is supposed to be forwarded in the SCN. Their distributions and connectivity characteristics of the reference system are presented, as they significantly impact the network topology and emerging traffic patterns. The ground segment characteristics are summarized in Table 2.2.

Table 2.2.: Summary of ground segment and coverage of reference system.

Ground segment characteristic	SCN-288	SCN-1440
Number of potentially simultaneously active UTs	[1 000, 10 000]	[1 000, 10 000]
User terminal minimum elevation angle $[\circ]$	20	30
Number of GWs	39	39
Gateway minimum elevation angle $[\circ]$	10	20

User Terminal Distribution

The distribution of UTs is modeled as non-homogeneous to reflect real-world dispersion patterns. Global connectivity is considered a crucial design aspect, which requires the provision of services to private UTs in remote areas. To accurately capture the spatial distribution of

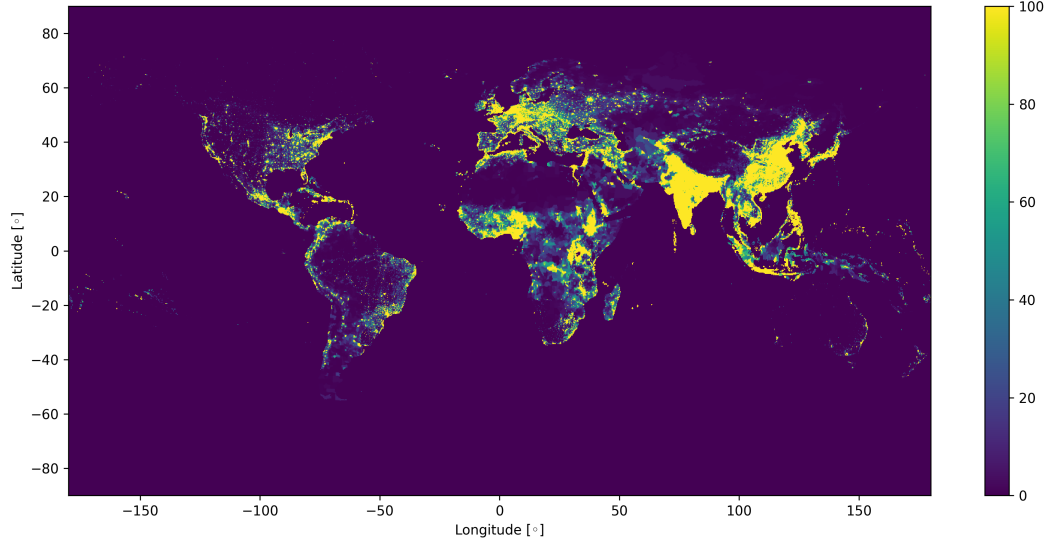


Figure 2.4.: Population density statistics in the year 2020, with a resolution of 15 arc-minutes [50]. Applied filtering function limits maximum number of persons per square kilometer to 100 [51].

users, a stochastic sampling process, which is based on worldwide geographical population density statistics, is employed. In Figure 2.4 the approximate global population density in the year 2020, with a resolution of 15 arc-minutes is shown.

As metropolitan and densely populated areas tend to be well connected, they need to be partly discounted in the creation of a realistic UT distribution. To this end, a filtering function is introduced which adjusts individual density values. The result after filtering is the input to the stochastic process generating relevant UT distributions. While relative interest in sparsely populated areas is expected to be higher, the overall number of users there is still assumed comparably low. Hence, the following filtering is applied: a linear relation up to a density of 100 persons per square kilometer. Then, a constant density of 100 is assumed.

Based on this adjusted population density, the terminal distribution shown in Figure 2.5 (UTs in blue) has been generated. To this end, the two-dimensional density distribution has been sampled using the corresponding cumulative density functions. 10 000 end user terminals were generated for the figure.

Gateway Station Distribution

GWs serve as interfaces between SCNs and terrestrial networks. In particular, they provide the constellation with high-speed Internet access via feeder ESLs. In general, most traffic is assumed to flow from or to GWs (as discussed in Section 2.1.4). The number and distribution of GWs are critical factors influencing the overall network load and traffic scenarios.

To optimize performance, each gateway campus should be strategically located near existing infrastructure such as data centers. As low latency communication is relevant for certain

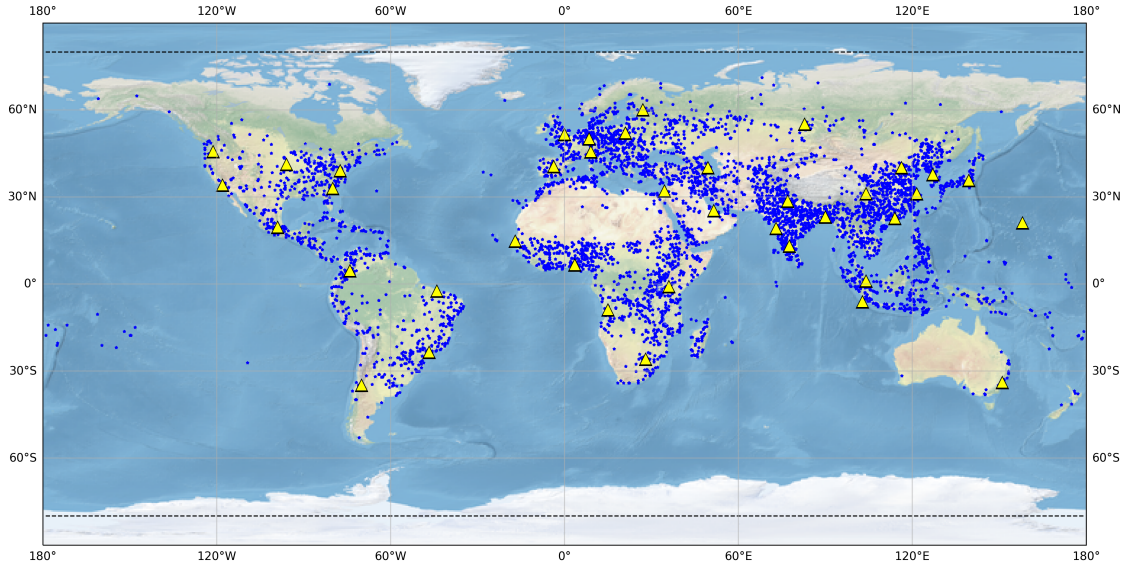


Figure 2.5.: Visualization of UT and GW distributions. 10 000 UT locations (in blue) have been sampled from the filtered population density distribution. 39 GW locations (in yellow) have been chosen deliberately.

services, minimizing the physical distance between UTs and GWs is of interest. However, UT locations can vary significantly, including remote areas such as islands, while viable GW positions are constrained by geopolitical factors which limit where they can be placed. Given these constraints and diverse regional needs, a deliberate approach to dispersing GWs is used. This strategy yields the GW distribution shown in Figure 2.5 (GWs in yellow).

2.1.3. Connectivity

The SCN itself is intended to function as a transport network which enables different types of services. A direct forwarding or encapsulation of IP packets is assumed. As end-to-end paths are considered, without involvement of ground-based infrastructure in between, the network provides direct interconnections between users. This service type may be compared to a leased line known from terrestrial networks. In addition, direct connections into the public Internet are enabled by routing packets via GWs. In this context, UTs are assumed to be provided with public IP-addresses by network control entities.

The satellites deployed in the constellations are equipped with four ISLs and two ESLs. Two intra-plane and two inter-plane ISLs are considered. All user data traffic is aggregated on one ESL, while the other ESL serves a feeder link to ground-based GWs. These connections correspond to what is used in real-world systems [4, 6].

Inter-satellite Links

Similar to current and planned constellations, optical ISLs are assumed. No concrete assumptions about the wavelength are necessary, but typical wavelengths within the near-infrared spectrum around 1550 nm are considered for this use case [52].

The received Power_{rx} is calculated using the Friis transmission equation [45]:

$$\text{Power}_{rx} = \text{Power}_{tx} \cdot G_{tx} \cdot G_{rx} \cdot \left(\frac{\lambda}{4\pi \cdot \text{dist}(tx, rx)} \right)^2 \cdot \eta_{\text{eff},tx} \cdot \eta_{\text{eff},rx} \quad (2.9)$$

Where Power_{tx} denotes the transmitted power, and G_{tx} and G_{rx} the transmitter (tx) and the receiver (rx) gains, respectively. The free-space loss is defined by the wavelength λ and the distance $\text{dist}(tx, rx)$. For optical systems, the transmitter and receiver efficiencies, $\eta_{\text{eff},tx}$ and $\eta_{\text{eff},rx}$ respectively, are also significant. They are sensitive to noise (e.g., thermal and shot noise).

Given the orbital stability, lack of unfavorable environmental conditions, and expected pointing and tracking accuracy, static link characteristics are assumed. Each link can thus be represented by a fixed capacity $c(i, j)$:

$$c(i, j) = \text{Fixed bandwidth of ISL between satellite } i \text{ and } j \quad (2.10)$$

Earth-satellite Links

ESLs connect the satellites with the UTs and GWs on ground. For these links, Radio Frequency (RF) connections are assumed. Working systems as well as research projects typically use frequencies within the Ka-band [4, 47]. Next-generation systems may also utilize Q/V-bands for increased bandwidth [15].

A predictable handover scheme is assumed which is discussed in more detail in Section 3.1.1. ESLs are modeled with constant data rates in this investigation, omitting aspects such as adaptive coding and modulation to isolate in-space routing effects. No assumptions are made about the antenna parameters, link budgets are considered sufficient to maintain the target ESL data rate of 100 Mbps for UTs (see Table 2.3). This simplification ensures that ESL dynamics do not confound the analysis of intra-constellation load balancing. The aggregated maximum feeder uplink and downlink defines the total capacity a satellite can receive from or transmit to ground stations. The connectivity characteristics are summarized in Table 2.3.

A single UT is expected to connect to a single satellite at a time. Each UT is assumed to have one principal network interface to connect to the SCN. It is also assumed that the UTs use phased array antenna technology, which enables rapid beam steering and splitting. So, a search beam can be used to prepare the connection with another satellite. In this fashion, a handover can be prepared beforehand.

This corresponds to a make before break strategy. A new connection is established before an old one is terminated. Using this approach, UTs are capable of seamless handovers, characterized by fast switching which is imperceptible to users. In contrast, hard handovers are assumed to significantly impede services by breaking connections before establishing new ones. This has detrimental effects on transmission quality metrics, and thus negatively impacts QoS profiles. As an example, hard handovers directly impact the delay jitter of a transmission, regardless of the applied routing protocol of the satellite network.

One campus may serve multiple feeder links. One feeder ESL is capable of serving the user ESL of the satellite directly, and four neighboring satellites in terms of data rate.

Table 2.3.: Summary of connectivity and data rates of reference system.

Connectivity characteristic	Value
ISL technology	Optical (at 1550nm)
Number of ISLs per satellite	4
Maximum ISL transmission rate [Mbps]	1000
ESL technology	RF (Ka, V, Q bands)
Number of user ESLs per satellite	[1, 10]
Maximum aggregate user ESL transmission rate [Mbps]	1000
Number of feeder ESLs per satellite	1
Maximum aggregate feeder ESL transmission rate [Mbps]	5000
Maximum aggregate data rate per UT [Mbps]	100

Similar to the discussion of handovers for the UTs, the handovers of GWs are assumed to be seamless.

Reference Transmission Rates

Optical ISLs with transmission rates of 1 Gbps are assumed. For the feeder ESL, the reference transmission rate is set at 5 Gbps, which has been proposed as an operational benchmark [47]. This rate aligns with the maximum output capacity of a single satellite, comprising four ISLs that collectively provide 4 Gbps and one user ESL offering 1 Gbps.

Given these link capacities, it is apparent that forwarding all feeder traffic via a single ISL may not be feasible due to inherent bandwidth limitations. Thus, distributing traffic across multiple paths becomes essential for efficient network operation. Suitable routing solutions must be designed to avoid bottlenecks in the ISL network. By placing this level of demand on the network, a comprehensive evaluation of the performance and scalability of routing and management protocols is enabled.

Telecommunications Payload

The telecommunication payload features a packet routing system, also called satellite router. These on-board routers are considered responsible for efficiently directing data packets within the network. The design is based on an embedded system which integrates both hardware and software components to facilitate advanced packet processing and routing capabilities. It is assumed capable of executing software applications of different scopes.

2.1.4. Traffic Models

The scenario involves examining data exchanges between UTs and GWs, and between UTs. Distinct traffic shares with individual requirements and corresponding QoS classes are assumed.

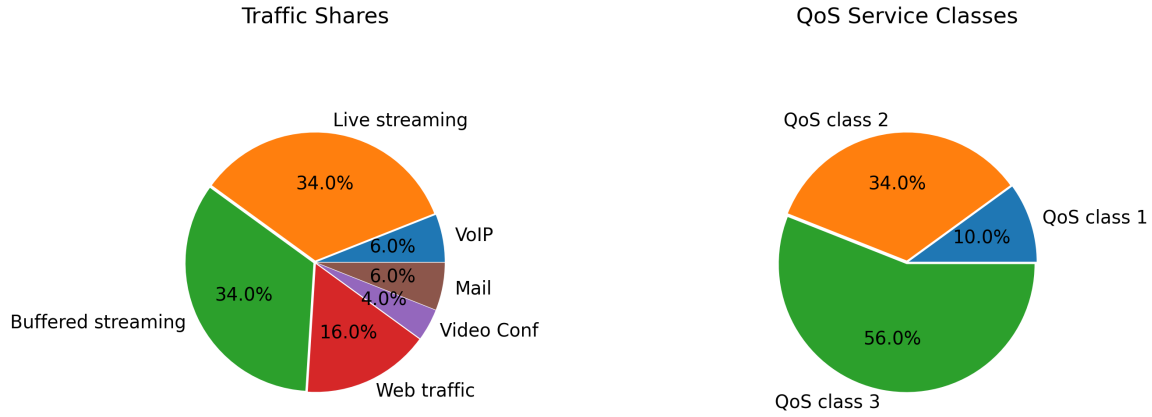


Figure 2.6.: Pie chart of assumed traffic shares and resulting QoS service class shares.

Based on the requirements of specific end-user applications, suitable QoS classes have to be formulated. The considered traffic shares between these classes correspond to system assumptions which have been made in the context of related projects and publications [34, 35, 47]. Since this research primarily focuses on protocol characteristics, solely the presented set of traffic model parameters is investigated.

Traffic Shares

Depending on the provided service, the required data rate of a user might vary. As a general indicator, several data rate requirements for 5G have been quantified [53]. According to their calculations, web browsing traffic normally ranges between 1000 bps and 16 Mbps, while ultra-high-definition video streaming ranges between 3.32 Mbps and 20.75 Mbps.

As there are multiple viable use cases for such a constellation, the considered traffic shares are based on the following assumptions:

- Most traffic, i.e. $> 50\%$, is not delay-critical. Typically, the transfer of larger quantities of data, e.g. in the form of video streaming or file sharing, is buffered. For this traffic, called "best effort", higher delay budgets are possible.
- Importantly, some traffic relies on the short end-to-end transmission delay enabled by SCNs [8]. This type has stringent delay and delay variation requirements. It is however assumed to be a relatively small share due to the limited traffic volume, assumed around 10% . A limited number of frame drops for services using this priority class are assumed acceptable, given its primary focus on end-to-end latency. Thus, an elevated packet dropping rate is considered compliant.
- Some services fall between these classes. They are represented by QoS class 2. While delay-tolerant, the maximum latency is considered lower than the assumed budget for best effort. The class also includes a more relaxed jitter requirement. The rest of the traffic share, i.e. around 30% to 40% , is assumed to belong to this category. For

2. Routing Fundamentals for Satellite Constellation Networks

signaling traffic, low latency and dropping rates are of importance. To this end, it is assumed that signaling packets are prioritized by the scheduler and use dedicated output buffers, which are large enough to effectively avoid packet drops for the considered signaling volume.

Naturally, for dedicated use cases or specific scenarios, more suitable traffic shares can be formulated. The proposed assumptions are intended to reflect a plausible working example of SCNs, which allows for an in-depth analysis of QoS-based routing protocols.

In a related project [47], the following average shares were assumed for UT to GW and vice versa: most traffic is streaming related (68%), web traffic is also significant (16%) and mail (6%), video conferencing (4%) and VoIP (6%) also have a small share. About half of the streaming traffic is assumed to consist of live streaming, which profits from a guaranteed data rate. The other half corresponds to buffered streaming. The classification is illustrated in Figure 2.6. The division of the overall traffic volume is summarized in Table 2.4.

Quality of Service Classes

The considered QoS classes are loosely based on the QoS profiles of the 5G NR networking standard [14]. In the SCN context, only a simplified subset of the given profiles is considered relevant. The proposed values are based on the end-user multimedia QoS categories of the International Telecommunication Union (ITU) [54, 55]. The assumed latency and delay variance budgets roughly follow these recommendations, namely objectives for IP-based services, Y.1541 [56]. Some concrete values have been adapted based on observations and previous investigations into SCNs [8, 35]. From a conceptual point of view, the classifications are similar to the differentiated services (DiffServ) model [57], which is reflected in the names of the considered classes. An overview of the QoS classes is provided in Table 2.4.

2.1.5. Unique Challenges in Satellite Constellation Networks

First and foremost, any LEO SCN represents a **dynamic network topology**. There are frequent ESL as well as ISL handover events due to the relative motion of the space-borne network entities, i.e. the satellites. Different nodes of the constellation network will connect to the same UT (or GW) on the ground within minutes.

The space-borne ISL handovers are highly predictable – the relative motion of satellites is based on orbital mechanics. As the ISL shutdown latitudes are predefined, the deactivation of ISLs can be prepared for in advance. The ESL handovers can be more flexible. Depending on the considered handover mechanisms, there may be a clear mapping of certain areas and satellite coverage. However, such a strategy is rather inflexible. In various circumstances, including failures, overloads, or blocked coverage, dynamic adjustments can be advantageous. For mobile UTs, signaling traffic to clarify ESL association is likely necessary. The general activation and deactivation of connections is also dynamic. In contrast to GWs, UTs may be turned off. So, in this investigation, aggregating all ESL connectivity information of the complete system is expected to result in massive signaling overhead. A quantitative analysis is provided in Section 3.1.2.

This dynamic ESL topology leads to non-static traffic patterns. In contrast to terrestrial

Table 2.4.: Summary of considered QoS classes. Some values from standards have been slightly adapted based on observations and SCN investigations [8, 35].

QoS class	Prio- rity	Packet delay budget	Packet loss rate	Delay vari- ation	Exemplary services	Relative traffic share
0: Signaling	4	<150 ms	$< 10^{-6}$	< 30 ms	Control plane signaling.	N/A
1: Delay-critical forwarding	3	150 ms [54–56]	10^{-2} [54, 56]	< 30 ms [56]	Conversational voice, real-time applications.	10%
2: Delay-tolerant forwarding	2	200 ms	10^{-4} [54, 56]	< 50 ms [56]	Non-conversational video (e.g. live-streams).	34%
3: Best effort	1	300 ms [54, 56]	10^{-6} [56]	N/A	Video (e.g. buffered streaming), file sharing, mail, web activity, multicast.	56%

networks, hot spots are not determined by network topology. Rather, they are related to the geography on Earth. As a result, the temporal drift of traffic characteristics must be considered when designing efficient solutions for these networks.

An additional important difference is the **limited on-board processing capability** of the satellites. On-board routing and network management software must operate on the computational resources of the available hardware. In contrast to terrestrial networks, the power budget and processing power on-board satellites is much more constrained. Given the distributed nature and design patterns of constellations, locally reinforcing network infrastructure is not possible. Moreover, due to the movement of the constellation, the processing requirements can vary. In the case of an SDN-based approach, both control plane and data plane software need to adhere to processing constraints. Exceeding capabilities is assumed to cause immediate performance degradation, through queuing and processing delays in the data plane, or through delayed management changes in the control plane. Similarly, exceeding existing memory limits can result in dropped packets, also degrading the performance immediately.

In general, no assumptions about the specific hardware utilized in the space-borne network are made. However, a suitable processor, the NXP Layerscape 2160 [58], has been investigated for on-board processing in related projects [47, 48]. An overview of this exemplary hardware is provided in Appendix A. It is also used by the hardware demonstrator presented in Section 3.4. As the design of this processor is intended to support network control solutions for 5G applications, it is considered representative for state-of-the-art and potential next-generation satellite hardware.

Nevertheless, maintaining low computational complexity remains an important design aspect, in particular for more complex tasks such as traffic engineering. In a related research

project, routing was only allowed a maximum relative processing load of 30 % [47]. Other applications are assumed to require the remaining resources. So, while the considered hardware is generally able to provide the computational capabilities, reduced complexity is critical in the context of resource management and energy conservation.

2.2. Routing Protocol Overview

2.2.1. State-of-the-art Routing Protocols

This concise overview of the state of the art focuses only on the most relevant protocols and approaches for the considered scenario. Rule-based and ML-based routing solutions are separated as their conceptual architectures typically differ. Whereas rule-based schemes may apply heuristics to simplify complex decision-making, the ML-based approaches learn policies directly from the observed data. For both strategies, several centralized, distributed, and fully decentralized solutions have been proposed.

Rule-based Routing Solutions

Relying on a virtual topology, representing a snapshot of the constellation in time, **source-routing** approaches compute paths through the pseudo-static network at the ingress node. Shortest Path First (SPF) approaches focus on lowest cost paths, typically based on the propagation delay between nodes, for instance, by using Dijkstra's algorithm [59]. Protocols such as Multi-Protocol Label Switching (MPLS) [60] can be used for the forwarding between satellites. QoS-aware source-routed approaches have also been investigated [8, 35]. The idea is to choose paths, potentially from a set of viable paths, according to their expected characteristics.

However, these approaches are prone to hot spots in high-traffic mesh networks [61]. Key limitations include information retrieval, as forwarding end-to-end link states to every ingress node results in significant signaling overhead [35]. Strict handover rules are required, as changes in ESL connectivity can cause packet drops if handovers occur during transmission. In this case, packets are forwarded according to a deprecated list of hops. Nevertheless, the approach represents a relevant benchmark. It reflects the behavior of known terrestrial protocols, e.g., Open Shortest Path First (OSPF). In general, comparisons with these protocols are difficult as they oftentimes face challenges in dynamic topologies [8]. In addition, due to the lack of dedicated backbone networks, to which all satellites are connected, terrestrial protocols are not applicable without significant modifications making them difficult to include as benchmarks [62]. A more flexible source-routed solution is **Segment Routing (SR)** [63]. This routing paradigm is based on the idea of encoding a sequence of segments, i.e., specific network functions, instead of individual hops.

In virtual-node and **geographical** routing, packets are forwarded based on the topology of a virtual network. Logical nodes with fixed geographical locations are used. This approach provides stability and predictability in the routing process, even in the face of a constantly changing network topology. Geographical routing uses static, physical locations as destinations. The Earth is partitioned into geographical areas with unique identifiers [64, 65]. Packets are forwarded in the direction of a corresponding destination region [39, 66, 67]. To

circumvent ambiguities in the last hop, additional mechanisms, such as local signaling, are required. As mobile terminals may change their geographical identifier, mobility management strategies are also important [68].

Moreover, **decentralized and distributed routing** schemes have been proposed for satellite or similar networks [69, 70], including distance vector approaches [71]. Some also provide decentralized load-balancing capabilities [72, 73]. Conceptually they rely on information exchanges between neighboring nodes to avoid saturated links. Due to the limited scope, only local decisions are made. The lack of coordination can also result in conflicting decisions or frequent adjustments.

A distributed architecture based on Intermediate System to Intermediate System (IS-IS) has also been published [62]. In this context, Segment Routing Traffic Engineering (SR-TE) can also be applied to actively steer traffic, e.g. to avoid known hot spots [74]. To maximize throughput in SCNs, additional approaches for steering traffic efficiently have been proposed, for instance, by applying heuristics to solve the linear-programming problem underlying a static MCFP [24]. However, distributed SDN is generally more flexible and adaptable given its softwarization of routing logic and fine-grained network control.

SDN is applied by many recent routing schemes for satellite networks [25, 75, 76], and is important for proposed 6G ecosystems [15, 77, 78]. Different architectures have been investigated, but typically, the choice is to use a control entity on ground or to have in-orbit controllers. When using a ground-based approach, the control unit acts as an enabler for satellites to make informed autonomous routing decisions [76].

For in-orbit routing architectures, distributed SDN is considered particularly promising [79, 80]. Corresponding research has also focused on the SDN controller placement problem [81, 82]. Given the flexible and fine-grained switching rules, which can be established by SDN, it provides a robust framework for QoS-aware routing [83].

Machine Learning-based Solutions

In recent years, various ML-based routing approaches have been proposed for wireless networks [84–86] and in the context of SDN [87, 88]. For ML-enabled network control, DRL and representation learning are typically used.

Representation learning techniques have been proposed for routing, predominantly in the context of predicting traffic demands [89]. To this end, Recurrent Neural Network (RNN) designs, such as Long Short-Term Memorys (LSTMs), are often considered [85, 90, 91]. Graph Neural Networks (GNNs)-based approaches are of particular interest, as they are able to extract features of the network more efficiently, which can provide more reliable predictions [92]. The technique has also been proposed for low-complexity on-board routing in SCNs [93]. Using topological feature extraction, a trained model has strong generalization performance and can handle dynamic topologies. GNNs are also used to find easier representations of the network, which are then used by a DRL agent as part of its environment [94].

For optimizing network control directly, **DRL** techniques are considered the most promising methods [85, 87]. For DRL-based routing in SCNs, a common approach is to apply hop-based

DQN schemes [95–97]. Designs have also included capabilities for QoS-aware forwarding and heterogeneous networks [98] as well as load balancing [99]. The approach has also been applied in the context of the MCFP [100–102].

Hierarchical architectures have been proposed as well [103,104], where routing decisions are made on different levels. Similar to distributed SDN architectures, group leaders control sub-networks. In this context, heuristics are used to decrease the complexity of the learning problem. In general, system-specific assumptions or pre-processing steps are oftentimes used to facilitate DRL-based solutions.

Assessment

In conclusion, the overview suggests that distributed SDN is a highly promising approach for enhancing load balancing and routing in autonomous next-generation SCNs. Given its flexibility, adaptability, and ability to support arbitrary flow allocations, it is an attractive framework for integrating rule-based as well as ML-based routing schemes.

The distributed SDN approach applies locally centralized control, which is expected to distribute traffic more efficiently. In contrast, decentralized load-balancing schemes are limited due to their restricted scope. However, a good understanding of the network and its entities is required to solve the MCFP effectively.

Moreover, DRL-based approaches have shown significant potential in diverse routing scenarios, including SCNs. While such methods have been able to enhance QoS-aware routing, there is a notable gap in existing research: there is currently no approach designed to specifically maximize network throughput in distributed SDN-enabled SCNs. So, there is an opportunity to develop a tailored DRL-based solution which makes use of the strengths of distributed SDN and addresses the unique challenges of efficient routing in SCNs.

2.2.2. Standardization Activity

Routing in SCNs is becoming increasingly crucial as these networks are expected to play a significant role in future communication frameworks, particularly in the context of 5G Beyond and 6G technologies [105]. The integration of LEO satellite networks into 6G systems is seen as a promising technology due to their characteristics of low latency, seamless coverage, and high bandwidth. In the context of standardization, the following design aspects are of particular interest:

- **Seamless integration with Terrestrial Networks (TNs):** requiring the development of satellite-terrestrial cooperative routing strategies.
- **Handling increasing traffic loads:** new standards typically support growing data volumes and demand by providing higher system throughput.
- **Ensuring QoS:** supporting various 5G Beyond and 6G applications which require high reliability and low end-to-end latency.

End-to-end routing is thus considered an open technical challenge in the context of enabling NTN-based 5G Beyond and 6G networks [19, 77, 106]. The diverse characteristics

of network segments as well as the dynamic topology complicate optimal path-finding using existing approaches, particularly in the **distributed network of networks** envisioned by 6G. Routing protocols may consider content-aware routing, semantic communications, and distributed control, especially in multi-orbit multi-layer satellite networks [19]. Additional open challenges include orbital edge computing, network orchestration, and network slicing.

There is also standardization progress in the direction of **ML for Network Management** [107]. To achieve this, both hierarchical and distributed intelligent architectures are being explored. Among others, important optimization aspects include traffic steering, QoE prediction, and QoE assurance. Notably, 6G is envisioned as an ML-native network, where protocols and platforms are designed to seamlessly support ML services, enabling efficient and adaptive management of complex networks.

In this context, the possible functional splits in 6G with regard to the NTN integration are of interest. Here, functional split refers to the division of responsibilities between different network entities, such as the user equipment, the radio access network, and the core network. Accordingly, the placement of data paths as well as routing and forwarding functions in 6G-NTN networks is determined [108–110].

Certain architectures envision distributed in-orbit network control which seamlessly integrates with terrestrial infrastructure and networking nodes [107]. For instance, variants of a functional split option are being investigated, which enable on-board routing functionality, allowing traffic flows to be routed within the space segment without going to the ground [111]. The resulting distribution of entities and control domains may strongly resemble the distributed SDN approach presented in this work.

Naturally, as the standardization is being developed, the concrete viability of the proposed approaches is hard to assess. Thus, an in-depth discussion or placement of the investigated methods is omitted. Nevertheless, so far many developments and trends are congruent with the concepts presented in this work. The proposed protocol design, functionality, and ML-based enhancements are enablers for autonomous SCNs. As such, they are of significant interest for standardization efforts. The proposed architectures deliver on the formulated design guidelines for next-generation satellite systems. So, the presented approaches are well-positioned to be of relevance for the development of 6G NTN schemes. The ability to distribute flows efficiently can help achieve the high-throughput requirements of future satellite networks.

2.2.3. Key Performance Indicators

The efficiency of a given routing solution can be evaluated using various distinct metrics, which are described by KPIs. Based on these indicators, approaches can be compared with each other and with benchmarks. The considered KPIs are tailored to the requirements of the reference system and the intended optimization goals (in Section 1.2). For different services or service classes, some KPIs are more important than others. Thus, performance analyses will consider the corresponding service or QoS classes. An overview of all considered KPIs is provided in Table 2.5. In the following, the different metrics are described and formalized.

Table 2.5.: Overview of KPIs. Considered primary and secondary metrics.

KPI	Unit	Relevance to Routing in SCNs	Impact Drivers	Evaluation
Primary KPIs: defining QoS classes, optimization goals				
Latency	[ms]	End-to-end requirement for delay-critical services	Constellation design; terminal distribution; traffic model	Box plots
Jitter	[ms]	Reduction of re-ordering for delay-critical services; sensitive to handovers	Constellation topology; terminal distribution; traffic model	Histograms
Packet Dropping Rate	[%]	Critical for throughput maximization; reduction of re-transmissions	Load-balancing performance; terminal distribution; traffic model	Values
Link Utilization	[%]	Critical for throughput maximization; indirect driver of packet loss	Load-balancing performance; terminal distribution; traffic model	Values, visualizations
Secondary KPIs: protocol design impact				
Signaling Overhead	[%]	Relative share of control to data packets	Network control distribution and placement; update frequency	Values
Routing Convergence	[ms]	Determines reactivity and network resilience	Network control distribution and placement	Values
Complexity	–	Critical to classify algorithmic viability	On-board processing; task distribution	Big $O(\cdot)$ notation; runtime

Latency

From a routing perspective, maintaining an adequate end-to-end latency is essential. To achieve this, paths of suitable lengths have to be formulated, as propagation delays constitute a fundamental component of latency in wireless communication networks.

In addition, queuing delays can play a substantial role in latency. Depending on the queue dimensioning, the repeated buffering of packets may introduce significant delays. Therefore, load-balancing decisions can be particularly important. Specifically, in large constellations, with typically longer multi-hop paths, small queuing delays at each node can lead to substantial additional latency.

The propagation delay between two nodes is denoted by $l_{p,t}$. It is time-dependent since the distance, and thus the delay between connected nodes, is constantly changing. Only

for intra-plane ISLs, the distance can be assumed to be constant. The propagation delay between nodes i and j can be expressed as follows:

$$l_{p,t}(i, j) = \frac{\text{dist}_t(i, j)}{c} \quad (2.11)$$

Here, $\text{dist}_t(i, j)$ describes the physical Euclidean distance between nodes i and j . c denotes the speed of light in a vacuum, which is assumed to be an adequate approximation of the actual physical velocity. The distance can be estimated using the approximate Cartesian coordinates of the satellites.

The queuing delay of node i is represented by $l_{q,t}(i)$. In general, the maximum queuing delay varies based on the chosen packet sizes and on-board buffer sizes. Packet sizes may vary, but cannot exceed the Maximum Transmission Unit (MTU). To describe the maximum possible buffer allocation, the maximum amount of packets M_{max} of packet size MTU which fits the on-board buffer capacity $b(i)$ is used:

$$M_{max} \cdot \text{MTU} \leq b(i) \quad (2.12)$$

If cut-through switching is assumed or the buffer capacity is a multiple of the packet size, this value is equal to $b(i)$. By setting this maximum buffer allocation in relation to the corresponding ISL data rate $\eta_{ISL}(i, j)$, it is possible to formalize the maximum queuing delay of a single ISL hop:

$$l_{q,max}(i) = \frac{M_{max} \cdot \text{MTU}}{\eta_{ISL}(i, j)} \quad (2.13)$$

In general, buffers tend to be near-empty or in an overloaded state (at maximum buffer allocation). For this reason, the current link utilization is oftentimes preferred over the buffer allocation to serve as a load metric.

There are additional switching and routing delays to consider. The routing delay l_r represents the time required to determine the next hop of a packet, so it can be directed to a corresponding output port. Depending on the considered approach, this may consist of the reading of a label (e.g., in MPLS) and/or a routing table lookup. Although table sizes and lookup procedures may vary, an average value is assumed for this metric. The switching delay l_s is the typical time required to switch a packet from the input buffer to the corresponding output buffer. Both of these delays are assumed to be relatively constant and, for all relevant scenarios, to be in the domain of μs . It is not assumed that these values vary significantly over time.

Therefore, the time-dependent multi-hop latency $L_t(p)$ of path p can be described by (with observed time domains below):

$$L_t(p) = \sum_{(i,j) \in p} \left[\underbrace{l_s(i)}_{[\mu s]} + \underbrace{l_r(i)}_{[\mu s]} + \underbrace{l_{q,t}(i)}_{[\mu s]} + \underbrace{l_{p,t}(i, j)}_{[ms]} \right] \quad (2.14)$$

For the considered end-to-end latency in this analysis, the ingress and egress ESLs are also added to this latency.

Jitter

Jitter is used to quantify inter-packet delay of consecutive packets. It is of interest for various delay-critical applications, e.g. for real-time voice and video applications. In SCNs, significant delay variations due to handover events occurring during transmissions are expected. Moreover, the alteration of routes during a transmission can also cause significant jitter. In large constellation networks, there are often multiple routes available of comparable latency. So, large variations may be mitigated. In scenarios with high network loads, the expected jitter is greater due to the increased traffic demand.

In this investigation, the maximum and average observed instantaneous jitter J_{inst} are used. In the given context, this is the maximum latency variation of subsequent packets, so the difference between $L_{\rho-1}$ and L_{ρ} for the M packets of a session:

$$J_{\text{inst, max}} = \max_{\rho \in \{2, \dots, M\}} |L_{\rho} - L_{\rho-1}| \quad (2.15)$$

Packet Dropping Rate

The packet dropping rate is a useful indicator to evaluate the effectiveness of load management. QoS profiles typically set thresholds on the dropping rate, which solutions have to comply with. In SCNs, packet drops are particularly costly. Re-transmissions of missing packets are problematic, given the potentially long end-to-end latency in the domain of more than 100 ms. On the other hand, due to the dynamic network topology, avoiding packet drops by design is difficult for some approaches, e.g., source-routing.

Link Utilization

Link utilization, commonly referred to as link load, is used to assess the impact of traffic load. This metric is typically applied when investigating how to minimize packet dropping rates. So, the KPI impacts QoS compliance indirectly. The observed link utilization is formally described in Section 2.3.

Secondary Key Performance Indicators

In addition, several secondary metrics are considered. The differentiation is made since these aspects are of relevance, but not reflected directly by the QoS requirements.

Signaling overhead: In addition to the user traffic, network management signaling and updates have to be transmitted as well. In general, this overhead should be minimal while providing all required network management functionalities. It can be expressed as the ratio of the number of forwarded signaling packets N_{signal} to the number of forwarded data packets N_{data} :

$$\text{Overhead} = \frac{N_{\text{signal}}}{N_{\text{data}} + N_{\text{signal}}} \quad (2.16)$$

A formulation based on management cost will be introduced in Section 3.1.2.

Routing convergence: The delay of establishing a coherent routing environment after changes is called the routing convergence. In practice, this is the amount of time required

for a routing table entry to take effect. This duration between the instant in time introducing a change t_{change} and the moment of stability t_{stable} is thus:

$$T_{\text{conv}} = t_{\text{stable}} - t_{\text{change}} \quad (2.17)$$

This KPI strongly depends on the utilized protocol architecture. The amount of relevant nodes which require coherence varies significantly. For instance, in a fully decentralized approach, a local update between neighbors can be sufficient.

Complexity: For rule-based approaches, complexity in time and space can typically be estimated based on the applied algorithms. For ML-based approaches, model size and corresponding processing can also provide a rough estimation of the computational requirements. In any case, evaluating and comparing the amount of actually required resources provides valuable insights into the feasibility and applicability in real-world systems. $\mathcal{TC}$ is utilized for the complexity in time, and $\mathcal{SC}$ for the complexity in space.

2.3. Network Optimization Theory

This section provides an overview of the general optimization methods used for routing. Routing problems can typically be described by graphs, and are thus categorized as graph theory problems. The complexity of the routing problem depends on its formulation, involving two key aspects: path selection and routing configuration.

Path selection is crucial as it determines the characteristics of the end-to-end connection. If the selection depends on a single metric, such as finding the shortest path, Dijkstra's algorithm [59] or similar approaches can be employed. The individual edge costs, which are used by SPF algorithms, can also be combinations of multiple metrics. However, for flexible and efficient handling of various input parameters, additional algorithms or different approaches may be required. In dynamic scenarios, finding optimal routes typically requires more sophisticated algorithms which incorporate additional techniques, e.g., adaptive load-balancing.

The individual path selection is but one aspect of the larger goal of finding optimal routing configurations which use the available network resources effectively. If critical resources, such as high-demand links, are not handled carefully, subsequent requests may not be routed correctly. To address this challenge, this section provides a formal description of this problem using the MCFP. The analytical basis of related Linear Programming (LP) and DP techniques to solve such problems is introduced as well.

2.3.1. Graph-Theoretic Representations

To provide a theoretical background for the proposed routing solutions, models and techniques from graph theory are applied. A graph can be described by $\mathcal{G}$, consisting of the set of vertices $\mathcal{V}$ and the set of edges $\mathcal{E}$.

$$\mathcal{G} = (\mathcal{V}, \mathcal{E}) \quad (2.18)$$

2. Routing Fundamentals for Satellite Constellation Networks

In SCNs, links are full-duplex. Thus, $\mathcal{G}$ describes a directed graph. Each edge can be described by its adjacent vertices and direction. Therefore, edges are denoted by $(i, j) \in \mathcal{E}$. The capacity of each edge is assumed to be a positive, real number:

$$c(i, j) \in \mathbb{R}_{>0} \quad (2.19)$$

A directed graph with edge capacities is commonly described as a flow network [32]. A flow represents a commodity from a source node to a destination node with a certain demand. The following notions are used to describe a flow f :

- $v_{src} \in \mathcal{V}$ is the source node,
- $v_{dst} \in \mathcal{V}$ is the destination node,
- $d \in \mathbb{R}_{>0}$ is the demand to be routed from v_{src} to v_{dst} .

The adjacency matrix $\mathbf{A}$ of a graph defines the connections between vertices. The general adjacency is typically indicated by values of 0 or 1:

$$\mathbf{A}[i, j] = \begin{cases} 1, & \text{if } (i, j) \in \mathcal{E} \\ 0, & \text{otherwise} \end{cases} \quad (2.20)$$

It is also possible to formulate an incidence matrix, which captures relationships between edges and vertices. However, most protocols and algorithms rely on adjacency matrices.

Graph Properties

Formally, a path p is a sequence of edges which connects two vertices v_{src} and v_{dst} . By traversing each edge of the path in order, the destination node v_{dst} is supposed to be reached from the source v_{src} . The nodes traversed in between are commonly called hops.

If two subsequent nodes forward packets between them repeatedly, a deadlock occurs. In communication networks, this is costly: the destination node is never reached and the packet is typically dropped after its Time To Live (TTL) is reached.

A cycle or loop is a path or sub-path in which the source node is identical to the destination node with no repeated edges in between. Typically this is undesired as v_{src} and v_{dst} are supposed to be different nodes. For routing in communication networks, negative edge weights are typically not considered, as the weight represents positive metrics such as latency or link load. Without negative (and zero) edge weights, a pathfinding algorithm which minimizes accumulated edge weight should thus not include cycles. Acyclic graphs represent directed graphs in which loops or closed paths are not possible. This assumption does not hold for the reference scenarios. In general, deadlocks and loops represent an avoidable depreciation of performance. Therefore, they are strong indicators of unsuitable routing solution design.

The degree $\deg(v)$ of a vertex v is the sum of incoming and outgoing directed edges.

Dynamic Graphs

SCNs are characterized by time-dependent topologies as described in Section 2.1.5. Therefore, a dynamic graph model, in which properties change over time, has to be used to capture these dynamics:

$$\mathcal{G}_t = (\mathcal{V}_t, \mathcal{E}_t) \quad (2.21)$$

Edge weights in particular are expected to vary over time:

$$w_t(i, j) = w(i, j) \text{ at instant } t \quad (2.22)$$

Due to handover events, a time-dependent adjacency matrix $\mathbf{A}_t$ has to be considered.

Snapshot Definition

While SCNs represent dynamic networks, a common approach is to use snapshot-based routing schemes. A snapshot is a static representation of the dynamic graph in a single moment in time. Typically, for the duration of the snapshot, no significant changes to the network characteristics are assumed. Consequently, suitable solutions to the routing problems at hand can be formulated based on considerations in temporarily static graphs. $\{[t_n, t_{n+1})\}$ is used to describe the series of unit snapshots. Snapshots of future network states due to predictable changes can be aggregated to enable proactive routing decisions.

Pathfinding Algorithms

For the formulated graphs, pathfinding algorithms are applicable to find suitable routes. SPF algorithms are designed to find the shortest path based on given edge weights w . As no negative weights are present, SPF algorithms are not susceptible to cycles in this context. It is important to differentiate whether a shortest path from a source to a destination or an Shortest Path Tree (SPT) from a source to any other vertex is required. In general, for SPF, the lowest total weight $W_{p,tot}$ is of interest. It is the sum of the weights of all edges of the path p :

$$W_{p,tot} = \sum_{(i,j) \in p} w(i, j) \quad (2.23)$$

For dynamic graphs, time-dependent SPF algorithms have to be utilized. As described, edge weights are no longer constant in this approach. Hence, only currently applicable weights w_t are used.

Dijkstra's Algorithm

For Dijkstra's algorithm [59], the idea is to use a priority queue which contains the smallest tentative distance, i.e. the current cost, to the destination. At first, all tentative distances are set to $+\infty$ except for the source node, for which it is 0. Then, the distances to the neighbors of the node with the currently lowest value (initially the source node) are computed. If a node has already been visited and no lower distance was found, it is not updated. This process is repeated until the destination node is reached or until the priority queue is

empty. If the destination was reached, the path can be reconstructed recursively. If there is no breaking condition after finding the destination, an SPT for the complete network is constructed.

The complexity in time of Dijkstra's algorithm can be estimated using the fact that in each iteration, vertices which have not yet been visited are processed. So, a maximum of $|\mathcal{V}|$ computations for $|\mathcal{V}|$ iterations [32]:

$$\mathcal{TC}_{\text{dijkstra}} \leq O(|\mathcal{V}|^2). \quad (2.24)$$

Using advanced data structures, a significantly lower time complexity can be achieved [32, 112]:

$$\mathcal{TC}_{\text{dijkstra}} = O(|\mathcal{E}| + |\mathcal{V}| \log |\mathcal{V}|). \quad (2.25)$$

A*-Algorithm

Since this approach is not directed, it is oftentimes not the most efficient solution to find the least-cost path. The A*-algorithm is a heuristic-based extension of the approach. A heuristic h is used, which estimates a general cost to the destination:

$$w_h(i, j) = w(i, j) + h(i, j) \quad (2.26)$$

A common example for a heuristic is the Euclidean distance to the destination as a general direction. In any case, the value of the heuristic has to be chosen carefully to guarantee the intended behavior.

Constrained Approaches

By introducing constraints, paths which contain undesired properties can be discounted. For instance, link utilization thresholds can be used to avoid highly loaded edges preemptively. To this end, edges in violation of predefined constraints are pruned or rendered more costly. To avoid potentially dissecting the graph, the latter approach is generally preferable. As the constraints reduce the number of viable paths, SPF algorithms can converge faster toward the destination. The inclusion of constraints makes the algorithm suitable for additional objectives which are based on thresholds.

2.3.2. Multi-Commodity Flow Problem

In order to propose paths resulting in routing configurations which enable the highest possible throughput, approaches which solve or approximate solutions to the MCFP have to be formulated. The MCFP provides a framework for modeling and solving the network flow distribution when multiple traffic demands (commodities) must be routed through a network with limited capacity.

The problem represents a generalization of the **Max-Flow Min-Cut theorem** [32]. The core insight of this statement is the fact that the maximum achievable flow through a network, from a source to a sink, is equal to its minimum cut. In this context, the minimum cut describes the set of edges which bi-partites the graph $\mathcal{G}$ with the minimum aggregated capacity. In other words, the tightest combined bottleneck when considering all edges.

It is possible to consider paths based on the edges of the minimum cut to maximize throughput. However, the approach is only valid for a single source-destination pair. A computed path may be sub-optimal when considering additional commodities which rely on related edges. Hence, the more general MCFP has to be considered in the context of this investigation.

If integer flows are considered, as in the Integer Multi-Commodity Flow Problem (I-MCFP), the problem is NP-complete [32]. The Fractional Multi-Commodity Flow Problem (F-MCFP), on the other hand, can be solved in polynomial time using LP [32]. Nevertheless, for increasingly large networks, the problem size grows rapidly. Therefore, approximate solutions may be considered for large networks since complete LP solutions are costly.

As the overall objective is to support as many incoming flows as possible without saturating links, a minimization of the maximum link utilization is fundamental. The *utilization* $u(i, j)$ is the total flow on the edge (i, j) divided by its capacity $c(i, j)$:

$$u(i, j) = \frac{f(i, j)}{c(i, j)} \quad (2.27)$$

There are other objectives which can be considered, such as minimizing the average link utilization. However, avoiding saturation remains the principal focus, as packet drops are particularly costly in SCNs.

Multi-Commodity Flow Problem in Static Networks

A set of commodities K with a cardinality of $|K| > 1$ is assumed. Each commodity $k \in K$ can be split and forwarded on separate paths. On edge (i, j) , the flow of commodity k is $f^k(i, j) \in \mathbb{R}_{\geq 0}$. The total flow on edge (i, j) is the sum over all commodities:

$$f(i, j) = \sum_{k \in K} f^k(i, j) \quad (2.28)$$

As mentioned, the considered objective is to minimize the maximum utilization among all edges in the network. Thus, the objective function is:

$$\min \max_{(i, j) \in \mathcal{E}} u(i, j) \quad (2.29)$$

The MCFP can be formulated as linear programs and solved efficiently using LP techniques. The problem is subject to the following constraints.

Flow conservation constraints, for all $k \in K$ and $i \in \mathcal{V}$:

$$\sum_{(i, j) \in \mathcal{E}} f^k(i, j) - \sum_{(j, i) \in \mathcal{E}} f^k(j, i) = \begin{cases} d^k, & \text{if } i = v_{src}^k, \\ -d^k, & \text{if } i = v_{dst}^k, \\ 0, & \text{otherwise} \end{cases} \quad (2.30)$$

Capacity constraints, for all edges $(i, j) \in \mathcal{E}$:

$$f(i, j) \leq c(i, j) \quad (2.31)$$

Non-negativity constraints:

$$f^k(i, j) \geq 0, \quad \forall k \in K, \forall (i, j) \in \mathcal{E} \quad (2.32)$$

Implicitly, Equation (2.30) requires that the complete flow demand originates from the source node and is routed toward the destination:

$$\sum_{(i,j) \in \mathcal{E}} f^k(i, j) = d^k \quad \text{for } i = v_{src} \quad \text{and} \quad \sum_{(j,i) \in \mathcal{E}} f^k(j, i) = d^k \quad \text{for } i = v_{dst} \quad (2.33)$$

This corresponds to the assumption of ingress and egress satellites. Based on these constraints, an LP solver can solve the MCFP for a given network and set of flows.

In the integer version of the MCFP, the I-MCFP, flow splits are restricted to integer values:

$$f^k(i, j) \in \mathbb{Z}_{\geq 0}, \quad \forall k \in K, \forall (i, j) \in \mathcal{E} \quad (2.34)$$

The I-MCFP is often considered more representative of real-world network flows, as packets are typically treated as the smallest indivisible unit of data transmission. However, due to its NP-completeness, the problem is frequently relaxed to the F-MCFP [23, 32].

Linear Programming

As the considered objective function is linear, linear or integer programming can be used. LP is used for continuous optimization problems, while integer programming is used for discrete problems. For the sake of simplicity, the used term LP includes integer programming.

In LP, decision variables are used. In the context of the MCFP, these variables are the flow values $f^k(i, j)$. In addition, there are auxiliary variables which are used in the program formulation, but are not decided upon directly. For instance, $f(i, j)$ represents an auxiliary variable. As the MCFP is formulated as a linear program, the same objective function and flow conservation constraints are applicable:

$$\min \max_{(i,j) \in \mathcal{E}} u(i, j) \quad \text{s.t.} \quad \sum_{k \in K} f^k(i, j) \leq c(i, j), \quad f^k(i, j) \geq 0 \quad (2.35)$$

There are various algorithms to solve LPs. Commonly used approaches are based on the Simplex Method [32]. The idea of this approach is to iteratively improve the solution by moving from one vertex of the feasible region to another, until the optimal solution is reached. So, the algorithm traverses edges of the multi-dimensional optimization region to find an optimum.

Dynamic Multi-Commodity Flow Problem

In dynamic networks, the set of flows follows a temporal order. Accordingly, the graph is expanded over time steps t to model these dynamics. Thus, flows on edges now depend on t :

$$f_t^k(i, j) \quad \text{for } t \in T \quad (2.36)$$

Similarly, the link utilization $u_t(i, j)$ is now time-dependent. For certain scenarios, the link capacities also change over time, hence $c_t(i, j)$ is used.

$$u_t(i, j) = \frac{f_t(i, j)}{c_t(i, j)} = \frac{\sum_{(i, j) \in \mathcal{E}} f_t^k(i, j)}{c_t(i, j)} \quad (2.37)$$

There are two options to formulate the objective function of minimizing the maximum link utilization over time. First, by extending the problem over all time steps. This represents the general approach of minimizing over all time steps:

$$\min \max_t \max_{(i, j) \in \mathcal{E}} u_t(i, j) \quad (2.38)$$

Or, secondly, by minimizing the time-averaged maximum link utilization over the time horizon T :

$$\min \frac{1}{T} \sum_{t=1}^T \max_{(i, j) \in \mathcal{E}} u_t(i, j) \quad (2.39)$$

Dynamic Programming

As the network states are directly impacted by the chosen route, any path selection affects subsequent routing decisions. So, when optimizing the overall performance over time, the sub-problems of individual routing decisions overlap. For such sequential decision-making, dynamic programming is a suitable approach [113]. To find solutions, good knowledge of the network and its state evolution is required.

The state variable s_t describes the network state at time t . In addition, a matrix of flows $\mathbf{F}_t$ is considered, which contains all flows $f_t^k(i, j)$ at time t . As before, the decision variables are $f_t^k(i, j)$. Moreover, a transition function between time steps is now required, which updates the current state based on the flows.

$$s_{t+1} = \Phi(s_t, f_t) \quad (2.40)$$

To formulate a suitable policy, the value function V_t is used to evaluate potential next states. This function considers the resulting cost, represented by a cost function C_t for the current moment in time, as well as the impact on the value of future states. So, the value function of the current state is based on the value function of subsequent states. Consequently, the computation relies on solving recursive sub-problems. Leveraging the Bellman equation enables a formulation of an optimal policy:

$$V_t(s_t) = \min_{f_t} [C_t(s_t, f_t) + V_{t+1}(s_{t+1})] \quad (2.41)$$

Given the objective function, the immediate cost function C_t is:

$$C_t(s_t, f_t) = \max_{(i, j) \in \mathcal{E}} u_t(i, j) \quad (2.42)$$

2. Routing Fundamentals for Satellite Constellation Networks

So, for the complete time horizon from time t to T , the optimal value function representing the minimal total cost is:

$$V_t(s_t) = \min_{f_t, \dots, f_T} \left(\max_{t' \geq t} \left[\max_{(i,j) \in \mathcal{E}} u_t(i,j) \right] \right) \quad (2.43)$$

This DP formulation implicitly assumes several simplifications. It does not include delay functions, queuing models, and most importantly stochastic traffic models. Therefore, formulating and solving DP for the considered SCN scenario is difficult, which will be detailed later in Section 3.2.1.

In addition, using DP results in high computational complexity due to the curse of dimensionality [27]. The state space grows exponentially with the number of input parameters. So, for large state spaces DP approaches may not be applicable in practice. Given the amount of parameters, heuristics are typically considered to make the problem manageable [24, 113].

3. Distributed Load-balanced Software-Defined Networking

In this chapter, a novel protocol based on distributed SDN is introduced, which provides load-balancing capabilities and enables advanced traffic engineering methods. As described in the overview of the state of the art (in Section 2.2.1), this framework is particularly promising in the context of SCNs. The specific design considerations and trade-offs which have shaped the proposed protocol, named Independent Distributed Load-balanced routing (IDLB), are discussed.

To provide a strong background to the approach, precursor concepts which have informed the design of the approach are also presented. Key insights from geographical routing and set-based path selection techniques are highlighted.

The performance of the approach is then thoroughly evaluated by extensive simulations. The evaluation is enabled by a system-level simulator specifically designed for this investigation. In addition, the feasibility of the protocol is validated by demonstrating its functionality on a hardware test bed.

3.1. Protocol Design

The developed routing protocol, called IDLB, represents a load-balancing routing scheme based on distributed SDN tailored specifically to LEO SCNs. It is based on a space-borne sub-division of the network into clusters, i.e. different domains. These sub-networks are fixed within the topology of the constellation. Thus, from a terrestrial point of view, the clusters move, as the involved satellites orbit Earth. The positions of the clusters are independent of the infrastructure on ground. Typically, a satellite near the center of a cluster takes on the role of the cluster controller. For the design of IDLB, insights from a variety of routing approaches are applied to formulate a streamlined protocol for the reference systems and traffic scenarios. IDLB consists of a distributed space-borne architecture for flexible, adaptive decision-making, which enables load-balanced intra-cluster routing, and heuristic inter-cluster routing based on geographical information.

The architecture of the proposed protocol is illustrated in Figure 3.1. Rectangular clusters of 12 nodes are visualized using different colors. The satellite near the center, which is acting as the cluster controller, is highlighted by an orange ring. A potential path through the satellite network is shown in Figure 3.2. A terminal in the red geographical area, which is served by the red cluster, sends packets to a terminal in the blue geographical area, which is served by the blue cluster. The resulting path along the ISLs is decided by the combined decisions of intra- and inter-cluster routing mechanisms. The core concepts which have influenced the design of the proposed protocol are presented in the following.

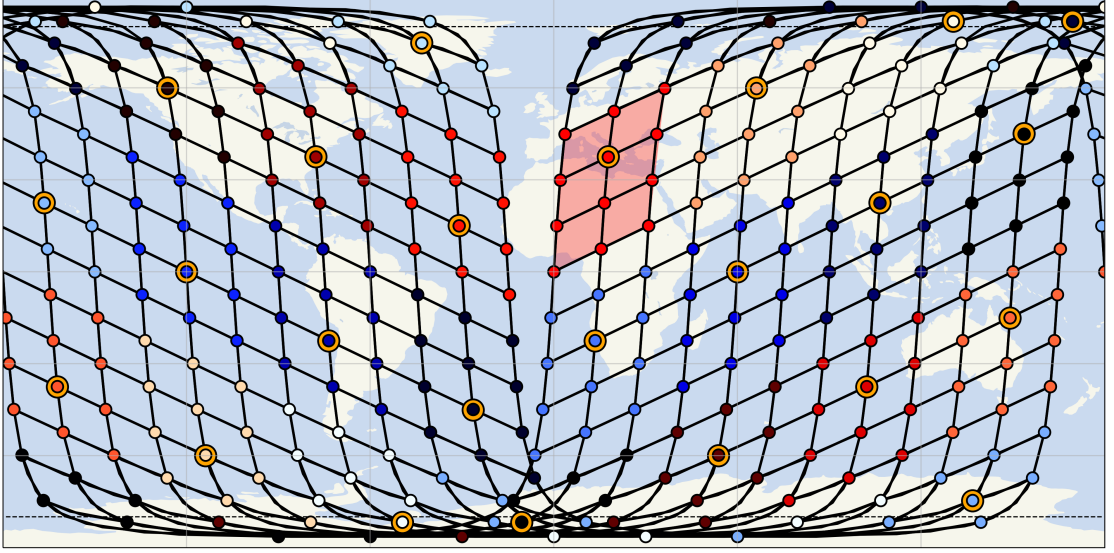


Figure 3.1.: Rectangular arrangement of clusters for SCN-288 using a cluster size of 12. Nodes of the same cluster are colored identically. The region spanned by a cluster is shown (in red). Satellites with SDN controller role are highlighted (orange ring). [51]

The first key concept of IDLB is to use the **geographical information** of the nodes on ground to approximate the position of a destination in the network topology, similar to geographical routing schemes [39, 66–68]. In general, the LEO SCN can be described by a fast moving in-space mesh network connected by ISLs, and slow moving or static nodes on ground connected by ESLs. By routing towards a geographical location, no global tracking of ESL handovers is required, except at the seams depending on the routing approach. Only near the destination is a concrete mapping of which satellite is connected to the destination terminal on ground necessary. Therefore, routing based on geographical information can significantly reduce the signaling overhead [39, 67]. To facilitate this strategy, a geographical address resolution scheme is used. Destination addresses are aggregated based on their geographical position.

The second concept is **distributed network control**. The routing scheme has to be able to react and adapt to network changes in a timely manner. Due to the physical size of the networks, end-to-end propagation delays over multiple hops of more than 100 ms are common. Outsourcing all network control to a single control center on ground is thus expected to negatively impact the QoS compliance. Even with multiple control entities on ground, the overall reactivity may be limited. So, to comply with delay-sensitive updates, the network is thus sub-divided into smaller independent domains. This sub-division of the in-space ISL network into clusters is shown in Figure 3.1. Each rectangular 12-node domain is colored accordingly. In this investigation, a flat in-orbit hierarchy of clusters is considered. However, the protocol can be used in multi-level SDN hierarchies, e.g., with a master SDN controller on ground.

Network information is aggregated at the dedicated SDN controller node of each cluster, enabling proactive and load-aware routing decisions. In Figure 3.1 and Figure 3.2, the satel-

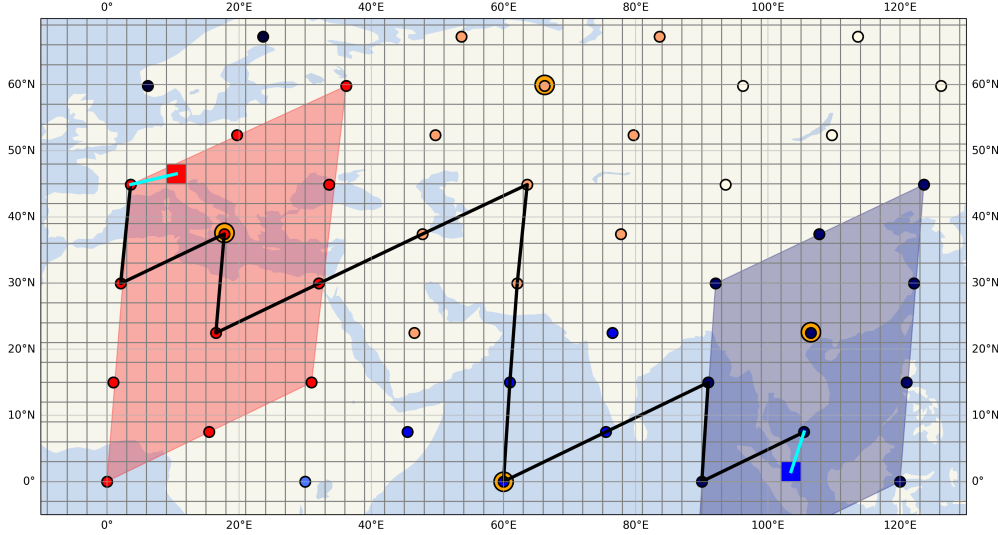


Figure 3.2.: Exemplary path from Munich to Singapore across several clusters in SCN-288. From geographical source cell (red rectangle) to destination cell (blue rectangle). Forwarded via ISLs (black lines) and ingress/egress ESLs (cyan lines). [51]

lites acting as controllers are highlighted by orange rings. This **autonomous, space-borne decision-making** enables improved reactivity, routing convergence and reduced signaling overhead. The size of the cluster determines the amount of signaling and propagation delays within the domain. Distributed approaches also inherently scale well, as additional clusters can be added to an existing setup. From a practical perspective, this space-borne autonomy is meaningful as well: presence on ground does not necessarily have to scale with the size of constellations.

To mitigate congestion, particularly for broadband traffic, the protocol is designed to enable effective **load-balancing** strategies. The design facilitates the desired maximization of throughput by efficiently using available network resources. Moreover, distributing traffic on diverse paths does not necessarily result in significantly increased latency. Because of the inherent grid structure of the network, there are often multiple paths with similar characteristics [8, 35]. As relevant information is aggregated at the control unit, proactive load-aware routing decisions are made and then forwarded to the other nodes of the cluster. In fact, a QoS-based distribution of the traffic can improve the latency of delay-critical classes: the shortest paths are not blocked by established flows with lower priority. Therefore, a proactive low-complexity intra-cluster routing algorithm is designed which is adjustable to different QoS classes.

The key design aspects that enable the approach are outlined in Table 3.1 and include:

- **Architectural details:** cluster arrangement, geographical tiling, handover events, routing table setup, and suitable multicast extension.
- **Geographical routing insights:** introduces geographical forwarding and the resulting reduction in signaling overhead. The approach is a direct precursor to the inter-cluster routing of IDLB.

3. Distributed Load-balanced Software-Defined Networking

- **Set-based source-routing insights:** introduces essential source-routing benchmark and quantifies network throughput characteristics as well as route diversity. Insights provide baseline for design of intra-cluster routing of IDLB.

Table 3.1.: Summary of IDLB protocol design components and features.

Category	Design aspect	Key characteristics
Architecture (Section 3.1.1)	Cluster arrangement	Size; shape
	Geographical tiling	Size; shape
	Handover events	Coherent cluster-specific adjustments
	Routing table setup	Two-tiered routing tables
	Multicast extension	SDN-enabled multicast solution
Inter-cluster	Geographical routing insights (Section 3.1.2)	Geographical forwarding; reduced signaling; applied in Section 3.2.2
Intra-cluster	Set-based source routing insights (Section 3.1.3)	Proactive load-balancing via path diversity; applied in Section 3.2.1

3.1.1. Protocol Architecture Details

Building upon the introduced reference scenario, this section provides an overview of essential design aspects of IDLB. Specifically, the considered scaling and arrangement of geographical areas and in-space clusters are presented. Moreover, the handling of handover events and the intended routing table setup are discussed, which represent key elements of the protocol. Finally, a potential multicast extension is proposed, highlighting the capacity for further developments and extensions.

Cluster Arrangement

In general, clusters should be larger than the geographical tiling on ground in order to reduce ambiguities. However, the specific cluster size represents a key design trade-off, which can be tailored to the system requirements. For intra-cluster routing, a larger scope is expected to enable more efficient load-balancing. However, the principal drawback is that latency between the SDN controller and its forwarding devices increases, which negatively impacts reactivity. Signaling overhead is increased as well. To find appealing trade-offs, the upper limit of the cluster size should be determined by the maximal permissible intra-cluster latency. This duration determines the routing convergence within a cluster, and thus the reactivity of the protocol.

In addition to the sizing of clusters, it is also important to consider their shape. Any viable arrangement of domains of adjacent satellite nodes is supported by IDLB, including differently sized clusters. The proposed protocol can thus enable specific cluster designs to meet certain criteria, if so required. For faster convergence, designs need to consider reducing the average number of intra-cluster hops. Accordingly, the cluster arrangement should follow the structure of the ISL topology. In the context of the Walker star reference constellations, this corresponds to diamond-shaped clusters. This shape follows the North, South, East, West directions of the ISLs.

For the reference scenarios, rectangular cluster shapes are considered as shown in Figure 3.1 and Figure 3.2. Rectangular clusters are used in order to enable coherent cluster-based statistics later on in the investigation. While diamond-shaped cluster arrangements improve routing convergence, they result in irregular, non-uniform clusters due to the seams of the reference systems. In the figures, a 4×3 arrangement is shown, which results in 12 nodes per cluster and 24 clusters overall. Moreover, 6×4 (24 nodes per cluster) and 8×6 (48 nodes per cluster) arrangements are investigated to evaluate the described sizing trade-off. For SCN-1440, clusters of a similar physical size are constructed to maintain comparable system characteristics. In total, 30 rectangular clusters of identical shape of size 6×8 , i.e. 48 satellites, are considered.

Geographic Tiling

The geographical tiling determines the number of logical geographic areas. The length of the geographical identifier in the header of a packet has to be large enough to support this number. A high number of areas is generally preferable, as it decreases the number of potential serving satellites. Local ambiguities can be resolved more easily. However, more areas also result in longer routing tables for geographical identifiers, which in turn increases the average required lookup time. This can be mitigated by employing tree-based aggregation to generate geographical identifiers, such as radix-trees or geocast [64,65]. Given that table lookups typically occur within microseconds, the impact on the overall latency is relatively small.

As with the cluster shapes, the geographical tiling can also be tailored to specific scenarios. The handover procedures and geographical table setup of IDLB support any meaningful partitioning into logical areas. Naturally, the tiling should comply with the coverage characteristics of the considered constellation. Moreover, the geographical identifier has to be long enough to correctly represent each area. A generic approach is to use equally sized areas.

Another option is to use equirectangular areas of a fixed size. The coordinates of each geographical cell are defined by regular latitude and longitude intervals, providing a straightforward mapping between geographic positions and cell indices. So, areas of n° latitude and m° longitude can be chosen. Near the poles, it makes sense to use single unified cells starting at $\pm(90^\circ - n^\circ)$. However, this results in areas whose physical size varies: the cells are larger at equator, and smaller near the poles.

For the reference scenario, such equirectangular areas of 3° latitude and 3° longitude until $\pm 87^\circ$ are used. This tiling is illustrated in Figure 3.2 by the gray grid. It results in 6962 areas. To represent this number of addresses, at least 13 bit are required for the geographical identifier.

Handover Events

To guarantee the correctness and stability of a routing solution in an SCN context, the handling of handover events is crucial. There are frequent ISL and ESL handover events which should result in new coherent path choices. For both ISLs and ESLs, we assume handovers to be triggered by the relative (elevation) angle. Typically, ISL handovers are

predictable and can thus be prepared accordingly. For the considered polar constellations, ISLs are switched when a satellite enters or leaves the inter-plane shutdown latitudes. The resulting changes in the network topology are precomputed by the routing solution. Updates can be timed with handovers, so that all relevant routing tables are adjusted accordingly. To provide enough time for the remaining packets which are still forwarded to the node to arrive at the destination, an inter-plane shutdown countdown flag is used. This timing can be based on the maximum expected queuing and intra-cluster signaling delays.

ESL handover events can be more unpredictable or ambiguous, depending on the utilized handover scheme. For the proposed distributed space network, handovers between clusters may introduce ambiguity. Due to a handover, it may be possible that a packet arrives in the apparent destination cluster, but cannot reach the egress satellite - deadlocks may occur. To resolve this, cell-based handovers are used: all UTs of a geographical cell are handed over to a new cluster at a defined moment in time. So, the geographical routing tables within the involved clusters can be adjusted accordingly simultaneously. A locally unambiguous mapping between clusters is established.

As multiple quasi-simultaneous handover events are considered by the terminals of a geographical cell, there are potential collisions without coordination. However, based on the scaling of the geographical cells and the density of available satellites, this caused no issues during the investigations.

At the seams, geographical information can be particularly ambiguous. There may be uncertainty about which side of the seam is connected to a terminal on ground. Using planned, timed handover events known by all control entities, this ambiguity can be resolved. This strategy is used in all simulations. Alternatively, previous literature also proposed the inclusion of orbital identifiers in addition to the geographical identifier [68].

A handover strategy for the ESLs based on satellite proximity and visibility is considered. When a satellite descends below the predefined minimum elevation angle threshold, it is no longer considered visible to the ground station. At this point, a handover to a different nearby satellite is initiated. It is not assumed that routing protocols trigger ESL handovers.

Routing Table Setup

To accommodate the proposed two-level routing hierarchy, IDLB leverages a two-tiered routing table system. While a hierarchical SDN controller architecture presents an alternative, it deviates from the proposed decentralized cluster setup. Network virtualization and overlay networks, though viable, introduce complexities that outweigh the benefits of two-tiered routing tables in this context. Notably, the SDN foundation of the proposed protocol allows for flexible and granular manipulation of routing tables, enhancing its adaptability and efficiency.

On the one hand, a so-called "geographical routing table" is proposed which maps each geographical identifier (for each cell), to a destination node (a satellite). The geographical cells of this table should be constant. The table is used to identify whether a cell is served by the current cluster or not. In addition, it indicates over which link the cluster is to be exited. The inter-cluster routing utilizes a dedicated function to compute viable Next-Hop Cluster Nodes (NHCNs). These external border nodes, which connect to the given cluster

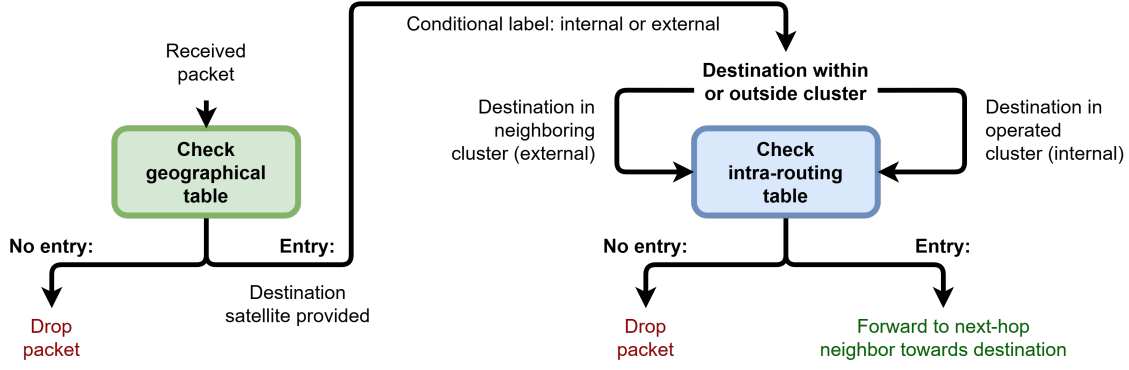


Figure 3.3.: Conceptual flowchart of packet handling: two-tiered routing table lookup. A geographical and an intra-routing table are used.

but are not a part of it, are the destination nodes listed in the geographical routing table. If the packet is in the apparently correct cluster, this is indicated by a special entry. There is no global mapping of every terminal and its serving satellite.

Only local resolutions are considered to maintain minimal signaling overhead. For these mappings, cluster-specific internal routing tables are used which relate destinations to next hops. MAC addresses of GWs and UTs served by the cluster, of satellites within the cluster, and of NHCNs are mapped to suitable next hops. As this mapping only contains relevant destinations within the operational scope of a cluster, a practically manageable size is typically achieved.

An abstracted flowchart of the two-tiered routing table lookup is shown in Figure 3.3. As shown, after the geographical table, packets are either destined for a satellite within the cluster or an NHCN, so a satellite from a neighboring cluster. In both cases, a subsequent intra-routing table lookup is required to find the correct next hop.

To maintain a coherent ESL overview within the clusters, each handover triggers a cluster-internal broadcast of related routing table updates. In order to avoid ambiguities, cell-based handovers between clusters are used and the geographical tables of the involved clusters are adjusted accordingly. This design mitigates false positives and false negatives. As a result, packets are correctly routed within the cluster or forwarded to the correct neighboring cluster.

Multicast Extension

Due to the special requirements of SCNs, many traditional terrestrial multicast protocols are not well-suited. A key problem is the spanning of efficient multicast trees in large-scale dynamic networks. As a light-weight extension, a design based on the Protocol Independent Multicast (PIM) protocol family is proposed, which uses existing unicast routes for multicasting.

As the relevant connectivity changes are considered highly predictable, a tailored PIM Sparse Mode (PIM-SM) [114] approach with timed switch-overs according to expected handovers is

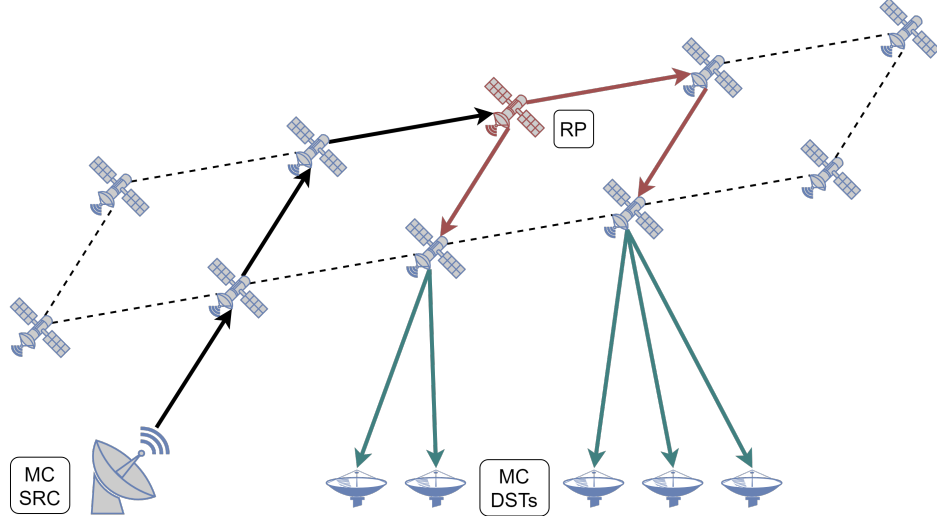


Figure 3.4.: Schematic representation of considered multicast extension. The black arrows highlight how multicast traffic is forwarded from its source (a GW) to the Rendezvous Point (RP) (a cluster controller). The red arrows show the traffic split at the RP towards egress satellites. The green arrows represent downlink ESLs to multicast destinations on ground (UTs).

considered.

PIM-SM is based on RPs, which are the roots of distribution trees for multicast groups [114]. In general, multicast traffic is forwarded from the source, the first-hop router, to the RP. If a receiving entity is interested in a multicasting service, it has to request the multicast service from the RP. To this end, the last-hop router sends an Internet Group Management Protocol (IGMP) request to the closest RP.

Based on the proposed distributed SDN architecture, designating the SDN controllers as RPs is practical. Multicast traffic is forwarded to the SDN controllers which act as RPs for their respective clusters. With this setup there is also no need for PIM-SM bootstrapping procedures. Since the controllers distribute forwarding instructions with each network update, additional signaling required for multicasting is minimal. Given the nature of multicast traffic, it is assumed to be widely available or cached. Thus, the GW which is closest to the SDN controller (the RP) of the cluster is expected to be the primary source of such traffic.

The considered multicast extension works as follows:

- UT requests to subscribe to multicast group
- Multicast session established between closest GW and interested UT
- If RP does not yet receive MC traffic: GW triggered to start transmission to RP
- If GW already sending MC traffic to RP: forwarding from RP to UT
- Forwarding from RP to UT follows unicast routing entries

With this setup, all RPs are relatively close to the receiving entities as well as the sources. Traffic on the connections between GWs and SDN controllers is minimized, there is a multicast gain on the trunk links. There are potential multicast gains via a duplication of packets at the SDN controller and at the egress satellite. A diagram of the intended functionality is shown in Figure 3.4.

In summary, the proposed multicast solution integrates well into the presented IDLB architecture. However, as the principal focus of this investigation is on unicast traffic, the multicast capabilities enabled by this potential extension fall outside the current scope. Nevertheless, it represents an intriguing research direction in which the protocol can be further developed. Since the PIM protocol family re-uses unicast table entries, it relies on the flow distribution of the unicast routing algorithms. The insights and principles on which these routing schemes have been formulated are presented in the following sections.

3.1.2. Geographical Routing Insights

Geographical routing offers a promising solution for reducing signaling overhead, which makes it particularly relevant to the inter-cluster routing scheme of IDLB. In this context, minimizing control plane traffic over multiple clusters is essential to limit overhead. Notably, IDLB uses geographical identifiers embedded in Media Access Control (MAC) addresses in its forwarding and routing tables. This setup is directly derived from a previously proposed geographical routing scheme [39]. Thus, the key contributions and insights which have shaped the design of the inter-cluster routing scheme of IDLB are presented.

In general, geographical routing is well-suited for LEO SCNs, as most terminals on ground are relatively static. Even if they move, their relative speed is low in comparison with the satellites. Thus, packets are simply forwarded to a satellite which is currently covering the destination area. To this end, the complete surface of the Earth is split into a grid, which is not necessarily homogeneous. Each cell in the grid has a unique area identifier, which is used to forward packets to the destination area. Due to overlapping coverage, there may be ambiguity in which satellite is serving the destination terminal on ground. Moreover, node mobility poses an issue as the area identifier has to change. This has to be signaled to all nodes to maintain a coherent geographical mapping. The analysis is based on previous investigations of these limitations [67, 68]. The corresponding scheme has been published in the literature [39].

Geographical Forwarding

The investigated scheme proposes key adjustments to improve geographical forwarding:

- Embedding of geographical identifier in MAC address
- Local address resolution approach to mitigate ambiguities in the last hop
- Local re-routing mechanisms to avoid saturated links

In contrast to conventional geographical routing schemes, a decoupling of topological information from the public identity of the terminal, i.e., its IP address, is proposed. This

3. Distributed Load-balanced Software-Defined Networking

implies that the MAC addresses of terminals are mutable; the embedded geographical identifier changes when moving to different cells of the grid. Moreover, the decoupling implies a dedicated address resolution scheme.

As the geographic location can be considered private information, the proposed decoupling inherently increases the privacy of user terminals. In comparison with schemes that embed the area identifier in the IP addresses, this information is not publicly available. The MAC addresses are only used within the SCN and the terminals on ground.

By periodically computing current next hops for geographical destinations, updated switching tables are generated. For the proposed geographical routing scheme, the mechanism for table generation and updating is shown in Algorithm 1.

From the perspective of a terminal on ground, packets traverse a distributed switch until arrival in the destination area. To find the individual next hops, the great-circle distance (dist_c) based on the coordinates of the sub-satellite points ($\text{lat}_1, \text{lon}_1$, and $\text{lat}_2, \text{lon}_2$) can be used, which corresponds to the shortest path on the surface of a sphere:

$$\text{dist}_c = r_{\text{Earth}} \cdot \arccos(\sin(\text{lat}_1) \cdot \sin(\text{lat}_2) + \cos(\text{lat}_1) \cdot \cos(\text{lat}_2) \cdot \cos(\text{lon}_1 - \text{lon}_2)) \quad (3.1)$$

Alternatively, the Euclidean distance (dist_e) of every hop (straight line) to the destination can be used (using x, y, z coordinates of sub-satellite points). Given the relative distances in LEO systems with respect to the radius of the Earth, the resulting shortest path should be identical:

$$\text{dist}_e = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2} \quad (3.2)$$

As seams pose a problem to this heuristic, the pathfinding algorithm has to consider orbital direction to steer traffic accordingly if necessary (as in [68]).

In order to minimize the dropping of packets, a rerouting mechanism based on thresholds is used. Surrounding nodes, which are informed of saturated links, then regenerate their switching tables accordingly. The previous hop cannot be a valid option as a new next hop, since deadlocks may be introduced.

However, the approach only slightly outperforms source-routing without link load information. While Algorithm 1 considers an overload threshold, the re-routing mechanism is limited. It cannot introduce loops or deadlocks. Moreover, to avoid route flapping, alternative next hops must not be computed for all destination areas using the overloaded link. Only simple strategies, such as round-robin, are feasible. Proactive flow adjustments cannot be made efficiently using this decentralized approach. Due to the limited scope and re-routing options available, no informed load-balancing decision-making is possible. Thus, geographical routing schemes tend to create hot spots similar to SPF approaches [39]. Nevertheless, as the approach significantly reduces signaling costs, it provides important insights for an efficient design.

Signaling Cost

In principle, the minimization of signaling overhead is a core advantage of this approach. As signaling is introduced by the routing as well as the address resolution, both aspects are considered.

Algorithm 1 Geographical routing: table generation [39]**Require:** satellite positions, geographical areas, overload information

```

1: function GENERATE_GEO_TABLE(node, geographic_cells, geographic_table)
2:   for each cell area  $\in$  geographic_cells do
3:     dist_to_area  $\leftarrow$  distance(node.location, area)
4:     orbital_dir  $\leftarrow$  orbital_direction(node.location, area)
5:     best_value  $\leftarrow$  max_float
6:     best_hop  $\leftarrow$  null
7:     for each neighbor ngr  $\in$  node.neighbors do
8:       value  $\leftarrow$  cost(node, dist_to_area, orbital_dir, overload(node, ngr))
9:       if value < best_value then
10:        best_value  $\leftarrow$  value
11:        best_hop  $\leftarrow$  ngr
12:       end if
13:     end for
14:     geographic_table.insert_or_assign(area, best_hop)
15:   end for
16: end function

```

As a well-suited address resolution procedure, an initial broadcast of related information is assumed. The mapping of IP address and corresponding MAC addresses, which include the geographical area, is thus broadcast proactively. As each node is in possession of all relevant information, an effectively immediate address resolution is provided. Updates are only required in three cases:

- When a UT joins or leaves the network
- When a UT changes its IP address
- When a UT moves to a new area, changing its area identifier in the MAC address

Alternatively, it is possible to request an address resolution when a connection is initiated. This can be realized by flooding the network or consulting a close-by address resolution directory. In either case, an initial setup delay is caused. Given the physical size of SCNs and their dynamic nature, such approaches are considered inefficient. To underline this issue, the signaling cost for a scheme using a localized registration entity, e.g. Mobile IP [115], is formulated.

The management cost, representing the signaling overhead, can be calculated as the product of the message size of the update and the number of hops, N_{hops} , which are required to reach the destination [116]. Using $\zeta(\rho)$ to denote the size of packet ρ , the cost can be described by [68]:

$$C = \zeta(\rho) \cdot N_{\text{hops}} \quad (3.3)$$

For the sake of simplicity, all update messages of all schemes are assumed to be of the same size ζ . The costs of required control updates are neglected in this analysis. $\Lambda_{t,h}$ is considered to be the time-dependent rate of inter-node handovers. The update as well as a response are sent. For approaches using a localized registration directory, an update has to be sent there

3. Distributed Load-balanced Software-Defined Networking

for every handover event. The number of hops required to reach the directory is denoted by $N_{\text{hops, dir}}$. Thus, the cost at time t for these schemes can be estimated by the following formula [68]:

$$C_{t,\text{dir}} = 2 \cdot \zeta \cdot N_{\text{hops, dir}} \cdot \Lambda_{t,h} \quad (3.4)$$

Signaling every handover to the registration directory is costly. Thus, decreasing the number of hops for such events is a key incentive. If all handovers are predictable, it is possible to forward based on an assumed mapping. However, this relies on strong assumptions and is not resilient to any dynamic adjustments or failures. Packets may be forwarded based on wrong ESL connectivity assumptions and may not reach their destination. So, in order to minimize signaling, local resolutions as used by geographical routing are advantageous.

In the proposed geographical scheme, signaling occurs according to the previously defined criteria. The resulting overall cost is split into an address resolution cost, $C_{t,\text{ar}}$, and a routing cost, $C_{t,\text{geo}}$:

$$C_{t,\text{overall}} = C_{t,\text{ar}} + C_{t,\text{geo}} \quad (3.5)$$

To avoid ambiguities in the last hop, a local resolution is required. Moreover, updates are required for cell crossing events by mobile terminals. The corresponding update frequency is denoted by $\Lambda_{t,c}$. The rate of terminals entering or leaving the network, $\Lambda_{t,n}$, is also considered. The resulting number of hops due to a network-wide broadcasting of messages is denoted by $N_{\text{hops, broad}}$. Orbital information is used to mitigate additional hops caused by seams, where no ISLs are established between counter-rotating planes. The rate of such changes in the direction of the orbit of the corresponding satellite, after an inter-node handover, is denoted as $\Lambda_{t,o}$. This results in a cost at time t of:

$$C_{t,\text{ar}} = \zeta \cdot N_{\text{hops, broad}} \cdot (\Lambda_{t,n} + \Lambda_{t,c} + \Lambda_{t,o}) \quad (3.6)$$

Depending on the applied local routing scheme, the resulting overhead varies. In the simulations, a local broadcasting of inter-node handover events was used. The number of hops of these updates is denoted as $N_{\text{hops, local}}$. The proposed adjusted geographical routing scheme [39] also introduces a re-routing procedure to reduce dropping rates. To this end, local link overload updates are exchanged. $N_{\text{hops, load}}$ and $\Lambda_{t,l}$ describe the number of hops of such updates and their rate respectively. Signaling due to node failures is omitted, because of their rarity and unpredictability. The cost of the signaling at time t by routing mechanisms becomes:

$$C_{t,\text{geo}} = \zeta \cdot (N_{\text{hops, local}} \cdot \Lambda_{t,h} + N_{\text{hops, load}} \cdot \Lambda_{t,l}) \quad (3.7)$$

As the scheme is decentralized, link load information is only shared between neighbors. So, $N_{\text{hops, load}}$ is equal to one. Similarly, depending on the LEO SCN, $N_{\text{hops, local}}$ is also only relevant in a neighborhood of typically three hops. Hence, for the reference system, the actual cost reduces to:

$$C'_{t,\text{geo}} = \zeta \cdot (3 \cdot \Lambda_{t,h} + \Lambda_{t,l}) \quad (3.8)$$

Therefore, the overall signaling cost at time t of the proposed scheme using the previously described address resolution mechanism is:

$$C_{t,\text{overall}} = \zeta \cdot (N_{\text{hops, broad}} \cdot (\Lambda_{t,n} + \Lambda_{t,c} + \Lambda_{t,o}) + (3 \cdot \Lambda_{t,h} + \Lambda_{t,l})) \quad (3.9)$$

The reduction of effective hops $N_{\text{hops, local}}$ for handover-related updates is key as $\Lambda_{t,h}$ is comparatively high. The proposed approach achieves this goal, efficiently resolving addresses based on geographical information. In the context of IDLB, this advantage can be re-used in the context of inter-cluster routing. By basing path selection on a cluster level on geographical information, signaling costs can be reduced in a similar fashion. So, the inter-cluster cost $C_{t,\text{inter}}$ is minimized. The management cost within each cluster, $C_{t,\text{intra}}$, has to be considered as well. Using M_{clusters} individual clusters, the signaling of the proposed protocol is described by:

$$C_{t,\text{IDLB}} = C_{t,\text{ar}} + (C_{t,\text{intra}} + C_{t,\text{inter}}) \cdot M_{\text{clusters}} \quad (3.10)$$

The signaling overhead introduced by intra-cluster routing mainly consists of node updates including link loads. This internal update rate is denoted by $\Lambda_{t,lc}$. The rate is significantly higher than the previously used $\Lambda_{t,l}$, as information is also exchanged even when no link is overloaded. However, the resulting overhead is balanced by the relatively small number of hops required to reach the control unit. $N_{\text{hops, ctrl}}$ is used to describe the (average) number of hops to reach the controller. Given the considered in-space architecture, the cluster size $M_{\text{in-cluster}}$ determines this value. The exchange between a forwarding device and the controller potentially results in all routing tables having to be adjusted. This is the case for handover events, but may also be triggered by a link load update. So, it is signaled to all forwarding devices of the cluster except the controller. However, only handover events relevant to the cluster have to be considered. Therefore, the resulting rate is $\Lambda_{t,hc}$. As the exchange is bi-directional, a factor of two is used. In general, the intra-cluster signaling overhead can thus be described by:

$$C_{t,\text{intra}} = \zeta \cdot 2 \cdot N_{\text{hops, ctrl}} \cdot (\Lambda_{t,hc} + \Lambda_{t,lc}) \cdot (M_{\text{in-cluster}} - 1) \quad (3.11)$$

Similar to geographical routing, inter-cluster information exchange is very limited in IDLB to reduce signaling overhead. Forwarding is based on proximity to the geographical area. Only for potential ambiguities, for instance after cell handovers between clusters, signaling is required. The rate of these events is described by $\Lambda_{t,i}$. The distance between cluster controllers is denoted by $N_{\text{hops, inter}}$. In the worst case, all clusters considered neighbors, $M_{\text{nbr-clusters}}$, have to be informed:

$$C_{t,\text{inter}} = \zeta \cdot 2 \cdot N_{\text{hops, inter}} \cdot \Lambda_{t,i} \cdot M_{\text{nbr-clusters}} \quad (3.12)$$

The resulting signaling costs are considered low relative to alternative options and the network control which is enabled. Based on the localization of dynamic handover information, IDLB is expected to cause reduced overhead. Advantageous signaling strategies from geographical routing have been used to facilitate inter-cluster information exchange. A quantitative evaluation of the resulting signaling in terms of control plane overhead is presented in Section 3.3.2.

The presented considerations concerning geographical schemes focus on inter-cluster routing. For the routing within the clusters, different approaches have shaped the design. As SDN relies on consistent information exchange, signaling overhead in clusters is less critical. Instead, using the available path diversity to enable flexible load-balancing is important. So, for intra-cluster routing, insights have been gained in the context of set-based source-routed approaches.

3.1.3. Set-based Source-Routing Insights

Set-based source-routing is an important precursor to the design of the intra-cluster routing scheme of IDLB. The gained insights provide a valuable baseline for the concrete design of the proposed routing algorithms. As IDLB is designed to enable both proactive as well as reactive intra-cluster load-balancing, it is important to quantify path diversity gains first.

Source-routing is an essential benchmark which provides an understanding of the throughput characteristics of the reference scenario. Load-agnostic source-routing is thus used to quantify how the UT distribution impacts congestion. Moreover, the set-based extension demonstrates the inherent path diversity in SCNs. To this end, the increase in throughput achieved by proactive route distribution is quantified.

Source-routing presents a compelling routing strategy that capitalizes on its inherent flexibility, adaptability, and local decision-making. In contrast to geographical routing, the approaches are based on the idea of computing end-to-end paths at the source. The resulting path is then included in the packet headers and read on every hop.

However, different strategies can be used for pathfinding, which have the potential to be less susceptible to bottlenecks. A common approach to load balancing is the preemptive distribution of routing requests on a set of viable paths [117]. In the context of SCNs, applying this strategy is particularly interesting, as the network structure typically results in multiple paths of similar characteristics [8]. Therefore, in this section, the viability of proactive load balancing using set-based source-routing is evaluated and quantified. The research presented in this section has been published in similar form in the literature [35].

As LEO constellations are characterized by dynamic and rapidly changing topologies, source-routing enables efficient route computation and selection by the source node (in the reference scenarios the ingress satellite) based on the current state of the network (a snapshot). Since the acquisition of all load information results in massive signaling overhead, it is assumed that no link load information is present at the ingress node. The approach is link load-agnostic, the resulting path distribution differs from active load-balancing. However, since the orbits of satellites are highly predictable, ISL connectivity and propagation delays can be readily computed at the source node, even in advance. An SPF algorithm can then be utilized to find a path between the ingress and assumed egress satellites. If geographical addressing is used, and ESL handovers are predictable, the ingress node can compute which node should currently serve the destination. These computations can be done periodically according to a timed handover strategy. An MPLS-like approach is assumed: the calculated path is added to the header of the packet and popped at every hop. The approach is generally considered viable, MPLS-based schemes are employed in real-world satellite networking solutions [6].

Proactive Route Distribution

For source-routing, the path selection algorithm is crucial to make use of the inherent path diversity of an SCN. By randomly choosing a path from the set of viable paths between the ingress and egress node, $\mathcal{P}_{\text{viable}}$, the load is intrinsically more balanced. This random path selection is flow-based in this analysis, to decrease jitter and the potential for packet

reordering. Therefore, the probability of selecting a path p is described by:

$$P(\text{selecting } p) = \frac{1}{|\mathcal{P}_{\text{viable}}|} \quad (3.13)$$

Due to the grid structure of the mesh network, the set of minimum hop paths contains the fastest path in terms of propagation delay. For the sake of simplicity, this investigation focuses on the minimum hop metric. Nevertheless, it is possible to employ the propagation delay for path selection without significant processing overhead. The set of paths can be computed in a similar fashion based on the fastest path and an additional time threshold which maintains QoS compliance. Such a propagation delay-based approach may provide slightly improved performance compared to the minimum hop approach. Nevertheless, in tests, the resulting path sets were mostly identical.

Most importantly, it is possible to relax the minimum hop constraint (or propagation delay constraint), for instance by allowing paths with two or more additional hops. If the resulting end-to-end latency is compliant with the requirements of the given QoS class, this relaxation can enable more intrinsic load balancing. Paths can be ranked by assigning a weight W , based on delay or other QoS metrics. However, as this implicitly favors certain paths, it reduces path diversity. The probability of choosing path p_i is then:

$$P(\text{selecting } p_i) = \frac{W_i}{\sum_{j=1}^{|\mathcal{P}_{\text{viable}}|} W_j} \quad (3.14)$$

For the SCN-288 constellation, the intra-plane propagation delay is approximately 6 ms, and the maximum inter-plane propagation delay is approximately 8 ms. So, the maximum additional delay can be coarsely estimated based on the constellation design. In a related investigation [118], the cost of additional hops and the relation to propagation delay is analyzed more thoroughly in a more flexible setting, including adjustable ISLs between more distant satellites.

It is important to note that computing the set of viable paths can be computationally complex in many scenarios. Depending on the applied limiting heuristic, e.g. the number of hops, the number of potential paths may be high in a large network. Let us assume the SCN is approximated by a grid network of dimensions $M_{\text{x-dim}} \times M_{\text{y-dim}}$. The number of nodes $|\mathcal{V}|$ is equal to the product of these dimensions. $|\mathcal{E}|$ is directly proportional to $M_{\text{x-dim}} \times M_{\text{y-dim}}$ as well. When considering a standard Breadth-First Search (BFS) approach, all nodes have to be explored in the worst case, for instance, when the destination node is in the furthest corner from the source. Consequently, the resulting complexity in time is:

$$\mathcal{TC}_{\text{set-based}} = O(|\mathcal{E}| + |\mathcal{V}|) = O(M_{\text{x-dim}} \cdot M_{\text{y-dim}}) \quad (3.15)$$

On the other hand, it is also possible to limit the number of candidate paths directly. For instance, the k -shortest-paths algorithm can be applied which results in [119, 120]:

$$\mathcal{TC}_{\text{k-paths}} = O(|\mathcal{E}| + |\mathcal{V}| \log |\mathcal{V}| + k) \quad (3.16)$$

However, such an approach significantly decreases the number of considered paths. The set of k candidate paths, $\mathcal{P}_k$, is typically only a sub-set of the set of viable paths:

$$|\mathcal{P}_k| \leq |\mathcal{P}_{\text{viable}}| \quad \text{since} \quad \mathcal{P}_k \subseteq \mathcal{P}_{\text{viable}} \quad (3.17)$$

3. Distributed Load-balanced Software-Defined Networking

More importantly, complexity in space can become a problem, depending on how paths are stored. In the worst case, the algorithm branches into N_{paths} different paths on every hop. For instance, using BFS, all unvisited neighbors of the current node represent potential next hops of new paths. Storing every single potential path results in an exponential growth. For a grid network of dimensions $M_{x\text{-dim}} \times M_{y\text{-dim}}$, the complexity in space becomes:

$$\mathcal{SC}_{\text{set-based}} = O(M_{x\text{-dim}} \cdot M_{y\text{-dim}} \cdot (N_{\text{hops}})^{N_{\text{paths}}}) \quad (3.18)$$

Therefore, smaller sets of variable paths may have to be considered to maintain suitably low complexity. By applying heuristics, the paths can be found more efficiently. For instance, a bidirectional A* algorithm [121] can be used. However, due to the seams and the ISL inter-plane shutdown latitudes, a suitable distance metric can be difficult to formulate. In this analysis, utilizing a BFS path discovery was efficient enough to compute all paths with the minimum amount of hops on an off-the-shelf laptop in milliseconds for SCN-288. However, for larger constellations, a significant amount of memory was required due to the exponential growth in space.

Hence, for increasingly large constellations, the scalability of source-routing is a significant issue. The physical size of these networks also leads to notable delays, further limiting reactive load-balancing schemes. Thus, on their own, proactive source-routed approaches do not scale well. However, their design is of interest for the intra-domain routing of hierarchical approaches, such as the proposed distributed SDN protocol, IDLB.

Performance Improvements

To validate the assumption that SCN grid structures offer diverse paths with similar characteristics, a quantitative comparison was conducted. For realistic results, a system-level simulator which allows for packet-based analyses was used. A detailed description of the corresponding software and its configuration is provided in Section 3.3.1. The simulation parameters are summarized in Table 3.2. An overview of additional simulation parameters is provided in Appendix B.

Table 3.2.: Simulation parameters of set-based source routing analysis.

Simulation parameter	Value
Simulation duration [s]	720
Number of sessions	$[5 \cdot 10^3, 13 \cdot 10^3]$
Average session duration [s]	100
Session data rate [Mbps]	10
Session begin distribution	uniform
Session duration distribution	normal
Packet size [kByte]	1.5
Number of active UTs	2000
UT-to-GW traffic share (and vice versa)	60 %
UT-to-UT traffic share	40 %

To maintain coherence, the same system scenario is applied for all simulations in this chapter. The setup corresponds to the reference scenario using the SCN-288 constellation. Simulated traffic consists of individual sessions between randomly chosen terminals on ground with constant bit rate. A more detailed explanation of this traffic model is provided in Section 3.3.1. Although the source-routed approaches investigated are QoS-agnostic, the introduced QoS classes and their respective traffic shares are applied for completeness. However, the source-routed schemes do not differentiate between QoS profiles.

In a first evaluation, the tendency of source-routing to create bottlenecks is illustrated. To this end, the reference density-based UT distribution (explained in Section 2.1.2) and a uniform UT distribution are compared. It is important to note that the locations of GWs remain unchanged for both scenarios. So, the 60 % UT-to-GW (and vice versa) traffic share is likely to cause hot spots even with a uniform UT distribution. Nevertheless, the remaining UT-to-UT traffic should be more uniformly distributed in this scenario.

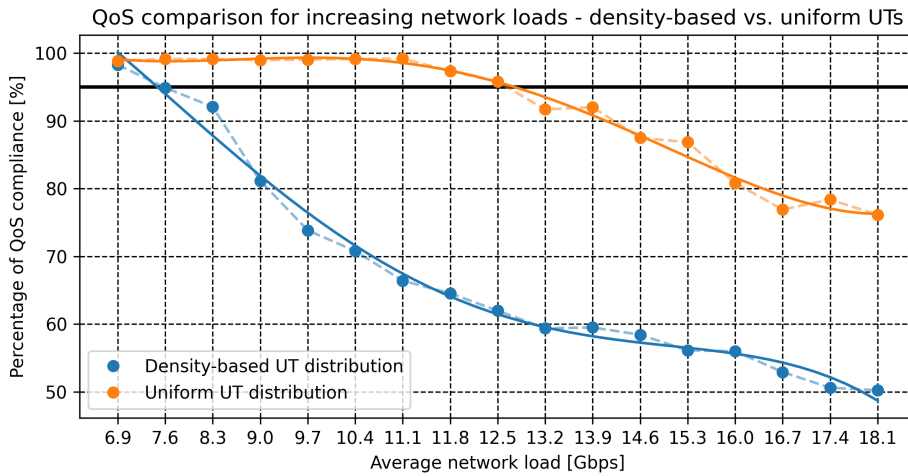


Figure 3.5.: QoS comparisons for increasing network loads for density-based UT distribution (in blue), and a uniform UT distribution (in orange). The latter derives from the reference scenario. SPF source routing used in both cases.

Figure 3.5 illustrates that the non-uniformly distributed traffic requirements cause the source-routed approach to be more susceptible to hot spots. Indeed, the sessions are non-compliant due to the increased packet dropping rates at higher network loads. A drop occurs if a packet cannot be queued in the corresponding output buffer of the satellite. This occurs either because the buffer is currently at maximum capacity or because the next hop defined in the header no longer exists due to a handover.

Using Little’s law [122], it is possible to estimate an average network load of 6.9 Gbps for 5500 sessions and 12.5 Gbps for 9000 sessions. These values correspond to the average network load measured during the simulation. As expected, the inherently more diverse routes for a uniform UT distribution enable higher loads. Approximately 64.5 % more load is supported with a QoS compliance of >95 %.

Figure 3.6 shows the resulting link loads based on the paths chosen for the density-based UT distribution. ISLs with higher link utilization appear thicker. While the traffic model

3. Distributed Load-balanced Software-Defined Networking

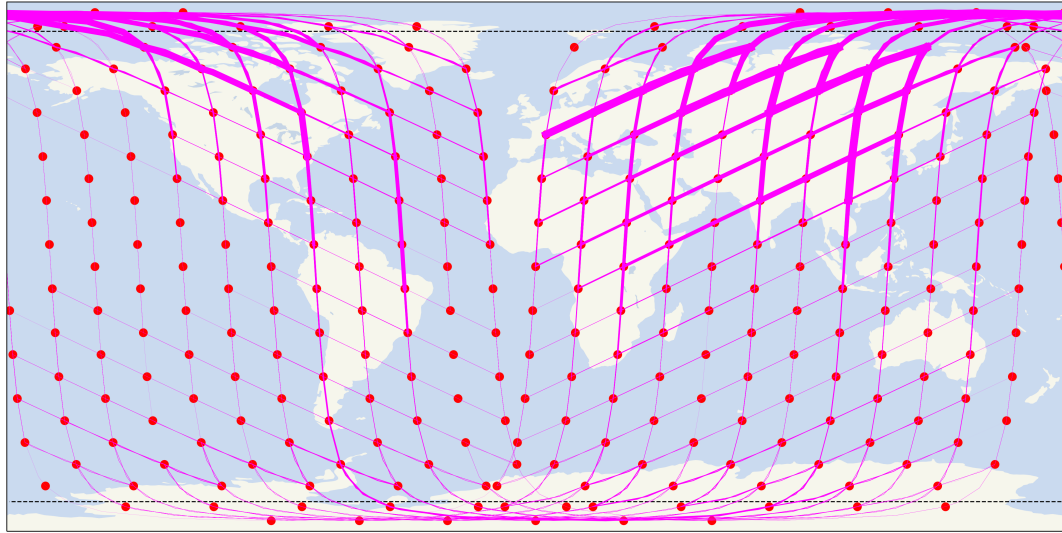


Figure 3.6.: Observed link utilization of ISLs of a snapshot using SPF source-routed scheme. Width of ISLs (magenta) scaled according to link utilization.

implies that certain paths are more commonly used than others, the resulting route diversity is of interest. The scheme does not avoid saturating links near hot spots, and disregards links with less favorable delay characteristics. For instance, longer paths across the Pacific Ocean are rare, due to the significant propagation delays.

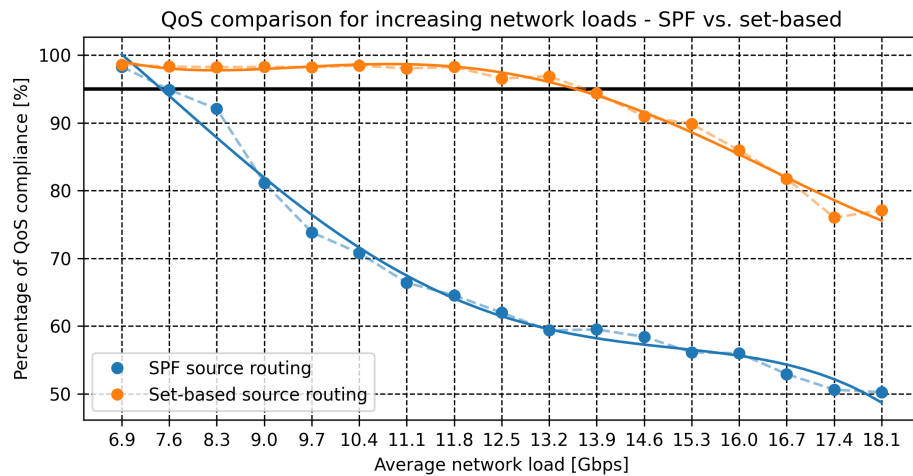


Figure 3.7.: QoS comparisons for increasing network loads for SPF source routing (in blue) and set-based source routing (in orange).

Additionally, the performance enhancements of using random paths with an equal number of hops, based on the minimum hop metric, are evaluated. The results of Figure 3.7 illustrate the performance improvements of the enhanced approach. By using a minimum hop metric instead of the propagation delay, and applying a random path selection per flow, the tendency for hot spots is reduced. A QoS compliance of $>95\%$ is still achieved for approximately

77.6 % more load. The inherent route diversity of the SCN is used more effectively, without considering load information at the source node at all.

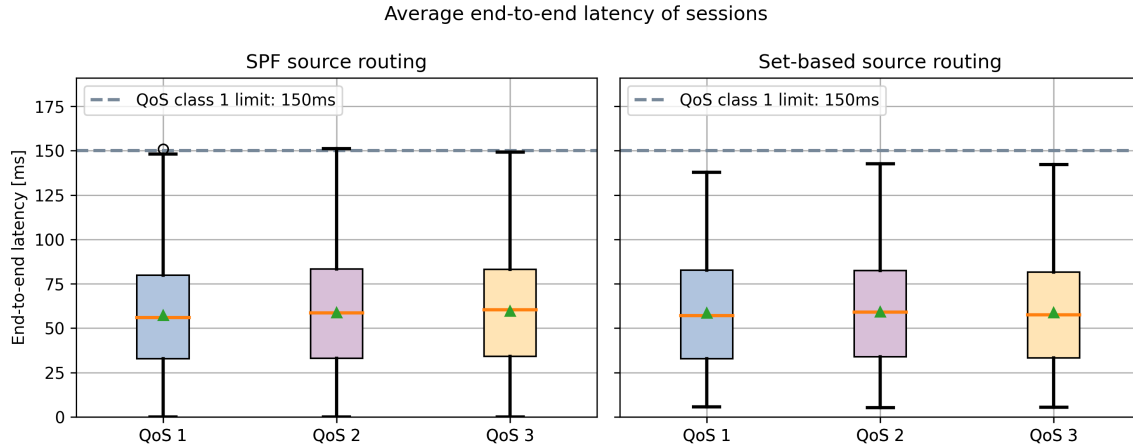


Figure 3.8.: End-to-end latency of sessions for SPF source routing (left plot) and a set-based source routing (right plot). Both schemes do not differentiate QoS classes. Comparing average (green triangle), median (orange line), minimum and maximum (caps).

Looking at the end-to-end latency depicted in Figure 3.8, both approaches perform similarly. The propagation delay-based SPF approach (on the left) can even cause longer delays than the set-based approach (on the right). Due to bottlenecks, multi-hop queuing delays can aggregate, resulting in longer end-to-end latency. For QoS class 3, which represents the bulk of the traffic, SPF source routing achieved on average 59.81 ms compared to the 60.74 ms of the set-based approach. The impact on latency of the more diverse path selection is minimal. Therefore, it should be the preferred option in most cases.

Drawbacks of source-routing were quantified as well in the investigation. A processing penalty was applied on each node, representing the effort to extract the next hop. However, its effect on the average end-to-end latency was not significant. A processing time of 50 μ s for each packet processing operation is assumed. So, even when longer routes consisting of 20 hops are considered, only an additional delay of 1 ms is observed. On the other hand, the amount of additional header operations increased linearly with the number of forwarded packets. At $3.5 \cdot 10^6$ packets, approximately $30.0 \cdot 10^6$ additional operations were required. So, this results in around 8.6 header label operations per packet.

Signaling messages and the resulting overhead are simulated as well. However, due to the assumed predictable handover strategies, no dedicated signaling is required for the source-routing approach. For decentralized approaches, unpredictable topology changes have to be broadcast to all nodes. This includes scenarios with mobile UTs or satellite failures. However, in all test cases, the relative share of signaling overhead was below 1 % of the overall traffic.

Overall, the investigation highlights that proactive route distribution by using a set of candidate routes is a promising strategy for SCNs. Even without knowledge of the link utilization within the constellation, significant improvements were achieved. In addition, the tendency

of the reference UT and GW distributions to create bottlenecks, unless actively mitigated, has been quantified. These insights are used in the design of the intra-cluster routing of the IDLB protocol.

3.2. Routing Schemes

In order to maximize the performance of IDLB, specifically tailored path computation and routing algorithms are proposed, complementing the considered design. While multiple suitable options can be applied, only the most promising solutions, which have been derived and implemented for the reference scenarios, are investigated. In any case, the softwarization of SDN allows changing algorithms or policies by adjusting the controller routing logic accordingly.

3.2.1. Intra-cluster Routing

The central goal of the intra-cluster routing algorithm is to balance the loads while complying with the QoS requirements of the traffic. Within a cluster, all relevant network state information is aggregated at the SDN controller. Therefore, load-aware proactive routing strategies are possible. To enable high system throughput within SCNs, an informed decision-making regarding which paths are best-suited to support incoming traffic is necessary. To achieve optimal performance, all of the resulting routes have to be chosen in a way which minimizes congestion. As described in Section 2.3.2, this optimization can be formulated by the MCFP [32]. Ideally, the controller solves this underlying MCFP to distribute low-priority traffic optimally.

Load-balancing Limitations

In order to avoid saturating links, the load-balancing objective is formulated as minimizing the maximum link utilization over time. So, over all time steps the following objective function is formulated:

$$\min_t \max_{(i,j) \in \mathcal{E}} u_t(i,j) \quad (3.19)$$

To highlight why formulating a solution to this problem is intractable, a corresponding value function $V_t(s_t)$ is formulated for a state s_t . This function considers the resulting cost, represented by a cost function C_t for the current moment in time, as well as the impact on the value of future states. To provide a generic formulation of the value function, the cost of the current network state s_t and the flow requests f_t are considered:

$$V_t(s_t) = \min_{f_t} [C_t(s_t, f_t) + V_{t+1}(s_{t+1})] \quad (3.20)$$

Given the objective function, the immediate cost function C_t can be described by:

$$C_t(s_t, f_t) = \max_{(i,j) \in \mathcal{E}} u_t(i,j) \quad (3.21)$$

So, for the complete time horizon from time t to T , the optimal value function representing the minimal total cost is:

$$V_t(s_t) = \min_{f_t, \dots, f_T} \left(\max_{t' \geq t} \left[\max_{(i,j) \in \mathcal{E}} u_{t'}(i, j) \right] \right) \quad (3.22)$$

However, this formulation implicitly simplifies important system characteristics. Crucially, propagation delays have to be considered as they are significant in SCNs. A corresponding function $\tau(i, j, u_t(i, j))$ is thus introduced to capture their impact. This delay function introduces non-linearities, creating a non-linear dynamic optimization problem. Moreover, this function is also impacted by additional delays for queuing and switching. To represent delays caused by queuing, models from queuing theory can be used.

Time-dependent topology changes, due to handover events, are also not included in the formulation. The time-varying set of active edges $\mathcal{E}_t$ is used to represent these changes in the state transition function.

Most notably, the reference scenario assumes stochastic traffic models. So, the problem becomes a stochastic dynamic programming problem: future states are probabilistic in nature. Given that these models are geographical rather than topological, the traffic requests are inherently dynamic. Furthermore, due to the distribution of traffic which follows the distribution of UTs and GWs, an analytical formulation of the traffic models is exceedingly difficult. Therefore, the estimation of future states may be quite imprecise, limiting the effectiveness of approximate solutions. ω_t is used to denote these incoming random traffic events. The value function of a current state s_t thus becomes:

$$V_t(s_t) = \min_{f_t} [C_t(s_t, f_t) + \mathbb{E}_{\omega_t} [V_{t+1}(s_{t+1}) | s_t, f_t]] \quad (3.23)$$

This combination of time-dependency, non-linearity, stochasticity, and dynamic topology represents a complex problem to solve formally. Similar problems have been determined to be unsolvable using polynomial-time algorithms [32]. The amount of parameters and their inter-dependencies result in a significantly large state space. The state transition function Φ highlights this complex interplay between observed network states, control actions, delays, random events, and topology changes:

$$s_{t+1} = \Phi(s_t, f_t, \tau_t, \omega_t, \mathcal{E}_t) \quad (3.24)$$

In addition, using DP to solve the problem results in high computational complexity due to the curse of dimensionality [27]. The state space grows exponentially with the number of input parameters, and direct DP approaches are thus not applicable in practice. Consequently, approximate solutions based on heuristics hold more promise for in-orbit application. For instance, heuristic assumptions for LEO SCNs have been formulated [24].

Given that the problem has to be solved repeatedly (e.g., for every flow request) and considering the on-board limitations of satellites, this work focuses on a low-complexity solution. So, instead of approaching the MCFP directly, a snapshot-based proactive path-finding algorithm is proposed, aimed at the current cost function.

Dynamic Best-of-k Algorithm

To maintain low complexity for varied cluster sizes, a dynamic best-of-k paths algorithm is used for intra-cluster routing. The approach follows similar ideas to a previously proposed top-k-paths algorithm based on SDN [123], but focuses on different metrics and objectives. By limiting the search to k suitable paths and utilizing a parameterized utility function, performance improvements are achieved while reducing computational complexity by orders of magnitude. This efficiency is crucial for real-time decision-making in space-based networks where computational resources are limited and latency requirements are strict.

By focusing on small values for k , the approach scales gracefully even for large cluster and constellation sizes. The approach enables QoS-aware forwarding by using a parameterized utility function which prioritizes relevant path characteristics. The proposed SDN architecture complements this approach, as it enables centralized policy definition.

A set-based approach is important, as traffic spikes may result in pressure on currently low-cost links. During a link load update period, the controller may be unaware of potential changes. If a high number of requests occur in this time frame, the routing logic has to account for the new demands and their interaction. When using sets of viable paths, choosing the best option decreases variance in path quality [124]. If paths have equal cost, a tie-breaking mechanism is applied similar to "Always-Go-Left" [125]. A nonuniform choice combined with such an asymmetry has been shown to improve load balancing [125].

For the different QoS classes, the following parameterized utility functions $Y_{\{1,2,3\}}$ are considered for a path p :

$$Y_1(p) = (L(p) + \epsilon)^{-1} \quad (3.25)$$

$$Y_2(p) = \psi_2 \cdot (N_{\text{hops}}(p) + \epsilon)^{-1} + (1 - \psi_2) \cdot (U(p) + \epsilon)^{-1} \quad (3.26)$$

$$Y_3(p) = \psi_3 \cdot (N_{\text{hops}}(p) + \epsilon)^{-1} + (1 - \psi_3) \cdot (U(p) + \epsilon)^{-1} \quad (3.27)$$

In these formulations, ϵ represents a small positive value to avoid division by zero. ψ_2 and ψ_3 are weighting factors for the link load. The values are chosen to comply with the QoS requirements and the system scenario. Path latency $L(p)$ is only used for the delay-critical class QoS1. The other classes focus on the number of hops N_{hops} and path utilization $U(p)$.

$$U(p) = \sum_{(i,j) \in p} u_t(i,j) \quad (3.28)$$

The approach corresponds to a dynamic k-shortest-path algorithm with a subsequent comparison step [119,120]. $\mathcal{V}_c$ denotes the number of nodes in a cluster. The complexity in time for the path computation for all flows F is thus:

$$\mathcal{TC}_{\text{intra, path}} = O(|F| \cdot (|\mathcal{E}| + |\mathcal{V}_c| \log |\mathcal{V}_c| + k)) \quad (3.29)$$

Paths are compared according to the utility function of the respective QoS class. This additional step has the following complexity in time:

$$\mathcal{TC}_{\text{intra, decision}} = O(|F| \cdot k) \quad (3.30)$$

Tracking all possible paths can lead to exponential growth in space depending on the connectivity of the graph. As described in the context of set-based source-routing, the complexity in space can grow exponentially. By limiting the choice to only a handful of paths, and not tracking others, the general complexity in space is reduced to:

$$\mathcal{SC}_{\text{intra}} = O(|\mathcal{V}_c| \cdot k) \quad (3.31)$$

These formulations assume the worst case, where every considered path has a length of at most $|\mathcal{V}_c|$ nodes (a path visits every node of the sub-network). The same approach is applied by the God's Eye View routing benchmark, presented in Section 3.3.2, due to the prohibitive space complexity of exhaustive path evaluation.

Per-packet considerations

A flow-based routing configuration enabled by SDN is generally assumed for IDLB. The control unit tracks all current flows and can set up rules individually, if necessary. Nevertheless, per-packet forwarding is also a potentially useful approach in the given context. Stateful information about flows is not required, which can result in a less complex routing logic.

Accordingly, packets are forwarded based on generic routing instructions independent of their flow (including details about source or demand). If a load-balancing update is triggered, all packets toward a certain destination adjust their path accordingly. Since high-priority traffic is not load-balanced, this poses no problem in terms of jitter for delay-critical traffic. Nevertheless, periodic updates may lead to path instability for load-balanced traffic. Link load hysteresis and thresholds have to be implemented to mitigate potential route flapping.

Given these considerations, flow-based routing is expected to enable significantly improved decision-making regarding network congestion compared to per-packet forwarding [123]. Effective load-balancing approximating the MCFP becomes challenging for the latter, due to the lack of distinction between flows. This assumption is evaluated in Section 3.3.2, where the QoS compliance under high network loads is compared. To distinguish per-packet IDLB, it is called **P-IDLB** in the following. For clarity, per-flow IDLB is called **F-IDLB**.

3.2.2. Inter-cluster Routing

For inter-cluster routing, the primary focus lies in limiting signaling overhead, while avoiding potential instabilities and congestion. Since geographical routing approaches for SCNs have the ability to strike attractive trade-offs with regard to signaling [39], a related approach is considered.

Based on the proximity of clusters to the geographical areas, the destination cluster is determined. Accordingly, an SPT is computed on a cluster-level providing all viable neighboring clusters. The key problem is now to identify on which specific link to exit the current one. The cluster-external node which is reached by this link, the NHCN, is used as the destination in the geographical routing table. The actual serving satellite is resolved in the destination cluster.

3. Distributed Load-balanced Software-Defined Networking

The NHCN choice is QoS-dependent, similar to intra-cluster routing. Delay-critical traffic should follow the shortest path, while any low-load link is valid for best effort traffic. Notably, the choice should be randomized for low-priority flows to mitigate bottlenecks. Algorithm 2 represents such an NHCN function.

For smaller cluster sizes, topological scenarios arise in which few viable links to leave the cluster may be available. For instance, near the poles, because of ISL shutdowns, bursty traffic spikes may result in saturated links to the next cluster. Naturally, forgoing end-to-end route estimations results in some zig-zagging between clusters. Nevertheless, the resulting end-to-end latency is acceptable for the given QoS classes.

Algorithm 2 IDLB inter-cluster routing: finding next-hop-cluster nodes

Require: current cluster, destination cell, destination cluster, QoS class

Ensure: Destination cluster is not current cluster

```

1: function NEXT_HOP_CLUSTER_NODE(current_cluster, dst_cluster, qos)
2:   next_clusters  $\leftarrow$  current_cluster.next_hops(dst_cluster)
3:   best_value  $\leftarrow$  max_float
4:   best_nhcn  $\leftarrow$  null
5:   candidates  $\leftarrow$  list()
6:   for each node nhcn  $\in$  current_cluster.nhcns() do
7:     ngbr  $\leftarrow$  get_neighbor_in_cluster(nhcn, current_cluster)
8:     if (nhcn  $\notin$  next_clusters  $\vee$ 
        isl_shutdown(ngbr, nhcn)  $\vee$ 
        isl_load_threshold(ngbr, nhcn, qos)) then
9:       continue ▷ Skip unsuited options
10:    end if
11:    dist  $\leftarrow$  distance(nhcn, dst_cell)
12:    link_load  $\leftarrow$  isl_link_load(ngbr, nhcn)
13:    value  $\leftarrow$  cost(dist, link_load, qos)
14:    if value < best_value then
15:      best_value  $\leftarrow$  value
16:      best_nhcn  $\leftarrow$  nhcn
17:    end if
18:    if to_randomize(qos) then
19:      candidates.add(nhcn) ▷ Add to candidates list
20:    end if
21:  end for
22:  if best_nhcn = null then
23:    return null ▷ No suitable next-hop cluster node
24:  end if
25:  if to_randomize(qos) then
26:    return sample(candidates)
27:  else
28:    return best_nhcn
29:  end if
30: end function

```

The complexity of the inter-cluster routing procedures is dominated by the cluster-based path selection and the NHCN choice. In both cases, the set size is relatively small. For SCN-288, there are at most 24 individual clusters. The complexity in time for a per-flow approach with $|F|$ flows, and $|\mathcal{C}|$ clusters (and $|\mathcal{E}|$ virtual links between them) is:

$$\mathcal{TC}_{\text{inter}} = O(|F| \cdot (|\mathcal{E}| + |\mathcal{C}| \log |\mathcal{C}| + N_{\text{NHCN}})) \quad (3.32)$$

The complexity in space linearly depends on the number of clusters and NHCNs. Due to the candidate list, space required of size $|N_{\text{NHCN}}|$ is counted twice. Depending on the cluster size, $|\mathcal{C}|$ or $|N_{\text{NHCN}}|$ is larger.

$$\mathcal{SC}_{\text{inter}} = O(|\mathcal{C}|) + O(|N_{\text{NHCN}}|) + O(|N_{\text{NHCN}}|) = O(|N_{\text{NHCN}}|) \quad (3.33)$$

Per-flow considerations

For the NHCN choice, per-flow and per-area decisions are possible. The former results in an NHCN computation for every flow. The latter periodically decides on suitable NHCNs for every geographical cell.

Due to the considered scenario, the per-area approach is expected to result in worse performance. While 6962 areas are used, the ones containing GWs or multiple UTs are the focus of a significant traffic share. Consequently, determining a single NHCN for all flows with the same destination area can result in bottlenecks. The assumption is investigated in Section 3.3.2. In a per-flow approach, this issue is avoided by selecting from a set of valid NHCNs, even for the same destination.

3.3. Comparative Quantitative Evaluation

Following the design of the protocol and the algorithms used for intra- and inter-cluster routing, the performance of IDLB is now evaluated quantitatively. To this end, it is implemented in a specifically designed network simulator, which enables varied analyses of design choices as well as comparisons to benchmark solutions.

3.3.1. System-level Constellation Routing Simulator

To evaluate routing protocols on a system-level basis, a simulator was developed to analyze routing protocols for SCNs. The software was primarily developed in the context of a corresponding ESA project [47]. It represents a discrete event-based network simulator for efficiently analyzing routing protocols in large-scale SCNs on a packet level.

The software was written in C++ using templated classes to enable generic and flexible data structures. The movement of the constellation, as well as the creation, handling, and forwarding of packets are realized by timed events. Using recursive functions, it is possible to include periodic operations in the time-ordered event loop. This allows for the tracking of packet objects propagating and queuing on nodes, while measuring metrics such as end-to-end latency. Python-based pre- and post-processing scripts are used to generate input files and to visualize simulation results. The overall design of the simulator is presented in

Figure 3.9. The complete parameter configuration, in accordance with the reference system, is provided in Appendix B.

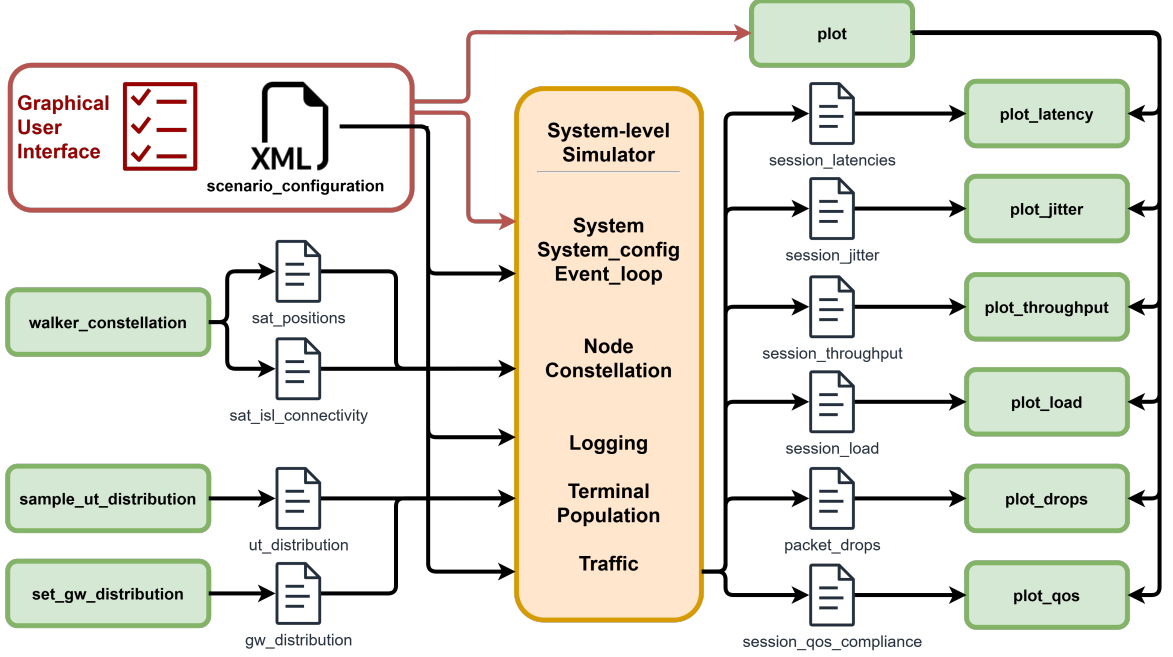


Figure 3.9.: Schematic diagram of the architecture of the system simulator, including input and output files, as well as processing scripts.

The generated traffic is based on sessions. A session represents the transmission between two terminals on ground, i.e., UTs and GWs. While the simulator includes both on-off as well as constant-bit-rate sessions, we focus on the latter for more coherent results in this investigation. Overall a session is defined by a unique identifier, its start time, its duration, the involved terminals, and its QoS class (determining its priority). The starting time is sampled uniformly across the simulated time. An exponential distribution is used for the end of the session with the defined average session duration. The priority of sessions is distributed to reflect the considered traffic shares. The sets of UTs and GWs are sampled accordingly. The parameters determining this sampling are based on the inputs defined in Section 2.1.4. Using this format, a diverse traffic model emerges with temporary spikes, and potential geographic hot spots depending on the ground segment. Nevertheless, over longer simulated time periods, which significantly exceed the average session duration, the overall network load does not vary too strongly. So, comparable traffic requirements are maintained for different simulation runs.

3.3.2. Performance Evaluation

To evaluate IDLB, suitable benchmarks are introduced. Then, the protocol performance is evaluated in terms of the end-to-end characteristics most relevant to QoS classes: latency, jitter, and dropping rate. The performance under increasing network loads in terms of QoS compliance is subsequently evaluated for different cluster sizes. In addition, the impact on performance of different design choices is investigated.

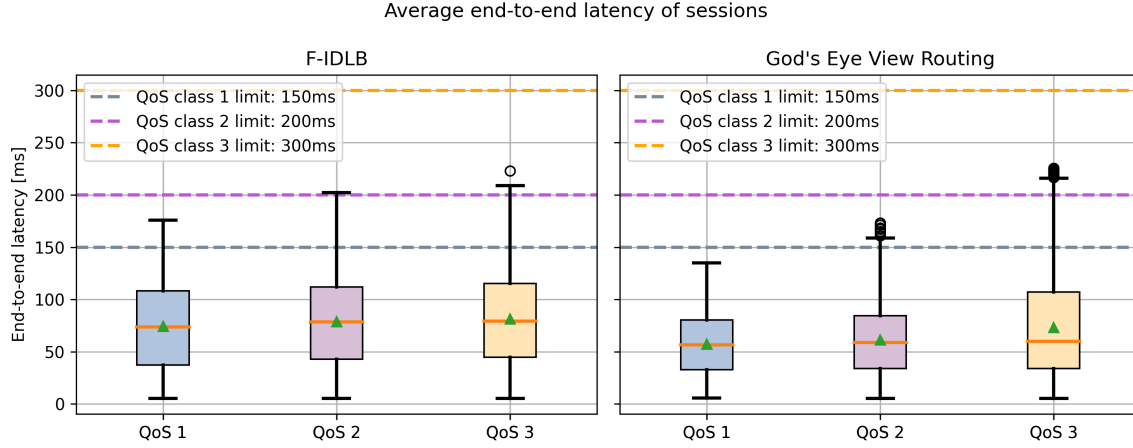


Figure 3.10.: Box plot of end-to-end latency of sessions for F-IDLB with a cluster size of 48 (left plot) and the upper theoretical benchmark GEVR (right plot). Both schemes use QoS-aware routing.

Benchmarks

As an upper benchmark, **God's Eye View Routing (GEVR)** is proposed. The approach is based on instantaneous perfect knowledge of all network states everywhere, while exploiting a set of suitable paths. There is no signaling traffic or delay. Naturally, such an algorithm cannot be implemented in the real world. Due to the inherent structure of the network, there is a diverse set of paths satisfying the QoS requirements for most ingress-egress pairs. To reduce the complexity in space, not all possible paths are tracked by the algorithm described in Section 3.2.1.

For a state-of-the-art benchmark solution, a **dynamic source-routed scheme** is used. It is based on Dijkstra's algorithm to provide the fastest routes in non-congested networks. As link load information is not shared, it is prone to packet drops in more demanding scenarios. It serves as a lower benchmark to quantify the advantages of load-balancing approaches in terms of latency and dropping rate.

Protocol Performance

Firstly, the evaluation must establish that the protocol design results in correct routing and forwarding behavior. So, core KPIs of IDLB are investigated and compared to the results of the GEVR benchmark solution. Overall, the protocol typically complies with QoS requirements. Latency, jitter, and later packet dropping characteristics are considered the most relevant metrics. The results of all sessions are analyzed, which includes the sampled UT-to-UT transmissions as well as UT-to-GW and vice versa.

End-to-end latency represents a fundamental drawback of the independent cluster design. While shortest paths are possible, additional delays are expected on average. Due to the independent load-balancing of clusters and the geographical inter-cluster routing, some zig-zagging is possible. Nevertheless, the comparison of end-to-end latency in Figure 3.10 shows that most sessions comply with the QoS requirements. For QoS class 1 and 2, there are

3. Distributed Load-balanced Software-Defined Networking

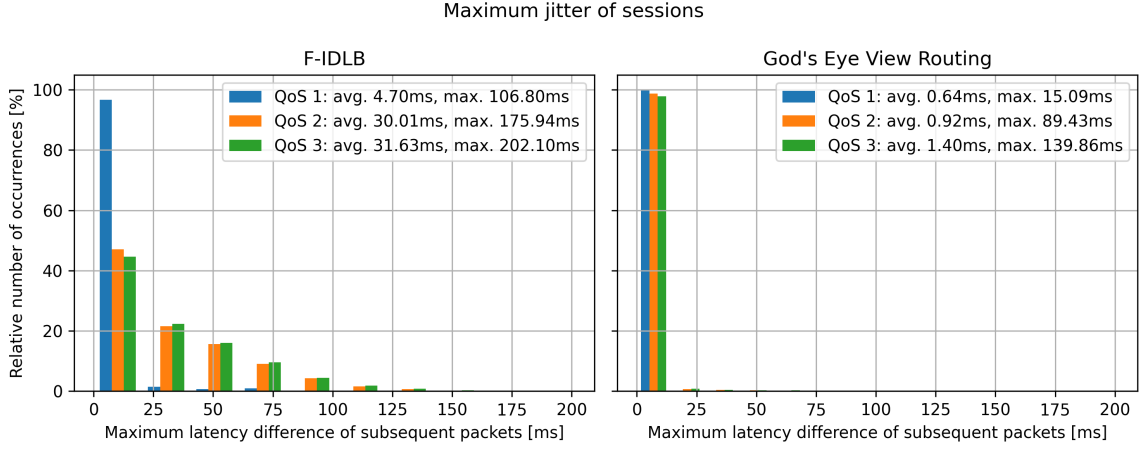


Figure 3.11.: Delay variance of sessions for F-IDLB with a cluster size of 48 (left plot) and the upper theoretical benchmark GEVR (right plot). Both schemes use QoS-aware routing. Outliers due to specific handover situations are expected.

less than 1% of sessions exceeding the maximum delay (150 ms for QoS1 and 200 ms for QoS2).

Moreover, the QoS-based forwarding is illustrated, as best effort traffic of QoS class 3 tends to follow longer paths, but still within the defined limit of 300 ms. For GEVR, this pattern is even more distinct: the latency distributions follow the end-to-end latency requirements of each QoS class. For the delay-critical class, very low values are achieved, while apparently longer paths are utilized for QoS class 3. As the path computation of GEVR is able to evaluate candidate paths on an end-to-end basis, any path which fulfills the latency requirement of the QoS class is an option. Thus, the approach can distribute best effort sessions among paths with low congestion.

The maximum jitter, in the form of the maximum delay between subsequent packets is considered as well. A comparison between the F-IDLB approach with a cluster size of 48 and GEVR is provided in Figure 3.11. Here, significant outliers with much higher values than the mean and average are observed for all routing approaches. To better understand the distribution of these delay variances, a histogram is used to illustrate their frequency and rarity. Since a resulting path of IDLB is under the control of multiple control entities, more frequent adjustments are expected. Cell handovers between clusters may result in path changes which are not considered by end-to-end schemes such as GEVR. Nevertheless, the impact is well within the requirements. For QoS class 1, the average jitter is 2.91 ms, and the maximum observed 85.15 ms. While this is significantly larger than the GEVR counterpart (on average 0.64 ms, maximum 15.09 ms), these variances are acceptable in the reference scenario. Even when GEVR is used, high jitter occurs for QoS class 3. The causes are handover events at the seams which coincide with path adjustments due to high link loads. As the routing of this best effort traffic considers paths with valid end-to-end latency, but does not rank them, paths of varying length occur.

It is important to note that signaling traffic has been omitted from this analysis given that it does not represent data plane traffic. Signaling does not follow end-to-end paths from

and to terminals on ground, thus its latency is not comparable to data plane traffic. As we consider individual signaling messages and not flows, there is no jitter. In general, signaling packets follow the same paths which have been computed for QoS class 1. However, the packets are scheduled using a dedicated signaling buffer which prevents packet losses for the encountered network update frequencies. The latency characteristics correspond to the convergence delays shown in Table 3.3.

Network Load Performance

To evaluate the load-balancing capacities of the proposed protocol, its behavior when encountering high traffic volumes is investigated. So, in this comparison, the resilience of the approaches in terms of QoS compliance to increasingly high network loads is tested. In this context, the increase in packet drops represents the main issue. In order to maximize throughput, the protocol has to balance the elevated traffic load to maintain QoS compliance. A QoS compliance of $>95\%$ is considered acceptable.

Given the large set of simulations, a shorter simulation window with the same average network load is used. Here, network load is defined as the aggregated data entering the SCN. So, every packet generated at a UT or GW and sent up to a connected satellite is counted in a sliding window. This constellation-wide uplink of packets to the constellation in terms of amount of data is then divided by the duration of the observation window. For a given window size T_{window} , using N_{uplink} to describe all the packets which were sent up and $\zeta(\rho_i)$ to denote the size of the i -th packet entering the SCN, the network load at t is thus:

$$\text{Network load}(t) = \frac{1}{T_{\text{window}}} \sum_i^{N_{\text{uplink}}(t)} \zeta(\rho_i) \quad (3.34)$$

While it is possible that packets have different sizes, in the presented simulations we assumed a general packet size of 12 kbit. Analogously, the system throughput describes the packets sent down from the SCN in the current window. The resulting system load and throughput for the link-load-agnostic source-routing approach is shown in Figure 3.12. Due to the ramp up and down of the traffic models, higher peak loads occur in the shorter simulation window. Notably, the throughput is significantly lower in Figure 3.12, indicating that packet losses occurred for the shown source-routed approach.

The comparison, illustrated in Figure 3.13, shows that the proposed approach (F-IDLB) is able to outperform source routing, in terms of QoS compliance for increasing network load. Looking at the overall QoS comparison, source-routing falls below the 95% compliance threshold at around 7.6 Gbps in this scenario. IDLB with 12-node clusters provides this compliance up to 12.5 Gbps, an increase of 64.5%. Larger cluster sizes improve load-balancing further. At network loads of approximately 15.0 Gbps and 16.2 Gbps, the 24-node and 48-node cluster approaches fall below the threshold in that order. Compared to source-routing, this corresponds to improvements of 97.4% and 113.2%, respectively.

These results highlight the load-balancing capabilities of IDLB for QoS-based routing. As all of the routing and pathfinding logic is within the controller software, it is possible to adjust or change intra- as well as inter-cluster algorithms by providing software updates to the controller logic. Therefore, solutions suitable to the system specifics and processing

3. Distributed Load-balanced Software-Defined Networking

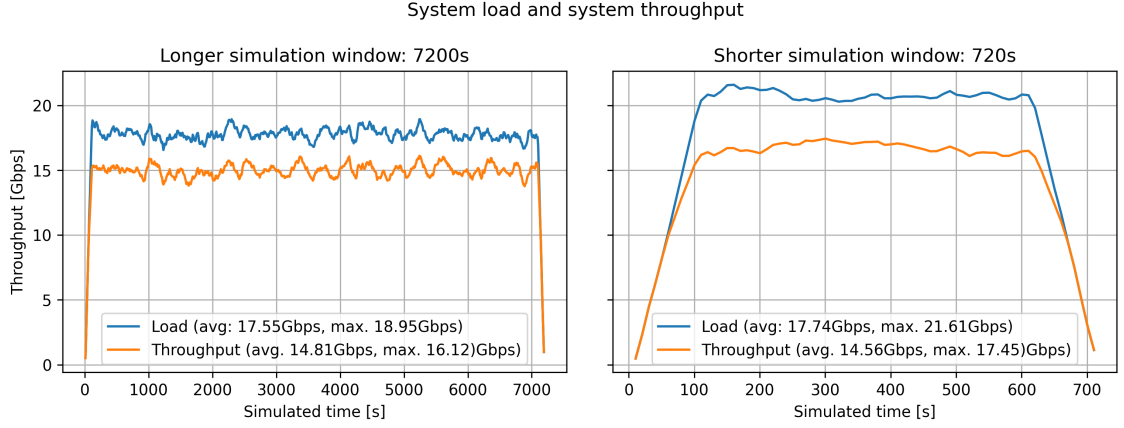


Figure 3.12.: Network load and system throughput for source-routing in SCN-288. Longer window with a simulated time of 7200s, and shorter window with simulated time of 720s are shown. Due to uniform session distribution and margins dictating simulation start/end times, shorter window results in slightly higher loads.

power of the satellites can be applied to IDLB. A further analysis of relevant design aspects is provided in the following.

Cluster size comparison

As expected, increasingly large cluster sizes result in improved load-balancing, as longer end-to-end paths in a larger scope are possible. However, there appear to be diminishing returns, as illustrated in Figure 3.13. Further KPIs for the different cluster sizes, under a network load of 13.2 Gbps, are summarized in Table 3.3.

The observed signaling overhead is significantly below the required ratio of 5 % for the given traffic load. As expected, there is a clear increase in the relative share of signaling packets for larger clusters. Most notably, Table 3.3 shows a significant increase in convergence delay. This metric is based on the latency of signaling packets within a cluster. For every packet, its creation time at the source and moment of arrival at the destination are tracked by the simulator. Importantly, the delay does not solely represent signaling to and from the controller. For instance, for ESL handovers, updates have to be forwarded to all other nodes of the cluster to adapt their intra-routing tables accordingly. As this can include two satellites at opposite ends of a cluster, the resulting convergence latency can thus be longer. Since the physical distances in large clusters introduce high propagation delays, the average increases from 56.968 ms for the 12-node cluster to around 273.526 ms for the 48-node setup.

For the given working point, the difference between approaches in terms of packet dropping ratios is significant. While the protocol is still able to comply with QoS requirements for most sessions, but there are more apparent bottlenecks resulting in increased dropping rates when using smaller clusters. For QoS3, a rate of 1.344×10^{-4} occurs in the 24-node cluster, while 2.285×10^{-6} is observed for the largest one. Since the simulation was configured to avoid packet drops for all QoS classes, there are similar dropping rates between them.

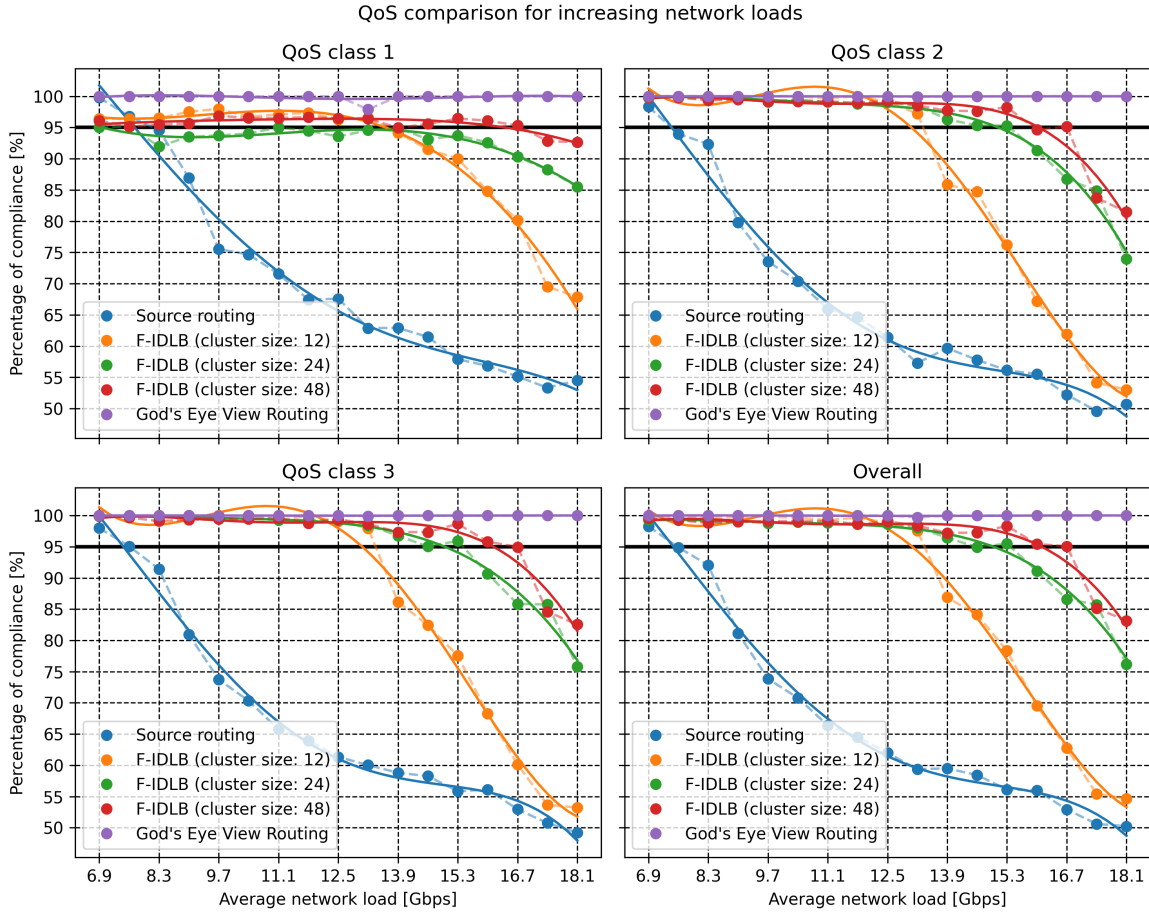


Figure 3.13.: QoS comparisons for increasing network loads for IDLB with a cluster size of 12 (in orange), a cluster size of 24 (in green), a cluster size of 48 (in red), as well as source-routing (in blue), and GEVR (in violet). Individual sub-plots for each QoS class: delay-critical forwarding (QoS1), forwarding (QoS2), best effort (QoS3). Source-routing approach is not QoS-aware.

Rare packet drops are occasionally observed in certain handover edge cases. While recorded separately, these events are included in the reported dropping rates to ensure completeness. Nevertheless, their influence on the overall results is considered negligible, with the maximum relative share being fewer than 5 % of observed drops.

Overall, the comparison illustrates the trade-offs for cluster sizes. In the 12-node cluster, load-balancing is limited. With larger cluster sizes, traffic is distributed more effectively and higher network loads supported. However, given the design of the approach, there are diminishing returns. The 48-node cluster only outperformed the 24-node cluster by 12 % more load supported (with >95 % compliance). Moreover, significant increases in maximum routing convergence delay are observed.

Using larger clusters, beyond the evaluated sizes, does not yield performance improvements. Instead, increased packet drops due to routing convergence delays and timing mismatches are observed. For instance, clustering 96 nodes results in three physically large domains with

Table 3.3.: Comparison of convergence rate and overall packet dropping rates for different cluster sizes in SCN-288 at a network load of 13.2 Gbps.

SCN-288 cluster size	Control plane share	Convergence delay [ms]: average, maximum		Packet dropping rates: QoS class 1, QoS class 2, QoS class 3		
12 (4×3)	0.062%	056.968	104.782	1.044×10^{-4}	3.571×10^{-4}	3.374×10^{-4}
24 (6×4)	0.112%	117.338	285.276	2.948×10^{-6}	7.050×10^{-5}	1.344×10^{-4}
48 (8×6)	0.208%	273.526	747.552	8.942×10^{-7}	2.557×10^{-6}	2.285×10^{-6}

significant intra-cluster delays and signaling overhead. To avoid packet drops in this context, significant adjustments to the timing and related configuration of the protocol are necessary. However, to maintain a consistent evaluation framework, these are not considered.

Impact of Design Choices

There are various relevant design aspects which can be evaluated in the context of this investigation. Most notably, comparisons of per-packet and per-flow intra-cluster routing, as well as per-area and per-flow NHCN choice, are performed.

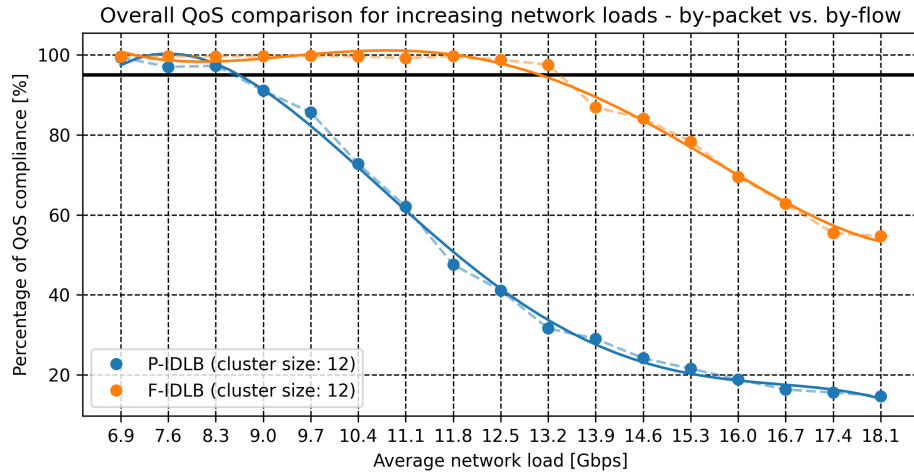


Figure 3.14.: QoS comparisons for increasing network loads for per-packet IDLB (P-IDLB in blue) with a cluster size of 12, and per-flow IDLB (F-IDLB in orange) with a cluster size of 12.

Per-packet vs. per-flow IDLB: To quantify the viability of a per-packet approach, it is compared to the per-flow version. The results are shown in Figure 3.14. The P-IDLB approach is clearly outperformed by its F-IDLB counterpart. The former crosses the 95 % compliance threshold at around 8.6 Gbps, which is better than source-routing. Additionally, performance degrades significantly. F-IDLB represents an improvement of 45.3 % in terms of supported network load. This result clearly demonstrates the advantage of using flow-based

SDN strategies.

As described in Section 3.2.1, P-IDLB sets non-flow-specific routing table entries. Therefore, it is more likely to create bottlenecks for flows with similar destinations. In-depth analyses show that this traffic pattern is prevalent in multiple clusters, resulting in packet drops. For this reason, increasing cluster sizes does not significantly improve performance for P-IDLB. This is in contrast with F-IDLB, which exhibits improved load-balancing as cluster sizes increase, as illustrated in Figure 3.13.

Notably, the choice of NHCN is also limited by this design. One specific NHCN is chosen for each destination cell, which can lead to congestion at adjacent links. The impact of generic and flow-based NHCN choice is investigated in the following section.

Inter-cluster routing - NHCN choice: As described in Section 3.2.2, a specific NHCN mapping to each destination cell is expected to cause bottlenecks. Thus, this design is compared with a per-flow NHCN choice. The results are shown in Figure 3.15. As expected, the performance of per-area choice is significantly worse and varies much more at higher network loads. An analysis of the results showed significant drops occurring at the borders of the clusters.

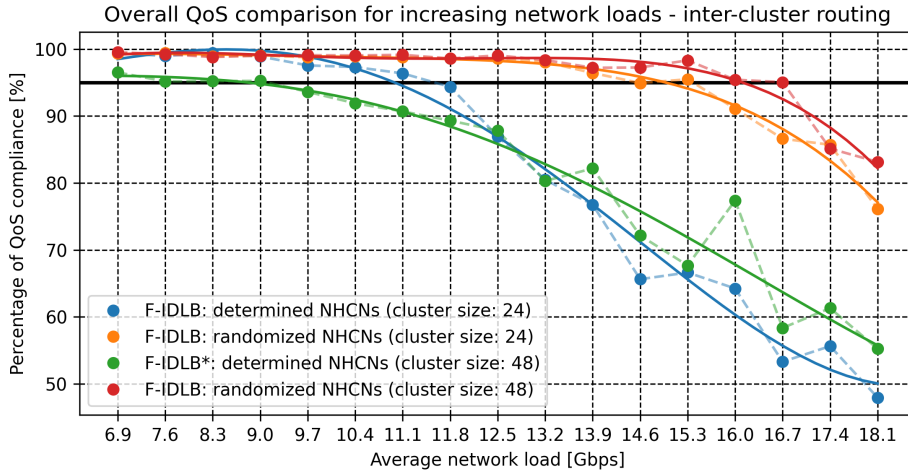


Figure 3.15.: QoS comparisons for increasing network loads for a per-area determined NHCN mapping with a cluster sizes of 24 (blue) and of 48 (green), as well as randomized per-flow NHCN mapping with a cluster size of 24 (orange) and of 48 (red). The determined mapping with a cluster size of 48 also shows buffer limitations (F-IDLB* in green).

It is important to note, that the shown 48-node cluster approach with the per-area mapping also shows buffer limitations. Using the same buffer size of 360 kbit causes issues in this scenario. When specific instructions are forwarded to 47 different nodes, a larger buffer or careful timing is required to avoid overflows. Signaling traffic is prioritized by the scheduler, so packets of other QoS classes may be dropped. Figure 3.15 illustrates this systematic error consisting of periodic drops due to peaks in high-priority signaling traffic. Consequently, this faulty approach rarely achieves more than 97% QoS compliance. For better results, a buffer size of 1.08 Mbit was used for all other simulations using 48-node clusters.

SCN-1440 comparison

To evaluate the impact of constellation size, additional simulations with SCN-1440 are of interest. The same configurations are used. However, for this constellation, a cluster of similar physical size to the previous 12-node cluster now consists of 48 nodes due to the increased satellite density of SCN-1440. So, while the number of nodes increases, the intra-cluster propagation delays are comparable as the physical cluster dimensions remain the same. Accordingly, the observed maximum routing convergence is 222.165 ms. The comparison of QoS compliance for increasing network loads is shown in Figure 3.16. Importantly, the x-axis range is now shifted: network loads from 13.9 Gbps to 25.0 Gbps are observed.

Given the increased number of satellites and links, much higher network loads can be achieved in the SCN-1440 constellation. However, given that the same cluster size and structure are used, the comparison is of interest. With 95 % compliance, a data rate of approximately 20.2 Gbps was supported. Thus, an increase of approximately 25 % is achieved for the same number of nodes in a cluster. Compared to a similarly sized cluster, which in the case of SCN-288 is a cluster size of 12, the increase is even more significant: approximately 61 %. In SCN-288, F-IDLB with a cluster size of 12 drops below 95 % at 12.5 Gbps (see Figure 3.13).

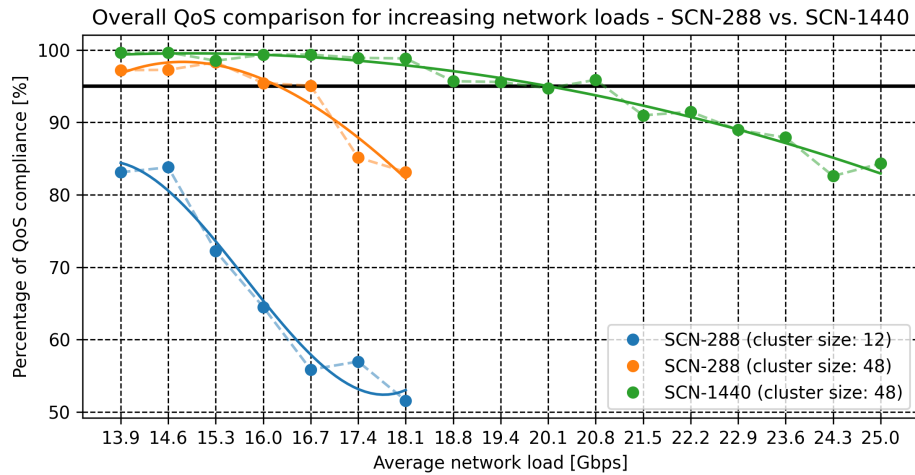


Figure 3.16.: QoS comparisons for increasing network loads for SCN-288 and SCN-1440 constellations. SCN-1440 constellation uses 48-node clusters (in green), while for SCN-288 both 48-node clusters (in orange) and 12-node clusters (in blue) are used. The latter included due to the similar physical size to 48-node clusters in SCN-1440.

In general, scenario-specific adjustments to the configuration of the algorithms are possible. However, to enable a direct comparison, the same parameters used for SCN-288 were applied. The promising simulation results obtained in this section demonstrate the potential of IDLB for efficient and scalable routing in satellite constellations. To further validate these findings, the next section presents a demonstration of the protocol on a hardware test bed.

3.4. Demonstration of Feasibility

To assess the functionality of the developed IDLB routing protocol, a comprehensive demonstration was set up to evaluate its feasibility for in-space applications. The protocol was implemented on equipment representative of current on-board satellite processing hardware. The Devices Under Test (DUTs) were equipped with LX2160A boards from NXP [58]. This hardware was chosen for the test bed as it is intended for on-board usage [47]. Further information about this processor is provided in Appendix A. For the proposed architecture, Technology Readiness Level (TRL) 5 has been attained [47], indicating that it has been validated in a relevant environment.

3.4.1. Demonstrator Setup

In this demonstration, instead of a complete constellation, four individual platforms were used to represent satellite nodes in a LEO SCN. Network management was performed on-board these satellite routers by network management services, which implemented the necessary functionality for the routing protocols. In the context of this demonstration, these four satellite routers are represented by the DUTs.

To establish a realistic environment, support equipment was provided as part of the demonstrator, including traffic generators, link emulators, a control system, and a monitoring system. This support equipment was hosted on hardware separate from the DUTs to ensure unbiased testing. The system entities, summarized in Table 3.4, were inter-connected via Ethernet links. These connections run through a central switch, allowing for the reconfiguration of the network without the need to physically alter the connections of the devices in the network. An overview of the demonstrator test-bed is provided in Figure 3.17.

In the following, the focus is on the integration of the IDLB routing logic into an SDN controller framework. To demonstrate that the protocol is able to deliver on its key design aspects, tests regarding processing limitations, handover management due to dynamic topology changes, and QoS-based load-balancing are included. The tests are summarized in Table 3.5.

3.4.2. Software-Defined Networking Controller

To provide the control unit functionality intended by IDLB, the `ryu` Python library [126] was used as an SDN controller framework. The software provides a southbound interface to create and process control packets, e.g., flow instructions. Routing logic implemented in the software simulator (described in Section 3.3.1) was extracted and wrapped into the control software. Several procedures and interface functions were adjusted to work with the input formats presented by the modified controller framework. The controller software was required not to exceed, on average, a CPU load of 30 % and a memory usage of 50 %. In all tests, except during the maximum processing power stress test, these requirements were fulfilled.

Notably, aggregated routing table updates, as envisioned in the simulator, are not possible with OpenFlow. The OpenFlow protocol only enables individual messages for table updates. While this causes no issues impacting the performance, it results in more, smaller control

Table 3.4.: Summary of devices involved in hardware test bed.

Device	Designation	Function
Devices under test	sat0, sat1, sat2, sat3 (as well as IP addresses)	Satellite nodes based on hardware considered for in-orbit utilization.
Traffic generators	acm0, acm1, ..., acm6 (as well as IP addresses)	Source and destination of IP/UDP traffic. Enables packet analysis.
GW nodes	gw1, gw2	Virtually connected to traffic generators. In virtualization cluster.
UT nodes	ut0, ut1, ut2, ut3	Virtually connected to traffic generators. In virtualization cluster.
Ground-satellite ESL emulator	em-gsl-0	Emulating link characteristics of ESL from ground to satellite. In virtualization cluster.
Satellite-ground ESL emulator	em-usl-0	Emulating link characteristics of ESL from satellite to ground. In virtualization cluster.
ISL emulator	em-isl-0	Emulating link characteristics of ISLs. In virtualization cluster.
Demonstrator control system	dcs-0	Control of virtualization cluster.
Monitoring system	monitoring-0	Aggregator of metrics, readings.
Management Laptop	mgmt-laptop	Management of devices, procedures.

plane packets. Overall, the resulting controller software conceptually reflects the capabilities included in the simulator.

3.4.3. Functional Evaluation

Three tests are of particular interest, as they evaluate essential functionality for a real-world application of the approach. While more tests were conducted within the demonstrator framework, the results of these setups are the most informative for the evaluation of IDLB. Firstly, the impact of the controller software on the considered hardware is quantified. Then, handovers and the correctness of dynamic route adjustments are validated. Thirdly, the QoS-based load-balancing capabilities of the protocol are evaluated. Table 3.5 summarizes the functional tests and results.

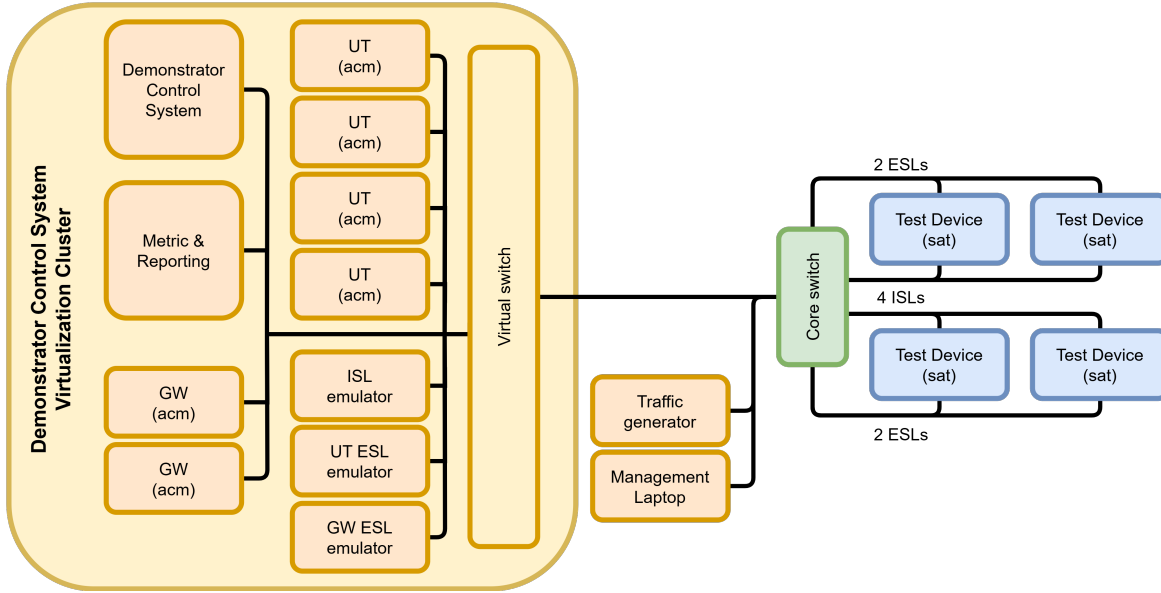


Figure 3.17.: Schematic of demonstrator. Test devices, representing the satellites, are connected to core switch which connects to virtualization cluster.

Table 3.5.: Summary of functional evaluation of IDLB demonstrator.

Test	Description	Passed	Result
Processing limitations	Stress test controller: identify maximum update frequency.	✓	Correct behavior until periods of ≤ 500 ms.
Dynamic handovers	Prepare handover: adapt routing logic, send out timed table updates.	✓	Correctly timed signaling with event.
	Adjust routes: new table entries have to reflect correctly updated routes.	✓	Correctly adjusted routes to topology.
	Limit packet drops: minimize interruption due to handover.	✓	No significant packet drops observed.
QoS-based load-balancing	Identify bottleneck: saturating flow induces congestion, triggering flow-based routing changes.	✓	Correctly identified sudden link saturation.
	QoS-based route adjustment: balance loads based on QoS class.	✓	Correctly re-routed low-priority flows.
	Limit packet drops: minimize interruption due to load-balancing.	✓	No significant packet drops observed.

Processing Limitations

A stress test was designed to evaluate the ability of the SDN controller to process information and to respond accordingly. The goal was to identify the maximum control plane update rate that the controller can handle. To this end, apparent topology changes, which result

3. Distributed Load-balanced Software-Defined Networking

in a cluster-wide re-routing operations, were sent to the control unit with increasingly high frequencies.

The test revealed that the software implementation of the controller, based on a single-threaded `Python` library, represents the limiting factor. Due to the Global Interpreter Lock, a safety mechanism in the `Python` interpreter, the controller runs on a single CPU thread. This represents an inherent limitation of the controller framework, since the memory management of `Python` is not thread-safe. Parallel path computations and update generations were thus not feasible. Thus, the maximum observed load corresponded to a single thread being at full capacity.

The test results showed that the controller was able to send out completely new flow instructions every 500 ms before reaching 99 % thread utilization. However, in the considered scenarios, update periods for such adjustments are typically much longer, in the order of seconds. Tests with logical cluster sizes of 12 and 48 satellites confirmed that the controller can handle the required update frequencies without overloading.

In summary, the results indicate that the on-board SDN controller can achieve significantly higher routing update frequencies than required for the test cases. Therefore, the controller should not overload the processing capabilities of suitable satellites. If needed, the software can be optimized with multi-threading functionality to ensure correct behavior in a real-world system.

Dynamic Handovers

Further testing was conducted to evaluate the ability of the protocol to handle handover events in an SCN. The setup is shown in 3.18, and is based on an ISL topology change. The four DUTs are connected in a grid-like manner mirroring the connectivity of the considered SCNs. A flow was established from `acm1` to `acm3`.

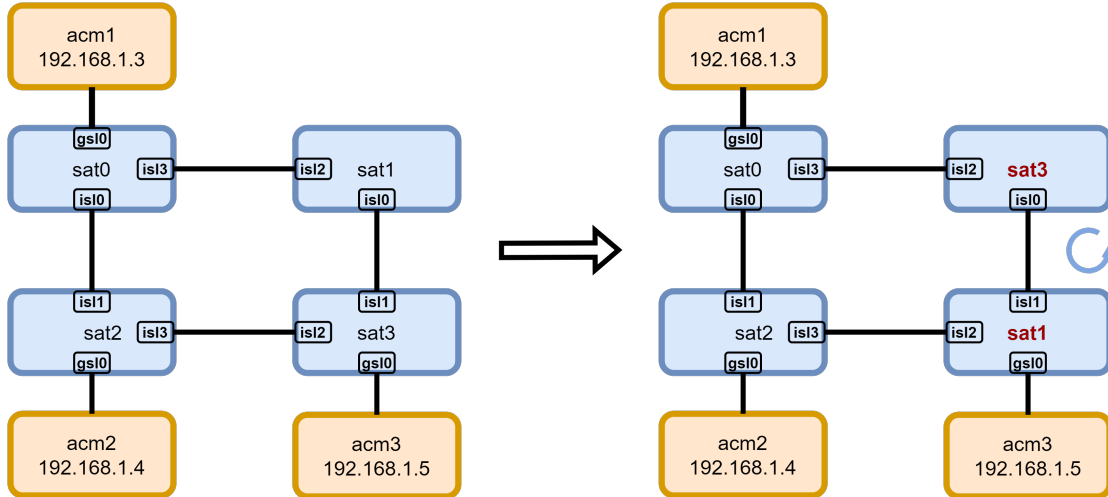


Figure 3.18.: Test setup for dynamic handover: topology before (on the left) and after (on the right) topology change. Satellite router 1 (`sat1`) and satellite router 3 (`sat3`) change relative positions.

After around 180s, a handover occurs, changing the logical positions of **sat1** and **sat3**. By design, the controller was informed of the impending change in the virtual topology at that moment. Due to this change in the internal routing topology, routes were adjusted accordingly. The physical setup then changed 186s after the start of the experiment. This 6s offset was added to provide enough time to prepare the handover. No significant packet losses were observed. Routes were adjusted according to the topology change: **sat1** was correctly identified as the node connected to **acm3**. Therefore, this test was passed, as the routing logic was updated correctly for both ESL as well as ISL handover events.

QoS-based Load-balancing

To evaluate QoS-based load-balancing, a setup consisting of a high-priority and two low-priority flows was tested. The core idea was to observe adaptive adjustments with multiple simultaneous transmissions.

To this end, a high-priority saturating flow was generated, which followed a link used by two low-priority flows. Load-balancing mechanisms should re-direct the best effort traffic accordingly. The network topology is similar to the setup shown in Figure 3.18, but there are two new traffic generators **acm0** and **acm4**.

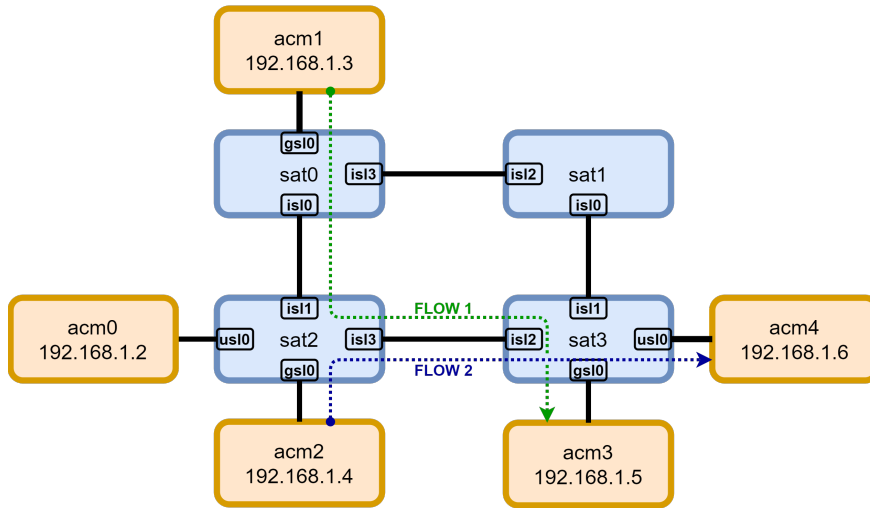


Figure 3.19.: Test setup for QoS-based load-balancing: initial flow state. Two low-priority flows are routed through the network. Flow 1 from **acm1** to **acm3**; flow 2 from **acm2** to **acm4**.

As shown in Figure 3.19, a flow from **acm1** to **acm3** was established. This flow is called flow 1. In addition, there was a similar flow from **acm2** to **acm4**, called flow 2. Figure 3.19 highlights that both flows used the ISL between **sat2** and **sat3**. Since a link load threshold of 40 % was used for this test, the utilization on this link did not trigger load-balancing.

After some time, a third flow was added as visualized in Figure 3.20. It was of high, delay-critical priority, and therefore not subject to load-balancing. Due to its significant demand of approximately 98 % of the link capacity, a bottleneck was created. While a short spike of dropped packets was observed, the controller re-routed flows 2 and 3 as expected. Alternative

3. Distributed Load-balanced Software-Defined Networking

paths were used avoiding the saturated link. Both low-priority flows now reach **sat3** via the link from **sat1**.

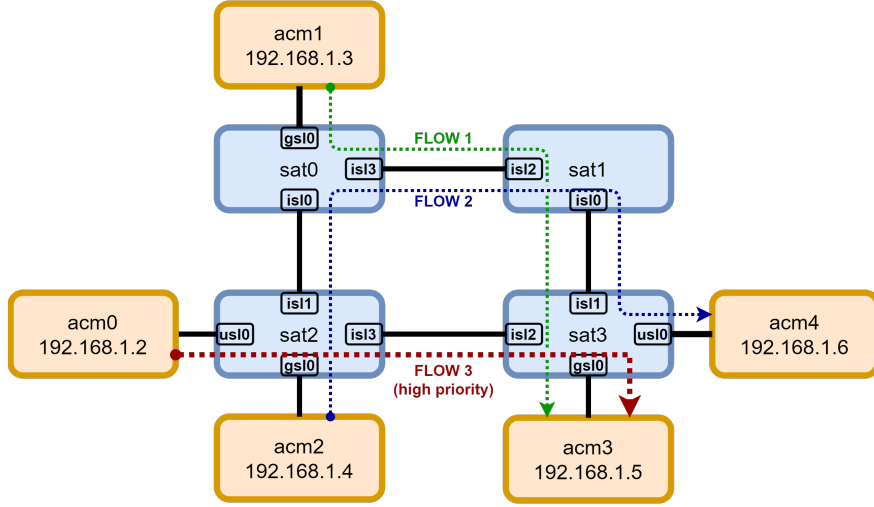


Figure 3.20.: Test setup for QoS-based load-balancing: adjusted routes after saturating flow. Two low-priority flows were re-routed to avoid saturated link between **sat2** and **sat3**, which was then used by high-priority flow 3.

Figure 3.21 shows the observed loads received at **sat2** and **sat3** over time. On **sat2** (left plot), a constant influx of approximately 0.199 Gbps on port **gsl0** was measured, corresponding to the incoming flow 2. Moreover, flow 1 of the same demand was received on port **isl1**, corresponding to the expected path shown in Figure 3.19.

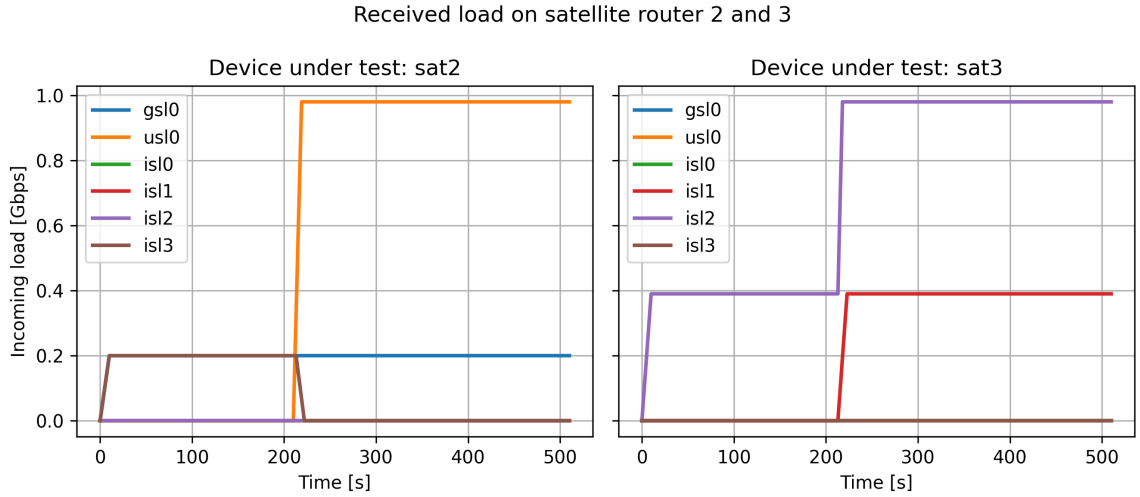


Figure 3.21.: Received load during QoS-based load-balancing test on **sat2** and **sat3**. Device under test **sat2** (left plot) and **sat3** (right plot) are shown.

At around 210 s, the saturating flow was observed on port **usl0** of approximately 0.984 Gbps. Shortly after (time granularity of this plot is 1 s), traffic from flow 1 stopped as it had been re-routed. On **sat3** (right plot), corresponding changes occurred. Initially, the combined

flows 1 and 2 were received on port `is12`. Starting at 211 s, these low-priority flows were received on `is11`. Incoming flow 3 was observed on port `is12` as expected.

The test was passed, demonstrating the QoS-based load-balancing capacities in the hardware test bed. Flows were adjusted correctly according to their QoS classes to avoid link saturation. The controller was able to react to the emerging bottleneck in a timely manner, highlighting this advantage of the in-space protocol design. These results demonstrate the effectiveness of IDLB in ensuring reliable and efficient data transmission in satellite networks.

3.5. Key Findings and Discussion

This discussion summarizes the key aspects and evaluations of the proposed load-balanced distributed SDN-based routing protocol, designed to enhance the performance of autonomous SCNs. The design of IDLB combines ideas from related routing approaches to achieve attractive trade-offs.

Protocol Design

The proposed protocol adopts conceptually similar link-state updates to those of IS-IS. However, its architecture and design choices are tailored to meet the specific needs of dynamic SCNs. In addition, the hierarchical structure, comprising inter- and intra-cluster schemes, is reminiscent of the Border Gateway Protocol (BGP) and Interior Gateway Protocol (IGP) split. However, BGP incorporates a range of features which are not necessary or adaptable for routing within satellite networks. Consequently, a protocol designed for autonomous constellations needs to be streamlined to optimize performance in this specific environment.

The distributed SDN paradigm is also generally more flexible and adaptable given its softwarization of routing logic and fine-grained network control, enabling arbitrary flow allocations [76]. For SCNs evolving in size and alongside standards such as 6G, this flexibility is a key enabler [15, 19]. Standard IP routing protocols are more difficult to override, and each device needs to be upgraded. The comparatively lightweight IDLB protocol is tailored to SCNs and designed to be adjustable to diverse routing policies and extensions.

Based on the characteristics of the reference scenario, the concrete design of the IDLB architecture is derived. To this end, key design choices are outlined, such as the sizing and arrangement of SDN clusters or geographical areas. Notably, crucial elements ensuring correct and stable functionality are clarified, namely the handover management and routing table setup. In addition, a suitable multicast extension is presented which aligns with this architecture. These descriptions provide an insight into implementation details, highlighting the actual feasibility of the approach. Building on these design choices, dedicated routing schemes are proposed for efficient and QoS-compliant forwarding within and between clusters.

Routing Schemes

The initially investigated geographical and set-based source-routing approaches are characterized by distinct advantages and disadvantages. While both schemes are viable for real-world systems, they fall short in fully resolving the research problem at hand. Despite the employment of basic load-balancing techniques, their designs limit overall effectiveness and leave little room for improvement. Consequently, they are unable to reliably achieve near-optimal performance which balances high system throughput with stringent QoS requirements. Nevertheless, both approaches provide valuable insights and heuristic strategies which influenced the design of IDLB.

In more detail, the specific characteristics of these methods are as follows. On the one hand, the use of geographical addressing and forwarding offers significant reductions in signaling overhead. But, the load-balancing capabilities of decentralized geographical routing are limited. Still, a similar approach is viable on an inter-cluster level, where minimal signaling is more significant. Set-based source-routing, on the other hand, introduces proactive traffic distribution among viable paths in SCNs. As the grid ISL network of satellite constellations results in multiple routes of similar characteristics, the effective increase in supported network load by choosing from sets of paths is significant. For the reference scenario, a 73 % higher traffic volume can be supported compared to an SPF scheme. Similar concepts are thus used in intra-domain routing for proactive congestion mitigation.

Accordingly, for intra-cluster routing in IDLB, a tailored dynamic best-of-k-paths algorithm was proposed. This method makes use of proactive load distribution and can be configured to reflect the requirements of the QoS classes. Given its low complexity, the scheme is expected to be feasible for on-board processing. By configuring suitable cluster sizes, fast routing convergence in response to handovers or failures is achieved. For inter-cluster routing, geographical information is used to find suitable links between clusters. The proposed QoS-based approach results in reduced signaling overhead, while avoiding potential bottlenecks at the links between SDN domains.

Performance Analysis and Limitations

Using a system-level simulator, in-depth performance analyses of the proposed protocol were provided. The QoS compliance of IDLB was investigated under increasing network loads to evaluate its load-balancing capacities, which significantly outperformed a source-routing benchmark.

The protocol was able to load-balance effectively, enabled by its low-complexity intra-cluster routing scheme. The resulting link utilization of ISLs, shown in Figure 3.22, demonstrates the effective use of available network resources (in particular compared to Figure 3.6). The link loads are more balanced among the ISLs, while complying with QoS requirements.

Additionally, various design aspects were investigated in isolation to show their impact on QoS compliance. Insights into the performance across different cluster sizes illustrate the load-balancing potential within clusters. A comparison between per-flow and per-packet-based routing highlighted the advantages of flow-based SDN decision-making. The benefits of flow-specific inter-cluster choices over fixed link strategies were also quantified, showing that load-aware routing is also necessary between domains. The performance of IDLB in SCN-

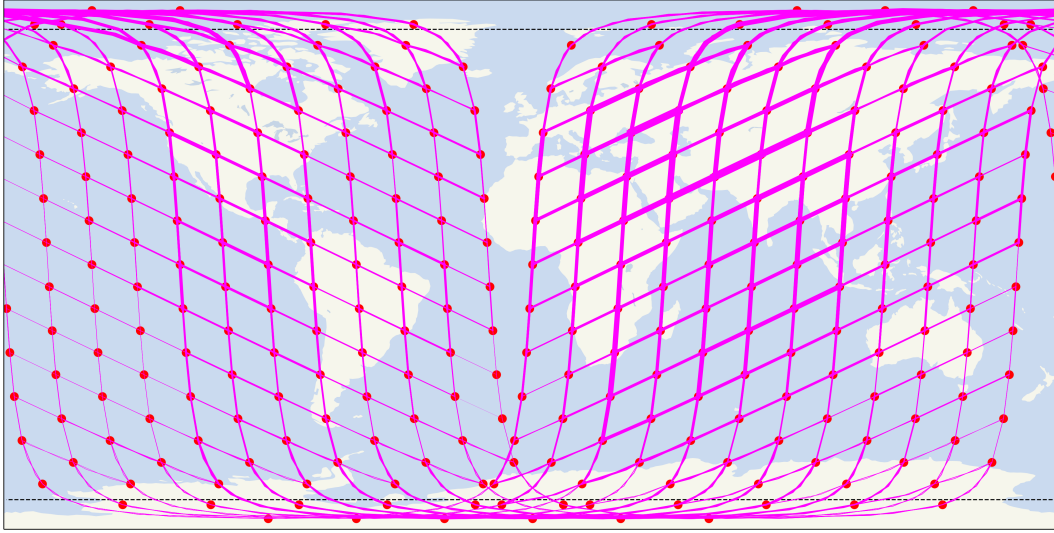


Figure 3.22.: Observed link utilization of ISLs of a snapshot using IDLB with 24-node clusters. Width of ISLs (magenta) scaled according to link utilization.

1440 demonstrated its capacity to scale to dense satellite networks, a critical requirement for next-generation mega-constellations.

Impact of Reference System

The evaluations also provided insights into how the reference system impacts performance. Some of the KPIs have been assessed in different scenarios directly, for others general trends were observed. The most apparent impacts of reference system parameters are summarized in Table 3.6.

Additional simulative evaluations can be considered for specific scenario parameters. The impact of different reference constellations depends on their characteristics. In general, simulations with other Walker constellations with comparable numbers of satellites provided similar results. The effectiveness of considered protocol design is by no means limited to the investigated scenarios.

Demonstration Results and Limitations

As relevant hardware was used in the test bed, TRL 5 was achieved. So, the demonstrated suitability to on-board processing capabilities underlines the applicability for real-world satellite systems. While the setup with four DUTs, which represent satellite nodes, is limited in its scale, relevant tests were nevertheless possible. Three key tests were of particular interest: quantifying processing limitations, validating dynamic handover adjustments, and demonstrating QoS-based load-balancing. All tests were passed by the implemented protocol.

Notably, the test setups only investigated intra-cluster routing procedures. However, given the general viability of the protocol in the test bed, inter-cluster capabilities should cause

3. Distributed Load-balanced Software-Defined Networking

Table 3.6.: Overview of impact of reference system parameters on individual KPIs.

Reference parameter	Impacted KPIs	Observed impact	Comments
Number of satellites	Latency	medium	Primarily influenced by physical distances and chosen paths, but also by multiple queuing delays.
	Jitter	low	More frequent handover events, depending on handover mechanisms.
	Packet dropping rate	high	Fewer drops due to more networking resources: 64.5 % more load supported due to fewer drops in SCN-1440 for same physical cluster dimensions (Section 3.3.2).
	Link utilization	low – high	Load-balancing approaches required to make use of additional network capacity.
Number of UTs	Latency	low	Limited variation if same distribution.
	Packet dropping rate	low	Few UTs: more focused bottlenecks.
	Link utilization	low	Few UTs: elevated near last-hop ISLs.
UT distribution	Latency	high	Variations according to distribution, resulting end-to-end paths.
	Packet dropping rate	high	Uniform: 64.5 % more load supported due to fewer drops for load-agnostic source-routing (Section 3.1.3).
	Link utilization	high	Inherently more diverse paths, improved link distribution.

no issues. While any application in real-world systems is expected to imply various changes, the protocol is considered a valid baseline as shown by the demonstrations.

Further Considerations

Overall, the evaluation demonstrates the effectiveness and scalability of IDLB in supporting QoS-compliant routing in satellite systems. While the presented analyses focus on the Walker star reference constellations, the protocol design is also valid for different topologies. The hierarchical design of the protocol is extensible to multi-orbit, multi-layer systems. Naturally, the impact of cluster sizing and other design choices may vary depending on the architecture and orbital layer. For other constellations, adjustments to handover policies and cluster boundaries may be required to align the protocol. In order to seamlessly integrate multiple constellation shells, the handling of inter-layer links and the corresponding inter-cluster routing are of particular interest. Comparative studies of load-balancing strategies across diverse systems would also further validate the versatility of the proposed solution. For

SDN hierarchies, it is possible to establish primary SDN controllers on ground, with the in-space controllers acting as secondary control units. In particular, in the context of 6G NTN integration and future constellation designs, extended IDLB schemes are thus promising options for autonomous next-generation satellite networks.

Looking at the reference scenarios at hand, significant performance improvements have been achieved. Still, the proposed low-complexity intra-cluster routing algorithm does not explicitly maximize throughput. In order to find approximate solutions to the MCFP, a more direct focus on how to optimize flow management within a cluster is required. While the impact of inter-cluster routing on the overall throughput is also significant, most load-balancing decision-making is handled by intra-cluster routing. Given the fine-grained routing and switching capabilities enabled by SDN, more complex flow distributions and splits are possible. However, designing efficient and adaptive algorithms which are capable of establishing near-optimal routing configurations is difficult. Formulating feasible LP and DP problems requires extensive heuristics, as mentioned in Section 2.3.2 and in Section 3.2.1.

In order to avoid design choices relying on specific system characteristics, the application of data-driven methods is proposed. Since SDN is a promising enabler of ML-enhanced traffic engineering [87], IDLB represents a well-suited, extensible baseline for ML. Most notably, since a network control problem is being investigated, DRL techniques are of particular interest. So, in the following chapter, it is explored how DRL can be applied to enhance the performance of intra-cluster routing of IDLB.

4. Reinforcement Learning-based Traffic Engineering

A promising extension to enhance the performance of the proposed distributed SDN-based protocol is the application of DRL. Given the on-board limitations of satellites and the inherent complexity of traffic engineering, such data-driven methods can be trained to maximize throughput. So, in this chapter, suitable DRL-based routing approaches and extensions are presented and evaluated. As the application of ML methods over conventional approaches is advisable for system optimization only in the presence of model or algorithmic deficits [29], the core use case of enhancing traffic engineering is identified first.

The considered principal policy is thus to apply ML techniques on specific routing problems while leveraging existing well-suited models and algorithms. Therefore, instead of applying a black-box solution to the research problem at hand, the viability of DRL is systematically derived from first principles. Based on this mathematical background, promising DRL architectures suitable for the system requirements are investigated. This involves exploring hop-based and path-based DRL approaches for flow allocation decisions.

For hop-based routing, DQNs for solving multi-objective functions for finding minimum cost paths are investigated. To this end, two relevant architectures, a centralized one and a fully-decentralized one, are considered. Since correct paths without loops or deadlocks are non-trivial to find in a hop-based context, corresponding adjustments are proposed. Thus, constraint-based centralized and decentralized multi-agent DQN solutions are analyzed.

In addition, a path-based approach which relies on the actor-critic method is proposed. The principal reasoning is to learn the underlying MCFP directly by providing a set of paths and their characteristics. Consequently, the training of the model focuses solely on maximizing throughput, without the need to learn how viable paths are formulated. The method is also compared to various benchmark techniques.

The chapter concludes with a discussion of the investigated approaches, evaluating their viability for the problem at hand while highlighting drawbacks and limitations for the reference scenario.

4.1. Machine Learning for Traffic Engineering Enhancements

When improving the performance of the distributed routing protocols discussed in Chapter 3, the principal focus was the mitigation of packet drops by formulating effective and adaptive load-balancing strategies (see Section 3.2.1). To optimize the traffic distribution, TE

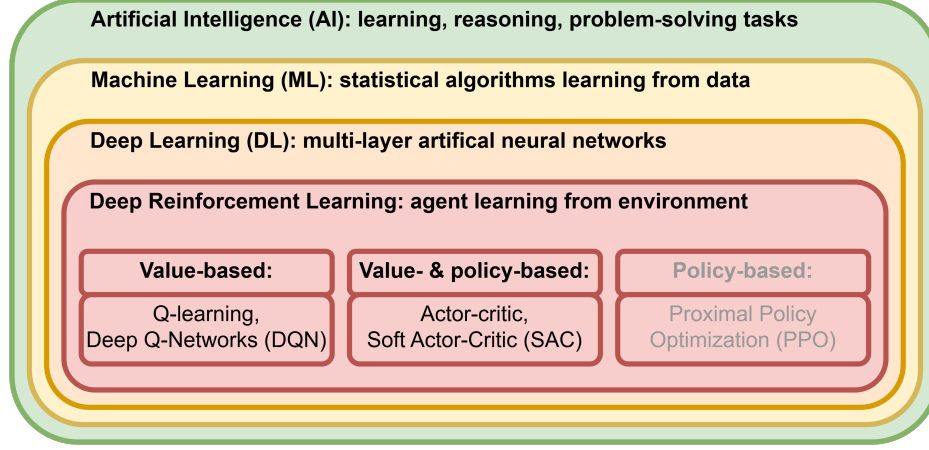


Figure 4.1.: Overview of classification of DRL algorithms investigated in this work. Policy-based DRL methods grayed out, as they are not considered.

approaches are applied. This means that flows are actively steered to achieve routing configurations enabling high throughput.

As described in Section 2.3, the underlying problem can be formulated as a dynamic MCFP. However, solving the complete dynamic problem using Stochastic Dynamic Programming (SDP) is computationally complex. When using snapshot-based approaches to solve quasi-static MCFPs, the temporal dynamics of the problem must be considered. While computing solutions in every time step or for every new flow is already complex, it is also difficult to estimate the impact of routing decisions. Even with advanced hardware on-board the satellites, approximate techniques have to be considered for real-world systems.

Given the ability to learn underlying traffic patterns to make suitable predictions, ML techniques are of significant interest for traffic engineering. While ML solutions have proven powerful in various domains, their real-world applicability can be debatable, in particular if well-functioning rule-based solutions are available. As mentioned, the application of ML algorithms is of interest **only if there is an algorithmic or model deficit** [29]. This is the case in this context, as the underlying problem of finding the best routing configuration is NP-hard [32]. Using ML, effective policies which adapt to the current and changing network conditions can be learned. Since the approach is trained on network data directly, no direct assumptions about traffic characteristics are made. There is no explicit model which has to be formulated to represent the state evolution of the dynamic network and its resources. Besides optimizing the routing configuration, the adaptability and robustness demonstrated by ML-based schemes are also attractive given the scenario.

An overview of the hierarchical relationship between AI, ML, DL, and DRL is provided in Figure 4.1 [28,127]. In general, ML involves training algorithms to learn from data, enabling them to make predictions or decisions accordingly. A key component in learning complex tasks is the application of DNNs, upon which DL is based [128]. Typically, DNNs rely on Multilayer Perceptrons (MLPs) with multiple hidden layers or similar architectures [28]. An in-depth derivation and formulation of the general deep learning framework is described in Appendix C.

In short, during training, the loss between ground truth (the known correct outcome) and estimation is propagated back through the neural network as a gradient to adjust its weights via an optimization algorithm, such as stochastic gradient descent. This is called backpropagation (see Appendix C). An approximation of complex functions, even if no analytical formulation is possible, can be achieved. Conceptually, the topology of DNNs enables the learning of lower-dimensional manifold representations, that means geometric structures, capturing complex, non-linear relationships in data, thereby effectively modeling patterns [31]. The training process is terminated, either by achieving a sufficiently low loss value or by reaching a predefined maximum number of training iterations (epochs).

If low loss has been achieved, it indicates that the model can appropriately approximate the underlying function. For every input, an output is computed by passing through the adjusted weights of the model. Such forward pass through the network, without backpropagation, is called inference. Notably, research suggests that in high-dimensional spaces, models are mostly performing some form of extrapolation during inference [33]. So, well-trained models are generally considered to robustly handle inputs which were not part of the training data set [129].

4.1.1. Viability for Satellite Constellation Networks

In order to provide context on the viability of DL approaches for SCNs, this section considers their compliance with the requirements. First and foremost, the function approximation capabilities are of interest in the context of the described NP-hard problem at hand. However, ML-based solutions have to comply with the specific requirements posed by the satellite network, such as on-board limitations and dynamic topology changes.

Training and Inference

The traffic engineering capacities are expected to be integrated on-board the satellites into the IDLB architecture. Thus, feasible solutions must not add significant computational complexity. For ML techniques, it is necessary to differentiate the complexity in time during training and during execution. Ground-based training is assumed to be feasible, and only on-board execution is considered. Consequently, the complexity of inference is more relevant for this analysis. A forward pass through the network corresponds to an element-wise matrix multiplication in each layer, with additional operations, e.g., activation functions. For a fully connected network of $I = |\mathcal{L}|$ layers with N_n^ι neurons for each layer ι , the complexity can be formulated as:

$$\mathcal{TC}_{\text{inference}} = O\left(\sum_{\iota \in \mathcal{L}} N_n^{\iota-1} N_n^\iota\right) \quad (4.1)$$

If all layers are of the same size with N_n neurons, the complexity in time is described by:

$$\mathcal{TC}_{\text{inference}} = O(|\mathcal{L}| N_n^2) \quad (4.2)$$

Moreover, processing algorithms are typically applied which cause additional complexity. The impact of these algorithms is provided when investigating the overall complexity of the proposed methods.

4. Reinforcement Learning-based Traffic Engineering

The training process requires significantly more computational resources and is expected to be performed on ground. It consists of forward and backward passes through the network, which is accounted for by a factor of two. Depending on the problem, thousands or even millions of iterations (N_{its}) are required for a model to converge (see Section 4.3.3). For N_{eps} denoting the total number of epochs, and $N_{\text{it, ep}}$ denoting the iterations per epoch, this results in a complexity of:

$$\mathcal{TC}_{\text{training}} = N_{\text{its}} (\mathcal{TC}_{\text{inference}} + \mathcal{TC}_{\text{backward}}) = O\left(N_{\text{eps}} \cdot N_{\text{it, ep}} \cdot 2 \cdot \sum_{\iota \in \mathcal{L}} N_n^{\iota-1} N_n^{\iota}\right) \quad (4.3)$$

Given the described typical scaling of N_{its} , the processing power required in training is significant. Therefore, to ensure viability for satellite systems, training on ground has to be assumed. But, this raises the question of how trained models are distributed, and where learning and control entities are placed.

Placement of Learning Entities

Relying on learning on ground implies periodic model updates if the problem is non-stationary. Underlying trends in the traffic patterns may change over time. Thus, to maintain high accuracy, regular updates can be necessary. To this end, an exchange of data and trained models is necessary between the NTN and the training entities on ground, e.g. Network Control Centers (NCCs). While this results in additional signaling overhead, the expected time frame is in a non-critical domain. Trained models should remain sufficiently accurate for extended periods on the order of days.

Inversely, the distribution of control entities using the models is also of interest. In the context of IDLB, trained models are integrated into the software on the SDN controller. However, architectures which provide each satellite node with an individual model are also of interest. For this reason, both centralized and fully-decentralized setups, within an SDN domain, are investigated. In this context, the intended DRL solutions are based on suitably large models with feasible complexity.

Autonomous Network Control and Traffic Engineering

As illustrated by the results of IDLB (in Section 3.3), a proactive usage of the available network resources is essential for efficient load-balancing. Consequently, optimal approaches should be able to effectively predict and adapt to the real-world requirements. The emerging traffic patterns and their dynamics are a key aspect of optimizing routing in the concrete reference scenarios. Consequently, ML techniques for traffic prediction are common approaches to improve performance.

The key idea is to anticipate traffic patterns in order to enhance a rule-based routing algorithm. To this end, a **supervised learning** approach is typically applied which directly learns on a labeled data set representing real-world conditions. For sequential data, RNNs or LSTMs approaches can be applied [90]. For graph-structured data, GNNs are of interest [92].

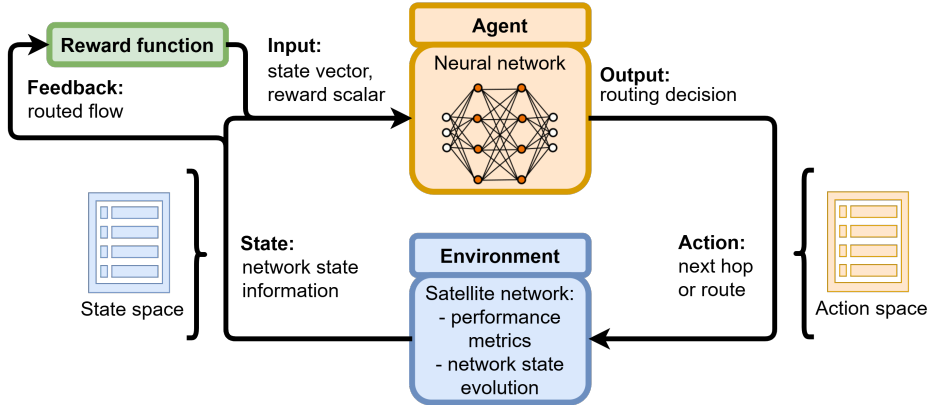


Figure 4.2.: Generic DRL-based architecture for routing in satellite networks. Depending on approach, action is either next hop or route.

However, modeling and predicting broadband Internet traffic is challenging even for ML approaches [89]. As there remains significant uncertainty in this context, even an efficient solution may be discarded due to disputable assumptions. The performance of the approach depends on the accuracy and size of the data set. Given that no high-fidelity data set of traffic patterns for SCNs is available, training data has to be synthesized using simulative environments. Moreover, when solely relying on traffic prediction, algorithms which effectively distribute flows accordingly are required. If links expected to be of high load are avoided proactively, viable resources may be underused.

Therefore, a more holistic approach which incorporates expected future patterns into the decision-making process is typically better suited for network control. Using **reinforcement learning**, an agent adapts to unpredictable changes in traffic patterns, and learns from experience [27]. Relying on historical data is advantageous, but not necessary – optimal routing strategies are identified through self-learning and exploration [85, 87]. Depending on the definition of the state and action spaces as well as the environment, different DRL architectures are of interest. The generic DRL-based approach to routing in SCNs is shown in Figure 4.2. The inputs of the DNN are shown: the relevant state information and reward. The outputs are also illustrated, i.e., the actions corresponding to routing decisions.

The derivation for the optimization problem as well as promising candidate solutions are discussed in the subsequent sections of this chapter. In Table 4.1, the considered approaches are summarized. Why DRL is formally well-suited for the problem at hand is derived in the following section. The mathematical framework is systematically derived to provide a theoretical basis for the formulation of suitable environments, state spaces, action spaces, and rewards required to enable efficient routing.

4.1.2. Derivation from First Principles

A fundamental aspect of the MCFP optimization problem is the fact that choices impact subsequent choices. So, a modeling of state transitions is required for its description. Models which are able to optimize the policy which influences state evolution are needed. Moreover, the considered state space is typically large with hard to predict, stochastic traffic demands.

Table 4.1.: Summary of investigated DRL approaches.

Actions	Architecture	DRL technique	Core concept
Hop-based	Centralized controller (Section 4.2.1)	Constrained Deep Q-Network (DQN)	Controller decides path on hop-to-hop basis. Infers best valid next hop until destination.
	Fully-decentralized (Section 4.2.2)	Deep Q-Networks (DQNs)	Each node decides best valid next hop, using local information in decision-making.
Path-based	Centralized controller (Section 4.3.2)	Soft actor-critic (SAC)	Controller distributes incoming flows on sets of pre-computed candidate paths.

The dynamic NTN characteristics pose additional constraints (see Section 2.1.5). The considered solutions have to provide distribution, adaptability, as well as resilience.

To capture state evolution, Markovian models are well-suited. However, Markov chain models are not applicable due to the overlap of sub-problems. A more general formulation, in the form of an MDPs has to be used. Therefore, based on the principal requirement of capturing the state evolution of the optimization problem, the MDP is formulated accordingly.

Markov Decision Processes

In general, an MDP can be formulated by its state, action, transition probability, and reward. At first, the transition between states is examined. The transition probability from state s to the following state s' by taking action a and getting reward r is:

$$P(s', r | s, a) = Pr(s_t = s', r_t = r | s_{t-1} = s, a_{t-1} = a) \quad (4.4)$$

Using time step t , the transition from the previous state s_{t-1} to the current state s_t occurs. If all transition probabilities are known, an MDP can be solved: for every state, the action yielding the best subsequent reward can be chosen. However, for most practical problems this is not the case. In the context of routing for SCNs, the transition probabilities are not known. To the network control unit, future incoming flows are unknown. Choosing the best actions in terms of flow allocation corresponds to the problem described by the MCFP.

In this mathematical framework, the optimal solution provides the highest possible reward. Thus, the principal goal of an agent is maximizing its cumulative gain, R_t , over time by making effective subsequent decisions. The cumulative gain can be described by the expected reward of the next action, as well as the discounted expected rewards of subsequent decisions. Since the estimation of subsequent rewards becomes increasingly difficult, the farther the planning horizon is extended, a discount factor γ is employed to decrease the impact exponentially. Hence, R_t can be formulated as follows:

$$R_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \dots = \sum_{i=0}^T \gamma^i r_{t+i+1} = r_{t+1} + \gamma R_{t+1} \quad (4.5)$$

This formalization reflects the core difficulty of the MCFP: making decisions which do not negatively impact later states. In each state, the long term effects are estimated in order to approximate an ideal routing configuration.

Policy and Q-Functions

Based on the cumulative reward, which in our case is designed to represent the load-balancing efficiency, it is possible to formulate policy and value functions. To evaluate a policy, i.e. a mapping of states and actions, the gradient of the expected cumulative reward for a policy π is examined. A step size for the gradient is used: the learning rate α . The resulting function is called the policy gradient θ :

$$\theta = \alpha \nabla_{\theta} \mathbb{E}_{\pi}[R_t] = \Delta \alpha \nabla_{\theta} \mathcal{J}(\pi_{\theta}) \quad (4.6)$$

For the estimation of state-action pairs, the Q-function is used. It represents the expected cumulative reward, given action a in state s . For the policy π , the Q-function is thus described by:

$$Q_{\pi}(s, a) = \mathbb{E}_{\pi}[R_t | s_t = s, a_t = a] = \mathbb{E}_{\pi}\left[\sum_{i=0}^T \gamma^i r_{t+i+1} | s_t = s, a_t = a\right] \quad (4.7)$$

The Q-function directly relates to the state-value function $V_{\pi}(s)$. For a given policy, the functions are interchangeable. The value of a state corresponds to the sum of Q-values for all actions, which are weighted by the policy:

$$\sum_a \pi(a|s) = 1 \Rightarrow V_{\pi}(s) = \sum_a \pi(a|s) \cdot Q_{\pi}(s, a) \quad (4.8)$$

Similar to dynamic programming, a generalized policy iteration can be applied to these functions for optimization. The optimization is driven by two interacting processes, namely the approximation of the policy function, and the approximation of the value or action-value function. Since R_t considers immediate and discounted future rewards, it is possible to decompose the state-action function accordingly. Thus, the problem is split into simpler, recursive sub-problems which enables easier computation. Using the Bellman equation [27], the state-action function can thus be described by:

$$Q_{\pi}(s, a) = \sum_{s'} P_{s,s'}^a (r(s, a) + \gamma V_{\pi}(s')) = \sum_{s'} P_{s,s'}^a (r(s, a) + \gamma \sum_{a'} \pi(a'|s') \cdot Q_{\pi}(s', a')) \quad (4.9)$$

This formalization is also applicable in the routing context. However, the transition probabilities and rewards are unknown a priori for the considered routing problems. Recall the value function for the MCFP in the context of an SDP formulation, with the immediate cost function C_t for flow f_t :

$$V_t(s_t) = \min_{f_t} (C_t(s_t, f_t) + \mathbb{E}_{\omega_t} [V_{t+1}(s_{t+1}) | s_t, f_t]) \quad (4.10)$$

Given the stochastic incoming traffic, described by ω_t , estimating values of long-term effects is difficult. Therefore, models must learn from underlying patterns in the data how to evaluate states and actions. The exploration of the state-action space is required to make informed decisions.

Q-learning

To enable a value-based estimation from input data, Q-learning is the logical extension of the provided formalization. In order to find the best solution, the optimal state-action function Q^* is considered. In general, this describes the maximum expected return over all policies.

$$Q^*(s, a) = \max_{\pi} Q_{\pi}(s, a) \quad (4.11)$$

Thus, the optimal value function, V^* , for the optimal policy is described by:

$$V^*(s) = \max_a Q^*(s, a) \quad (4.12)$$

The Bellman equation of optimality defines the optimal Q-function as:

$$Q^*(s, a) = \sum_{s'} P_{s,s'}^a (r(s, a) + \gamma \max_{a'} Q^*(s', a')) \quad (4.13)$$

Now, by calculating the temporal difference error [27], the update rule for Q-learning is derived. The principal idea is to adjust Q-values using observed rewards, iteratively improving the approximation. For a given moment in time t , the Q-value is computed and combined with the expected best current reward and expected subsequent Q-value. As before, a learning rate α is used to limit the step size for smoother convergence. The subtraction of $Q(s_t, a_t)$, so the current Q-value, removes the impact of the current estimate, allowing us to update the Q-value based solely on the difference between the expected and observed outcomes. This helps to prevent overestimation or underestimation of the true value. The Q-learning update rule is thus:

$$Q^{new}(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha (r_t + \gamma \max_{a'} Q(s_{t+1}, a') - Q(s_t, a_t)) \quad (4.14)$$

Q-routing

Based on this Q-learning function, a corresponding approach for packet routing, called Q-routing, can be derived directly [130, 131]. The approach follows a hop-by-hop decision-making process, which focuses on the queuing delay l_q and the propagation delay l_p . On each node, a link to a potential next node is considered. So, the next hop choice v_n can be interpreted as the action, while the current node v_c represents the state. For node v_c , there is a set of neighbors $\mathcal{N}_c$. With each hop, states and actions change: each hop can thus be seen as a time step.

The core idea is to evaluate and estimate the viability of a potential next hop j , with regard to the destination node v_{dst} . So, on node i , the applicable queuing delay, $l_q(i, j)$, propagation delay, $l_p(i, j)$, and estimated remaining latency to the destination node, $L(v_{dst}, j)$, are computed. The latter is an estimation based on Q-values of the subsequent neighbors v_n of potential next hop j [130]:

$$L(v_{dst}, j) = \min_{v_n \in \mathcal{N}_j \setminus i} Q_j(v_{dst}, v_n) \quad (4.15)$$

Here, $\mathcal{N}_j$ describes the set of next hop neighbors for j . Since the packet comes from i , this node is typically not considered; the set $\mathcal{N}_j \setminus i$ is assumed. The resulting update rule is described by [130]:

$$Q_i^{new}(v_{dst}, j) \leftarrow Q_i(v_{dst}, j) + \alpha \left(\underbrace{l_q(i, j) + l_p(i, j) + L(v_{dst}, j)}_{\text{new estimate}} - \underbrace{Q_i(v_{dst}, j)}_{\text{old estimate}} \right) \quad (4.16)$$

Note that due to the hop-by-hop formulation, the next Q-values (as shown in Equation (4.14)) are not discounted, so $\gamma = 1$.

As finding a Q-function which appropriately represents the problem is difficult, these approaches are based on exhaustive exploration. Naturally, this is infeasible for large, high-dimensional state spaces. However, by applying a DNN, it is possible to approximate Q-tables effectively. So, the function approximation capacities of DNNs are specifically used for the estimation of a large set of Q-values. The corresponding method of approximating Q-values using DNNs is called DQN.

Deep Q-Networks

The previous formulation of Q-routing (4.14) is adapted to include the model parameters θ of the DNN: $Q(s, a; \theta)$. To improve convergence and stability during training, DQNs incorporate additional design aspects:

- **Experience replay:** improves sample efficiency and stability by storing experiences (s, a, r, s') in a replay buffer D . During training, an efficient sampling of mini-batches is enabled [132].
- **Target networks:** by using a secondary DNN, $\bar{\theta}$, robustness against over-estimations and noise is increased. The target network is periodically synchronized with the primary network [133].

Stochastic gradient descent is used to train the DNNs (see Appendix C). The temporal difference error in Equation (4.14) is multiplied by the gradient $\nabla_{\theta} Q(s_t, a_t; \theta)$ with respect to the model parameters θ . The update rule for the parameters of the primary network becomes:

$$\theta \leftarrow \theta - \alpha \nabla_{\theta} Q(s_t, a_t; \theta) (r_t + \gamma \max_{a'} Q(s_{t+1}, a'; \bar{\theta}) - Q(s_t, a_t; \theta)) \quad (4.17)$$

With this update rule, the foundation for learning routing solutions using deep learning is established. DNNs approximate a function representing state-action values, allowing the model to be trained and an adequately accurate Q-function to be learned. Through training, the model can thus be used to derive routing solutions from observed data. The complex state transitions that make the MCFP challenging in SCNs are addressed by providing estimates of their values through the learned Q-table. As a result, decisions can be made directly based on these estimations, facilitating integrated routing decision-making.

In the following, DQN-based approaches are applied and evaluated for routing in SCNs. Based on the mathematical framework, the challenge is to formulate suitable environments, state spaces, action spaces, and rewards so that efficient routing is enabled. Overall, the actual performance of ML-based routing approaches crucially depends on the chosen DRL architecture.

Implicit Assumptions

As a side note, there are various implicit assumptions in the intended derivation. Model-free methods, which do not use an explicit model of the environment, are the focus. Learning is based on direct, raw experience gained by interacting with the environment. Therefore, they are better suited for environments which are complex or poorly understood. Such approaches typically require more data to learn efficiently [27].

Since the problem at hand is difficult to formalize, there is no available model which represents it in all its relevant aspects. As a correct modeling of NP-hard routing with a dynamic network topology is considered overly complex, model-free techniques are applied.

Moreover, only off-policy methods are considered in this section. This means the agent learns from data which was collected when using a different policy than the one being currently optimized. The main reason is sample-efficiency: off-policy methods approximate the Q-function, which is used to determine the best action, even if the actions taken by the agent do not follow the current policy. For routing in particular, this makes sense as learning from previous decisions is valuable regardless of the applied policy. Historical data is used more effectively, leading to better decision-making and adaptation in dynamic environments.

4.2. Deep Reinforcement Learning for Hop-based Routing

In this section, the viability of DRL-based routing schemes for SCNs is investigated, following the hop-by-hop path-finding strategy introduced by Q-routing. To this end, a constrained-centralized and a fully-decentralized solution are presented.

In the context of maximizing throughput, models must learn the multi-objective optimization problem of simultaneously discovering stable paths while balancing loads. Although adjustments to improve model convergence are proposed, such as constrained action spaces, both centralized and decentralized approaches have significant drawbacks. The key disadvantages are scalability and robustness, which are highlighted in this section. Although hybrid methods offer a potential compromise, learning to approximate the MCFP directly is difficult for hop-based methods. Based on these findings, alternative DRL formulations, such as path-based approaches, are thus explored. So, this section provides a precursor to path-based DRL-enabled routing, building upon the Q-routing framework derived earlier.

The presented research has been published in corresponding works [36, 38].

4.2.1. Centralized Constrained Deep Q-Networks

When using a Centralized DRL (CDRL)-based approach, solely shaping the reward function does not guarantee correct pathfinding and optimized load-balancing. In fact, the latter objective may introduce instability: depending on the formulation of rewards, an incentive to avoid overloaded links may result in a routing loop. To mitigate these issues, it is possible to alter the action space by masking unsafe actions during the action selection. Hence, the viable action space is constrained based on routing-specific insights. Only unsafe actions are filtered, and no changes to the Q-update step are made.

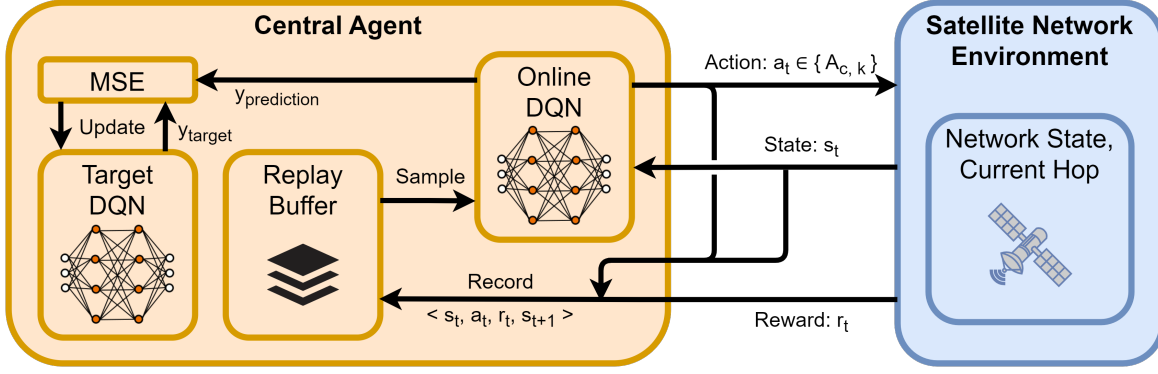


Figure 4.3.: Diagram of centralized constrained DQN architecture with satellite network environment. Action set constrained to correct routing paths.

To expand on these concepts, a constrained CDRL approach is proposed, called C-DQN, which adapts action and state spaces accordingly. As constraints are independent of the scaling of the immediate reward function, a safe policy is learned.

Centralized Architecture

A general centralized DQN architecture is illustrated in Figure 4.3. The approach relies on two DNNs, the training or online network represented by its parameters θ_c , which is used for action selection, and the target network represented by $\bar{\theta}_c$, which is used to compute stable target values for training. The online network is trained by minimizing the MSE between its predictions $y_{\text{prediction}}$ and the target values y_{target} generated using the target network. To maintain training stability, the parameters of the target network are periodically updated by copying or slowly tracking the parameters of the online network. This dual-network approach helps mitigate the moving target problem in Q-learning and improves convergence [133].

State: The state space $\mathcal{S}$ is based on the number of active satellite nodes $\mathcal{V}$ operating in the considered mesh topology. A rectangular cluster corresponding to the IDLB design is assumed. The connectivity between satellite nodes can be described by an adjacency matrix $\mathbf{A}$. The link utilization is represented similarly by a utilization matrix $\mathbf{U}$. Thus, on an individual hop i , the state may consist of the node information, the destination, v_{dst} , the relevant adjacency $\mathbf{A}[i]$, and the relevant utilization $\mathbf{U}[i]$. So, a potential state $s_i \in \mathcal{S}$ can be formulated as:

$$s_i = (i, v_{dst}, \mathbf{A}[i], \mathbf{U}[i]) \quad (4.18)$$

In order to constrain the potential next hops to avoid loops, rule-based pathfinding algorithms can be used. Using the k -shortest paths and k -simple path algorithms in the mesh network between source node, v_{src} , and destination node, v_{dst} , a set of valid, safe paths is computed. The set of safe states that operate when actions are chosen within a constrained action space is:

$$\mathcal{S}_{\text{con}} = \{a \in \mathcal{A}_{\text{con}} | (s_t, a)\} \quad (4.19)$$

Action: In an SCN topology, the connectivity to neighboring satellites changes dynamically,

4. Reinforcement Learning-based Traffic Engineering

which is difficult to reflect in an action space. The output layer of the DNN is generally static, so action spaces of variable lengths are hard to represent correctly. Therefore, a generic, finite action space for each satellite node corresponding to the directions of its four ISLs is used. As each satellite possesses four ISLs at maximum, the action space consists of four possible actions. In accordance with the ISL setup of the reference scenario the following directions are assumed: North, East, South, and West. So, for hop-based approaches in general, the action set corresponds to these potential directional ISLs:

$$\mathcal{A} = \{a_{\text{North}}, a_{\text{East}}, a_{\text{South}}, a_{\text{West}}\} \quad (4.20)$$

When a certain directional link is not present or is not active, its value is set to zero. Given that a constrained approach is used, the representation of infeasible options does not cause issues. The selection of such actions is mitigated a priori. The set of safe actions $\mathcal{A}_{\text{safe}}$ for a given state s_t can thus be described by:

$$\mathcal{A}_{\text{safe}}(s_t) = \{a \in \mathcal{A}_{\text{con}}(s_t) \mid a \in p \text{ with } p \in \mathcal{P}_{\text{safe}}\} \quad (4.21)$$

Here $\mathcal{P}_{\text{safe}}$ corresponds to the set of all safe routing paths for the given source and destination. As mentioned, it is possible to focus on k -shortest or k -simple paths. In both categories, C-DQN ensures sufficient exploration of the nodes in the mesh topology while focusing on valid paths.

Reward: The design of the reward function r_t is a crucial aspect in learning suitable policies. For C-DQN, it is formulated by considering the factors that lead to selection of an action at each hop [95]. The local reward function can be formulated according to a desired link utilization threshold, u_{th} , as follows:

$$r_t(a_t|s_t) = \begin{cases} \Xi & \text{shortest path with } \max_{(i,j) \in p} u(i,j) < u_{th} \\ \Xi - \xi_1 & \text{simple path with } \max_{(i,j) \in p} u(i,j) < u_{th} \\ \Xi - \sum_{(i,j) \in p} \xi_2 & \text{exceeding threshold with } u_{th} \leq u(i,j) \quad \forall (i,j) \in p \\ 0 & \text{any other case} \end{cases} \quad (4.22)$$

Here, Ξ denotes the highest reward which is received when a shortest path has been found without exceeding the link utilization threshold u_{th} . The formulation of a suitable simple path, which complies with the utilization constraint, results in a lower reward decreased by ξ_1 . The value of ξ_2 can also be scaled by the increase in distance compared to the shortest path. In addition, if the utilization threshold is exceeded on individual links, the reward is reduced accordingly by ξ_2 .

Complexity in Time

As described, inference is the focus of the complexity evaluation. For this constrained DQN-based approach, the previously described complexity in time, in Equation (4.1), is applicable on every hop. However, for each incoming flow, if constraints are computed, the complexity

of the k -shortest-paths algorithm has to be considered as well. For $|\mathcal{E}|$ edges, and $|\mathcal{V}_c|$ nodes within the assumed sub-network, this results in:

$$\mathcal{TC}_{\text{constraints}} = O(|\mathcal{E}| + |\mathcal{V}_c| \log |\mathcal{V}_c| + k) \quad (4.23)$$

Overall, for $|F|$ incoming flows, this results in a complexity of:

$$\mathcal{TC}_{\text{C-DQN}} = O\left(|F| \cdot (|\mathcal{E}| + |\mathcal{V}_c| \log |\mathcal{V}_c| + k + N_{\text{hops}} \sum_{\iota \in \mathcal{L}} N_n^{\iota-1} N_n^{\iota})\right) \quad (4.24)$$

The resulting complexity in time is comparable to the dynamic best-of-k algorithm. Moreover, given the shape of the reward, an improved proactive load-balancing capabilities are not expected. Similar to a rule-based multi-cost pathfinding algorithm, rewards are provided according to threshold criteria. Traffic is allocated based on the current network state, foresight in decision-making is not a direct objective of the optimization.

Overall, the presented C-DQN architecture is capable of computing safe routing paths by ensuring adequate node exploration and dynamic load-balancing in changing connectivity conditions [38]. The ability of this algorithm to explore safe states and actions renders it more stable in comparison to the conventional DQN based approach. However, the cost of this path stability is a significant increase in complexity. In addition, the agent passes multiple times through the network to find a single path: every hop represents a new state. To reduce this complexity, the decision-making can also be distributed among the networking devices. In such a fully-decentralized scheme, each satellite only focuses on its own forwarding, the resulting state space is much smaller.

4.2.2. Fully-Decentralized Multi-Agent Deep Q-Networks

The idea of Fully-Distributed Multi-Agent DRL (FD-MADRL) is that each satellite acts as an individual agent and makes decisions based on its local observations. However, a careful design and reward shaping are required to avoid loops and to find coherent paths. Learning coordinated behaviors can pose a challenge due to the complex interactions which can arise from the policies of individual agents.

Thus, the problem at hand is best described by a Decentralized Partially Observable MDP (Dec-POMDP). The state space is not only partially observable, the local observations also differ between agents. As each agent acts independently, the scheme is well-suited to dynamic environments as changes and disruptions can be handled locally. However, it can be difficult to converge towards effective routing policies, as the emergent behavior is only optimized indirectly. The end-to-end coherence and stability become questionable, the agents have to learn the behavior of other agents as well. Complex and unexpected interactions between agents can emerge.

Decentralized DQN Architecture

While CDRL is based on a singular agent, in FD-MADRL, each satellite is assumed to act as an independent agent with its own model and only partial knowledge of the enviro-

4. Reinforcement Learning-based Traffic Engineering

onment. However, the formulation of the DQN architecture is similar to the centralized approach.

It is assumed that, for FD-MADRL, information about the adjacent link loads $\mathbf{U}[i]$, the previous hop v_{pre} , and the destination v_{dst} , are of relevance. Since the complete state is no longer considered and individual observations are used instead, an observation of node i , denoted by o_i , is defined:

$$o_i = (v_{dst}, v_{pre}, \mathbf{U}[i]) \quad (4.25)$$

The action space encompasses the available ISLs, so the potential next hops. While the action space can be specific for each node, an identical one can be assumed for the reference scenario:

$$\mathcal{A} = \{a_{\text{North}}, a_{\text{East}}, a_{\text{South}}, a_{\text{West}}\} \quad (4.26)$$

Reward Shaping

Defining suitable rewards is imperative to enforce the intended optimization behavior, in particular for Multi-Agent DRL (MADRL). Composite rewards can be formulated, which split feedback into global and more immediate local rewards [103]. At time step t , the global reward of an agent can capture relevant end-to-end characteristics. Specifically, the flow-specific latency L_t , the aggregated link utilization U_t , and the resulting packet dropping rate D_t can be considered. Each metric is weighted by a factor, denoted as w_1 , w_2 , and w_3 . However, this reward is only received after a significant time lag, denoted by T , due to the multi-hop propagation delays. The global reward, r^g , can thus be represented as:

$$r_{t+T}^g = w_1 U_t - w_2 L_t - w_3 D_t \quad (4.27)$$

The local reward represents immediate feedback consisting of directly observable metrics in the scope of a node and its neighborhood. Thus, it is assumed to be received in the subsequent time step $t + 1$. In this context, local link thresholds $u_{th,t}$, local delays l_t , and packet losses d_t due to saturated links can be considered. As before, these metrics are weighted by factors, denoted as w_4 , w_5 , and w_6 . The local reward, r^l , can thus be described by:

$$r_{t+1}^l = w_4 u_{th,t} - w_5 l_t - w_6 d_t \quad (4.28)$$

The effectiveness of local rewards is examined in detail. For fully-decentralized designs, composite rewards are typically not considered [91, 95]. The limited scope makes it difficult to evaluate the impact of individual actions of independent agents. Instead of using local delays, the estimated decrease in distance toward the destination is employed. The resulting path reduction is denoted by ξ_1 . Most importantly, to form coherent paths, loops have to be punished, but only at the nodes which are actually causing them. In this case a punishment $-\Xi_1$ is received, which should be at least a magnitude higher than ξ_1 . Including a similarly large reward, so $+\Xi_1$, for a successful path, which improved training performance - even though it was not propagated to all nodes along the path.

To incentivize decentralized load-balancing, a threshold-based approach proved effective. The utilized values of ξ_2 are in the domain of ξ_1 for balanced paths. Depending on which policy is more desired, the values for ξ_2 and ξ_3 can be varied. $-\Xi_2$ punishes the usage of

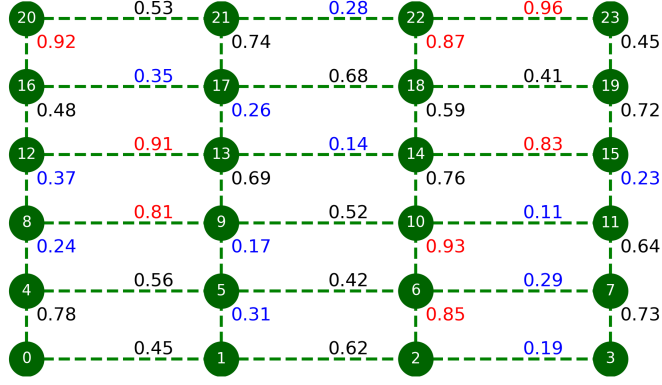


Figure 4.4.: Abstracted sub-network representation: satellites (green) with ISLs (green-dotted), and link load (color according to load level).

saturated links, which results in packet drops. In order to strongly discourage this behavior, $-\Xi_2$ is in the domain of $-\Xi_1$. The resulting local reward considered for FD-MADRL is thus the sum of:

$$r_{t+1}^l = \begin{cases} +\Xi_1, & \text{if goal reached} \\ +\xi_1, & \text{if overall path reduction} \\ -\xi_1, & \text{if overall path extension} \\ -\Xi_1, & \text{if path not reduced \& loop detected} \\ +\xi_2, & \text{if link load} < 0.4 \\ -\xi_2, & \text{if } 0.4 < \text{link load} \leq 0.8 \\ -\xi_3, & \text{if link load} > 0.8 \\ -\Xi_2, & \text{if link saturated} \end{cases} \quad (4.29)$$

Robustness Evaluation

In order to consolidate the expected shortcomings of FD-MADRL approaches, the performance and resilience is analyzed in a simplified model network.

To evaluate the considered approaches, a specifically designed environment was created using the `gymnasium` library [134]. The DQN functionality was implemented using `Keras` [135] and `Tensorflow` [136]. For the comparisons clusters of 12 and 24 satellites are used with varied link loads. The larger cluster, with an assumed initial link utilization, is shown in Figure 4.4. While the loads are set randomly, they comply with the observed link characteristics of the considered traffic scenario. For this performance comparison, the latency on a per-hop basis, i.e., the hop count, as well as the link loads are measured.

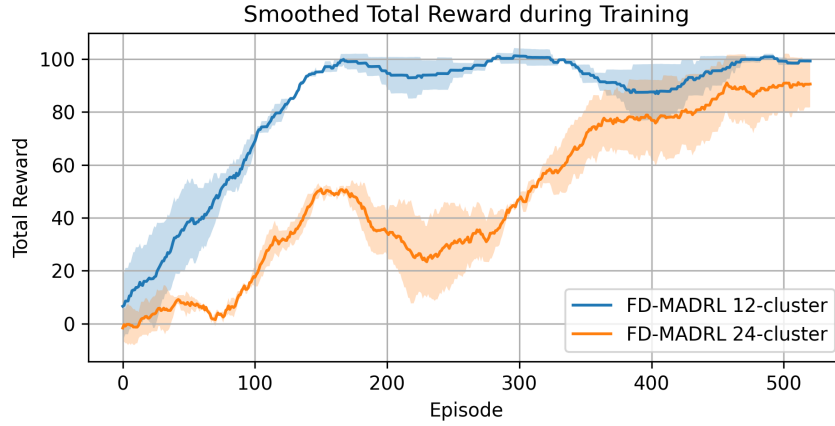


Figure 4.5.: Smoothed rewards over training for different DQN-based approaches in static scenarios, highlighting differences in convergence.

Performance Evaluation: Static Pathfinding

Firstly, a static case is investigated. The goal is to find paths with suitable latency and link load characteristics, as described by the rewards. To this end, the total rewards received during training in each episode are compared for the 12-node and 24-node clusters to gain insights into scalability. As shown in Figure 4.5, more than twice as many episodes are required to achieve maximum rewards consistently.

When analyzing the results, it is observed that sub-optimal decisions are often made in the last hop. For instance, in the 24-node cluster shown in Figure 4.4, to reach node 23 from node 4, going up to node 8 was learned, as the adjacent link has a lower load. However, this can result in a route going over node 22 which then takes the almost-saturated link to node 23. As assumed, the limited scope of individual nodes can result in costly end-to-end decisions.

The performance of the approach in comparison to a dynamic SPF scheme using a dynamic multi-cost approach based on Dijkstra's algorithm [59] is shown in Figure 4.6. The rule-based method corresponds to the presented dynamic best-of-k-paths algorithm for intra-cluster routing (Section 3.2.1). Links with high utilization, 80 %, are actively avoided. The results show that the current reward design achieves short paths: for most routes FD-MADRL requires fewer hops. However, the approach tends to saturate links. While this static example may not reveal significant issues, it is crucial to consider potential consequences in dynamic scenarios where a lack of foresight can result in decreased overall system performance.

Performance Evaluation: State Evolution - Dynamic Link Loads

By reflecting the impact of previous decisions in the link loads, differences in training can be observed, as shown in Figure 4.7. The idea is to test the robustness of the approach by evolving the state space according to previously made decisions. This corresponds more to an actual SCN scenario, where routing decisions impact future states and link loads change dynamically. Thus, in principle, subsequent constellation snapshots are studied. Consequently, the network is modified to reflect the previous action trajectory, with an

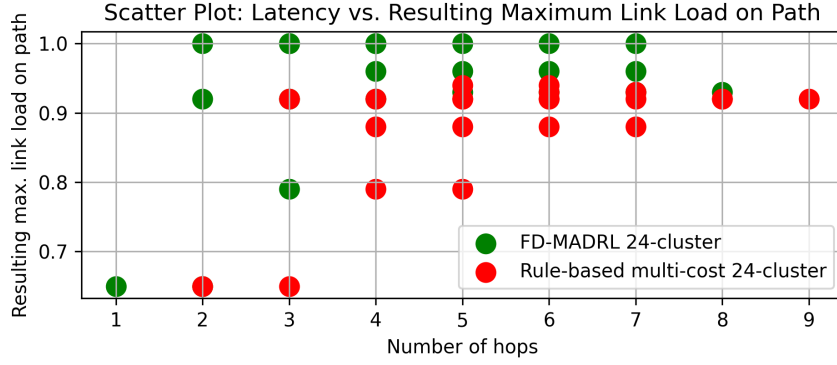


Figure 4.6.: Comparison of FD-MADRL (in green) and a rule-based multi-cost approach (in red) in terms of latency (number of hops, on the x-axis) and the resulting maximum link load on the chosen path (y-axis).

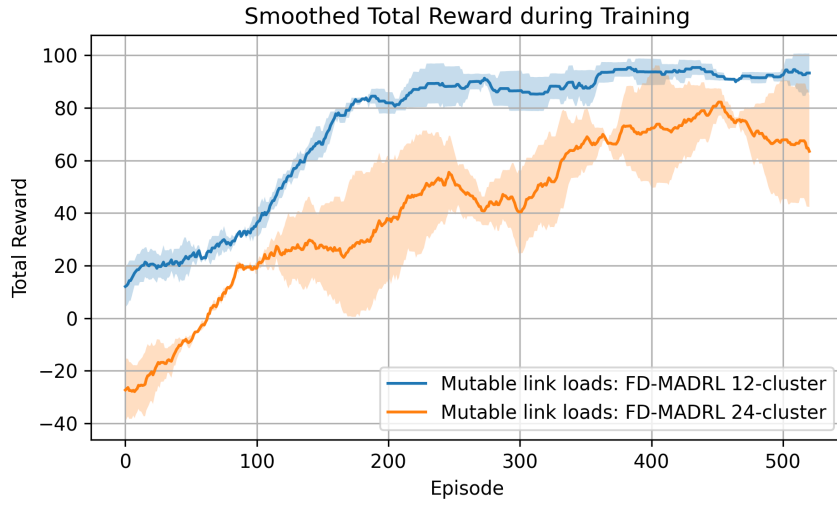


Figure 4.7.: Smoothed rewards over training for FD-MADRL with dynamic, mutable link loads.

extra 20 % utilization applied to every link used in the prior episode. These additional loads are only episode-specific, to avoid saturation on all links.

In comparison to Figure 4.5, the achieved rewards are lower and less stable in the given training setup. While this can be explained by higher link loads resulting in more negative rewards, the current setup is also not able to robustly respond to the more varied state space. While a high performance can be achieved for the 12-node cluster nonetheless, the results are unstable for the 24-node cluster. In this larger network, more scenarios can arise which lead to significant negative rewards: either by selecting saturated links or by inadvertently introducing loops. Moreover, learning to proactively avoid potential bottlenecks is difficult with fully-decentralized agents. These results highlight the necessity for adjustments such as information exchange between nodes, or schemes specifically designed to enable cooperation between agents.

4.2.3. Limitations and Further Perspectives

While the presented hop-based approaches converge, and are capable of handling intra-cluster routing, their potential to maximize throughput is limited. Core aspects, namely stability, robustness, and scalability pose problems in the reference scenario.

FD-MADRL is capable of approximating multi-objective routing, in terms of hop count and load-balancing. However, with increasing cluster size and state complexity, finding suitable solutions becomes more challenging. The limited scope of each agent negatively impacts the end-to-end routes, in both static and dynamic scenarios. Moreover, with increasingly complex policies, the agents have to learn the behavior of other agents, which is counter-productive for scalability. C-DQN provides a larger scope and is able to improve stability by leveraging rule-based pathfinding algorithms. Nevertheless, its hop-based design and the resulting complexity in time limit its viability. Varied QoS requirements and network dynamics will also require more complex state representations and reward structures.

To balance these concepts, a hybrid approach based on Centralized Learning - Decentralized Control (CL-DC) can be considered. Such a scheme can improve coherence and end-to-end performance while maintaining the flexibility, robustness, and scalability of FD-MADRL. In this setup, a central controller provides guidance, while decisions are made locally by the agents. Local actions and rewards are forwarded to the central entity for learning. The centralized guidance improves stability by facilitating coordination between agents. Based on local conditions and constraints, the decentralized agents can make informed local decisions. However, to enable such an interaction and to overcome the presented limitations of the DQN methods, alternative techniques must be considered.

In this context, actor-critic methods are particularly promising, as they combine the value-based approximation with direct policy learning. Unlike pure Q-learning approaches, which require consistent state-action representations during both training and execution, actor-critic architectures can separate the information needed for training (used by the critic) from the information required during execution (used by the actor). This separation enhances robustness in dynamic environments where complete information may not always be available at execution time [137].

Actor-Critic Methods

Actor-critic approaches combine policy gradient methods, utilized by an actor, with value function approximation, i.e., Q-learning, by a critic. They consist of two main components: an actor (or policy) $\pi(a|s; \theta_a)$, parameterized by θ_a , which maps states to actions, and a critic, represented by the state-action value function $Q(s, a; \phi_c)$ (or the value function), parameterized by ϕ_c , which estimates the expected return or value of taking an action in a given state. In contrast to pure Q-learning methods which derive policies from state-action values, actor-critic methods explicitly learn a policy via the actor while the critic provides value estimates to guide this learning.

The actor's goal is to learn an optimal policy that maximizes the cumulative reward over time. To achieve this, the actor updates its parameters using the policy gradient theorem, which provides an expression for the gradient of the objective function $J(\pi_{\theta_a})$ with respect

to the policy parameters θ_a :

$$\nabla_{\theta_a} J(\pi_{\theta_a}) = \mathbb{E}_{\pi} [\nabla_{\theta_a} \log \pi(a|s; \theta_a) Q^{\pi}(s, a)] \quad (4.30)$$

Here, the expectation is taken with respect to the policy π , and $Q^{\pi}(s, a)$ represents the true state-action value function under the current policy. Accordingly, the role of the critic is to provide an estimate of this value function, which can then be used as a baseline for updating the actor. The update rule for the parameters θ_a is directly derived from the policy gradient theorem:

$$\theta \leftarrow \theta + \alpha \nabla_{\theta} J(\pi_{\theta}) \quad (4.31)$$

The critic typically learns by minimizing the Mean Squared Error (MSE) between its estimate $Q(s, a; \phi_c)$ and the true value of taking action a in state s . This can be represented as:

$$\ell(\phi_c) = (r + \gamma Q(s', a'; \phi_c) - Q(s, a; \phi_c))^2 \quad (4.32)$$

The combination of these two components allows the actor-critic algorithm to learn effective policies in complex environments while avoiding some of the challenges associated with pure policy-based or value-based methods. For DRL-enabled traffic engineering, this facilitates a more direct approximation of the MCFP. The learning focus can be shifted from finding suitable paths to identifying optimal routing policies. However, to isolate the problem, path-based decision-making is necessary.

4.3. Deep Reinforcement Learning for Path-based Routing

In this section, a path-based routing approach using Soft Actor Critic (SAC) is investigated and evaluated. Building on the presented centralized DQN architecture, a similar setup is considered with a central agent distributing incoming flows while trying to approximate solutions to the MCFP.

In order to avoid varying action spaces, sets of candidate paths of fixed size are considered. Moreover, flows are assumed to be splittable across multiple paths. Consequently, a flow may be distributed across several paths, and not always on a single one. Additionally, to represent flow fractions, a continuous action space may be required, instead of a discrete one. It is conceptually similar to a probability simplex.

To enable this design, a SAC-based architecture is proposed. The particular viability of this actor-critic method is formally described, and a formal link to the MCFP is established. For the design, various sets of candidate paths and state-representation schemes are considered. During the evaluation, two principal questions are examined:

- MCFP approximation: can a trained model approximate an optimal flow distribution?
- Universal validity: can a trained model outperform rule-based approaches in scenarios with random sets of incoming flows?

Accordingly, two separate scenarios are investigated: training with a static flow set and training with random flow sets. The approach is compared with various state-of-the-art schemes and a benchmark solution.

The research presented in this section has been published in a corresponding publication [37].

4.3.1. Soft Actor-Critic Framework

To handle the intended path-based action space and stochastic policies, a SAC architecture [138, 139] is proposed. SAC is an off-policy actor-critic algorithm which maximizes the expected return while also maximizing entropy to encourage exploration. A stochastic policy is learned, as well as value functions, and a temperature parameter to balance exploration and exploitation. This encouragement of exploration aims to avoid premature convergence to sub-optimal policies, which is a concern in the given scenario.

It consists of an actor (policy) network, parameterized by θ_a , and two critic (Q-value) networks, parameterized by ϕ_{Q1} , and ϕ_{Q2} . While the original SAC formulation [138] applied a V-critic network ϕ_V , the utilized implementation relies on the now more commonly used twin Q-critic approach [140]. This method can help reduce overestimation of Q-values, improving stability.

The temperature parameter $\alpha_{\mathcal{H}}$ adjusts the weight of the entropy term. It thus controls the trade-off between reward maximization (exploitation) and entropy maximization (exploration). When the policy entropy is below the target, $\alpha_{\mathcal{H}}$ increases to encourage exploration. Inversely, it can also decrease when entropy is too high. $\bar{\mathcal{H}}$ denotes the target entropy, which is set to $-|\mathcal{A}|$, so the negative cardinality of the action space. The value is tuned automatically using the following objective:

$$\mathcal{J}(\alpha_{\mathcal{H}}) = \mathbb{E}_{a \sim \pi_{\theta_a}(\cdot|s)} [-\alpha_{\mathcal{H}} \log \pi_{\theta_a}(a|s) - \alpha_{\mathcal{H}} \bar{\mathcal{H}}] \quad (4.33)$$

The objective functions $J_{Q1}(\phi_{Q1})$ and $J_{Q2}(\phi_{Q2})$ for updating the soft Q-function parameters correspond to the minimization of the soft Bellman residual. So, they are a measure of the difference between the current estimate of the action-value function and its updated value based on the Bellman equation:

$$\mathcal{J}_{Q1}(\phi_{Q1}) = \mathbb{E}_{(s_t, a_t, r_t, s_{t+1}) \sim \mathcal{D}} \left[\frac{1}{2} \left(Q(s_t, a_t; \phi_{Q1}) - \hat{Q}(s_t, a_t; \phi_{Q1}) \right)^2 \right] \quad (4.34)$$

$$\mathcal{J}_{Q2}(\phi_{Q2}) = \mathbb{E}_{(s_t, a_t, r_t, s_{t+1}) \sim \mathcal{D}} \left[\frac{1}{2} \left(Q(s_t, a_t; \phi_{Q2}) - \hat{Q}(s_t, a_t; \phi_{Q2}) \right)^2 \right] \quad (4.35)$$

Here, $\mathcal{D}$ denotes the replay buffer. Lastly, the actor is updated to maximize the expected reward plus the entropy term. The objective of the actor network is to encourage the policy to take actions with high Q-values while maintaining diversity for exploration through entropy maximization:

$$\mathcal{J}_{\pi}(\theta_a) = \mathbb{E}_{(s_t, a_t, r_t, s_{t+1}) \sim \mathcal{D}} \left[\mathbb{E}_{a_t \sim \pi_{\theta_a}(\cdot|s)} \left[\min(Q(s_t, a_t; \phi_{Q1}), Q(s_t, a_t; \phi_{Q2})) - \alpha_{\mathcal{H}} \log \pi(a_t|s; \theta_a) \right] \right] \quad (4.36)$$

These objective functions highlight the self-regulating design of SAC, in terms of its exploration-exploitation trade-off. This balancing of a primary objective, i.e., reward maximization, with a regularization term which promotes diversity represents a formal relationship with the MCFP.

Relationship to Multi-Commodity Flow Problem

Based on the presented objective functions, a formal relationship between the MCFP and the SAC framework can be established. This connection demonstrates the theoretical viability of SAC for the optimization problem at hand. While the suitability of DRL-based approaches for routing has been derived, this framework shares additional properties with throughput maximization under constraints.

The distinguishing feature of SAC is its entropy regularization term. It prevents the policy from becoming deterministic too quickly and encourages exploration of the state space. A similar approach can be applied in the MCFP to promote flow diversity by adding a regularization term.

In the context of throughput maximization in the MCFP, the objective function can be formulated as:

$$\mathcal{J}_{\text{MCFP}}(f^k) = \max \sum_{k \in K} \sum_{(i,j) \in \mathcal{E}} f^k(i,j) \quad (4.37)$$

The described capacity and flow conservation constraints apply (see Section 2.3). Using entropy regularization, a formal connection is visible. To control the importance of flow diversity, α_R is introduced, and the flow-specific entropy term $\mathcal{H}(f)$ is defined as:

$$\mathcal{H}(f) = - \sum_{k \in K} \sum_{(i,j) \in \mathcal{E}} f^k(i,j) \log f^k(i,j) \quad (4.38)$$

Accordingly, the regularized MCFP objective function becomes:

$$\mathcal{J}_{\text{MCFP}}(f^k) = \max \sum_{k \in K} \sum_{(i,j) \in \mathcal{E}} f^k(i,j) - \alpha_R \mathcal{H}(f) \quad (4.39)$$

This formulation shares structural similarity with the SAC objective of the actor network, provided in Equation (4.36). In both cases, the primary objective, throughput or expected reward, is balanced with an entropy term that promotes diversity in solutions. The temperature parameter $\alpha_{\mathcal{H}}$ in SAC serves a similar role to α_R in the regularized MCFP, controlling the trade-off between the primary objective and solution diversity. The latter is an important aspect of proactive load-balancing.

Overall, this formal relationship indicates that SAC has the potential to adequately approximate solutions to the MCFP. The automatic tuning of the exploration-exploitation trade-off is considered a significant advantage, as converging towards promising policies is non-trivial.

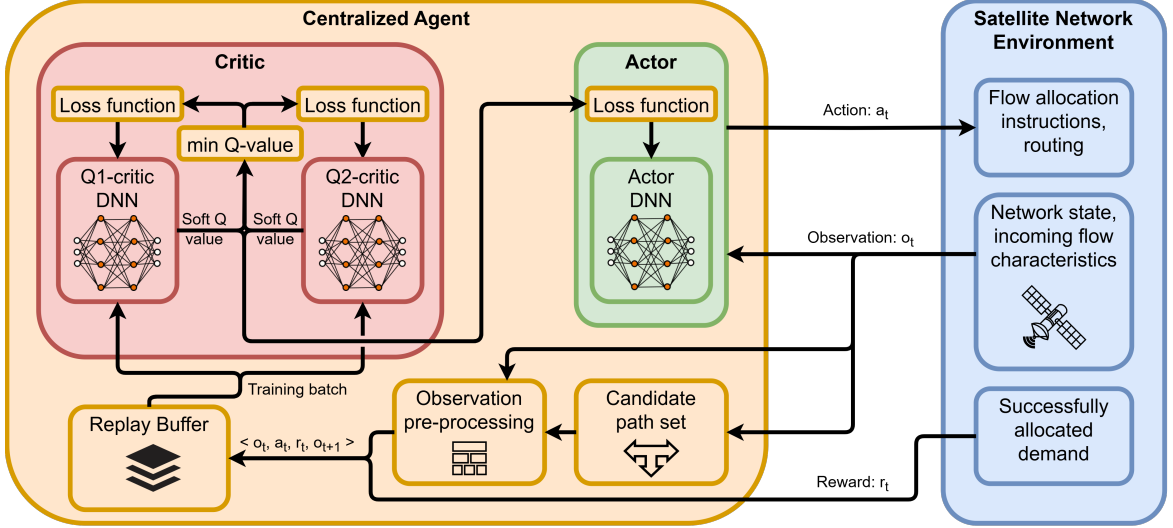


Figure 4.8.: Diagram of proposed path-based routing architecture. A centralized agent using a soft actor-critic framework interacts with the satellite network environment.

4.3.2. Path-based Architecture

Instead of learning path-finding, existing path-selection algorithms are leveraged to generate sets of candidate paths. The core assumption of this approach is that an optimal solution can be approximated by splitting incoming flows on these candidate paths. In principle, the idea is to learn underlying traffic characteristics which enables the anticipation of bottlenecks. Consequently, higher network loads can be supported thanks to informed decision-making. The components of the SAC framework and their interaction are summarized in the diagram shown in Figure 4.8.

Environment

As before, the environment corresponds to an IDLB cluster, a grid sub-network. At each step t , an incoming flow request must be routed through the network. This request is represented by a tuple consisting of a source node, target node, and demand, denoted as (v_{src}, v_{dst}, d_t) , respectively. Using efficient path-finding algorithms, a set of k candidate paths from source node v_{src} to target node v_{dst} is pre-generated. The source and target node of flows are sampled from a fixed heat map, which is illustrated in Figure 4.9. The demand of a flow is sampled from a uniform distribution. Link utilization is updated according to the flow allocations.

Action Space

The task of the agent is to split the incoming flow request among the pre-generated candidate paths. Thus, the action space is defined as a vector of length k over $\mathbb{R}_{\geq 0}$, where k corresponds to the number of candidate paths pre-generated for the current flow request. To ensure the action represents valid fractions of the flow, the agent's chosen action vector is normalized such that its entries sum to one. This normalization ensures that the flow is

divided proportionally among the set of candidate paths, $\mathcal{P}_{\text{candidates}}$, based on the output of the agent. The action space is thus:

$$\mathcal{A} = \mathcal{P}_{\text{candidates}} \quad \text{with} \quad |\mathcal{P}_{\text{candidates}}| = k \quad (4.40)$$

For example, if $|\mathcal{A}| = 4$ and the agent outputs an action vector $\mathbf{a}_t = [3.0, 1.0, 2.0, 4.0]$, it is normalized to $\mathbf{a}'_t = [0.3, 0.1, 0.2, 0.4]$. This means that 30 % of the flow is allocated to the first path, 10 % to the second, 20 % to the third, and 40 % to the fourth. This approach guarantees that the full flow is distributed among the candidate paths in proportion to the agent's decisions.

Set of Candidate Paths

In order for the agent to learn effective policies, it is important that the action space and its impact in terms of reward are varied. Since the key assumption of approximating optimal flow distribution relies on this set, its design is critical. Naturally, the set should include less congested paths to offer viable options. However, the paths should only minimally overlap, so that the effects of choices are distinguishable. The investigated three sets of candidate paths are listed in Table 4.2. As indicated, training with different sets resulted in varied behaviors.

To approximate the least congested path, the min-max problem is transformed into a min-sum problem. For each edge $(i, j) \in E$ with utilization $u(i, j)$, an auxiliary weight $w(i, j) = \exp(u(i, j))$ is introduced. This transformation is inspired by the Jacobian logarithm approximation, where

$$\log(\exp(a) + \exp(b)) = \max(a, b) + \log(1 + \exp(-|a - b|)) \quad (4.41)$$

$$\approx \max(a, b). \quad (4.42)$$

To find the path $p \subset E$ that minimizes the sum of these auxiliary weights

$$\sum_{(i,j) \in p} w(i, j) = \sum_{(i,j) \in p} \exp(u(i, j)),$$

Dijkstra's algorithm [59] is used. The logarithm of this sum

$$\log\left(\sum_{(i,j) \in p} \exp(u(i, j))\right) \approx \max_{(i,j) \in p} u(i, j),$$

minimizes the sum of exponentials. Thus, it provides an effective approximation for minimizing the maximum edge utilization along the path. Therefore, this approximation is used as a link weight for a baseline method called "least-congested-first". It is also used in the computation of candidate paths for the proposed DRL approach.

State Space

To enable coherence with the action space, the observation space of the agent has to include the path of the considered candidate routes, which are the actions. Thus, the agent can map

4. Reinforcement Learning-based Traffic Engineering

Table 4.2.: Considered sets of candidate paths. The overlap indicates the proportion of nodes a path can share with other paths.

Name	Set (a)	Set (b)	Set (c)
Path 1	shortest path in hops	least-weight*	least-weight*
Path 2	no edge with path 1	50% overlap	70% overlap
Path 3	least-weight*	50% overlap	70% overlap
Path 4	2 nd least-weight*	50% overlap	70% overlap
*: weight on hop is $w(i, j) = \exp(u(i, j))$			

topological characteristics to actions. Since the path size is variable with a maximum viable path length, this information has to be embedded.

One-Hot Encoding: A simple approach is to consider edge-based one-hot encoding to represent paths. In addition, path characteristics are included in the observation. Namely, the maximum, average, and variance of the current link utilization on the path. Moreover, the demand of the flow as well as the current link utilization matrix are embedded. The state space vector consists of real-valued elements, and its length, denoted by $|s_1|$, is given by

$$|s_1| = k \cdot (|\mathcal{E}| + 3) + 1 + |\mathcal{E}|, \quad (4.43)$$

where k is the number of candidate paths, and $|\mathcal{E}|$ is the number of edges in the network graph. The three comes from the three path characteristics and the one is for the current demand.

Histogram State Space: An alternative state space represents the system using a histogram of the current edge utilization, k histograms for the utilization of the candidate paths, and the demand. The length of this state representation, denoted by $|s_2|$, is given by:

$$|s_2| = (k + 1) \cdot |\mathbf{n}_{\text{hist}}| + 1 \quad (4.44)$$

Here, $|\mathbf{n}_{\text{hist}}|$ is the length of the histogram vector. Each histogram captures the relative appearance of utilization values on the edges, represented as a list of values between zero and one, where the entries sum to one. Importantly, the length of s_2 is independent of the number of edges in the graph, making it more scalable to different graph sizes. The value of $|\mathbf{n}_{\text{hist}}|$ depends on the number of bins N_{bins} used for discretizing utilization values. A larger number of bins provides a finer-grained representation but increases the size of $\mathbf{n}_{\text{hist}}$ and, consequently, s_2 . This histogram-based state space is compared with the one-hot encoding state space under different scenarios.

Reward

The rewards are shaped according to the intended characteristics of the MCFP. Correctly allocated flows are rewarded, while the saturation of links is punished. The highest new link utilization introduced by the flow allocation is also included. The reward function at time

step $t + 1$ is defined as

$$r_{t+1} = -\xi_1 \sum_{v=1}^k \sum_{(i,j) \in p_t^{(v)}} \left(1 + \frac{u(i,j)}{c(i,j)}\right) \cdot \mathbb{1}[u(i,j) > c(i,j)] \quad (4.45)$$

$$+ \xi_2 \left(1 - \max_{(i,j) \in \bigcup_{v=1}^k p_t^{(v)}} u(i,j)\right) \quad (4.46)$$

$$+ \xi_3 \sum_{v=1}^k \sum_{(i,j) \in p_t^{(v)}} d_t a_t^{(v)} \cdot \mathbb{1}[u(i,j) \leq c(i,j)], \quad (4.47)$$

where

- $p_t^{(1)}, p_t^{(2)}, \dots, p_t^{(k)}$ are the k candidate paths at time step t , and each $p_t^{(\cdot)}$ is a set of edges.
- $a_t = (a_t^{(1)}, a_t^{(2)}, \dots, a_t^{(k)}) \subseteq \mathbb{R}_{\geq 0}^k$ is the action of the agent at time step t , with $a_t^{(\cdot)}$ being the fraction of demand d_t allocated to path $p_t^{(\cdot)}$, satisfying $\sum_{v=1}^k a_t^{(v)} = 1$.
- $\mathbb{1}[\cdot]$ is the indicator function, which equals 1 if the condition in the argument is true and 0 otherwise.
- ξ_1, ξ_2 , and ξ_3 are positive weight factors to balance the components of the reward.

The term in Equation (4.45) penalizes paths with edges whose utilization exceeds capacity, scaled by ξ_1 . Similarly, the term in Equation (4.46) incentivizes minimizing the maximum utilization across all edges, weighted by ξ_2 . Lastly, the term in Equation (4.47) rewards successful allocation of demand to paths without exceeding capacity, scaled by ξ_3 .

If the set of candidates consists of less congested paths, the agent will only encounter punishments at high network loads. Other rewards of varying complexity, which applied exponential functions or further inputs have been tested, but did not outperform the presented reward shape.

Complexity

Once trained, the complexity of the proposed approach consists of the complexity of computing the candidate paths and the complexity of model inference, i.e.:

$$\mathcal{TC}_{\text{deploy}} = \mathcal{TC}_{\text{paths}} + \mathcal{TC}_{\text{inference}} \quad (4.48)$$

As before, using a k-shortest paths algorithm [119], the following complexity in time can be achieved:

$$\mathcal{TC}_{\text{paths}} = O(|\mathcal{E}| + |\mathcal{V}| \log |\mathcal{V}| + k). \quad (4.49)$$

Since the candidate paths may be computed using a mix of path-finding algorithms, the resulting complexity is slightly higher, but assumed to be in a similar domain. As described, the inference consists of a forward pass through the actor network:

$$\mathcal{TC}_{\text{inference}} = O\left(\sum_{\iota \in \mathcal{L}} N_n^{\iota-1} N_n^{\iota}\right) \quad (4.50)$$

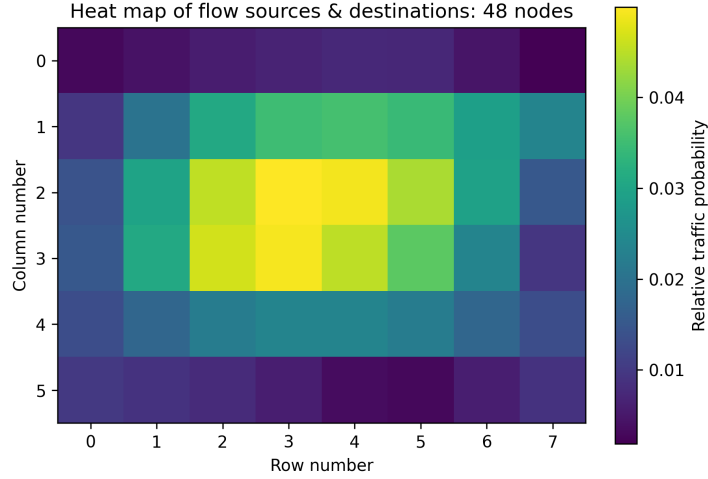


Figure 4.9.: Heat map of traffic sources and destinations for 48-node network. Shows normalized probability of each node to act as source or destination. Node indices represented by row and column numbers.

Training complexity, on the other hand, is much higher. It is the product of the number of iterations, N_{its} , and the time complexity of a forward and backward pass through the actor network ($\mathcal{TC}_{\text{actor}}$) and the two critic networks ($2\mathcal{TC}_{\text{critic}}$), so:

$$\mathcal{TC}_{\text{training}} = O(N_{\text{its}} (\mathcal{TC}_{\text{actor}} + 2\mathcal{TC}_{\text{critic}})) \quad (4.51)$$

4.3.3. Performance Evaluation

Simulative Environment

The environment is implemented in Python using the `Gymnasium` library [134], with SAC algorithms provided by `Stable-Baselines3` [140]. The underlying graph of the 2D grid network and path-finding routines are implemented using the Python library `NetworkX` [141].

For this investigation, a network of 48 satellite nodes, in an 8×6 arrangement, as in the previous evaluations, is considered. Flows from a source to a destination node are sampled randomly according to the heat map shown in Figure 4.9. The demand, i.e. the utilization of the flow, is uniformly sampled from the interval $[0.1, 0.5]$. In each step, a new flow is either read from a pre-existing set or generated. Then, candidate paths are computed and provided as part of the observation along with path characteristics as described in Section 4.3.2, with $k = 4$. Sets of $N = 50$ flows are considered, as the resulting load proves challenging for rule-based approaches. Nevertheless, optimal solutions which do not surpass the maximum capacity on any link are still possible in many cases. The used hyperparameters are summarized in Table 4.3.

Hyperparameter	One-Hot Encoding	Histogram Policy
Algorithm	SAC	SAC
Policy	MlpPolicy	MlpPolicy
State Space Size	$S_1 = 833$ (see (4.43))	$S_2 = 51$ (see (4.44))
Policy Architecture	[512, 256, 128]	[128, 128]
Critic Architecture	[512, 256, 128]	[128, 128]
Action Space Size	$k = 4$	$k = 4$
Activation Function	ReLU	ReLU
Learning Rate	$1 \cdot 10^{-3}, 1 \cdot 10^{-4}$	$1 \cdot 10^{-3}$
Buffer Size	1,000,000	1,000,000
Batch Size	512	512
Target Update Rate	0.01	0.01
Discount Factor	0.999	0.999
Entropy Coefficient	Automatic	Automatic

Table 4.3.: Hyperparameters of the SAC model for one-hot encoding and histogram policies using $k = 4$ pre-computed candidate paths.

Comparison of Candidate Path Selection

To evaluate the effects of the considered path selection schemes (in Table 4.2), the highest achieved mean reward during training is investigated. As the scenarios are varied, it is difficult to make universal statements about the performance of the path selection sets. In Figure 4.10, the mean rewards while training on a deterministic set of flows for 300,000 iterations is shown. For these flow requests, similar rewards were achieved by the different methods.

Although path set (a) provided slightly higher rewards, its actual performance in inference was worse than the solution using path set (b). In general, the result strongly depends on the considered flow set. For randomly sampled flow sets in particular, candidate path set (b) yields the best performance. Candidate set (c) is outperformed by set (b) in almost all scenarios. Therefore, the increased path diversity of set (b) is typically assumed preferable, even if there are specific scenarios where set (a) can achieve similar performance.

Baseline Methods

The methods, to which the proposed approach is compared, are intended to serve as lower, upper, and state-of-the-art benchmarks.

SPF Algorithm: A delay-based shortest-path-first approach is included as a lower benchmark. While the approach is link load-agnostic, it introduces the lowest overhead in terms of network load. As it relies on the shortest paths only, it introduces the lowest overall network utilization since no additional hops are used. The flow requests are not split in this approach.

Least Congested Path First (LCF) Algorithm: Essentially the same as the previous

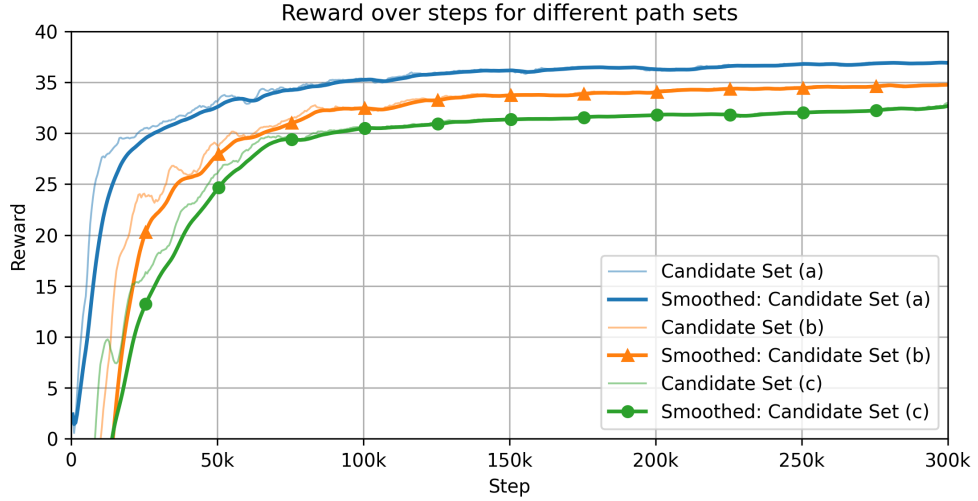


Figure 4.10.: Comparison of mean episode rewards for candidate path algorithms in scenario using deterministic set of flows. The plot shows both the raw reward data (lighter lines) and the smoothed reward data (solid lines). Sets (a) (in blue), (b) (in orange), and (c) (in green) are shown. The smoothed data is obtained using a moving average with a window size of 50 steps.

method. However, the edge weights represent link loads. Therefore, the lowest-cost path is the path with the lowest aggregated link load. The flow requests are also not split in this approach. This approach is important to show that the agent is able to learn a better policy than just always choosing the currently least-congested path.

Step-wise LP: It is possible to find an allocation for each incoming flow using LP. The flow is allocated to different paths based on the current state of the network. However, the approach does not consider how its decisions may impact future network states. Therefore, it can be outperformed by a well-trained approach which is familiar with underlying traffic patterns and able to choose paths accordingly.

Equal Multi-Path (EMP) Algorithm: To assess the impact of the trained policies, a benchmark that uniformly distributes incoming flows among the candidate paths is also presented. It is included in the comparison in Section 4.3.3.

Upper Benchmark - Ideal Allocation: By providing an LP solver with the complete set of incoming flows, one can compute the optimal flow allocation accordingly. It thus represents the upper benchmark which the RL-based algorithm is trying to approach. Naturally, this approach is purely theoretical as all incoming flows are known and allocated simultaneously. Effectively, it operates with perfect knowledge of all future flow requests.

Performance Comparison: Deterministic Set of Flows

As a first step, the convergence and performance of the considered approach is examined by training on a static, single set of incoming flows. The resulting box plot using this set of flows is shown in Figure 4.11. For this scenario, the DRL approach is able to achieve

similar statistics (median, average, and maximum utilization) as the theoretical benchmark solution. This result is remarkable, given that the agent is constrained to a set of candidate paths and solves the problem in a step-wise fashion. The agent implicitly learns the structure of future incoming flows, as these are identical across episodes during training. This enables the agent to optimize its decisions based on this implicit knowledge, explaining its ability to closely approximate the benchmark performance.

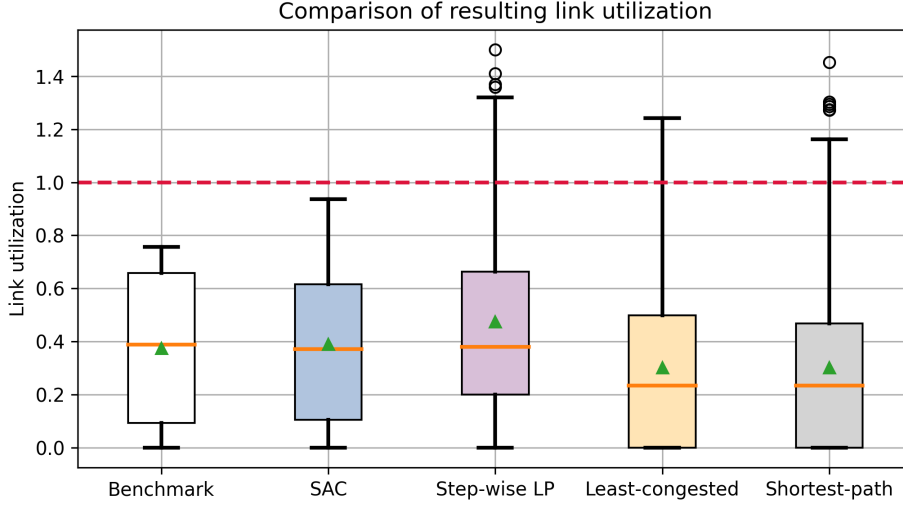


Figure 4.11.: Static scenario: box plot for link utilization of deterministic set of 50 flows. Benchmark is solution to MCFP using complete set, other approaches make step-wise decisions.

Accordingly, the resulting link utilization resembles the optimal distribution, as illustrated by histograms in Figure 4.12. However, the benchmark has complete knowledge of all flows upfront and can allocate them directly to minimize maximum utilization. In contrast, the DRL method includes some links with slightly higher load, as it does not have access to global information about all flows. Nevertheless, the results illustrate that the model has learned to approximate the MCFP for this deterministic set of flows.

Performance Comparison: Random Sets of Flows

While the previous scenario validated that a near-optimal approximation is possible in a static setup, a more general scenario consisting of randomly sampled incoming flows is of interest. Thus, the agent must learn policies that are valid for a diverse and unpredictable set of potential incoming flows. In this setting, the environment seen by the agent is inherently noisy, as the sampled flows vary significantly between episodes. Unlike the deterministic setting, the agent cannot rely on implicit learning of future flows but instead must base its actions on general traffic patterns dictated by the heat map. This added complexity introduces greater uncertainty in the impact of individual actions. Compared to a curriculum learning approach, where the agent is gradually exposed to increasingly complex scenarios, this approach confronts the agent with a diverse set of environments at once. While this may lead to slower convergence, it has the advantage of training the agent to be robust to variability in the environment, which is critical for real-world applications.

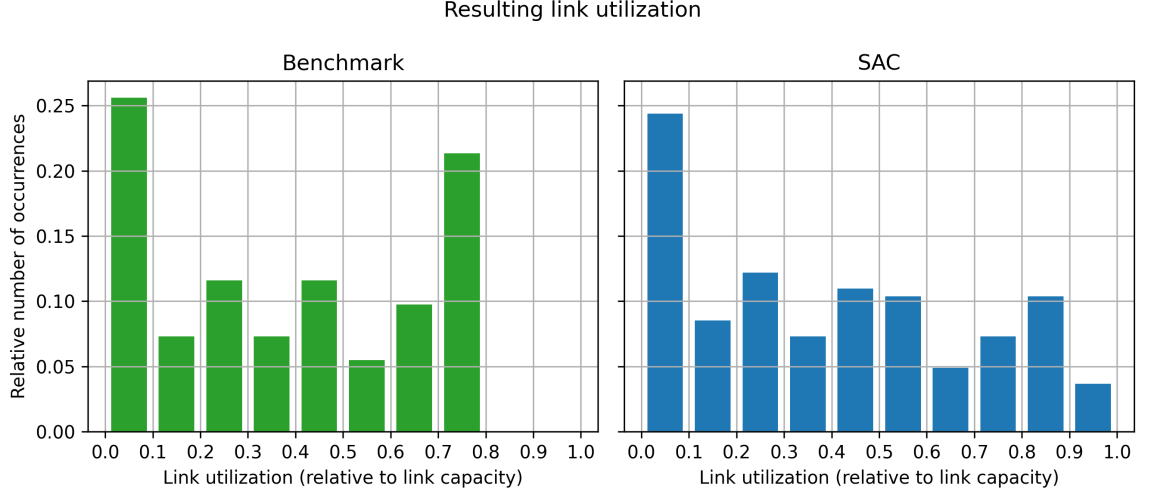


Figure 4.12.: Histogram of link utilization for exemplary set of 50 flows. Resulting distribution of benchmark solution (left) and proposed SAC approach (right).

No prior knowledge of the flows is assumed, so the theoretical benchmark is not applicable in this scenario. Moreover, for some flow sets, there may even be no feasible LP solution. In this case, the LP solver reports infeasibility, as the problem is overconstrained. So, instead, the results are compared to the EMP method. This comparison provides information on whether the agent simply distributes flows evenly or follows a more refined policy.

To better evaluate the performance, the total demand routed by each method is measured until saturation is reached. For this scenario, a separate policy is trained which terminates early whenever an incoming flow request cannot be routed due to link saturation. A corresponding reward function is defined:

$$r_{t+1} = \begin{cases} d_t & \text{if the flow is successfully allocated,} \\ 0 & \text{otherwise.} \end{cases} \quad (4.52)$$

This reward function encourages the agent to maximize the number of successfully routed flow requests, thereby increasing the total demand routed and earning a higher cumulative reward. In this particular context, this simpler formulation achieved similar performance as the previously introduced reward function.

Two policies are examined: one utilizing the one-hot encoding strategy for the state space and the other using the histogram-based state space, as described in Section 4.3.2. The training progression, including reward evolution for both policies, is depicted in Figure 4.13. The hyperparameters used for both policies are shown in Table 4.3. For the histogram state space, a binning size of 0.1 is used which results in a state space of size $|s_2| = 51$, as described by Equation (4.44).

As the flow sets are randomized, Monte Carlo simulations are applied to evaluate the proposed approach and compare it to EMP, LCF, and SPF. The results, based on $10 \cdot 10^3$ sampled episodes, are shown in Figure 4.14 and demonstrate that the agent learned viable

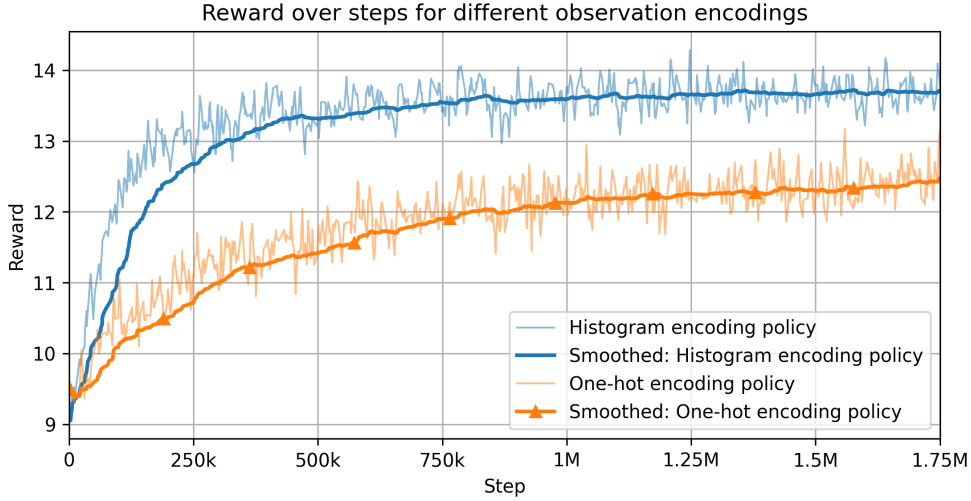


Figure 4.13.: Reward progression during training of the policies using the SAC algorithm. The plot shows both the raw reward data (lighter lines) and the smoothed reward data (solid lines) for each policy: the histogram policy (in blue) and the one-hot encoding policy (in orange). The smoothed data is obtained using a moving average with a window size of 100 steps.

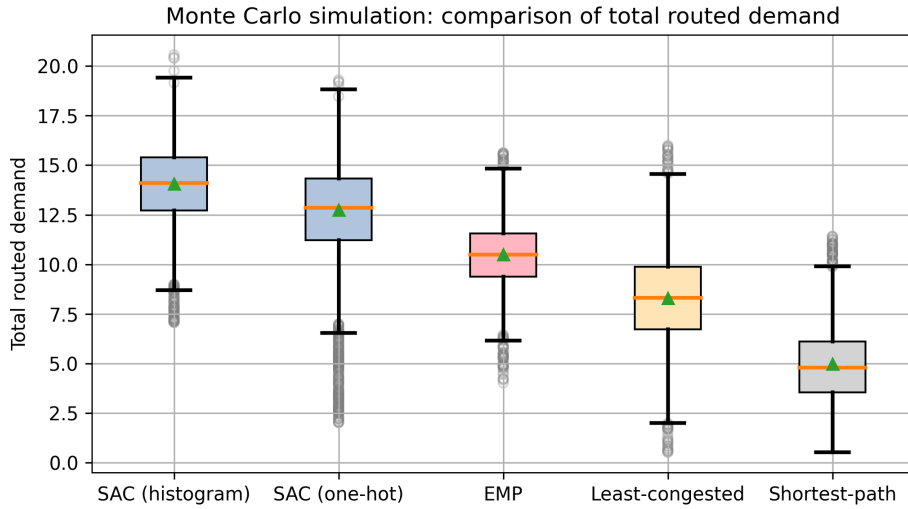


Figure 4.14.: Randomized scenario: box plot for total routed demand. Comparison of different routing methods for iteratively routing demands until overflow.

policies. Each method operates on the same set of flows per episode until network capacities are saturated. The figure also compares SAC policies (DRL-1 with the histogram state space and DRL-2 with the one-hot encoding state space) to these baselines. The DRL approaches outperform the baselines by routing higher total demand, demonstrating the effectiveness of learned policies over heuristic methods.

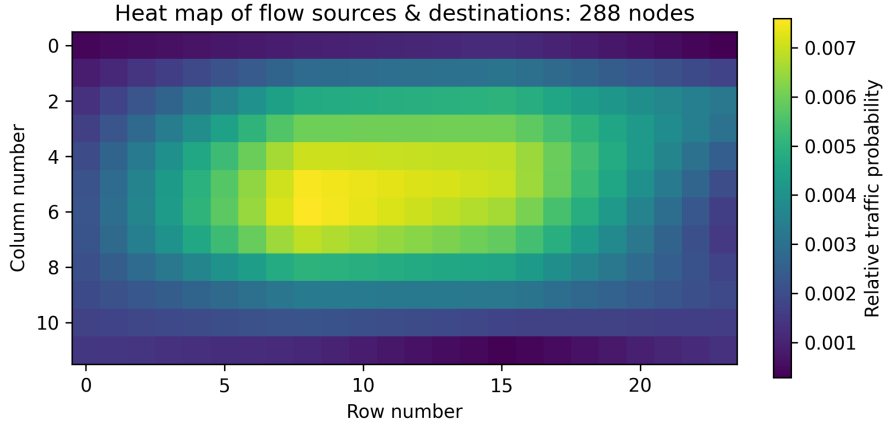


Figure 4.15.: Large heat map of traffic sources and destinations for 288-node network. Shows normalized probability of each node to act as source or destination. Node indices represented by row and column numbers.

Performance in Larger Clusters

In order to quantify how the approach scales, an additional randomized scenario is considered with more nodes. While the main evaluation focused on routing within SCN sub-networks, the proposed DRL-based method is equally applicable to ground-based or centralized control of the full constellation. So, to assess its potential in larger-scale domains, the evaluation is extended to a 24×12 grid topology, comprising 288 nodes. The corresponding heat map is shown in Figure 4.15.

While the size and setup of this network are based on SCN-288, they do not replicate the precise layout of a Walker constellation. The orientation and directionality of ISLs are simplified. Nevertheless, the grid structure provides a relevant approximation of path diversity, scale, and congestion effects observed in SCN-288.

In addition, a softmin path benchmark is included, which evaluates paths based on an exponential cost transformation:

$$w(i, j)_{\alpha_s} = \exp(\alpha_s \cdot u(i, j)) \quad (4.53)$$

The resulting path choices differ from the least-congested approach, which sums utilization on each link. For this benchmark, paths with individual links with high load are more costly. A weighting factor of $\alpha_s = 10$ is used.

Figure 4.16 shows the total routed demand until saturation for the considered baseline and DRL methods, as well as the additional benchmark. The same Monte Carlo sampling procedure as before is used. Notably, the Softmin(10) benchmark outperforms the least-congested and EMP methods. The SAC agent again achieves the highest throughput, confirming that the approach scales well with network size and maintains a performance advantage even under increased path complexity and capacity pressure.

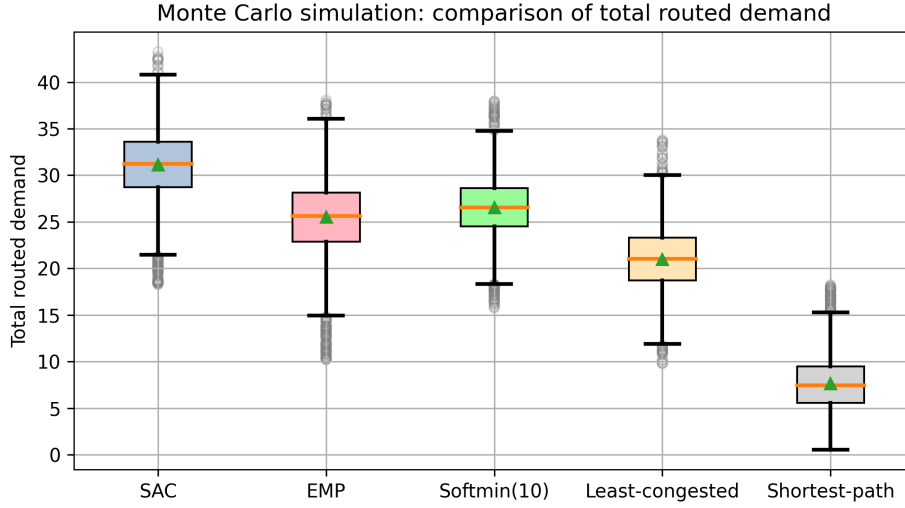


Figure 4.16.: Monte Carlo simulation: comparison of total routed demand for 24×12 grid.

Summary

Overall, the proposed and examined SAC-based DRL approach is well-suited to approximating solutions to the MCFP in SCNs with low complexity. By distributing traffic on sets of candidate paths, the scheme is able to significantly outperform rule-based methods in different scenarios. For a deterministic set of flows, the results show that DRL is fundamentally able to approach a benchmark solving the MCFP. When using randomly sampled sets of flows, the learned policies enabled more flow allocations until link capacity was exceeded, thanks to their knowledge of underlying traffic characteristics. Moreover, the performance improvements remained consistent even in a larger network, demonstrating the flexibility and robustness of the path-based architecture.

4.4. Implications and Discussion

This chapter provided a comprehensive overview of viable DRL-based routing strategies for SCNs. From first principles, the relevance of DRL for in-space routing was derived, highlighting its ability to learn dynamic traffic patterns and to approximate complex functions. Key RL concepts and architectures were presented, suitable for extending the proposed distributed load-balancing SDN solution, IDLB.

Hop-based Approaches Analysis and Limitations

The hop-based approaches represent a direct application of the derived DRL-based routing using DQNs. However, important drawbacks of both centralized and decentralized schemes were identified. The presented C-DQN approach avoids stability issues by constraining pathfinding. However, these constraints introduce complexity. In addition, it is unclear how proactive decision-making in the context of the MCFP can be learned. FD-MADRL demonstrates questionable robustness and scalability. Information exchange and coordina-

tion between nodes is required to overcome these problems.

To mitigate some of these issues, hybrid approaches and actor-critic methods can be considered. However, due to the necessity of learning pathfinding, the overall research goal of maximizing throughput is not the singular optimization objective. Inherently, the hop-based architecture complicates a direct approximation of the MCFP. For these reasons, an actor-critic approach relying on paths instead of hops was identified as a more promising follow-up.

Path-based Approach Analysis

The proposed path-based method exhibits several key strengths that make it an attractive solution for flow distribution in SCNs. Notably, the approach demonstrates flexibility in learning complex traffic patterns and distributing flows through predictive modeling, allowing it to anticipate and mitigate potential bottlenecks. The scheme consistently outperforms state-of-the-art methods, including the least-congested path approach.

Furthermore, the improved performance achieved by the method could not be replicated by mimicking its behavior with rule-based decision-making. This demonstrates that the agent is actually learning effective policies and not generic strategies.

In addition, the observed computational complexity of the approach was manageable, even during training. The majority of the complexity is caused by computing the candidate paths. Relative to heuristics, no strong assumptions about network characteristics are necessary. Additional tests also indicate that even smaller models may perform similarly well. Moreover, the method seamlessly integrates into SDN frameworks, making it a viable and practical solution for real-world deployment.

Path-based Approach Limitations

The proposed approach has two key limitations: candidate path efficiency and assumed traffic pattern consistency.

Firstly, a core assumption underlying the approach is that distributing flows among a set of candidate paths will result in near-optimal performance. While the results show its capacity to route more demands, the given sets of paths may represent a ceiling to the effectiveness of the approach. Consequently, additional investigations with differently sized sets of candidate paths are of interest.

Secondly, a significant limitation of the investigated setup is the focus on a specific heat map pattern. In order to quantify the robustness of the approach, diverse heat maps have to be evaluated. For instance, clusters which mostly experience transit traffic patterns are not represented. Additionally, the learning of underlying traffic patterns is currently tied to static heat maps, which may not capture the full complexity of real-world traffic dynamics. Given the movement of the constellation and dynamically changing network topology, satellites may be faced with diverse traffic scenarios. More complex, time-dependent environments have to be considered. While the results demonstrate some degree of robustness and generalization, the effectiveness of the method may decrease for such environments and related models.

Further Considerations

As the path-based SAC approach has shown potential for optimizing flow distribution in SCNs, it is essential to consider its implications for real-world deployment and potential extensions. Overall, once trained, the approach represents a relatively low-complexity solution to distribute incoming flows. Its design is also well-suited for SDN-enabled routing optimization in SCNs. Notably, the on-board hardware used in the demonstrator is advertised as capable of ML applications (see Section 3.4 and Appendix A).

Real-world applications are expected to require more sophisticated and fine-tuned models, depending on the scenario. In order to exploit observed historical traffic patterns more explicitly, an adjusted DRL architecture can be provided with estimated heat maps as inputs. To efficiently handle such larger inputs, additional pre-processing methods including Convolutional Neural Networks (CNNs) or GNNs can be considered for feature extraction.

Given the adaptable nature of the method, there are interesting implications for further research. The approach is expected to generalize for various scenarios including terrestrial networks. For more complex environments, convergence may be achieved by employing curriculum learning approaches, where the agent is first trained on simpler contexts and then gradually introduced to more realistic scenarios. With these extensions, it may be possible to train agents to master diverse environments while maintaining high performance.

5. Conclusion

5.1. Summary

This thesis presents a novel routing protocol for SCNs optimized for maximum network throughput and scalability. The key research goal was to develop an approach which enables efficient routing configurations by making use of the existing network resources. As a result, higher network loads can be supported while maintaining high QoS compliance. So, the system can potentially handle more customers by avoiding the creation of bottlenecks in the ISL topology.

To achieve this, an in-orbit routing protocol based on distributed SDN, called IDLB, was designed. It is based on fixed sub-domains within the SCN topology, controlled by on-board SDN controllers. This cluster-based approach applies concepts from geographical routing to reduce signaling overhead, and insights from set-based source-routing for intra-cluster routing. Suitable QoS-aware algorithms were formulated for both routing within and between clusters.

An exhaustive quantitative evaluation using a system-level simulator demonstrated the performance of the approach, highlighting its load-balancing capabilities as well as the impact of various design choices. The results showed how different cluster sizes and routing strategies influence QoS compliance. In addition, a demonstration in a hardware test bed consolidated the viability of the design for real-world satellite constellations.

As IDLB is a promising enabler of data-driven approaches, the potential of ML techniques to approximate solutions to the MCFP within clusters was explored as well. The viability of DRL for routing was derived systematically, and suitable approaches were formulated. Based on insights from the limitations of hop-based schemes, a path-based approach using SAC was proposed. This architecture is based on the idea of intelligently distributing traffic among sets of candidate paths.

The evaluation of the trained models showed their ability to approximate solutions to the MCFP in SCNs with low complexity. Notably, this approach outperformed various benchmarks in a more general setting, demonstrating its potential as an intriguing solution for diverse traffic conditions.

Overall, the research contributes to the development of efficient and scalable space-borne routing protocols. The combined approach of the IDLB protocol and the SAC-based extension represents a promising, robust, and adaptive solution for routing in autonomous, next-generation satellite systems.

Research Questions

The initially posed research questions have thus been answered as follows:

- What low-complexity, on-board load-balancing strategies can effectively approximate optimal solutions for the MCFP in SCNs?

The answer to this question is provided in Chapter 3 and Chapter 4. The proposed dynamic best-of-k routing algorithm represents a rule-based scheme to indirectly approximate solutions to the MCFP using proactive route distribution. However, the scheme does not attempt to solve the problem directly. Thus, a path-based DRL solution was proposed which learns to distribute flows in order to maximize network throughput. As such, the approach approximates solutions with low complexity in inference. Only a set of viable paths and a forward pass through the network are required.

- Which frameworks and architectures can enable these techniques, and how should suitable protocols be tailored for application in dynamic SCN scenarios?

In Chapter 3, a suitable architecture is proposed. The IDLB protocol utilizes the promising distributed SDN framework and enables intricate load-balancing techniques. IDLB is specifically tailored to the requirements of SCNs.

- How can data-driven methods be leveraged for network control to formulate and dynamically adjust effective load-balancing strategies, and what are the implications for network performance?

This question is answered in Chapter 4. DRL approaches are proposed and analyzed in the context of finding dynamically adjustable load-balancing strategies. Insights from centralized as well as decentralized methods were used to formulate a performant path-based solution. The implications and limitations of the approaches were highlighted.

5.2. Future Perspectives

This work primarily focused on the maximization of throughput, as well as related system-relevant performance metrics, such as signaling overhead and convergence time. Future research may consider varied scenarios with additional or different optimization goals.

Most notably, the applicability and integration into 6G frameworks are of interest. Given that its underlying building blocks align with proposed functional splits, an extended version of IDLB holds significant potential for standardization. This includes the direct integration of IDLB into terrestrial and other heterogeneous networks. Moreover, the combination with network slicing and orchestration mechanisms is an intriguing research direction. Further investigations may also focus on multicast solutions, potentially based on the proposed extension.

For practical deployments, additional analyses are likely necessary. While the general viability has been demonstrated in a hardware test bed, it is also of interest to investigate larger setups. Accordingly, inter-cluster routing procedures can be tested and validated. Testing actual protocol stacks on hardware devices is of particular interest to quantify achievable

performance. The interplay between protocol mechanisms, such as congestion control and IDLB, represents an important aspect to study.

There are various directions in which the presented SAC-based approach can be extended. The most important aspect with regard to SCNs will be its applicability to dynamically changing traffic characteristics. To this end, pre-processing and refined state space formulation may be considered. In addition, the robust performance of DRL in dynamic systems is also of interest for future applications in SCNs. For instance, the intelligent handling of unexpected link outages and handovers can be considered. In addition, further investigations across diverse scenarios, including system-level simulators and hardware test beds, are considered promising. Thus, the viability of DRL-based traffic engineering for real-world deployments can be demonstrated.

In the following, promising future perspectives for related research with different optimization goals are provided. The following topics are directly related to the presented approaches:

System resilience: Besides performance, building resilient satellite-based communication systems is a key research topic. To this end, various techniques and methods employed in this work can be used as well. However, the overall optimization problem is different: inefficiencies are encouraged if they are able to increase the resilience of the system against failures, but also against external influences. Nevertheless, core aspects of the underlying optimization problem of proactively avoiding saturated or unavailable links remain. In the context of denial-of-service attacks, load-balancing approaches are a key countermeasure to maintain routing stability and QoS compliance. Therefore, research in this direction may build upon the presented protocol and methods, with additional features for anomaly detection and mitigation. As novel approaches oftentimes rely on ML, the IDLB framework can be used to enable these techniques, as demonstrated in this study.

ESL handover strategies: ESLs represent an important aspect of the SCN connectivity. For systems considering optical ESLs, cloud blockage is a significant obstacle resulting in link loss. In order to maintain end-to-end routes, it is thus essential to formulate alternative backup paths, which are immediately used in the event of a blockage. In this context, it is possible to use generic downlink conditions, for instance in the form of segments. IDLB can be extended to support segment routing for ESLs. Given its in-orbit network control, fast adjustments are possible. Moreover, future research may also consider the integration of complex handover strategies into IDLB.

Multi-layer multi-orbit constellations: Similarly, handover strategies and corresponding routing adjustments are relevant when considering topologies with inter-layer links. In this context, the design of IDLB can be extended to seamlessly integrate layers, but also heterogeneous devices. For particularly complex and dynamic topologies, DRL extensions similar to the proposed methods may be investigated.

Overall, the results of this investigation provide a wide foundation for further research. There is great promise to impact future developments and to inspire next-generation satellite constellation network designs.

A. Exemplary On-Board Processing Hardware

In research projects related to this investigation, the principal choice of on-board processing hardware utilized the NXP Layerscape LX2160A processor [58]. Accordingly, all information is provided by NXP. This processor solution is considered representative of current hardware capabilities of state-of-the-art satellite designs.

The product is designed for 5G packet-processing operations. Representative applications and features are summarized in Table A.1 [142, 143]. Notably, the processor is intended for operating Open Radio Access Network (O-RAN) Central Units, Distributed Units, and for enabling ML. It is thus designed to support network control solutions such as the proposed distributed SDN-based protocol and DRL-based extensions. The block diagram of this processor is illustrated in Figure A.1.

Table A.1.: Applications of LX2160A as 5G packet processing component [142, 143].

Example applications	Relevant features
O-RAN Central Unit O-RAN Distributed Units, Radio Units Machine learning Cloud-RAN	Linux-based networking applications Hardware QoS Sufficiently large caches 100 Gbps Ethernet, 30 Gbps high-volume flows

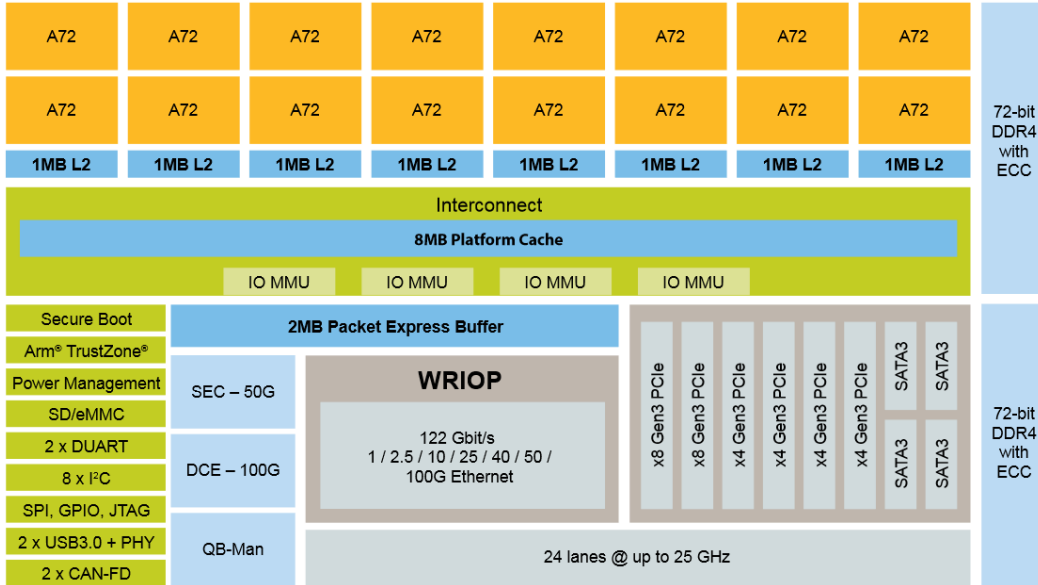


Figure A.1.: Block diagram of Layerscape LX2160A provided by NXP [58, 142].

B. Configuration of Simulator

The simulator generates C++ objects for the satellite nodes, user nodes (i.e., the terminals), and gateway nodes according to configurable parameters. So, it is possible to simulate different scenarios by creating or editing dedicated XML files as inputs. The used simulation parameters are listed in Table B.

Overall, the simulator relies on abstracted objects. Therefore, no actual packets are forwarded. Instead, a packet header object is instantiated with the corresponding characteristics and passed through different functions. Delays are established by setting the timing of events in the event loop accordingly. Consequently, no actual protocol stacks are utilized. Instead, relevant characteristics of IP, MAC, and MPLS are modeled to enable representative simulations.

Table A.1.: General configuration used by simulator for routing protocol analyses.

Simulation parameter	Value
Simulation duration [s]	720
Satellite movement step [s]	5.0
Routing scheme	flow-based-idlb
Scaling factor	10.0
Satellite amount	288
Satellites per plane	24
Movement timeseries file	inputs/scn288_movement_10s_43200s.txt
Sat input step [s]	10
ISL connectivity indices file	inputs/scn288_isls.txt
ISL data rate [Mbps]	1000
On-board buffer size [Mbit]	0.36
Max downlink data rate [Mbps]	5000
Max uplink data rate [Mbps]	5000
Processing delay [μ s]	50
ISL load update period [s]	2
ISL load window duration [s]	10
ESL load window duration [s]	10
Update deployment timer [ms]	100
Cluster size	12
QoS Class 2 load weight	6

A. Exemplary On-Board Processing Hardware

QoS Class 3 load weight	15
Intra-cluster algorithm	set-based
User terminal amount	2000
User terminal positions file	<code>inputs/uts_density_10k.txt</code>
User terminal minimum elevation [deg]	15
Mobile terminal amount	0
Mobile terminal maximum speed [m/s]	300
Gateway amount	39
Gateway positions file	<code>inputs/gateways_39.txt</code>
Gateway minimum elevation [deg]	10
Geographical area tiling	equirectangular
Geographical area amount	6962
Session amount	5000
Session average duration [s]	100.0
Session data rate [Mbps]	10
UT-GW traffic share	0.6
UT-GW traffic multicast share	0.0
Session on-off distribution	exponential
QoS Class 1 relative share	0.1
QoS Class 2 relative share	0.34
QoS Class 3 relative share	0.56
QoS Class 1 on-off share	0.3
QoS Class 2 on-off share	0.0
QoS Class 3 on-off share	0.0
QoS Class 1 average talkspurt duration [s]	3.0
QoS Class 1 average silence duration [s]	1.0
QoS Class 2 average talkspurt duration [s]	3.0
QoS Class 2 average silence duration [s]	1.0
QoS Class 3 average talkspurt duration [s]	3.0
QoS Class 3 average silence duration [s]	1.0
Network update step [s]	60.0
Packet size [kByte]	1.5
Maximum number of hops	90
Generate random sessions	true
Predetermined sessions file	<code>outputs/sessions.txt</code>

C. Deep Learning Framework

Deep learning describes the process of adjusting the weights between nodes in a network to achieve a desired behavior through backpropagation [28, 128].

The general operation of a neural network proceeds from an input layer through potentially multiple hidden layers to an output layer. The set of layers is denoted by $\mathcal{L}$. Each layer represents a set of nodes which perform mathematical operations on input vectors resulting in outputs. Every node is connected with other nodes via weights, w , and has a bias, b . For each node in layer ι the following operation is performed:

$$\mathbf{z}^{[\iota]} := \mathbf{w}^{[\iota]} \mathbf{a}^{[\iota-1]} + \mathbf{b}^{[\iota]} \quad (\text{C.1})$$

Here, $\mathbf{w}^{[\iota]}$ denotes the weights of layer ι , while $\mathbf{b}^{[\iota]}$ denotes the biases. $\mathbf{a}^{[\iota-1]}$ is used to describe the output from the previous layer $\iota - 1$ which is passed through an activation function σ . The network is assumed to be fully connected. The output of layer ι is thus:

$$\mathbf{a}^{[\iota]} := \sigma(\mathbf{z}^{[\iota]}) = \sigma(\mathbf{w}^{[\iota]} \mathbf{a}^{[\iota-1]} + \mathbf{b}^{[\iota]}) \quad (\text{C.2})$$

The resulting estimation, $\hat{\mathbf{y}}$, corresponds to the output of the last layer I :

$$\hat{\mathbf{y}} := \mathbf{a}^{[I]} \quad (\text{C.3})$$

Now the cost or overall loss function for the DNN, $\mathcal{J}$, in relation to these parameters is defined as the loss between the estimation, $\hat{\mathbf{y}}$, and the target, $\mathbf{y}$. It can be formulated as follows, with ℓ denoting the loss function:

$$\mathcal{J}(\mathbf{w}, \mathbf{b}) = \ell(\hat{\mathbf{y}}, \mathbf{y}) \quad (\text{C.4})$$

Common examples for loss functions include mean-square error and cross-entropy loss. The specific choice of function depends on the problem type and data characteristics. In training, the weights and biases, i.e. the parameters of the network, are adjusted towards an approximate optimal solution using gradient descent:

$$\mathbf{w} := \mathbf{w} - \alpha \frac{\partial \mathcal{J}(\mathbf{w}, \mathbf{b})}{\partial \mathbf{w}} \quad (\text{C.5})$$

$$\mathbf{b} := \mathbf{b} - \alpha \frac{\partial \mathcal{J}(\mathbf{w}, \mathbf{b})}{\partial \mathbf{b}} \quad (\text{C.6})$$

The step size of the gradient, denoted by α , is also known as the learning rate. Now, to compute these gradients efficiently, backpropagation is used. This method is based on the

C. Deep Learning Framework

application of the chain rule to calculate how changes in model parameters impact the cost function:

$$\frac{\partial \mathcal{J}}{\partial \mathbf{w}^{[\iota]}} = \frac{\partial \mathcal{J}}{\partial \mathbf{a}^{[\iota+1]}} \frac{\partial \mathbf{a}^{[\iota+1]}}{\partial \mathbf{z}^{[\iota+1]}} \frac{\partial \mathbf{z}^{[\iota+1]}}{\partial \mathbf{w}^{[\iota]}} \text{ assuming } \frac{\partial \mathbf{a}^{[\iota+1]}}{\partial \mathbf{z}^{[\iota+1]}} = \sigma'(\mathbf{z}^{[\iota+1]}) \quad (\text{C.7})$$

It is assumed that the activation function is derivable at $\mathbf{z}^{[\iota+1]}$ with respect to its input, denoted by $\sigma'(\mathbf{z}^{[\iota+1]})$. To illustrate the applicability of the chain rule, $\hat{\mathbf{y}}$ is represented as a composition of layer-wise functions:

$$\hat{\mathbf{y}} = \mathbf{a}^{[I]} = \sigma_{out}(\mathbf{z}^{[I]}(\sigma(\mathbf{z}^{[I-1]}(\dots(\sigma(\mathbf{z}^{[1]}))\dots))) = (\sigma_{out}\mathbf{z}^{[I]}) \circ (\sigma\mathbf{z}^{[I-1]}) \circ \dots \circ (\sigma\mathbf{z}^{[1]}) \quad (\text{C.8})$$

So, a gradient adjusting the parameters can be passed backward through the network. Consequently, backpropagation computes the impact of the weights and biases from the last to the first layer:

$$\frac{\partial \mathcal{J}}{\partial \mathbf{w}^{[\iota]}} , \frac{\partial \mathcal{J}}{\partial \mathbf{b}^{[\iota]}} \text{ recursively from last to first } I \rightarrow 1 \text{ with } \iota \in \mathcal{L} \text{ and } \iota = \{1, \dots, I\} \quad (\text{C.9})$$

For hidden layers, non-linear activation functions are typically used to introduce complexity in the model. These functions have to be non-linear to avoid linear transformations, and to enable a representation of complex patterns in the resulting manifold [33]. Common choices include:

- ReLU: $\sigma(\mathbf{z}^{[\iota]}) = \max(\mathbf{0}, \mathbf{z}^{[\iota]})$
- tanh: $\sigma(\mathbf{z}^{[\iota]}) = \frac{\exp(\mathbf{z}^{[\iota]}) - \exp(-\mathbf{z}^{[\iota]})}{\exp(\mathbf{z}^{[\iota]}) + \exp(-\mathbf{z}^{[\iota]})}$

Based on this general functionality, deep learning architectures are capable of approximating complex functions, even if no analytical formulation is possible. The presented adjustment of parameters only occurs during training.

List of Figures

1.1.	Schematic representation of a sub-section of an SCN. Labels illustrate GWs, UTs, ISLs, and ESLs. Distances and entity sizes not to scale. Further ISLs have been omitted.	3
1.2.	Exemplary scenario of a space segment consisting of a LEO satellite constellation (red dots) inter-connected via ISLs (black lines) and a corresponding ground segment consisting of UTs (blue dots) and GWs (yellow triangles). Map projection skews distances and angles.	4
1.3.	Diagram of logical SDN architecture. The SDN controller interacts with the application plane via a dedicated northbound interface, and with the forwarding devices via a southbound interface.	7
1.4.	Schematic illustration of a general RL framework, depicting the interaction of an agent with an environment.	8
1.5.	Diagram of a generic DRL agent with a satellite network environment.	9
1.6.	Overview of the proposed approach, including the DRL-enhanced on-board SDN controller architecture, for an exemplary path from Munich to Singapore across several clusters.	10
2.1.	Top-down view of SCN-288 (left) and SCN-1440 (right) satellite constellations [46].	17
2.2.	Coverage of SCN-288 with a minimum elevation angle of 20° [46].	19
2.3.	Coverage of SCN-1440 with a minimum elevation angle of 30° [46].	20
2.4.	Population density statistics in the year 2020, with a resolution of 15 arc-minutes [50]. Applied filtering function limits maximum number of persons per square kilometer to 100 [51].	21
2.5.	Visualization of UT and GW distributions. 10 000 UT locations (in blue) have been sampled from the filtered population density distribution. 39 GW locations (in yellow) have been chosen deliberately.	22
2.6.	Pie chart of assumed traffic shares and resulting QoS service class shares.	25
3.1.	Rectangular arrangement of clusters for SCN-288 using a cluster size of 12. Nodes of the same cluster are colored identically. The region spanned by a cluster is shown (in red). Satellites with SDN controller role are highlighted (orange ring). [51]	44
3.2.	Exemplary path from Munich to Singapore across several clusters in SCN-288. From geographical source cell (red rectangle) to destination cell (blue rectangle). Forwarded via ISLs (black lines) and ingress/egress ESLs (cyan lines). [51]	45
3.3.	Conceptual flowchart of packet handling: two-tiered routing table lookup. A geographical and an intra-routing table are used.	49

3.4.	Schematic representation of considered multicast extension. The black arrows highlight how multicast traffic is forwarded from its source (a GW) to the RP (a cluster controller). The red arrows show the traffic split at the RP towards egress satellites. The green arrows represent downlink ESLs to multicast destinations on ground (UTs).	50
3.5.	QoS comparisons for increasing network loads for density-based UT distribution (in blue), and a uniform UT distribution (in orange). The latter derives from the reference scenario. SPF source routing used in both cases.	59
3.6.	Observed link utilization of ISLs of a snapshot using SPF source-routed scheme. Width of ISLs (magenta) scaled according to link utilization.	60
3.7.	QoS comparisons for increasing network loads for SPF source routing (in blue) and set-based source routing (in orange).	60
3.8.	End-to-end latency of sessions for SPF source routing (left plot) and a set-based source routing (right plot). Both schemes do not differentiate QoS classes. Comparing average (green triangle), median (orange line), minimum and maximum (caps).	61
3.9.	Schematic diagram of the architecture of the system simulator, including input and output files, as well as processing scripts.	68
3.10.	Box plot of end-to-end latency of sessions for F-IDLB with a cluster size of 48 (left plot) and the upper theoretical benchmark GEVR (right plot). Both schemes use QoS-aware routing.	69
3.11.	Delay variance of sessions for F-IDLB with a cluster size of 48 (left plot) and the upper theoretical benchmark GEVR (right plot). Both schemes use QoS-aware routing. Outliers due to specific handover situations are expected.	70
3.12.	Network load and system throughput for source-routing in SCN-288. Longer window with a simulated time of 7200s, and shorter window with simulated time of 720s are shown. Due to uniform session distribution and margins dictating simulation start/end times, shorter window results in slightly higher loads.	72
3.13.	QoS comparisons for increasing network loads for IDLB with a cluster size of 12 (in orange), a cluster size of 24 (in green), a cluster size of 48 (in red), as well as source-routing (in blue), and GEVR (in violet). Individual sub-plots for each QoS class: delay-critical forwarding (QoS1), forwarding (QoS2), best effort (QoS3). Source-routing approach is not QoS-aware.	73
3.14.	QoS comparisons for increasing network loads for per-packet IDLB (P-IDLB in blue) with a cluster size of 12, and per-flow IDLB (F-IDLB in orange) with a cluster size of 12.	74
3.15.	QoS comparisons for increasing network loads for a per-area determined NHCN mapping with a cluster sizes of 24 (blue) and of 48 (green), as well as randomized per-flow NHCN mapping with a cluster size of 24 (orange) and of 48 (red). The determined mapping with a cluster size of 48 also shows buffer limitations (F-IDLB* in green).	75
3.16.	QoS comparisons for increasing network loads for SCN-288 and SCN-1440 constellations. SCN-1440 constellation uses 48-node clusters (in green), while for SCN-288 both 48-node clusters (in orange) and 12-node clusters (in blue) are used. The latter included due to the similar physical size to 48-node clusters in SCN-1440.	76

3.17. Schematic of demonstrator. Test devices, representing the satellites, are connected to core switch which connects to virtualization cluster.	79
3.18. Test setup for dynamic handover: topology before (on the left) and after (on the right) topology change. Satellite router 1 (sat1) and satellite router 3 (sat3) change relative positions.	80
3.19. Test setup for QoS-based load-balancing: initial flow state. Two low-priority flows are routed through the network. Flow 1 from acm1 to acm3; flow 2 from acm2 to acm4.	81
3.20. Test setup for QoS-based load-balancing: adjusted routes after saturating flow. Two low-priority flows were re-routed to avoid saturated link between sat2 and sat3, which was then used by high-priority flow 3.	82
3.21. Received load during QoS-based load-balancing test on sat2 and sat3 . Device under test sat2 (left plot) and sat3 (right plot) are shown.	82
3.22. Observed link utilization of ISLs of a snapshot using IDLB with 24-node clusters. Width of ISLs (magenta) scaled according to link utilization.	85
4.1. Overview of classification of DRL algorithms investigated in this work. Policy-based DRL methods grayed out, as they are not considered.	90
4.2. Generic DRL-based architecture for routing in satellite networks. Depending on approach, action is either next hop or route.	93
4.3. Diagram of centralized constrained DQN architecture with satellite network environment. Action set constrained to correct routing paths.	99
4.4. Abstracted sub-network representation: satellites (green) with ISLs (green-dotted), and link load (color according to load level).	103
4.5. Smoothed rewards over training for different DQN-based approaches in static scenarios, highlighting differences in convergence.	104
4.6. Comparison of FD-MADRL (in green) and a rule-based multi-cost approach (in red) in terms of latency (number of hops, on the x-axis) and the resulting maximum link load on the chosen path (y-axis).	105
4.7. Smoothed rewards over training for FD-MADRL with dynamic, mutable link loads.	105
4.8. Diagram of proposed path-based routing architecture. A centralized agent using a soft actor-critic framework interacts with the satellite network environment.	110
4.9. Heat map of traffic sources and destinations for 48-node network. Shows normalized probability of each node to act as source or destination. Node indices represented by row and column numbers.	114
4.10. Comparison of mean episode rewards for candidate path algorithms in scenario using deterministic set of flows. The plot shows both the raw reward data (lighter lines) and the smoothed reward data (solid lines). Sets (a) (in blue), (b) (in orange), and (c) (in green) are shown. The smoothed data is obtained using a moving average with a window size of 50 steps.	116
4.11. Static scenario: box plot for link utilization of deterministic set of 50 flows. Benchmark is solution to MCFP using complete set, other approaches make step-wise decisions.	117
4.12. Histogram of link utilization for exemplary set of 50 flows. Resulting distribution of benchmark solution (left) and proposed SAC approach (right).	118

List of Figures

4.13. Reward progression during training of the policies using the SAC algorithm. The plot shows both the raw reward data (lighter lines) and the smoothed reward data (solid lines) for each policy: the histogram policy (in blue) and the one-hot encoding policy (in orange). The smoothed data is obtained using a moving average with a window size of 100 steps.	119
4.14. Randomized scenario: box plot for total routed demand. Comparison of dif- ferent routing methods for iteratively routing demands until overflow.	119
4.15. Large heat map of traffic sources and destinations for 288-node network. Shows normalized probability of each node to act as source or destination. Node indices represented by row and column numbers.	120
4.16. Monte Carlo simulation: comparison of total routed demand for 24×12 grid.	121
A.1. Block diagram of Layerscape LX2160A provided by NXP [58,142].	129

List of Tables

2.1.	Summary of space segment of SCN-288 and SCN-1440.	18
2.2.	Summary of ground segment and coverage of reference system.	20
2.3.	Summary of connectivity and data rates of reference system.	24
2.4.	Summary of considered QoS classes. Some values from standards have been slightly adapted based on observations and SCN investigations [8,35].	27
2.5.	Overview of KPIs. Considered primary and secondary metrics.	32
3.1.	Summary of IDLB protocol design components and features.	46
3.2.	Simulation parameters of set-based source routing analysis.	58
3.3.	Comparison of convergence rate and overall packet dropping rates for different cluster sizes in SCN-288 at a network load of 13.2 Gbps.	74
3.4.	Summary of devices involved in hardware test bed.	78
3.5.	Summary of functional evaluation of IDLB demonstrator.	79
3.6.	Overview of impact of reference system parameters on individual KPIs.	86
4.1.	Summary of investigated DRL approaches.	94
4.2.	Considered sets of candidate paths. The overlap indicates the proportion of nodes a path can share with other paths.	112
4.3.	Hyperparameters of the SAC model for one-hot encoding and histogram policies using $k = 4$ pre-computed candidate paths.	115
A.1.	Applications of LX2160A as 5G packet processing component [142,143].	129
A.1.	General configuration used by simulator for routing protocol analyses.	131

Bibliography

- [1] A. Knopp, C. Hofmann, and H. Reder, “HEUMEGA - Unabhängige Trendanalyse zum Thema Megakonstellationen,” https://www.dlr.de/rd/desktopdefault.aspx/tabid-18004/28579_read-74147/, 2021.
- [2] D. Giggenbach, A. Knopp, W. Mohr, M. T. Knopp, S. Schuster, and S. Hoppe, “NeSC – NewSpace Communications,” <https://www.vde.com/de/itg/publikationen/studien/vde-positionspapier-nesc-new-space-communications>, Oct. 2023.
- [3] T. Butash, P. Garland, and B. Evans, “Non-geostationary satellite orbit communications satellite constellations history,” *International Journal of Satellite Communications and Networking*, vol. 39, no. 1, pp. 1–5, 2021.
- [4] N. Pachler, I. Del Portillo, E. F. Crawley, and B. G. Cameron, “An Updated Comparison of Four Low Earth Orbit Satellite Constellation Systems to Provide Global Broadband,” in *2021 IEEE International Conference on Communications Workshops (ICC Workshops)*, Jun. 2021, pp. 1–7.
- [5] R. Leopold and A. Miller, “The IRIDIUM communications system,” *IEEE Potentials*, vol. 12, no. 2, pp. 6–9, Apr. 1993.
- [6] Rivada, “Rivada - Our Technology,” <https://rivadaspace.com/our-technology>, 2025.
- [7] J. Ferragut, G. Montesanti, A. Ginesi, D. Roques, P.-D. Arapoglou, S. Cioni, M. Crisci, G. Dauron, N. Guillermin, A. Hanan, A. Mengali, A. Quiles, F. P. Romero, R. Sallantin, S. Scarda, J. Godet, and G. Salgado, “IRIS² – The New EU Programme Providing Secure Communications Via Satellites,” <https://publications.jrc.ec.europa.eu/repository/handle/JRC139380>, 2024.
- [8] M. Handley, “Delay is Not an Option: Low Latency Routing in Space,” in *Proceedings of the 17th ACM Workshop on Hot Topics in Networks*, ser. HotNets ’18. New York, NY, USA: Association for Computing Machinery, Nov. 2018, pp. 85–91.
- [9] V. Singh and A. Gadre, “On the Feasibility of Laser Inter-satellite Links for Low-latency High Frequency Trading,” in *Proceedings of the 2nd International Workshop on LEO Networking and Communication*, ser. LEO-NET ’24. New York, NY, USA: Association for Computing Machinery, Nov. 2024, pp. 37–42.
- [10] Rivada, “Rivada Tapped for Virtual Network Operator Contract with U.S. Navy,” <https://rivadaspace.com/news/us-navy-virtual-network-operator>, 2025.
- [11] S. Erwin, “Starlink set to hit \$11.8 billion revenue in 2025, boosted by military contracts,” <https://spacenews.com/starlink-set-to-hit-11-8-billion-revenue-in-2025-boosted-by-military-contracts/>, Dec. 2024.

- [12] 3GPP, “3GPP Release-19 5G,” <https://www.3gpp.org/specifications-technologies/releases/release-19>, 2025.
- [13] —, “3GPP Release-20 5G Advanced,” <https://www.3gpp.org/specifications-technologies/releases/release-20>, 2025.
- [14] 3GPP, “Solutions for NR to support Non-Terrestrial Networks (NTN),” 3GPP, Tech. Rep. TR 38.821 V16.0.0, Dec. 2019.
- [15] G. Giambene, S. Kota, and P. Pillai, “Satellite-5G Integration: A Network Perspective,” *IEEE Network*, vol. 32, no. 5, pp. 25–31, Sep. 2018.
- [16] K. T. Li, C. A. Hofmann, H. Reder, and A. Knopp, “A Techno-Economic Assessment and Tradespace Exploration of Low Earth Orbit Mega-Constellations,” *IEEE Communications Magazine*, vol. 61, no. 2, pp. 24–30, Feb. 2023.
- [17] L. Izhikevich, R. Enghardt, T.-Y. Huang, and R. Teixeira, “A Global Perspective on the Past, Present, and Future of Video Streaming over Starlink,” *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 8, no. 3, pp. 1–22, Dec. 2024.
- [18] J. Garcia, S. Sundberg, G. Caso, and A. Brunstrom, “Multi-Timescale Evaluation of Starlink Throughput,” in *Proceedings of the 1st ACM Workshop on LEO Networking and Communication*, ser. LEO-NET ’23. New York, NY, USA: Association for Computing Machinery, Oct. 2023, pp. 31–36.
- [19] T. De Cola, “Enabling Effective Multi-Link Data Distribution in NTN-based 6G Networks,” in *WSA & SCC 2023; 26th International ITG Workshop on Smart Antennas and 13th Conference on Systems, Communications, and Coding*, Feb. 2023, pp. 1–5.
- [20] A. U. Chaudhry and H. Yanikomeroglu, “Laser Intersatellite Links in a Starlink Constellation: A Classification and Analysis,” *IEEE Vehicular Technology Magazine*, vol. 16, no. 2, pp. 48–56, Jun. 2021.
- [21] M. Werner, J. Frings, F. Wauquiez, and G. Maral, “Topological design, routing and capacity dimensioning for ISL networks in broadband LEO satellite systems,” *International Journal of Satellite Communications*, vol. 19, no. 6, pp. 499–527, 2001.
- [22] D. Bhattacharjee, P. G. Madoery, A. U. Chaudhry, H. Yanikomeroglu, G. K. Kurt, P. Hu, K. Ahmed, and S. Martel, “On-Demand Routing in LEO Mega-Constellations With Dynamic Laser Inter-Satellite Links,” *IEEE Transactions on Aerospace and Electronic Systems*, pp. 1–17, 2024.
- [23] C. H. Papadimitriou and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity*. Courier Corporation, Jan. 1998.
- [24] P. Grislain, N. Pelissier, F. Lamothe, O. Hotescu, J. Lacan, E. Lochin, and J. Radzik, “Rethinking LEO Constellations Routing with the Unsplittable Multi-Commodity Flows Problem,” in *2022 11th Advanced Satellite Multimedia Systems Conference and the 17th Signal Processing for Space Communications Workshop (ASMS/SPSC)*, Sep. 2022, pp. 1–8.

- [25] L. Bertaux, S. Medjiah, P. Berthou, S. Abdellatif, A. Hakiri, P. Gelard, F. Plan-chou, and M. Bruyere, “Software defined networking and virtualization for broadband satellite networks,” *IEEE Communications Magazine*, vol. 53, no. 3, pp. 54–60, Mar. 2015.
- [26] R. Ferrús, H. Koumaras, O. Sallent, G. Agapiou, T. Rasheed, M.-A. Kourtis, C. Boustie, P. Gélard, and T. Ahmed, “SDN/NFV-enabled satellite communications networks: Opportunities, scenarios and challenges,” *Physical Communication*, vol. 18, pp. 95–112, 2016.
- [27] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. MIT press, 2018.
- [28] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.
- [29] O. Simeone, *Machine Learning for Engineers*. Cambridge University Press, Nov. 2022.
- [30] B. Christian, *The Alignment Problem: Machine Learning and Human Values*. W. W. Norton & Company, Oct. 2020.
- [31] M. T. Islam, Z. Zhou, H. Ren, M. B. Khuzani, D. Kapp, J. Zou, L. Tian, J. C. Liao, and L. Xing, “Revealing hidden patterns in deep neural network feature space continuum via manifold learning,” *Nature Communications*, vol. 14, no. 1, p. 8506, Dec. 2023.
- [32] C. Larsson, *Design of Modern Communication Networks: Methods and Applications*. Academic Press, 2014.
- [33] R. Balestriero, J. Pesenti, and Y. LeCun, “Learning in High Dimension Always Amounts to Extrapolation,” *arXiv:2110.09485 [cs]*, Oct. 2021.
- [34] M. M. H. Roth, H. Brandt, and H. Bischl, “Distributed SDN-based Load-balanced Routing for Low Earth Orbit Satellite Constellation Networks,” in *2022 11th Advanced Satellite Multimedia Systems Conference and the 17th Signal Processing for Space Communications Workshop (ASMS/SPSC)*, Sep. 2022, pp. 1–8.
- [35] M. M. H. Roth, “Analyzing Source-Routed Approaches for Low Earth Orbit Satellite Constellation Networks,” in *Proceedings of the 1st ACM Workshop on LEO Networking and Communication*, ser. LEO-NET ’23. New York, NY, USA: Association for Computing Machinery, Oct. 2023, pp. 43–48.
- [36] M. M. H. Roth, A. Hegde, T. Delamotte, and A. Knopp, “Shaping Rewards, Shaping Routes: On Multi-Agent Deep Q-Networks for Routing in Satellite Constellation Networks,” in *Proceedings of SPAICE2024: The First Joint European Space Agency / IAA Conference on AI in and for Space*. Harwell, UK: Zenodo, Oct. 2024, pp. 100–104.
- [37] M. M. H. Roth, T. Jerkovits, A. Hegde, T. Delamotte, and A. Knopp, “Deep Reinforcement Learning for Adaptive Traffic Engineering in Satellite Constellation Networks,” in *2025 12th Advanced Satellite Multimedia Systems Conference and the 18th Signal Processing for Space Communications Workshop (ASMS/SPSC)*, Feb. 2025, pp. 1–8.
- [38] A. Hegde, M. Roth, and H. Bischl, “Routing in Low Earth Orbit Satellite Networks:

- Constrained DQN-Based Approach,” in *2025 IEEE Wireless Communications and Networking Conference (WCNC)*, Mar. 2025, pp. 1–6.
- [39] M. M. H. Roth, H. Brandt, and H. Bischl, “Implementation of a geographical routing scheme for low Earth orbiting satellite constellations using intersatellite links,” *International Journal of Satellite Communications and Networking*, vol. 39, no. 1, pp. 92–107, 2021.
- [40] M. Roth, H. Brandt, H. Bischl, D. F. Piñas, and G. Acar, “IDLB: An SDN-based Load-balancing Routing Protocol for Autonomous Satellite Constellation Networks,” Apr. 2025.
- [41] J. G. Walker, “Satellite constellations,” *Journal of the British Interplanetary Society*, vol. 37, p. 559, 1984.
- [42] E. Lutz, M. Werner, and A. Jahn, *Satellite Systems for Personal and Broadband Communications*. Springer Science & Business Media, Dec. 2012.
- [43] R. Leopold, “The Iridium Communications Systems,” in *[Proceedings] Singapore ICCS/ISITA ‘92*, 1992, pp. 451–455 vol.2.
- [44] C. Fossa, R. Raines, G. Gunsch, and M. Temple, “An overview of the IRIDIUM (R) low Earth orbit (LEO) satellite system,” in *Proceedings of the IEEE 1998 National Aerospace and Electronics Conference. NAECON 1998. Celebrating 50 Years (Cat. No.98CH36185)*, Jul. 1998, pp. 152–159.
- [45] G. Maral, M. Bousquet, and Z. Sun, *Satellite Communications Systems: Systems, Techniques and Technology*. John Wiley & Sons, Apr. 2020.
- [46] L. Wood, “SaVi: Satellite constellation visualization,” Apr. 2012.
- [47] ESA, “Routing and management protocols for large constellations with inter-satellite links (ROPRO),” <https://connectivity.esa.int/projects/ropro>, 2025.
- [48] —, “AI for satellite 5G communications (AIComS),” <https://connectivity.esa.int/projects/aicoms>, 2025.
- [49] —, “Optical feeder link site-diversity protocols for satellite constellation networks (OFELIAS),” <https://connectivity.esa.int/projects/ofelias>, 2025.
- [50] Center for International Earth Science Information Network - CIESIN - Columbia University, “Gridded Population of the World, Version 4 (GPWv4): Population Density, Revision 11,” 2018.
- [51] Met Office, *Cartopy: A Cartographic Python Library with a Matplotlib Interface*, Exeter, Devon, 2010.
- [52] N. Kaur and G. Soni, “Performance analysis of inter-satellite optical wireless communication (IsOWC) system at 980 nm and 1550 nm wavelengths,” in *2014 International Conference on Contemporary Computing and Informatics (IC3I)*, Nov. 2014, pp. 1245–1250.
- [53] Y. Qi, M. Hunukumbure, M. Nekovee, J. Lorca, and V. Sgardoni, “Quantifying data

- rate and bandwidth requirements for immersive 5G experience,” in *2016 IEEE International Conference on Communications Workshops (ICC)*, May 2016, pp. 455–461.
- [54] N. Seitz, “ITU-T QoS standards for IP-based networks,” *IEEE Communications Magazine*, vol. 41, no. 6, pp. 82–89, Jun. 2003.
 - [55] ITU, “G.114 : One-way transmission time,” ITU-T, G.114, May 2003.
 - [56] —, “Y.1541 : Network performance objectives for IP-based services,” ITU-T, Y.1541, Dec. 2011.
 - [57] F. Baker, D. L. Black, K. Nichols, and S. L. Blake, “Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers,” Internet Engineering Task Force, Request for Comments RFC 2474, Dec. 1998.
 - [58] NXP, “LX2160A, LX2120A, LX2080A,” <https://www.nxp.com/products/LX2160A>, 2025.
 - [59] E. W. Dijkstra, “A note on two problems in connexion with graphs,” *Numerische mathematik*, vol. 1, no. 1, pp. 269–271, 1959.
 - [60] A. Donner, M. Berioli, and M. Werner, “MPLS-based satellite constellation networks,” *IEEE Journal on Selected Areas in Communications*, vol. 22, no. 3, pp. 438–448, Apr. 2004.
 - [61] N. Gvozdiev, S. Vissicchio, B. Karp, and M. Handley, “Low-Latency Routing on Mesh-Like Backbones,” in *Proceedings of the 16th ACM Workshop on Hot Topics in Networks*, ser. HotNets-XVI. New York, NY, USA: Association for Computing Machinery, Nov. 2017, pp. 136–142.
 - [62] T. Li, “A Routing Architecture for Satellite Networks,” Internet Engineering Task Force, Request for Comments RFC 9717, Jan. 2025.
 - [63] C. Filsfil, N. K. Nainar, C. Pignataro, J. C. Cardona, and P. Francois, “The Segment Routing Architecture,” in *2015 IEEE Global Communications Conference (GLOBECOM)*, Dec. 2015, pp. 1–6.
 - [64] C. Maihofer, “A survey of geocast routing protocols,” *IEEE Communications Surveys Tutorials*, vol. 6, no. 2, pp. 32–42, 2004.
 - [65] B. Meijerink, M. Baratchi, and G. Heijenk, “An Efficient Geographical Addressing Scheme for the Internet,” in *Wired/Wireless Internet Communications*, ser. Lecture Notes in Computer Science, L. Mamatas, I. Matta, P. Papadimitriou, and Y. Koucheryavy, Eds. Cham: Springer International Publishing, 2016, pp. 78–90.
 - [66] T. Henderson and R. Katz, “On distributed, geographic-based packet routing for LEO satellite networks,” in *Globecom '00 - IEEE. Global Telecommunications Conference. Conference Record (Cat. No.00CH37137)*, vol. 2, Nov. 2000, pp. 1119–1123 vol.2.
 - [67] H. Tsunoda, K. Ohta, N. Kato, and Y. Nemoto, “Supporting IP/LEO satellite networks by handover-independent IP mobility management,” *IEEE Journal on Selected Areas in Communications*, vol. 22, no. 2, pp. 300–307, Feb. 2004.

- [68] —, “Geographical and Orbital Information Based Mobility Management to Overcome Last-Hop Ambiguity over IP/LEO Satellite Networks,” in *2006 IEEE International Conference on Communications*, vol. 4, Jun. 2006, pp. 1849–1854.
- [69] E. Ekici, I. Akyildiz, and M. Bender, “A distributed routing algorithm for datagram traffic in LEO satellite networks,” *IEEE/ACM Transactions on Networking*, vol. 9, no. 2, pp. 137–147, Apr. 2001.
- [70] H. An, T. Zhang, and J. Yang, “Priority-based distributed dynamic multipath routing for LEO satellite networks,” in *9th International Conference on Communications and Networking in China*, Aug. 2014, pp. 316–320.
- [71] M. Marina and S. Das, “On-demand multipath distance vector routing in ad hoc networks,” in *Proceedings Ninth International Conference on Network Protocols. ICNP 2001*, Nov. 2001, pp. 14–23.
- [72] T. Taleb, D. Mashimo, A. Jamalipour, K. Hashimoto, Y. Nemoto, and N. Kato, “Sat04-3: Elb: An explicit load balancing routing protocol for multi-hop ngeo satellite constellations,” in *IEEE Globecom 2006*. IEEE, 2006, pp. 1–5.
- [73] P. E. Benjamin, S. Abishek, and C. Ananth, “Traffic Light Based Intelligent Routing Strategy for Satellite Network,” *International Journal of Advanced Research in Biology, Ecology, Science and Technology (IJARBEST)*, vol. 1, pp. 24–27, 2015.
- [74] L. Davoli, L. Veltri, P. L. Ventre, G. Siracusano, and S. Salsano, “Traffic Engineering with Segment Routing: SDN-based Architectural Design and Open Source Implementation,” Dec. 2015.
- [75] N. Alzaben and D. W. Engels, “End-to-End Routing in SDN Controllers Using Max-Flow Min-Cut Route Selection Algorithm,” in *2021 23rd International Conference on Advanced Communication Technology (ICACT)*. IEEE, 2021, pp. 461–467.
- [76] M. Corici, H. Buhr, H. Zope, and M. Zaboub, “An SDN-Based Solution for Mega-Constellation Routing,” in *2024 IEEE 35th International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, Sep. 2024, pp. 1–6.
- [77] M. Condoluci and T. Mahmoodi, “Softwarization and virtualization in 5G mobile networks: Benefits, trends and challenges,” *Computer Networks*, vol. 146, pp. 65–84, Dec. 2018.
- [78] M. Corici, M. Caus, X. Artiga, A. Guidotti, B. Barth, T. de Cola, J. Tallon, H. Zope, D. Tarchi, F. Parzysz, D. Naseh, and S. Sadashiv Shinde, “Transforming 5G Mega-Constellation Communications: A Self-Organized Network Architecture Perspective,” *IEEE Access*, vol. 13, pp. 14 770–14 788, 2025.
- [79] F. Bannour, S. Souihi, and A. Mellouk, “Distributed SDN Control: Survey, Taxonomy, and Challenges,” *IEEE Communications Surveys Tutorials*, vol. 20, no. 1, pp. 333–354, 2018.
- [80] B. J. Barritt and W. Eddy, “SDN Enhancements for LEO Satellite Networks,” in *34th AIAA International Communications Satellite Systems Conference*. Cleveland, OH: American Institute of Aeronautics and Astronautics, Oct. 2016.

- [81] A. Papa, T. De Cola, P. Vizarreta, M. He, C. Mas Machuca, and W. Kellerer, “Dynamic SDN Controller Placement in a LEO Constellation Satellite Network,” in *2018 IEEE Global Communications Conference (GLOBECOM)*, Dec. 2018, pp. 206–212.
- [82] A. Papa, T. De Cola, P. Vizarreta, M. He, C. Mas-Machuca, and W. Kellerer, “Design and Evaluation of Reconfigurable SDN LEO Constellations,” *IEEE Transactions on Network and Service Management*, vol. 17, no. 3, pp. 1432–1445, Sep. 2020.
- [83] X. Cao, Y. Li, X. Xiong, and J. Wang, “Dynamic Routings in Satellite Networks: An Overview,” *Sensors*, vol. 22, no. 12, p. 4552, Jan. 2022.
- [84] M. Á. Vázquez, P. Henarejos, I. Pappalardo, E. Grechi, J. Fort, J. C. Gil, and R. M. Lancellotti, “Machine Learning for Satellite Communications Operations,” *IEEE Communications Magazine*, vol. 59, no. 2, pp. 22–27, Feb. 2021.
- [85] C. Zhang, P. Patras, and H. Haddadi, “Deep Learning in Mobile and Wireless Networking: A Survey,” *IEEE Communications Surveys & Tutorials*, vol. 21, no. 3, pp. 2224–2287, 2019.
- [86] E. Muscinelli, S. S. Shinde, and D. Tarchi, “Overview of Distributed Machine Learning Techniques for 6G Networks,” *Algorithms*, vol. 15, no. 6, p. 210, Jun. 2022.
- [87] 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 Communications Surveys & Tutorials*, vol. 21, no. 1, pp. 393–430, 2019.
- [88] S. Troia, A. Rodriguez, I. Martín, J. A. Hernández, O. G. De Dios, R. Alvizu, F. Musumeci, and G. Maier, “Machine-Learning-Assisted Routing in SDN-Based Optical Networks,” in *2018 European Conference on Optical Communication (ECOC)*, Sep. 2018, pp. 1–3.
- [89] A. Valadarsky, M. Schapira, D. Shahaf, and A. Tamar, “A Machine Learning Approach to Routing,” *arXiv:1708.03074 [cs]*, Nov. 2017.
- [90] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.
- [91] X. You, X. Li, Y. Xu, H. Feng, J. Zhao, and H. Yan, “Toward Packet Routing With Fully Distributed Multiagent Deep Reinforcement Learning,” *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 52, no. 2, pp. 855–868, Feb. 2022.
- [92] F. Geyer and G. Carle, “Learning and Generating Distributed Routing Protocols Using Graph-Based Deep Learning,” in *Proceedings of the 2018 Workshop on Big Data Analytics and Machine Learning for Data Communication Networks*, ser. Big-DAMA ’18. New York, NY, USA: Association for Computing Machinery, Aug. 2018, pp. 40–45.
- [93] M. Liu, J. Li, and H. Lu, “Routing in Small Satellite Networks: A GNN-based Learning Approach,” Aug. 2021.
- [94] H. Wang, Y. Ran, L. Zhao, J. Wang, J. Luo, and T. Zhang, “GRouting: Dynamic Routing for LEO Satellite Networks with Graph-based Deep Reinforcement Learn-

- ing,” in *2021 4th International Conference on Hot Information-Centric Networking (HotICN)*, Nov. 2021, pp. 123–128.
- [95] B. Soret, I. Leyva-Mayorga, F. Lozano-Cuadra, and M. D. Thorsager, “Q-learning for distributed routing in LEO satellite constellations,” Jun. 2023.
 - [96] P. Zuo, C. Wang, Z. Yao, S. Hou, and H. Jiang, “An Intelligent Routing Algorithm for LEO Satellites Based on Deep Reinforcement Learning,” in *2021 IEEE 94th Vehicular Technology Conference (VTC2021-Fall)*, Sep. 2021, pp. 1–5.
 - [97] G. Xu, Y. Zhao, Y. Ran, R. Zhao, and J. Luo, “Towards Spatial Location Aided Fully-Distributed Dynamic Routing for LEO Satellite Networks,” in *GLOBECOM 2022 - 2022 IEEE Global Communications Conference*, Dec. 2022, pp. 1570–1575.
 - [98] X. Shi, P. Ren, and Q. Du, “Heterogeneous Satellite Network Routing Algorithm Based on Reinforcement Learning and Mobile Agent,” in *2020 IEEE Globecom Workshops (GC Wkshps)*, Dec. 2020, pp. 1–6.
 - [99] P. Zuo, C. Wang, Z. Wei, Z. Li, H. Zhao, and H. Jiang, “Deep Reinforcement Learning Based Load Balancing Routing for LEO Satellite Network,” in *2022 IEEE 95th Vehicular Technology Conference: (VTC2022-Spring)*, Jun. 2022, pp. 1–6.
 - [100] K.-C. Tsai, L. Fan, L.-C. Wang, R. Lent, and Z. Han, “Multi-Commodity Flow Routing for Large-Scale LEO Satellite Networks Using Deep Reinforcement Learning,” in *2022 IEEE Wireless Communications and Networking Conference (WCNC)*, Apr. 2022, pp. 626–631.
 - [101] K.-C. Tsai, T.-J. Yao, P.-H. Huang, C.-S. Huang, Z. Han, and L.-C. Wang, “Deep Reinforcement Learning-Based Routing for Space-Terrestrial Networks,” in *2022 IEEE 96th Vehicular Technology Conference (VTC2022-Fall)*, Sep. 2022, pp. 1–5.
 - [102] B. Soykan and G. Rabadi, “Optimizing Multi Commodity Flow Problem Under Uncertainty: A Deep Reinforcement Learning Approach,” in *2023 International Conference on Machine Learning and Applications (ICMLA)*, Dec. 2023, pp. 1267–1272.
 - [103] R. E. Ali, B. Erman, E. Baştuğ, and B. Cilli, “Hierarchical Deep Double Q-Routing,” arXiv, Tech. Rep. arXiv:1910.04041, Mar. 2020.
 - [104] A. Cigliano and F. Zampognaro, “A Machine Learning approach for routing in satellite Mega-Constellations,” in *2020 International Symposium on Advanced Electrical and Communication Technologies (ISAECT)*, Nov. 2020, pp. 1–6.
 - [105] T. Darwish, G. K. Kurt, H. Yanikomeroglu, M. Bellemare, and G. Lamontagne, “LEO Satellites in 5G and Beyond Networks: A Review from a Standardization Perspective,” Oct. 2021.
 - [106] F. Tang, B. Mao, Y. Kawamoto, and N. Kato, “Survey on Machine Learning for Intelligent End-to-End Communication Toward 6G: From Network Access, Routing to Traffic Control and Streaming Adaption,” *IEEE Communications Surveys & Tutorials*, vol. 23, no. 3, pp. 1578–1598, 2021.
 - [107] Q. Sun, N. Li, C.-L. I, J. Huang, X. Xu, and Y. Xie, “Intelligent RAN Automation

- for 5G and Beyond,” *IEEE Wireless Communications*, vol. 31, no. 1, pp. 94–102, Feb. 2024.
- [108] R. Campana, C. Amatetti, and A. Vanelli-Coralli, “RAN Functional Splits in NTN: Architectures and Challenges,” Sep. 2023.
 - [109] 3GPP, “Specification # 38.821,” Technical Specification Group Radio Access Network, Tech. Rep. 3GPP TR 38.821, Mar. 2023.
 - [110] —, “Specification # 23.501,” Technical Specification Group Services and System Aspects, Tech. Rep. 3GPP TS 23.501, Jun. 2025.
 - [111] —, “Specification # 23.700-29,” Technical Specification Group Services and System Aspects, Tech. Rep. 3GPP TR 23.700-29, Jun. 2024.
 - [112] M. H. Xu, Y. Q. Liu, Q. L. Huang, Y. X. Zhang, and G. F. Luan, “An improved Dijkstra’s shortest path algorithm for sparse network,” *Applied Mathematics and Computation*, vol. 185, no. 1, pp. 247–254, Feb. 2007.
 - [113] K. Salimifard and S. Bigharaz, “The multicommodity network flow problem: State of the art classification, applications, and solution methods,” *Operational Research*, vol. 22, no. 1, pp. 1–47, Mar. 2022.
 - [114] B. Fenner, M. J. Handley, I. Kouvelas, and H. Holbrook, “Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification (Revised),” Internet Engineering Task Force, Request for Comments RFC 4601, Aug. 2006.
 - [115] W. Ma and Y. Fang, “Dynamic hierarchical mobility management strategy for mobile IP networks,” *IEEE Journal on Selected Areas in Communications*, vol. 22, no. 4, pp. 664–676, May 2004.
 - [116] X. Zhang, J. Gomez Castellanos, and A. T. Campbell, “P-MIP: Paging Extensions for Mobile IP,” *Mobile Networks and Applications*, vol. 7, no. 2, pp. 127–141, Apr. 2002.
 - [117] J. Zhang, F. R. Yu, S. Wang, T. Huang, Z. Liu, and Y. Liu, “Load Balancing in Data Center Networks: A Survey,” *IEEE Communications Surveys Tutorials*, vol. 20, no. 3, pp. 2324–2352, 2018.
 - [118] D. Bhattacharjee and A. Singla, “Network topology design at 27,000 km/hour,” in *Proceedings of the 15th International Conference on Emerging Networking Experiments And Technologies*, ser. CoNEXT ’19. New York, NY, USA: Association for Computing Machinery, Dec. 2019, pp. 341–354.
 - [119] D. Eppstein, “Finding the k Shortest Paths,” *SIAM Journal on computing*, no. 28.2, p. 26, 1997.
 - [120] S. Hoceini, A. Mellouk, and Y. Amirat, “K-Shortest Paths Q-Routing: A New QoS Routing Algorithm in Telecommunication Networks,” in *Networking - ICN 2005*, ser. Lecture Notes in Computer Science, P. Lorenz and P. Dini, Eds. Berlin, Heidelberg: Springer, 2005, pp. 164–172.
 - [121] P. E. Hart, N. J. Nilsson, and B. Raphael, “A Formal Basis for the Heuristic Determination of Minimum Cost Paths,” *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, Jul. 1968.

- [122] J. D. C. Little and S. C. Graves, “Little’s Law,” in *Building Intuition: Insights From Basic Operations Management Models and Principles*, ser. International Series in Operations Research & Management Science, D. Chhajed and T. J. Lowe, Eds. Boston, MA: Springer US, 2008, pp. 81–100.
- [123] J. Li, X. Chang, Y. Ren, Z. Zhang, and G. Wang, “An Effective Path Load Balancing Mechanism Based on SDN,” in *2014 IEEE 13th International Conference on Trust, Security and Privacy in Computing and Communications*, Sep. 2014, pp. 527–533.
- [124] Y. Azar, A. Z. Broder, A. R. Karlin, and E. Upfal, “Balanced allocations (extended abstract),” in *Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing - STOC ’94*. Montreal, Quebec, Canada: ACM Press, 1994, pp. 593–602.
- [125] B. Vöcking, “How asymmetry helps load balancing,” *Journal of the ACM*, vol. 50, no. 4, pp. 568–589, Jun. 2003.
- [126] Ryu developers, “Ryu SDN Framework,” <https://ryu-sdn.org/>, 2025.
- [127] J. Schmidhuber, “Deep learning in neural networks: An overview,” *Neural Networks*, vol. 61, pp. 85–117, Jan. 2015.
- [128] Y. Bengio, “Learning Deep Architectures for AI,” *Foundations and Trends® in Machine Learning*, vol. 2, no. 1, pp. 1–127, Nov. 2009.
- [129] F. Chollet, “On the Measure of Intelligence,” Nov. 2019.
- [130] J. Boyan and M. Littman, “Packet Routing in Dynamically Changing Networks: A Reinforcement Learning Approach,” in *Advances in Neural Information Processing Systems*, vol. 6. Morgan-Kaufmann, 1993.
- [131] A. Bitailou, B. Parrein, and G. Andrieux, “Q-routing: From the Algorithm to the Routing Protocol,” in *Machine Learning for Networking*, ser. Lecture Notes in Computer Science, S. Boumerdassi, E. Renault, and P. Mühlethaler, Eds. Cham: Springer International Publishing, 2020, pp. 58–69.
- [132] M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, O. Pieter Abbeel, and W. Zaremba, “Hindsight Experience Replay,” in *Advances in Neural Information Processing Systems*, vol. 30. Curran Associates, Inc., 2017.
- [133] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, “Human-level control through deep reinforcement learning,” *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015.
- [134] M. Towers, A. Kwiatkowski, J. Terry, J. U. Balis, G. D. Cola, T. Deleu, M. Goulão, A. Kallinteris, M. Krimmel, A. KG, R. Perez-Vicente, A. Pierré, S. Schulhoff, J. J. Tai, H. Tan, and O. G. Younis, “Gymnasium: A Standard Interface for Reinforcement Learning Environments,” Nov. 2024.
- [135] F. Chollet, “Keras,” <https://github.com/keras-team/keras>, Aug. 2025.
- [136] T. Developers, “TensorFlow,” <https://github.com/tensorflow/tensorflow>, Aug. 2025.

- [137] R. Lowe, Y. Wu, A. Tamar, J. Harb, O. Pieter Abbeel, and I. Mordatch, “Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments,” in *Advances in Neural Information Processing Systems*, vol. 30. Curran Associates, Inc., 2017.
- [138] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, “Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor,” Aug. 2018.
- [139] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel, and S. Levine, “Soft Actor-Critic Algorithms and Applications,” Jan. 2019.
- [140] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann, “Stable-Baselines3: Reliable Reinforcement Learning Implementations,” *Journal of Machine Learning Research*, vol. 22, no. 268, pp. 1–8, 2021.
- [141] A. Hagberg, P. J. Swart, and D. A. Schult, “Exploring network structure, dynamics, and function using NetworkX,” Los Alamos National Laboratory (LANL), Los Alamos, NM (United States), Tech. Rep. LA-UR-08-05495; LA-UR-08-5495, Jan. 2008.
- [142] NXP, “Layerscape LX2160 Communications Processor Fact Sheet,” <https://www.nxp.com/docs/en/fact-sheet/LX2160AFS.pdf>, 2025.
- [143] —, “QorIQ® LX2160A Reference Design Board (RDB),” <https://www.nxp.com/docs/en/fact-sheet/LX2160A-RDB-FS.pdf>, 2025.