TECHNISCHE UNIVERSITÄT DRESDEN

Faculty of Computer Science Institute of Systems Architecture, Chair of Computer Networks

Prediction-enhanced Routing in Disruption-tolerant Satellite Networks

Dipl.-Ing. Felix Walter
Born on: 7 June 1992 in Großröhrsdorf

Dissertation

to achieve the academic degree

Doktor-Ingenieur (Dr.-Ing.)

First referee
Prof. Dr. rer. nat. habil. Dr. h. c. Alexander Schill
Second referee
Prof. ing. Carlo Caini
Advisor
Prof. Dr. Thorsten Strufe
Supervisor
Dr.-Ing. Marius Feldmann

Submitted on: 19 May 2020
Defended on: 2 July 2020

Abstract

This thesis introduces a framework for enhancing DTN (Delay-/Disruption-Tolerant Networking) routing in dynamic LEO satellite constellations based on the prediction of contacts. The solution is developed with a clear focus on the requirements imposed by the “Ring Road” use case, mandating a concept for dynamic contact prediction and its integration into a state-of-the-art routing approach. The resulting system does not restrict possible applications to the “Ring Road,” but allows for flexible adaptation to further use cases. A thorough evaluation shows that employing proactive routing in concert with a prediction mechanism offers significantly improved performance when compared to alternative opportunistic routing techniques.

Acknowledgments

At this point, I would like to express my profound gratitude to everyone who supported me during my work on this thesis. First and foremost, I thank Dr. Marius Feldmann for being my supervisor and mentor, not only concerning this thesis but also in several other contexts. During countless fruitful discussions, he always promoted my thoughts and taught me how to tackle substantial problems in a pragmatic way. I furthermore thank him for his initial proposal to consider a doctorate, without which this document would have never been written.

I would further like to express my gratitude to Prof. Alexander Schill for his continued support, in particular by granting me the opportunity to join his group, providing detailed and helpful feedback on my work, and performing a thorough review of the thesis manuscript.

Additionally, I feel highly honored that Prof. Carlo Caini agreed to be the second referee of this thesis, as I consider him one of the core experts in the satellite DTN field. I especially thank him for his detailed review of the thesis and for giving me the chance to present my work at the University of Bologna.

Moreover, I sincerely thank my partner Kristin Krebs for standing by my side during the last eight years and encouraging me to start, keep up, and successfully finalize this work.

I am also deeply grateful to my parents and grandparents, who always believed in me. They supported and taught me during my whole life and continue to do so. I would like to particularly express my gratitude to my late grandmother Ursula Walter, who was always there for me during my childhood and youth.

Finally, I thank my colleagues at the Chair of Computer Networks and D3TN, my former colleagues at Cloud&Heat, as well as my friends and family for their support, the continuous stimulation of my thoughts, and for providing me with a productive balance between thesis work and compensating activities.

Felix Walter
Dresden, July 2020

Statement of Authorship

I hereby certify that I have authored this Dissertation entitled *Prediction-enhanced Routing in Disruption-tolerant Satellite Networks* independently and without undue assistance from third parties. No other than the resources and references indicated in this thesis have been used. I have marked both literal and accordingly adopted quotations as such. There were no additional persons involved in the intellectual preparation of the present thesis. I am aware that violations of this declaration may lead to subsequent withdrawal of the degree.

Dresden, 19 May 2020

Dipl.-Ing. Felix Walter

Contents

List of Figures

List of Tables

1 Introduction

1.1 Motivation

Disruption-tolerant satellite networks are a promising approach to provide low-cost machine-to-machine communication and Internet access to remote areas. A particularly interesting example is the *Ring Road* concept discussed by Burleigh and Birrane in [BB11]. In addition to being cost-efficient, a Ring Road network is rapidly deployable by leveraging small low Earth orbit (LEO) satellites. As shown in [FFW18], it is possible to apply the same approach to sensor networks on further celestial bodies, e.g., the moon. This way, a Ring Road network may be used both for initial planetary exploration and as a backup for a primary communication network.

Conventional LEO satellite networks, such as Iridium, provide continuous connectivity but require sophisticated constellations with inter-satellite links (ISL) as well as an extensive ground network. A disruption-tolerant satellite network, however, allows hop-by-hop data transmission in a *store-carry-forward* manner, to counteract intermittent connectivity. By that, it has much lower requirements concerning the constellation as well as the used hardware.

The store-carry-forward mode of operation is enabled by the *Delay- and Disruption-Tolerant Networking* (DTN) architecture [RFC4838] and supporting protocols, most importantly, the *Bundle Protocol* [RFC5050]. The work on these building blocks formerly started within the Interplanetary Internet Research Group (IPNRG) of the Internet Research Task Force (IRTF). Though, after initial publications on the interplanetary networking use case, it was realized that similar challenges are present in specific terrestrial networks, and a more general solution, the DTN architecture, is preferable (see, e.g., [MF09], pp. 82-83). This finding led to the formation of the DTN Research Group (DTNRG) at IRTF in 2002. Thanks to the maturity reached by the DTN protocols, in 2014, the DTN Working Group (DTNWG)[1] of the Internet Engineering Task Force (IETF) was created, and the DTNRG concluded its work in 2016.

[1] See: `https://datatracker.ietf.org/wg/dtn/` (accessed: 7 May 2020)

In parallel, the Consultative Committee for Space Data Systems (CCSDS), an international body consisting of most major space agencies, promotes DTN standards for space exploration. This joint effort aims to pave the way to a future space internetwork.

One of the central issues of DTN research is routing, which has been proven to be an NP-hard problem. [BLV07] Due to intermittent connectivity and potentially massive delays, the application of state-of-the-art Internet routing protocols to the addressed *challenged networks* is impossible: The topology changes much faster than the pace at which a propagation of these changes can occur. Thus, researchers have investigated novel approaches to routing, targeting various use cases (see, e.g., Cao et al. [CS13], Kuppusamy et al. [Kup+19]). With few exceptions, these use cases can be divided into two categories, depending on the information available concerning future changes in connectivity. [Bur+16] The first category consists of deterministic networks, where links are intermittent but can be scheduled in advance, i.e., a priori information about future episodes of connectivity (*contacts*) is available. Primarily, scenarios from the space domain fulfill this criterion. The second category comprises opportunistic networks, which do not allow for such precise planning. In these latter networks, heuristics and replication have to be employed to achieve a high probability of data delivery and low end-to-end delays.

When compared to the mentioned two categories, however, some disruption-tolerant satellite networks are special. This set includes the sketched Ring Road networks. Though, theoretically, the deterministic orbits enable precise predictions of future contacts, in practice, such predictions are not always available, because of the peculiar characteristics of these networks. On the one hand, nodes (e.g., user terminals or satellites) may dynamically join and leave the network. On the other hand, especially when using low-cost *commercial off-the-shelf* (COTS) hardware, a large set of factors can result in opportunistic impairments, for example:

- Satellites may fail due to cosmic radiation or hardware faults.
- The power supply of ground stations in remote areas (either provided by the electrical grid, or by a renewable power source) may face disruption.
- Unknown topographical features near ground stations can lead to unexpected signal attenuation.
- Random atmospheric effects, e.g., clouds and rain, can reduce the link quality.
- Interference has to be expected, notably due to the use of shared radio bands (e.g., ISM bands).
- Overall, low-cost communication systems may vastly differ regarding the achievable link performance.

In consequence, while entirely deterministic routing does not fit these networks due to the presence of random factors, opportunistic techniques seem inappropriate as well, as they do not use available topological information. This thesis aims to fill this gap by proposing a new, intermediate routing approach based on a dynamic inference of the network topology.

1.2 Problem Statement

Figure 1.1 shows a small data-ferry satellite network to illustrate the central problem. Two ground stations, **Sens** and **Recv**, are interconnected via a LEO satellite labeled **Sat1**. **Sens** is a sensor node collecting data which need to be transmitted periodically to the receiver **Recv**. Applying the DTN store-carry-forward approach, the satellite accepts these data from **Sens**, stores them in its local buffer, and delivers them when it encounters **Recv**. Deterministic routing fits this simple network exceptionally well. Either a central entity or every node by itself may predict a *contact plan* (i.e., the schedule of future changes in connectivity) from the orbital track of **Sat1**. By that, nodes can infer the fastest path to deliver data to the destination and even estimate the best-case delivery time.

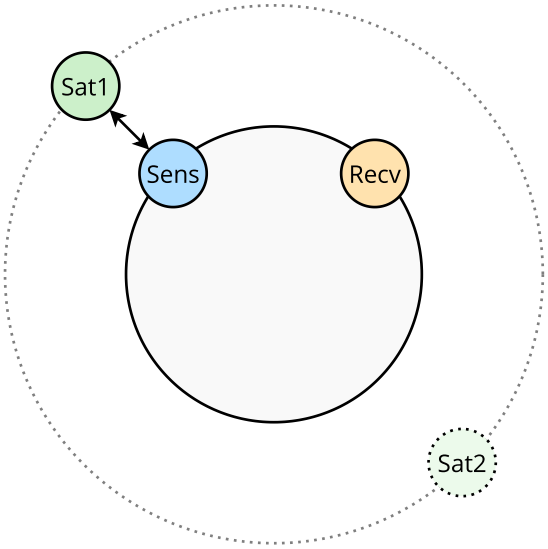

Figure 1.1: Minimal example scenario

As an extension to this three-node topology, after some time, a second satellite, **Sat2**, dynamically joins the network. Using the described deterministic approach, **Sens** would not be aware of **Sat2** until it receives the next contact plan or calculates it from updated orbital information. Of course, to generate this update, the responsible entity needs to be aware of the introduction of **Sat2**.

An alternative way to immediately make use of **Sat2** is the application of opportunistic routing techniques. Every time a contact with a neighbor is detected, an opportunistic algorithm would choose whether or not to forward data to it. Thereby, **Sat2** could be leveraged for data forwarding as soon as it joins the network. However, depending on their orbits, one of the two satellites might provide a faster route to **Recv**, even if it encounters **Sens** at a later point in time. The heuristic applied by **Sens** might select the wrong satellite or introduce unnecessary redundancy by replication. In brief, the main issue of an opportunistic algorithm is that it does not make any use of the available topological knowledge.

Neither fully-deterministic nor opportunistic routing can provide optimal performance in the sketched scenario, implying the need for an intermediate solution that dynamically adapts to changes.

1.3 Objectives

The central objective of the research presented in this thesis can be summarized as answering the following core research question.

> *Can routing based on the prediction of contacts improve the data delivery performance in disruption-tolerant satellite networks?*

This core question is further decomposed into the following three research theses.

/T1/

Leveraging the observed quality of the communication channel during contacts in concert with the underlying deterministic mobility pattern improves the accuracy of contact predictions.

/T2/

By distributing a numerical description of node characteristics throughout the network, future communication opportunities can be inferred by all nodes, enabling the decentralized calculation of end-to-end paths.

/T3/

The use of accurate contact predictions in conjunction with a routing approach that plans end-to-end paths based on future communication opportunities leads to improved data delivery performance.

This thesis applies the term *data delivery performance* to a set of characteristics relevant to the end-to-end delivery of data through a network. These characteristics include the end-to-end delay, the delivery probability, and the overhead induced by routing and forwarding.

Overall, a successful verification of the individual research theses would provide an affirmative answer to the core question. This verification is performed by introducing and evaluating concepts that, when combined, enable prediction-based routing in a concrete, practical use case.

1.4 Research Context

This thesis is part of a more extensive research effort in the field of Delay- and Disruption-Tolerant Networking, focusing on topologies that require the dynamic integration of new nodes. The overall goal of the work is to develop novel solutions for routing, node discovery, and contact prediction. Figure 1.2 gives an overview of relevant publications and university theses in which the author was involved.

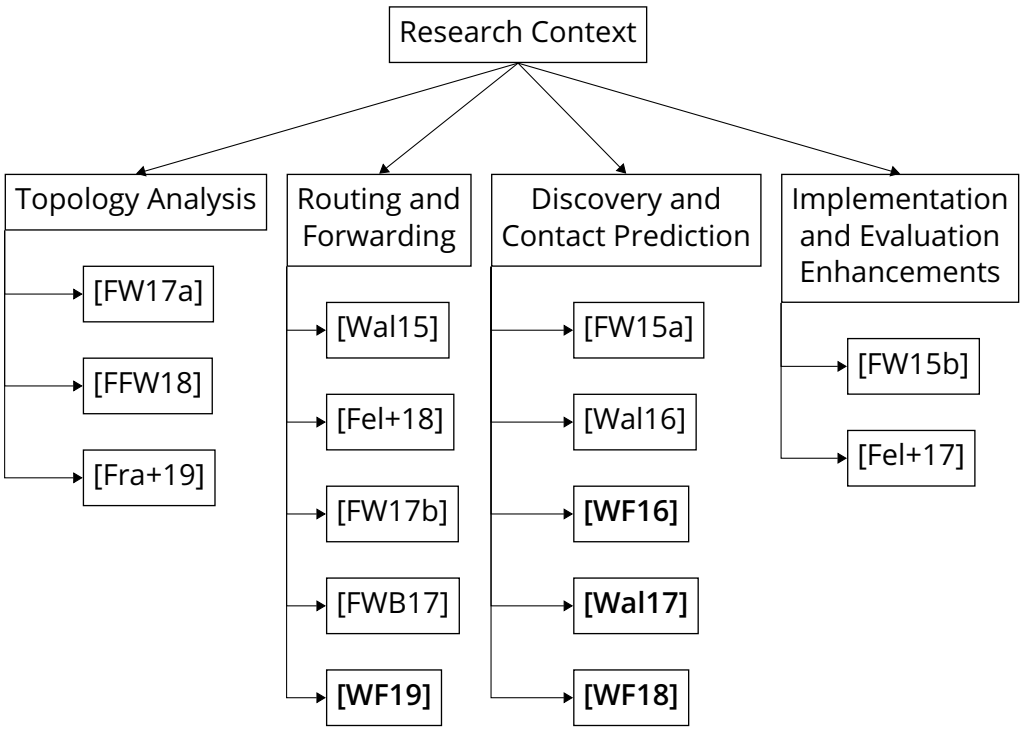

Figure 1.2: Research context of this thesis (the conceptual basis is marked in **bold**)

Most papers from the first three groups (from left to right) also include enhancements to the overall evaluation setup. Papers in the rightmost group primarily describe implementation aspects, especially concerning the µPCN DTN protocol implementation (see [FW15b] for an introduction). The conceptual basis of this thesis consists of four papers marked in bold in the figure: [WF16] gives an overview of the dynamic discovery system developed in the diploma thesis [Wal16]. In [Wal17], a technique for distributing probability metrics is introduced. The approach described in [WF18] allows for predicting characteristics of future contacts. Finally, in [WF19], a concept for routing to leverage probabilistic contacts is presented.

1.5 Outline

The remainder of this thesis is structured as follows: In chapter 2, fundamental aspects of routing and forwarding in disruption-tolerant satellite networks are outlined. Chapter 3 introduces the use case and derives core requirements from its characteristics. In chapter 4, the current state of the art in DTN routing is discussed, with a specific focus on the defined requirements. Chapter 5 presents a novel contact prediction approach and applies it to the targeted use case. Afterward, in chapter 6, techniques to leverage the generated predictions for routing are described. The concepts are evaluated in chapter 7, based on a technical realization and an extensive simulation toolchain. Finally, chapter 8 summarizes the core contributions and gives an outlook on possible future research.

2 Fundamentals

This chapter provides an overview of fundamental aspects relevant to routing and forwarding in disruption-tolerant satellite networks. After giving a brief introduction to DTN and the applied terminology in section 2.1, the contact characteristics specific to satellite use cases are described in section 2.2.

2.1 DTN Architecture and Terminology

[RFC4838] introduces an architecture for "internetworks having operational and performance characteristics that make conventional (Internet-like) networking approaches either unworkable or impractical" (see [RFC4838], p. 1), also called *challenged networks*. An introduction to this architecture and its history is provided by McMahon and Farrell in [MF09]. In [Cai+11], Caini et al. present a broad overview of DTN and specifically highlight its applicability to satellite networks. A concise summary of DTN features relevant to this thesis is given below.

The DTN architecture combines several techniques to tackle the problems presented by challenged networks. Firstly, it introduces the so-called *bundle layer*. The *Bundle Protocol*[1] allows the transmission of arbitrarily-sized application data units independently of lower layers. The adaptation to underlying protocols is enabled via *convergence layer adapters* (CLA) that provide bundle transmission and reception services. Figure 2.1 shows an example of applying the DTN bundle layer on top of layers from the Internet stack model (see also [RFC1122]). Although the Bundle Protocol typically operates on top of a transport-layer protocol, it may as well be used directly over lower-layer protocols. Furthermore, a *store-carry-forward* mode of operation is used, meaning that intermediate nodes store bundles until the next viable communication opportunity occurs.

[1] At the time of writing, two versions of the Bundle Protocol were available; version 6 which has been specified in 2007 as an experimental Request for Comments (RFC) [RFC5050] as well as a CCSDS Recommended Standard [CCSDS15], and the draft version 7 (BPbis) [BFB20] which is in the process of becoming an IETF *Internet Standard*.

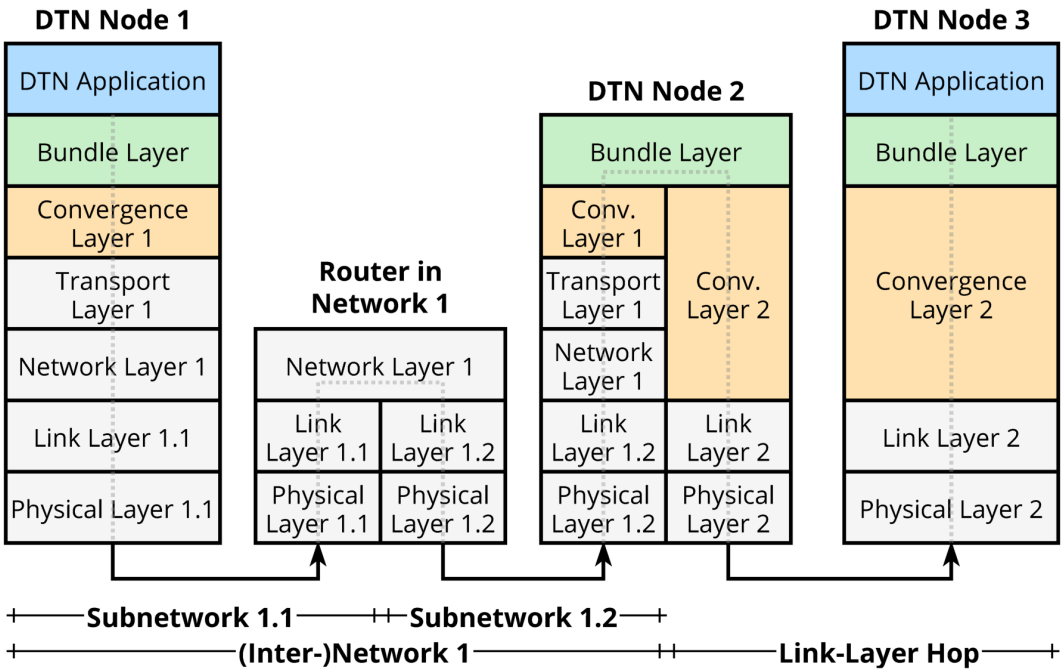

Figure 2.1: DTN bundle layer on top of the Internet stack model (example based on figure 2-1 in [CCSDS15])

These aspects require the development of novel approaches for many problems solved for a long time in conventional networks, including reliable end-to-end transfer, routing, buffer management, flow control, and congestion control.

2.1.1 Contact Definitions, Properties, and Terminology

The term *contact* is commonly used for referring to an individual data transmission opportunity between two DTN nodes. However, there is no unique definition across publications. [RFC4838] defines a contact as a "period of time (interval) during which the capacity is strictly positive, and the delay and capacity can be considered to be constant" ([RFC4838], pp. 16–17), concerning an edge in a multigraph having a time-varying delay and capacity. Jain et al. use a semantically equivalent definition in [JFP04], whereas Hossmann et al. [HSL10] refer to a contact as "the period of time during which two nodes are able to communicate." Conversely, the *Schedule Aware Bundle Routing* (SABR) standard by CCSDS [CCSDS19] defines a contact as "[an] interval during which it is expected that data will be transmitted by a sending node and that most or all of the transmitted data will be received by a receiving node."

The preceding examples show that the term *contact* may refer to an expectation, an observation, or a combination of both. It may also be associated with an instruction for the involved nodes to initiate transmission at a specific time, e.g., in the case of *on-demand* or *scheduled* contacts, as defined in [RFC4838] (p. 17).

Predicted and Observed Contacts

In this thesis, a clear distinction between *contact predictions*, i.e., planned or expected contacts, and *contact observations* is made according to the following two definitions.

Definition 1 *A contact prediction refers to a time interval during which unidirectional transmission of data from one node to another is expected to become possible.*

Definition 2 *A contact observation refers to a time interval during which the possibility of unidirectional data transmission from one node to another has been observed.*

Contact predictions, as well as contact observations, can be associated with further parameters that describe the properties of the (predicted or observed) contact. It should be noted that the mentioned interval may be unbounded concerning its end time (*continuous* and *on-demand* contacts as defined in [RFC4838]). Furthermore, the definitions solely refer to the *transmission* of data, as the reception interval may be shifted forward due to the propagation delay.

If bidirectional communication between two nodes is possible, this is represented by two distinct contact observations or contact predictions which can have different properties. This distinction is essential for space networks, in which asymmetric contacts are common in contrast to terrestrial challenged networks.

The term *contact* may be applied to either definition, depending on the time at which the network topology is examined, denoted by t_0. In the case of contact predictions, this instant is the prediction generation time and, thus, always before the predicted start time. Therefore, a contact prediction is only valid until the associated interval has elapsed. Conversely, contact observations become valid after the transmission opportunity has ended. Thus, for contact observations, the instant t_0 is always after the end of the associated interval as, otherwise, their properties could not be fully determined. The processing of incomplete contact observations is beyond the scope of this thesis.

Most importantly, contacts and all of their properties are considered from the perspective of the bundle layer. Both contact predictions and contact observations refer to intervals not overlapping with others of the same type (for the same ordered pair of nodes), following the note in [CCSDS19] (p. 2-4). Though multiple concurrent links can be available during a contact, there is only one data transmission opportunity that may use any of the links.

Contact Characteristics and Terminology

A contact is associated with a set of characteristics summarized in the following list. The terms applied in this thesis are provided along with the descriptions.

- **Data rate**. This property refers to the *rate of data transmission* (i.e., the amount of outgoing data divided per time unit) available at the bundle layer during a given contact interval in a given direction. The data rate may be time-dependent. It can be either an expectation as part of a contact prediction or a measured value associated with a contact observation.

- **Volume**. The *volume* of a contact prediction defines the expected maximum amount of data that can be transmitted during the contact interval. For a contact observation, it is the maximum amount of data that could have been transmitted, whether fully utilized or not.

- **Capacity**. In several documents (including [RFC4838]), the term *capacity* is related to the transmission rate. However, sometimes also the contact volume is referred to by the same term. Thus, to avoid any possible ambiguity, the term *capacity* is not applied to a contact characteristic in this thesis. Instead, the terms *data rate* and *volume* are used.

- **Probability**. Every contact prediction has a certain *probability of occurrence*. Even in theoretically deterministic networks, node failure may occur and prevent a predicted contact from being observed. However, in most cases, this mathematical probability can only be roughly estimated. Obviously, for a contact observation, there is no probability as its occurrence is known.

- **Confidence**. Instead of an accurate probability, a *confidence* metric may be associated with a contact prediction. This value may not be equal to the contact probability, but enables the use of probability-based heuristics for routing.

It becomes evident that contact predictions and contact observations have different characteristics. Thus, chapter 5 applies two distinct models (see sections 5.2 and 5.3).

2.1.2 DTN Routing and Forwarding

In any computer network, nodes[2] forward individual protocol data units (PDU, depending on the layer and protocol termed, e.g., “frames,” “segments,” or “packets”). In the process, each node on the path either sends a PDU to one or multiple other nodes or passes it to a higher-layer protocol or application. The same principle applies to DTN, though, in these networks, nodes also employ persistent local storage.

Definitions in Literature

The terms *routing* and *forwarding* do not have fully consistent definitions across literature. Sometimes they are even used interchangeably. However, several authors support a clear differentiation.

[2] In this context, a *node* is any entity capable of forwarding data on the considered layer.

Tanenbaum and Wetherall state in [TW11] (p. 363) that "[it] is sometimes useful to make a distinction between routing, which is making the decision which routes to use, and forwarding, which is what happens when a packet arrives.". In [PD12], Peterson and Davie indicate that there is "an important distinction, which is often neglected, between forwarding and routing. Forwarding consists of receiving a packet, looking up its destination address in a table, and sending the packet in a direction determined by that table. [...] Routing is the process by which forwarding tables are built.".

Consequently, in TCP/IP networks, the term *routing* refers to the selection of the next IP address on the path to a given destination. A *routing protocol* exchanges topological information among nodes for automating this selection. Some routing protocols (e.g., the Border Gateway Protocol specified in [RFC4271]) maintain a *Routing Information Base* (RIB) recording the current set of routes and accompanying information. With the inferred routes, a *Forwarding Information Base* (FIB) can be derived, which associates possible target prefixes with interfaces of the underlying layer. [RFC1812] defines the FIB as "[the] table containing the information necessary to forward IP Datagrams [...]. At minimum, this contains the interface identifier and next hop information for each reachable destination network prefix." ([RFC1812], p. 147).

In the context of DTN, [RFC4838] states: "As with the Internet architecture, we distinguish between routing and forwarding. Routing refers to the execution of a (possibly distributed) algorithm for computing routing paths according to some objective function [...]. Forwarding refers to the act of moving a bundle from one DTN node to another." ([RFC4838], 4.3, p. 26) However, in many publications introducing "routing algorithms" (especially from the opportunistic domain, see section 4.1 for an overview), no path computation takes place. In these cases, routing is reduced to a decision whether a given bundle should be forwarded to a specific node, or not. This decision can be based on different heuristics, e.g., using transitive metrics, single-hop information, properties of received bundles, or a random selection of nodes.

Applied Definitions

In this thesis, the following definitions for *routing* and *forwarding* are applied.

Definition 3 *Routing is the set of processes that determines one or multiple nodes to which one or multiple protocol data units shall be sent. This decision is termed the routing decision.*

Definition 4 *Forwarding is the process of sending a protocol data unit to one or multiple other nodes via appropriate lower-layer protocols.*

By that, forwarding is the execution of a routing decision for a specific PDU. These definitions aim to keep compatibility with the TCP/IP terminology and further allow referring to DTN routing algorithms as such. Compared to [RFC4838], the definition of routing is relaxed to cover opportunistic techniques as well.

Path Selection and Next-hop Selection

Similar to the differentiation between distance-vector and link-state routing protocols in IP networks, DTN routing algorithms can be characterized by whether or not a full path to the destination node is calculated. If no knowledge regarding the topology beyond the next hop (i.e., the set of current neighbors) exists, excluding the trivial case that the destination *is* one of these neighbors, a full path cannot be determined. In this case, however, the next hop can still be selected.

This thesis uses the following terminology. Routing techniques that only choose the next node on the path (without calculating such a path) are called *next-hop selection* approaches, while those computing a full path to the destination (thus, selecting the next hop based on that path) are called *path selection* approaches.

2.2 LEO Satellite Communication

LEO satellite networks face particular challenges due to the nature of the satellite communication channel. A network with low-cost hardware is often designed with lower margins to accommodate for random and time-varying effects (see subsection 2.2.2). Thus, it becomes important to also consider channel influences during the operation phase of the network as they dictate the occurrence and basic properties (such as duration, error rate, and volume) of contacts.

2.2.1 A Satellite Contact Example

Figure 2.2 shows the measured carrier-to-noise ratio (CNR) of a contact observed with a weather satellite of the National Oceanic and Atmospheric Administration (NOAA).[3] The data source was a satellite ground station created by the author using an RTL-SDR[4] software-defined radio receiver connected to an LNA4ALL[5] pre-amplifier and a 3-element Yagi antenna pointing south at an angle of 75 degrees towards the zenith. In addition to the CNR, the elevation of the satellite is depicted.

By inspecting the figure, it can be seen that the CNR highly fluctuates in time, but, on average, it reaches its maximum around the center of the observation interval, when the satellite passes almost exactly over the zenith of the ground station. In the first half of the observation interval, the CNR is stronger than in the second half and there is a steep decline at about 550 seconds after the observation start. It is assumed that this results from pointing the directional antenna towards the south and the satellite rising at an azimuth of 161.0 degrees.

[3] The observation of the "NOAA 19" satellite was recorded in the time frame 2019-10-23 15:25:24 to 2019-10-23 15:41:24 (UTC). Thus, the total duration of the observed time frame is 960 seconds. The satellite rose at an azimuth of 161.0 °, and set at an azimuth of 345.0 °. During the observation period, it had a maximum elevation of 89 ° above the horizon. The ground station is located at 51.175 ° N, 14.174 ° E at an altitude of 255 meters above sea level. The center frequency of the receiver was set to 137.100 MHz.

[4] See: `https://www.rtl-sdr.com/` (accessed: 7 May 2020)

[5] See: `http://lna4all.blogspot.com/` (accessed: 7 May 2020)

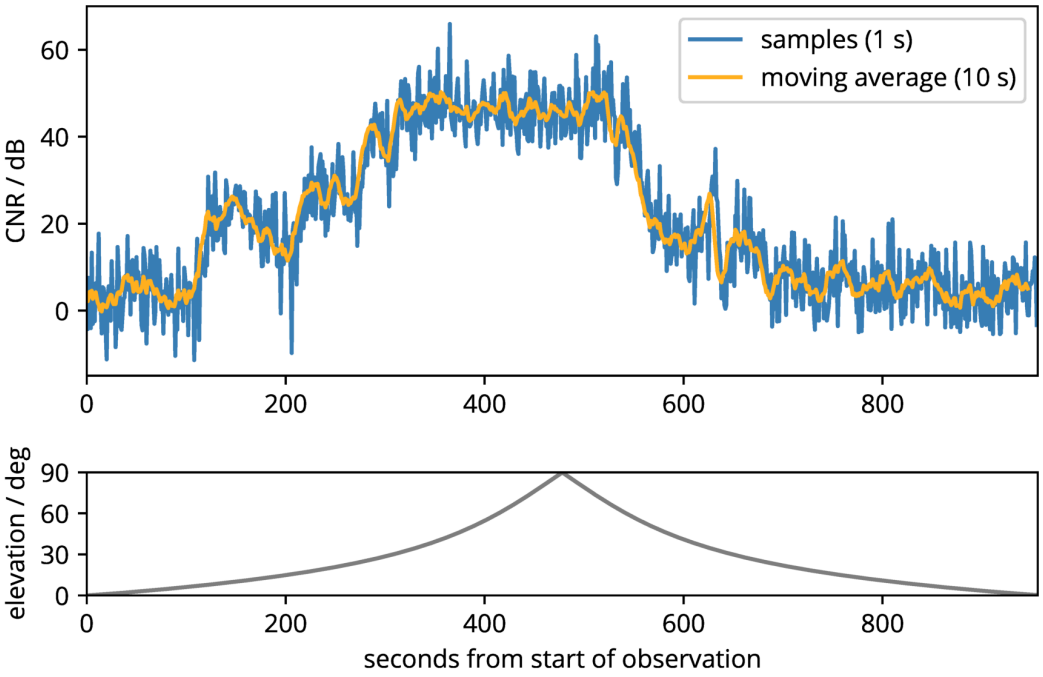

Figure 2.2: Measured CNR for example observation

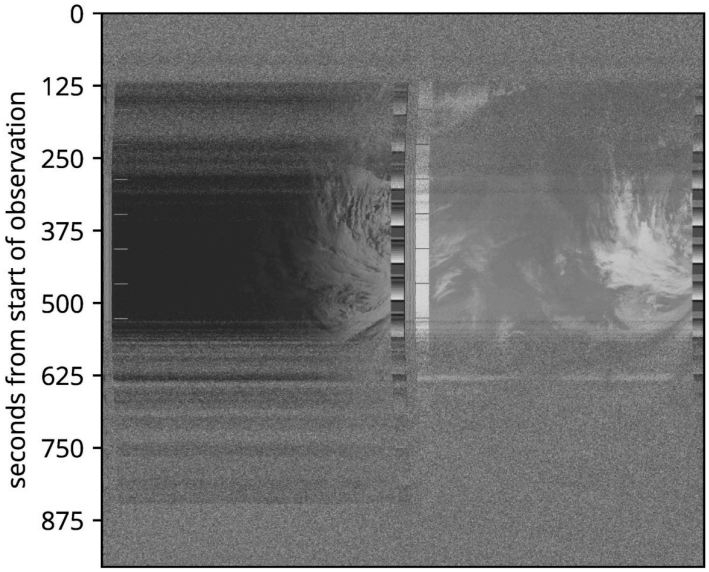

Figure 2.3: Decoded image for example observation

The decoded data reflect the variations in CNR. In this case, the weather image decoded from the Automated Picture Transmission (APT) format is shown in figure 2.3. The slightly curved appearance of some vertical lines around the start and end of the visible image is due to missing synchronization: When the CNR is low, the decoder does not correctly recognize the synchronization pattern. Though accommodating for Doppler shift, slight inaccuracies lead to a drift of the in-line pixel position, requiring continuous re-synchronization.

This example was introduced to clearly show the vast number of factors that influence the CNR and, consequently, the error rate during a LEO satellite contact. It should be noted that most satellite ground stations have much more sophisticated hardware than the one used in this example, e.g., steerable and highly-directive antennas. However, the associated challenges are the same. Besides that, the transceiver hardware used in a low-cost LEO satellite network might not be significantly different than that used for creating the example.

2.2.2 Communication Link Characteristics

This section gives an overview of the satellite link characteristics that are considered in this thesis. For a detailed introduction into the design of digital communication systems, readers are referred to [Skl17]. The specific case of satellite communication systems engineering is discussed in-depth in [Ipp17].

For estimating the performance achievable by a communication system, during its design, a *link budget analysis* is carried out. (see [Skl17], 5.4) By that analysis, impairments to the link are estimated, and an expected *signal-to-noise ratio* (SNR) at the receiver is calculated. For the case of constant envelope signaling (that is commonly used in satellite communication), this figure is equal to the CNR depicted in subsection 2.2.1. (see [Skl17], 5.4) Characteristics such as the applied modulation and coding schemes and the targeted error probability entail specific requirements for the SNR, from which a *link margin* (or safety margin) is derived. (see [Skl17], 5.4.1) This margin indicates the susceptibility of the link to any impairments that may occur randomly or vary over time.

A comprehensive overview of impairments to the satellite link is given by figure 5.1 in [Skl17]. This overview highlights the difference between *losses* (decreases in signal power) and *noise* (increases in noise power or interfering signal power). (see [Skl17], 5.2.2) As a LEO satellite moves relative to a ground station, some of the shown losses vary deterministically over the contact interval. Of those, as in [WF18], the following time-varying losses are considered in this thesis:

- **Free-space path loss**. The dominant loss in received signal power in the satellite link budget occurs due to the electric field strength at the receiver being dependent on the distance. This loss can be calculated from the distance between satellite and ground station and the used frequency. Higher frequencies are associated with a higher free-space path loss.

- **Atmospheric loss**. This impairment results from the signal propagation through the atmosphere. Attenuation due to water vapor and oxygen are major contributors to the figure and are especially relevant at specific frequencies. As the length of the signal path through the atmosphere varies over the contact duration, the atmospheric loss varies as well.

- **Antenna pointing loss**. If the antennas of the sender and the receiver are not perfectly pointed at each other, the pointing loss reduces the received signal strength. Especially if the ground station has a fixed (i.e., non-steerable) antenna, the pointing loss becomes a significant time-dependent factor.

In addition to these losses, link degradation due to *fading* because of multipath propagation and shadowing is taken into account. In [LS+19], Lopez-Salamanca et al. provide a comprehensive overview of available state-of-the-art link models for low-cost LEO satellites. The authors indicate that an additive white Gaussian noise (AWGN) channel model can only be used under line-of-sight (LOS) conditions, i.e., when there are no obstacles in the signal path. This is the case at high elevation angles. However, during a pass, a satellite is always observed at low elevation angles as well. Following the overview of available link models, the authors present an extended model taking into account the case of low elevation angles. Under that circumstance, the fading loss becomes a significant part of the link impairments.

For the example discussed in subsection 2.2.1, the fading loss has only a minor impact on the received signal. As the directional antenna is fixed, at low elevation angles, the pointing loss results in a huge CNR reduction, effectively preventing any signal from being decoded. However, for ground stations with steerable antennas, the fading loss is much more relevant. Due to topographical features, the magnitude of that loss not only depends on the elevation but also on the azimuth angle between satellite and ground station.

The authors of [LS+19] also discuss the relevance of the Doppler shift in the line-of-sight component. In this thesis, it is assumed that a frequency shift is added to account for the Doppler effect. The necessary prerequisite is that both the ground location and the satellite orbit are known to derive the Doppler frequency. The presence of this knowledge, however, can be safely assumed, as COTS positioning systems are readily available at low cost, and satellites can be configured to store and broadcast their orbital elements.

2.2.3 Concurrent Links and Multiple Access

Due to the large surface area of the Earth that is in the *field of view* of a LEO satellite at any instant (its *footprint*), the possibility of establishing concurrent links to multiple ground stations is not an exception, but the usual case. Thus, this has to be addressed either by scheduling the links to individual nodes or by allowing multiple concurrent links.

A huge quantity of multiple access schemes are available to tackle the latter challenge. Their evolution continues along with that of mobile communication systems. A survey of state-of-the-art multiple access schemes targeted at 5G networks can, e.g., be found in [Bas+18].

Independently of the multiple access scheme, a practical limit to the number of concurrent links is always present. Given a large number of concurrent links, either the transmission rate has to be reduced heavily, the computational effort becomes too high, or the available spectrum is exhausted. This means that even if an advanced multiple access scheme is leveraged, a scheduling approach is still necessary to limit the maximum number of concurrent connections and make effective use of the available resources.

2.2.4 Impact on DTN Contacts

In summary, the effects described in the preceding subsections define the nature of contacts observed at the bundle layer. Even if the convergence layer adapter offers reliable transmission of bundles to the next hop, random degradation on the radio link can impair or even prevent the contact. Concurrent links and their prioritization may also lead to a reduced contact duration, disruptions during the contact, a reduced data rate, or prevent the contact from occurring. Finally, contacts can be disrupted by technical failures or power outages. With low-cost hardware that may be located in remote, environmentally-extreme areas, this case becomes more likely.

Thus, with respect to a bundle-layer contact predicted without taking into account these effects, the following types of degradation might be observed:

- Shift, reduction, or disruptions of the contact time interval.
- Reduction of the contact volume due to either or both contact time reduction and/or data rate reduction.
- In the most extreme cases, total failure of the contact.

This implies that, when developing contact-based routing for low-cost LEO satellites as here, the characteristics of the communication links have to be specifically considered.

3 Ring Road Use Case

This chapter, first, gives an overview of the Ring Road use case in section 3.1. Afterward, an example topology is presented in section 3.2 to clarify the challenges these networks present for routing. In section 3.3, similarities and differences to further use cases are identified, and the applicability of concepts developed for Ring Road networks is discussed. Core characteristics of a Ring Road network are pointed out in section 3.4. From these characteristics, specific requirements for a routing algorithm are derived in section 3.5.

3.1 Ring Road Networks

The Ring Road concept is a proposal for a scalable, low-cost communication infrastructure. The idea was first introduced in a 2008 conference presentation by Krupiarz et al. [Kru+08] and was later discussed in detail by Burleigh and Birrane in [BB11]. A Ring Road network uses LEO picosatellites and DTN technologies to provide Internet access to remote areas in an efficient, low-cost, and scalable way. The Ring Road satellites are also called *data mules* since they carry data via physical motion. Two types of ground stations connect to the satellite network: *Hot spots*, which have Internet access, and *cold spots*, which provide access via the Ring Road satellites. Cold spots do not have to be endpoints of the communication. Rather, they can serve as gateways to the satellite network.

Initial discussions of the Ring Road concept focused primarily on connecting remote areas to the Internet for humanitarian relief. Though this still being an essential motivation for realizing a Ring Road network, further applications have been proposed, including low-cost infrastructures for sensor data forwarding or planetary exploration (see, e.g., [FFW18]). Their immense flexibility makes Ring Road networks an adequate platform for research. Besides enabling a diverse set of applications, Ring Road networks can be a viable testbed for the DTN architecture. They further provide educational value due to their intuitive simplicity and the possibility of straight-forward practical realization, as discussed in [FW17a].

Though most aspects of a Ring Road topology are deterministic, several factors influence Ring Road communication in stochastic ways, impeding the ability to plan all changes in topology. These factors include ground stations joining and leaving the Ring Road network, nodes autonomously choosing one of several available peers to connect to, nodes running out of power or experiencing hardware failure, and influences to the radio channel as outlined in section 2.2. Thus, a Ring Road network provides the need as well as an ideal use case for a contact prediction-based routing approach.

3.2 Exemplary Scenario

An example Ring Road network is displayed in figure 3.1. For the sake of simplicity, but without loss of generality, a two-dimensional projection of the Earth and the satellite orbits is assumed. The network initially consists of two satellites (**1**, **2**), one hot spot (**B**), and two cold spots (**A**, **C**). The satellite orbits, as well as the ground station locations, are known by all nodes. Thus, the visibility intervals of involved satellites from any ground station can be calculated, and possible future line-of-sight transmission opportunities can be determined.

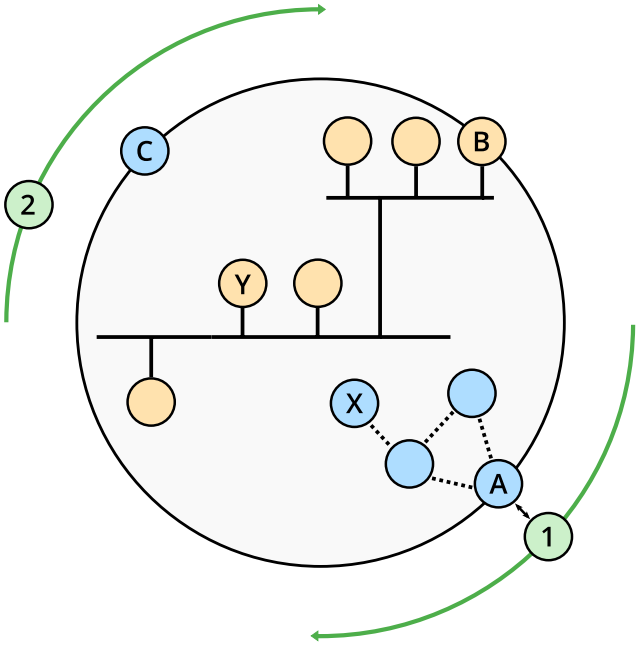

Figure 3.1: Exemplary Ring Road scenario

It is assumed that node **X** periodically emits a bundle to be transmitted to node **Y**. In the local ad-hoc network of node **X**, bundles are exchanged opportunistically. Cold spot **A** provides access to the Ring Road satellite backbone, serving as a gateway on the DTN bundle layer. In the network shown in figure 3.1, the shortest path from **X** to **Y** leads via satellite **1** and hot spot **B**.

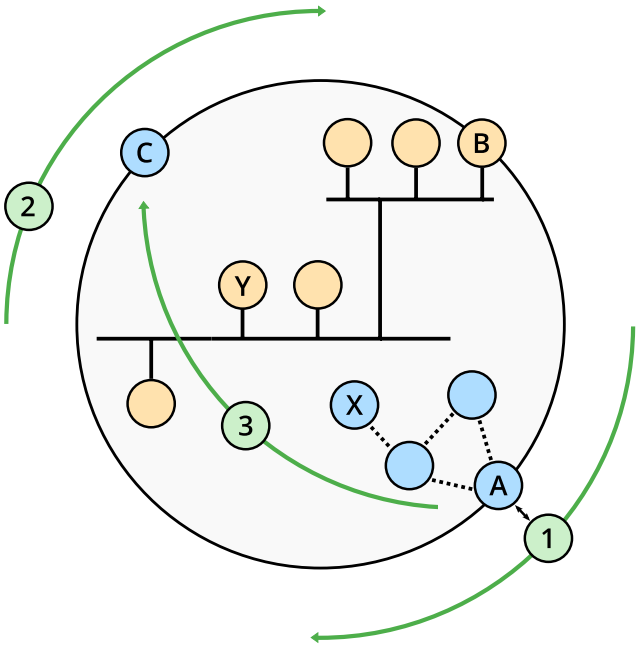

Figure 3.2: Exemplary Ring Road scenario, extension

At some point, a third satellite (**3**) joins the network, which repeatably encounters nodes **A** and **C**. The modified network is shown in figure 3.2. For this satellite, the time to reach **C** after it has encountered **A** is shorter compared to satellites **1** and **2**. In that case, the shortest path to forward bundles from **A** to **B** in terms of end-to-end delay may consist of satellite **3** and node **C** as intermediary nodes. After the nodes have discovered the new satellite, e.g., via broadcasted discovery beacons, bundles may take this alternative route.

However, ground station **C** may experience hardware failures or be located in a remote area where continuous electrical power is not available. It may, thus, temporarily or permanently cease operation without an announcement to the other nodes. One of the involved satellites may fail as well. In these cases, data cannot be forwarded via the respective path anymore. Such events have to be discovered, and the forwarding strategy needs to be adapted dynamically by the nodes.

3.3 Relationship to Further Use Cases

Further use cases exist that, with respect to their characteristics, are similar to Ring Road networks. Examples for that include public transport networks in which individual vehicles carry communication nodes. Such a network was, e.g., evaluated at the University of Massachusetts Amherst. [Bur+06] Due to available bus schedules, a rough prediction of contact intervals is possible. However, unanticipated delays and disruptions may occur as a result of traffic jams, construction work, or hardware defects. Compared to Ring Road networks, the available power and computing resources may not be as limited. Therefore, in such networks, larger buffer sizes and higher data rates on individual links can be expected.

As a second example, a low-cost LEO satellite network with satellites and ground stations operated by a single provider can be considered. Such a network would be very similar to the Ring Road use case concerning the periodicity of contacts and the properties of communication links. However, as all nodes are controlled by a single provider, anticipated topological changes (e.g., the addition or removal of nodes) can be planned. Thus, only unanticipated changes (e.g., failures or link disruptions) have to be handled.

These two examples show that, overall, Ring Road networks are no completely distinguished use case. However, as Ring Road networks face more constraints than alternative use cases with similar properties, it is expected that a concept addressing them provides a more general solution. It can be assumed that such a concept is applicable to a broader range of scenarios than one developed for a use case with fewer constraints.

3.4 Central Characteristics of Ring Road Networks

Ring Road networks have several characteristics that, when combined, enforce a very generic solution to routing and make them an ideal research testbed. The following list gives an overview of these characteristics.

Node Characteristics

/C1/ Predictability of node positions

Nodes participating in Ring Road networks are either stationary (ground stations) or have predictable trajectories (satellites). The time-dependent position of any given Ring Road satellite can be inferred precisely using orbital propagation techniques.

/C2/ Power limitations

Due to the limited power budget available on Ring Road satellites as well as resulting hardware limitations, the onboard processing capabilities are much more constrained than that of a terrestrial node. The power available to the communication system is also heavily limited.

Communication Link Characteristics

/C3/ Probabilistic influences to communication

Several probabilistic factors may influence data transmission in a Ring Road network. These can lead to varying levels of service disruption or, in rare cases, even unexpectedly improve communication (e.g., extend the range).

/C4/ Concurrent communication channels

Due to the large transmission range of satellite communication systems, nodes in a Ring Road network often get the opportunity to communicate with several neighbors at the same time.

/C5/ Heterogeneous communication systems

In satellite communications, the uplink and downlink data rates are often asymmetric. Furthermore, the minimum supported elevation and achievable data rates depend on the type and quality of components that are leveraged (e.g., the type of antenna and error correction capabilities of the receiver).

/C6/ Data rate limitations

Satellite links have to span huge distances. By that, sophisticated communication systems are required to achieve high data rates. In the case of picosatellites, hardware and power limitations constrain the achievable data rates. Though high-rate optical communication systems are in development (e.g., [Mat+19]), in the near future, data rates in LEO satellite networks can still be expected to be much lower than that of wireless communication on Earth.

Network Characteristics

/C7/ Dynamic variations in the set of active nodes

Ring Road networks are open topologies that allow new ground stations to join as well as active participants to cease operation at any time. Satellites may be decommissioned, or further satellites may be deployed to expand the network. It can, however, be assumed that such changes occur on a time scale longer than the average data delivery delay.

/C8/ Recurring contacts

Contacts in a Ring Road network are recurring, i.e., two nodes that come in contact once can be expected to have further contacts unless any of them leaves the network.

/C9/ Long and variable inter-contact delay

Ring Road topologies exhibit a low rate of successive contacts between a pair of nodes when compared to many other DTN scenarios. In addition to that, the delay between two successive contacts may vary by several hours.

/C10/ Huge hop count on optimal paths

As shown in [FW17a], in a Ring Road network, optimal paths in terms of data delivery delay are particularly long with respect to the hop count.

3.5 Requirements for a Routing Approach

For investigating the research theses stated in section 1.3, a further decomposition of the problem domain is performed. The starting point for this, however, lies not within the research theses themselves but is provided by the use case characteristics. In the course, requirements are derived, which need to be fulfilled when routing data based on an end-to-end path. These requirements allow the analysis of related work as well as the development of solution concepts to be use case driven. Additionally, by verifying their fulfillment, an assessment of the research theses becomes possible in a straight-forward manner.

The definition of requirements is split into three parts. In the first part, functional requirements for next-hop selection are defined. The second part identifies functional requirements for path selection-based routing (see subsection 2.1.2). Finally, the third part defines non-functional requirements relevant to any routing approach.

It should be noted that the requirements defined in this section have to be supported in addition to those present in DTN scenarios in general, e.g., buffer management, fragmentation, security, robustness, loop prevention, and handling of data expiration and priorities. The discussion of such general requirements and the development of appropriate concepts to fulfill them are beyond the scope of this thesis.

3.5.1 Functional Requirements for Next-hop Selection

First, functional requirements for performing next-hop selection are identified. These requirements have to be supported by any routing approach applied to the Ring Road use case.

/R1/ Support of scheduled contacts

Characteristic **/C1/** implies that contacts between nodes in a Ring Road network occur in predictable intervals. Thus, contact intervals can be scheduled for all nodes of which necessary characteristics (e.g., the trajectory) are known. Support for scheduling contacts is essential, at least in a part of the infrastructure, for allowing the satellite operator to ensure reliable delivery of command and monitoring data. Consequently, such support has to be present in the applied routing algorithm.

/R2/ Support of probabilistic contacts

Due to possible disruptions resulting from probabilistic influences (characteristic **/C3/**) and concurrent communication channels (characteristic **/C4/**), the complete failure of contacts as well as random interruptions need to be supported by a Ring Road routing approach.

/R3/ Dynamic integration of nodes

As implied by the *open access* characteristic of Ring Road networks (characteristic **/C7/**), the dynamic integration of new nodes has to be natively supported. It should be possible to consider these nodes in routing decisions, even if they are not endpoints of the communication. Conversely, if a node ceases operation, this fact should be recognized and handled by the routing algorithm appropriately.

3.5.2 Functional Requirements for Path Selection

Further functional requirements can be derived from the use case if path selection is assumed. The latter is necessary to assess research thesis **/T3/**.

/R4/ Estimation of link characteristics

For supporting probabilistically-influenced contact intervals (characteristic **/C3/**) and heterogeneous communication systems (characteristic **/C5/**), the path selection has to take into account link characteristics.

/R5/ Support of deterministic mobility patterns

Contacts in a Ring Road network are not only recurring (characteristic **/C8/**), but happen with potentially huge and variable inter-contact delays (characteristic **/C9/**), which is a result of the satellite orbits (characteristic **/C1/**). As knowledge of the contact intervals is required for path selection, it is essential to support the underlying deterministic mobility patterns.

/R6/ Application of transitive information

Due to dynamic variations in the set of nodes (characteristic **/C7/**) and the large number of hops on low-latency paths (characteristic **/C10/**), information has to be propagated transitively to enable the selection of appropriate end-to-end paths.

3.5.3 Non-functional Requirements

Non-functional requirements applicable to any routing approach can be additionally derived from the use case characteristics.

/R7/ Efficient use of available energy

The routing approach should not exhaust the limited power budget of picosatellite systems (characteristic **/C2/**).

/R8/ Low routing information exchange overhead

In case an exchange of information is required for a routing approach to operate, the induced overhead should not consume a major part of the available contact time. Keeping the overhead small is important because there is a low rate of successive contacts (characteristic **/C9/**), and the volume of individual contacts is limited (characteristic **/C6/**).

/R9/ Low replication overhead

The number of bundle replicas created by the routing approach should converge to the amount of redundancy required for reliable delivery. The volume of data that can be transmitted is limited due to characteristics **/C6/** and **/C9/**. Furthermore, a low number of replicas created at each hop allows for supporting paths with huge hop counts (characteristic **/C10/**) without quickly exhausting the network resources.

3.5.4 Relationship between Requirements and Characteristics

As the set of requirements results from the core characteristics of the use case, a mapping of this relationship can be derived. Table 3.1 displays this mapping. A bullet indicates that the requirement results from the given use case characteristic.

	/C1/	/C2/	/C3/	/C4/	/C5/	/C6/	/C7/	/C8/	/C9/	/C10/
/R1/	•									
/R2/			•	•						
/R3/							•			
/R4/			•		•					
/R5/	•							•	•	
/R6/							•			•
/R7/		•								
/R8/						•			•	
/R9/						•			•	•

Table 3.1: Relationship between requirements and characteristics

3.5.5 Relationship between Requirements and Research Theses

In addition to mapping the functional requirements to characteristics of the use case, they can be put in relation to the research theses introduced in section 1.3, for supporting their assessment. The corresponding mapping is shown in table 3.2 and is explained below. In this case, a bullet indicates that the requirement demands a solution supporting the given research thesis.

Requirement / Research Thesis		/T1/	/T2/	/T3/
/R1/	Support of scheduled contacts			•
/R2/	Support of probabilistic contacts	•		•
/R3/	Dynamic integration of nodes		•	
/R4/	Estimation of link characteristics	•		
/R5/	Support of deterministic mobility patterns	•		
/R6/	Application of transitive information		•	

Table 3.2: Relationship between functional requirements and research theses

/R1/ implies that a routing approach for Ring Road networks should support scheduled contacts. This support is needed to investigate **/T3/** as routing has to leverage concrete contact predictions.

The second requirement **/R2/** expresses the need to support probabilistic contacts. This provides the basis for assessing **/T1/** and **/T3/**: Whereas **/T1/** refers to the generation of probabilistic contact predictions, investigating **/T3/** requires them to be applied.

The dynamic integration and removal of nodes demanded by **/R3/** implies the need to provide topological information to them. Thus, a solution supporting **/T2/** is required.

Link characteristics should be included in the routing decision as expressed by **/R4/**. For that, an estimation of these characteristics is necessary. Hence, an approach fulfilling **/R4/** promotes the investigation of **/T1/**.

/R5/ demands that routing supports the deterministic mobility pattern present in a Ring Road network. This support is further necessary for assessing **/T1/**.

Finally, **/R6/** expresses the need to employ transitive information. This need implies the use of a distribution approach, which serves to verify **/T2/**.

4 State of the Art and Related Work

For a long time, the research on routing and forwarding techniques in the DTN field has been divided into two major areas – deterministic and opportunistic network scenarios. This differentiation was already made by Zhang in [Zha06]. For both areas, vastly different strategies to achieve reliable and efficient end-to-end delivery of bundles have been developed. In deterministic networks (e.g., interplanetary networks), connections between nodes can be planned, and knowledge of the time-varying topology is available. This way, *path selection* algorithms can be leveraged.

In contrast to that, opportunistic networks feature random and spontaneous contacts, i.e., transmission opportunities that cannot be fully predicted in advance. Thus, as *path selection* is not possible, *next-hop selection* techniques are applied that commonly replicate bundles to multiple neighbors. At the intersection of both areas, networks exist that have deterministic properties but are also influenced by random factors. Although in these networks (that include the Ring Road use case), opportunistic routing can still be applied, to make full use of the deterministic properties, more specific solutions are necessary.

This chapter gives a broad overview of the state of the art and current developments concerning DTN routing techniques. By applying a simplified taxonomy that differentiates between deterministic and opportunistic concepts, existing solutions and related work are discussed concisely. The focus lies on approaches applicable to end-to-end (unicast[1]) bundle[2] forwarding. All concepts and algorithms are investigated with respect to their fulfillment of the identified functional requirements (see section 3.5). As *next-hop selection* approaches may also support requirements for path selection, both groups of functional requirements are considered in all cases. At the end of the chapter, a comparison of existing work by the supported requirements is provided.

[1] As identified in [CCSDS19], "[no] specifications yet exist that would govern the forwarding of a bundle to a non-singleton endpoint (e.g., 'multicast' transmission)." Thus, the issue of non-unicast transmission is considered beyond the scope of this thesis as well.

[2] It should be noted that although a DTN routing implementation will handle *bundles*, many authors use the more general term *message*. This thesis applies the term *bundle* for consistency.

4.1 Opportunistic Next-hop-selection Approaches

A vast number of routing strategies has been developed for the domain of opportunistic networks. These reactive next-hop-selection techniques often operate based on data- or node-dependent characteristics. An overview is given in this section. Firstly, approaches are described which do not select a specific next hop for forwarding a bundle. Secondly, six groups of node-selective approaches are outlined, depending on the metrics they use in their forwarding decisions.

4.1.1 Non-node-selective

This group consists of approaches that do not respect the properties of individual nodes for selecting the next hop. The selection is either done based on properties of the bundle itself, on buffer considerations, or randomly. Techniques in this section only support requirements **/R2/** and **/R3/**: As they do not make use of planned contact intervals, probabilistic contacts and the dynamic integration of nodes are not an issue. However, advanced scheduling decisions or the inclusion of specific contact characteristics and transitive information are also not possible.

Epidemic Routing

A very basic next-hop selection strategy for opportunistic DTN is *Epidemic Routing*, specified in [VB00]. The approach does not need any knowledge of the topology. It forwards bundles comparably to passing an epidemic disease - any opportunity which arises is attempted, given the bundle is not already present in the buffer of the neighbor.

To determine whether a specific bundle should be forwarded to another node, Epidemic Routing performs an *anti-entropy session* at the start of each contact: So-called *summary vectors*, indicating the presence of specific bundles in a local hash table, are exchanged between the nodes. To limit the impact on the network resources, the authors propose to set a maximum number of bundles queued at each node (the *maximum queue size*) and a maximum hop count for a given bundle replica.

In summary, if no bundle is dropped due to buffer exhaustion or the imposed limits, all possible paths are attempted. This set, of course, also includes the optimal path to the destination, allowing Epidemic Routing to achieve the lowest-possible delivery delay. However, a considerable number of bundle replicas may be produced in this case, which can rapidly lead to congestion.

Because of its simple nature and the widespread support in simulation environments, Epidemic Routing is often used as a baseline for the comparison of novel routing and forwarding approaches.

Spray and Wait

The *Spray and Wait* routing scheme [SPR05] sets a fixed limit for the number of copies that may exist of a given bundle. In an implementation, the remaining number of copies is associated with the bundle and decreased during replication. The total is always equal to the limit set at the source.

The replication limit effectively splits the process of end-to-end bundle delivery into two phases. During the *spray phase*, replication occurs until each node having the bundle only holds a single copy of it, i.e., the associated counter is set to one. This marks the start of the *wait phase*, during which the bundle is only forwarded directly to the destination node.

The authors present different variants for setting the number of copies forwarded on every encounter during the spray phase. A variant highlighted as having the minimum expected delay is called *Binary Spray and Wait*. Here, a node gives half of the copies it has (rounded down to the nearest integer) to each encountered neighbor.

Though explicitly being designed for networks in which "connectivity is rather opportunistic and subject to the statistics of the mobility model followed by nodes" ([SPR05], p. 253), Spray and Wait has shown satisfactory performance when applied to Ring Road networks in an evaluation documented in [FW17b].

Further Non-node-selective Approaches

Further non-node-selective approaches exist, which are not used as often for comparison as those mentioned above. This includes, for example, *Direct Delivery* and *Two-hop Relay*. These two strategies are described in [GT02] and deliver data over either one or two hops. Direct Delivery only sends a bundle to the destination if it is directly encountered. In contrast, Two-hop Relay replicates bundles over a set of neighbors, which then only send it directly to the destination. By that, these two approaches are limited to topologies in which the destination may become reachable over either one or two hops.

4.1.2 Probability-based

The approaches in this group infer and update a probability metric associated with individual neighbors, but not directly referring to specific contacts.

Because of that, probability-based routing schemes support the same requirements as non-node-selective techniques. Additionally, as it is discussed below, many of them distribute the inferred probabilities to other nodes, allowing them to make use of transitive information. Thus, in some cases, support of requirement **/R6/** is present. In this subsection, the most important representatives of the category are outlined.

PRoPHET and Related Approaches

Probabilistic Routing Protocol using History of Encounters and Transitivity (PRoPHET) [RFC6693] is an opportunistic next-hop selection strategy published by the *Internet Research Task Force (IRTF)*. Lindgren and Doria submitted a first draft of the PRoPHET specification [LD05] to the IRTF in 2005. Since then, several refinements were performed, and various extensions have been published.

At its core, PRoPHET leverages a transitive probability estimation using the so-called *Delivery Predictability (DP)* metric. This metric is stored for every known neighbor and gets updated on each contact, starting with an initial value. Using an exponential weighted average approach, the DP increases if neighbors are encountered repeatedly. To account for neighbors that cease to have contacts with the local node, the metric decreases over time.

During a contact, both neighbors exchange the available DP values. Thus, transitive probabilities can be calculated. Both the locally-available and the transitive relations are merged and represented as a single probability estimation for reaching a given node.

A set of extensions to the basic PRoPHET algorithm was developed that are discussed in detail in [Gra+11]. Further variants of PRoPHET were proposed by various researchers, often improving upon a specific property. [PGR17] gives a comprehensive overview of these and further enhancements. Combinations with other routing schemes exist as well. For example, *Probability-based Spray and Wait* [Xue+09] employs the PRoPHET probability calculation to derive the maximum number of bundle replicas for Spray and Wait (see subsection 4.1.1).

All PRoPHET-based approaches support transitive probability estimations regarding data delivery. By that, requirements **/R2/**, **/R3/**, and **/R6/** are fulfilled. Additionally, deterministic patterns in the occurrence of communication opportunities are inferred and reflected in the probability metric. This indicates partial support of **/R5/**. Though an available analytic description of node mobility may not be leveraged, PRoPHET-based routing benefits from deterministic mobility patterns.

MaxProp

The *MaxProp* routing protocol, as published by Burgess et al. in [Bur+06], also calculates a probability metric per node and extends it with a sophisticated buffer management strategy.

The probability value, called *delivery likelihood* in the publication, is increased on each contact by an incremental averaging approach and then re-normalized, always ensuring that the sum of all probabilities stays 1.0. On contact, two neighbors exchange the values they have calculated for other nodes. A *path cost* can then be determined to each destination by transitively summing the inverse probabilities. The transmission order of bundles is derived considering this cost, sending low-cost bundles first.

In addition to that, MaxProp uses a specific queuing order to enhance forwarding decisions: Besides ordering bundles just by probability metrics, more recent bundles (i.e., those which did not traverse many hops before being received) are sent first during a contact. This way, bundles are prioritized at the beginning of their lifetime, spreading them over a larger group of nodes. An analysis using the ONE simulator has shown that this strategy leads to good results in terms of delivery rate and delay in a Ring Road network. [FW17b]

As in the case of PRoPHET, requirements **/R2/**, **/R3/**, and **/R6/** are fulfilled. MaxProp also benefits from deterministic mobility and, thus, partial support of **/R5/** is present.

Further Probability-based Approaches

In their paper "A novel contact prediction-based routing scheme for DTNs" [Zha+17], Zhang et al. present a next-hop selection technique specifically targeted at networks where the inter-contact time is distributed identically. This scenario is vastly different from those assumed by most other opportunistic routing schemes. For example, the *random walk* and *random waypoint* movement patterns result in an exponentially-distributed inter-contact time. [KLV10] The routing scheme by Zhang is, thus, more applicable to networks in which contacts are periodic with an extended inter-contact time, such as satellite networks. Especially in these networks, a recent contact does not imply that the nodes will have their next contact within the near future. A probability metric, the *limiting contact probability*, is distributed via all present neighbors. This value indicates the likelihood that two nodes will have an indirect contact in the future. Transitive values are combined, and the metric is updated similarly to PRoPHET. Forwarding is performed via the neighbor which is associated with the highest probability to reach the destination.

Thus, the routing scheme also supports requirements **/R2/**, **/R3/**, and **/R6/**. It further partially fulfills requirement **/R5/** by reflecting the periodic mobility pattern in the probability metric.

4.1.3 Time-based

Two noteworthy opportunistic next-hop selection approaches that directly employ a time-based metric are the *expected encounter based routing protocol* (EER) and the *community-aware routing protocol* (CAR) as proposed by Chen and Lou in [CL16]. Both techniques derive a metric based on inter-contact times to enable estimations regarding the frequency of contacts. This metric is also exchanged between nodes to allow for transitive calculations.

Time-based approaches perform basic predictions concerning the time and duration of upcoming contact occurrences. They mostly estimate timespans between consecutive contacts. Thus, as in the case of non-node-selective approaches, support of requirements **/R2/** and **/R3/** is present. Moreover, a transitive contact time estimation is done, supporting requirement **/R6/**.

4.1.4 Mobility- or Position-based

This group of routing schemes uses the mobility pattern or (relative/logical) positions of nodes for generating a routing decision.

History Based Prediction Routing (HBPR) [Dhu+13] was published by Dhurandher et al. in 2013. HBPR is specifically targeted at a human mobility pattern, i.e., applicable to devices carried by humans. The node positions are determined by dividing the geographic area into *cells*. A utility metric is derived from the stability of node movements, predicted future movements (using a Markov model), and the line-of-sight distance between source and destination. Bundles are only forwarded to nodes for which the calculated utility metric exceeds a predefined threshold.

Diana et al. [Dia+17] describe a next-hop selection algorithm called *DTN routing for quasi-deterministic networks* (DQN), which is targeted at LEO satellite constellations without onboard storage or inter-satellite links. The authors mention that, in these networks, the traffic characteristics and link outages compromise the otherwise deterministic character and, thus, the network becomes *quasi-deterministic*. Their algorithm defines a geographic direction in which data are forwarded and sends bundles via nodes that are (physically) closest to the destination.

In addition to supporting opportunistic contacts (requirements **/R2/** and **/R3/**), HBPR and DQN use mobility information and, thus, also support requirement **/R5/**.

4.1.5 Social-based

The *social-based* group of next-hop selection techniques aims to infer relations between nodes, which can be represented as a graph structure.

Daly and Haahr [DH07] proposed the *SimBet* routing scheme which employs two metrics, *similarity* and *betweenness*. Based on these metrics, a utility value is calculated and used to request bundles. The *betweenness* represents to which degree a specific node is important for information flows within the topology. Hence, it is influenced by the number of paths containing the considered node. The *similarity* measures how many common neighbors two nodes have. For performing the calculations, vectors describing the current contacts are exchanged between nodes when they encounter each other.

The *BUBBLE* algorithm, described by Hui et al. in [HCY11], also uses the *betweenness centrality* and combines it with a so-called *community* metric. BUBBLE focuses especially on human mobility: It assumes the nodes are part of *communities* that are interconnected much better than the communities with each other. The algorithm aims to detect these communities and forwards bundles upwards in a hierarchy until they reach the destination community. Nodes with greater connectivity are placed higher in the hierarchy.

Both SimBet and BUBBLE base their decisions on logical (termed *social*) relationships of nodes and apply transitive metrics. Thus, in addition to requirements **/R2/** and **/R3/**, they further support requirement **/R6/**.

4.1.6 Congestion-aware

Next-hop selection techniques from this group, for example, the *Backpressure-based Routing* protocol published by Dvir and Vasilakos in [DV10], base their decisions on the discovered or estimated network load. These considerations include evaluating the queue lengths and packet arrival rates at the local node and its neighbors. *Backpressure-based Routing* additionally prefers bundles with an associated delay greater than that of other bundles in the buffer.

The approach only uses local information and is applicable to opportunistic contacts. By that, it supports requirements **/R2/** and **/R3/**. Furthermore, link characteristics play a role in the routing decisions, which is achieved via the analysis of available queues and the packet arrival rate. Thus, partial support of requirement **/R4/** is present, though no advance estimation of the link quality takes place.

4.1.7 Combinations

Zhou et al. [Zho+17] published a probability-based next-hop selection scheme called *Maximum data delivery probability-oriented routing protocol in opportunistic mobile networks* (MDDPC). The authors define a probability metric per node relation. In addition to that, a centrality metric is derived from the maximum data delivery probabilities to all other nodes. This way, forwarding is performed either via the node that has the greatest probability of reaching the destination or via the node with the highest centrality value.

Context-aware Adaptive Routing (CAR), introduced by Musolesi and Mascolo [MM09], is another approach that allows for combining multiple metrics. The authors define a node's *context* by various attributes which can be chosen flexibly depending on the scenario. For example, this may include the current battery level of a sensor node. From these attributes, combined with a utility estimation, a probability for reaching the destination via a specific node is derived. The calculated values are stored within routing tables, which are also exchanged transitively. Between updates, a Kalman filter is employed to predict the metric values. Due to the flexible node context definition, many different parameters can be considered for deriving a delivery probability.

In [DP18], Dudukovich and Papachristou consider next-hop selection as a machine learning classification problem. Their approach leverages different historical data concerning a node's neighbors. According to the authors, relevant data include the set of connected nodes, the contact duration, the data rate, the buffer capacity, and the location of each neighbor. Furthermore, the path taken by received bundles is recorded. With the collected information, a machine learning model is trained, which later provides routing decisions.

Due to the possible use of many different parameters, in these approaches, in addition to requirements **/R2/** and **/R3/**, partial support of requirements **/R4/** and **/R5/** is present. Additionally, MDDPC and CAR make use of transitive metrics and, thus, support requirement **/R6/**.

4.2 Deterministic Routing

In some DTN scenarios, routing may leverage a priori knowledge of contacts. This information allows for *proactive* routing decisions, i.e., bundles can be *scheduled* for specific upcoming contacts. Because deterministic algorithms base their decisions on information provided in advance, they cannot support the requirement of probabilistic contact intervals (requirement **/R2/**).

4.2.1 Contact Graph Routing

Contact Graph Routing (CGR) is a path selection-based routing algorithm which has seen wide adoption in the DTN community. Especially in the domain of space networks and the interplanetary Internet, CGR is applicable, due to the scheduled nature of these networks.

After a long history of continuous evolution, the CGR algorithm has been standardized by the CCSDS in the *Schedule-Aware Bundle Routing* (SABR) blue book [CCSDS19]. Before, the most recent reference version of CGR was available through the *Interplanetary Overlay Network* (ION) DTN software implementation[3]. Nevertheless, earlier algorithmic descriptions of specific versions of CGR are available, e.g., an IRTF draft [Bur10] and a description of CGR in ION 3.5.0 by Fraire et al. [Fra+17]. Due to the availability of the SABR standard, which corresponds to the current ION reference implementation, this thesis exclusively considers the standardized version of CGR. Hence, every time CGR is mentioned without further explanation, the term refers to the version provided in [CCSDS19].

The core idea behind CGR is to represent the network topology as a graph structure. It is assumed that a *contact plan* is available at each node ([CCSDS19], 2.3.1). This contact plan consists of *contacts* and *range intervals* between nodes (the latter corresponding to a specific one-way delay) and can be provided, e.g., by a mission control center. In a first processing step of CGR ([CCSDS19], 3.2.1), the contact plan is used to build a so-called *contact graph*: Contacts become vertices in a directed acyclic graph and are connected via edges to successive contacts. Thus, an edge represents the temporary storage of bundles at a specific node. For the creation and the delivery of bundles, *notional* contacts (which connect a node to itself) are introduced into the graph. This representation enables the use of standard graph algorithms, such as Dijkstra's shortest path algorithm, to compute routes through the time-varying network topology.

It is essential to understand that the computed routes do not refer to a series of network nodes, but a series of contacts. Burleigh et al. [Bur+16] use the analogy of flight bookings for CGR route computation; bundles can be seen as passengers booked to a specific series of flights, i.e., contacts. The availability of flights (contacts) constrains the possible paths that passengers (bundles) may take. In this analogy, the edges of the contact graph are the connections (including the waiting time) between successive flights.

[3] The software and its documentation which describes the CGR algorithm are available online: `https://sourceforge.net/projects/ion-dtn/` (accessed: 7 May 2020)

A contact graph is constructed for each destination. Afterward, the following processing steps are performed to determine the neighboring node(s) to which bundles shall be forwarded ([CCSDS19], 3.1 and 3.2.2–3.2.8) using this graph:

1. **Contact plan check and route pruning** ([CCSDS19], 3.2.2–3.2.3). First, CGR checks whether a contact to the destination is available, discards routes in case the contact plan has changed, and removes elapsed contacts from the contact plan.

2. **Route computation** ([CCSDS19], 3.2.4 and 3.2.6.10). A list of routes is computed based on the earliest arrival time at the destination, independently of the properties of any specific bundle. Though the nature of route computation is explicitly implementation-dependent, Yen's algorithm (an extension of Dijkstra's shortest path algorithm) is proposed in [CCSDS19], 3.2.6.10. Route computation may occur in advance or on-demand as specified in [CCSDS19], 3.2.6.9.1.

3. **CGR preparation** ([CCSDS19], 3.2.5–3.2.6). For every bundle, the list of routes is reviewed, considering its properties. Routes that are not applicable (e.g., due to backward propagation) or do not fit the bundle are excluded. The result is called the *candidate routes list*. In the course, a *projected bundle arrival time* (PBAT, see [CCSDS19], 3.2.6.7) is computed for each route, which estimates the arrival time of the last byte at the destination.

4. **Candidate routes list check and CGR forwarding** ([CCSDS19], 3.2.7–3.2.8). Finally, if candidate routes are available, one or multiple of them are selected to forward the bundle. For a bundle that is not marked as *critical*, the best route from the list of candidate routes is selected. This selection is based on the earliest PBAT, the least hop count, and further factors (see [CCSDS19], 3.2.8.1.4). If the bundle is *critical* due to a quality-of-service flag being set in its header, it is queued for all candidate routes. However, if no candidate routes were found, the *candidate routes list check* fails. In that case, further implementation-dependent procedures can be used, or the route determination process fails altogether ([CCSDS19], C1).

As the properties of a bundle are only required starting from the CGR preparation step, the route computation can be executed independently of a specific bundle. By that, a list of routes can be cached for any possible destination, enabling the application of computationally-expensive algorithms for searching viable routes through the graph ahead of time.

It should be noted that CGR is a best-effort approach that has some important peculiarities. One of them is route re-computation: As no information about downstream traffic and the state of further nodes on the path is available, the algorithm has to be executed again on every node. This can result in a high computational overhead and may lead to loops and instabilities. [Bir12], [Ara+15] Birrane published a proposal to prevent route re-computation by adding an extension block with the route at the source in [Bir12]. Besides reducing computational overhead, such a mechanism may enable novel approaches to optimizing network utilization, reducing congestion, or distributing topological information. [Ara+15]

Several further variants and proposals to improve CGR are available in the literature. For example, one of the earliest proposals was the use of Dijkstra's algorithm by Segui et al. in [SJB11]. Bezirgiannidis et al. discussed two further enhancements to CGR in [BCT16]: CGR with *Earliest Transmission Opportunity* (CGR-ETO) additionally considers the expected queuing delay, and *Overbooking Management* (CGR+OM) aims to improve the handling of contact *oversubscription* by high-priority bundles. An overview of CGR enhancements is given by Araniti et al. in [Ara+15]. Many of these improvements were integrated into the ION reference version and became part of the SABR standard.

In summary, due to the graph representation of the whole network, support of requirement **/R6/** is present in CGR. Given appropriate up-to-date contact plans, CGR *may* also support requirements **/R3/**, **/R4/**, and **/R5/**. However, probabilistic information cannot be leveraged and, thus, requirement **/R2/** is not supported.

4.2.2 HotSel, SatSel, ColdSel

A group of deterministic routing techniques was published by Cello et al. in 2015 and 2016. *HotSel* [CMP15], *SatSel* [CMP16b], and *ColdSel* [CMP16a] are specifically targeted at Ring Road networks. These approaches aim to improve the selection of hot spots, satellites, and cold spots for data transmission.

HotSel is an algorithm that runs in a central node and plans the paths for bundles routed from the Internet to cold spots. The satellites continuously report their remaining buffer capacity, which is considered in the selection of an appropriate hot spot for forwarding the bundles. Scheduling of transmissions is performed such that reduced delivery delays can be achieved compared to the static or random selection of hot spots.

The *SatSel* algorithm aims to select suitable satellites at cold spots. It considers the buffer utilization as well. The satellite selection aims to minimize the delivery delay while not exhausting onboard buffers.

The aim of *ColdSel* is to mitigate congestion in the network by selecting cold spots where bundles can reside for a longer time compared to the buffers of satellites. The goal is to improve the utilization of contacts between hot spots and satellites and reduce satellite buffer allocation.

All three techniques assume deterministic contacts with a fixed duration and a fixed amount of data that can be transmitted. This way, only scheduled contacts (requirement **/R1/**) are supported. The contact plans are derived from the deterministic mobility pattern of the satellites. Thus, requirement **/R5/** is supported as well. Moreover, the buffer utilization is analyzed, indicating indirect support for the consideration of link characteristics (requirement **/R4/**). In comparison to CGR, however, path selection is only performed for a part of the network and only in the case of HotSel.

4.2.3 MARVIN

The MARVIN routing algorithm described by Sekhar et al. in [SMM04] is specifically targeted at interplanetary networks. It derives future communication intervals based on orbital trajectories (ephemeris data) of planets. By that, end-to-end routes can be derived using Dijkstra's algorithm. Due to the available trajectory data, routes can be re-configured before a link becomes unavailable.

MARVIN supports scheduled contacts (requirement **/R1/**), deterministic mobility (requirement **/R5/**), and transitive information (requirement **/R6/**). However, probabilistic contacts (requirement **/R2/**) cannot be leveraged. Furthermore, the dynamic integration of nodes (requirement **/R3/**) is not possible. Varying link characteristics (requirement **/R4/**) are supported partially: The propagation delay during future contacts is calculated, but no estimation of the link quality is performed. Focusing only on interplanetary networks, MARVIN has a very limited application area. The author is not aware of any continued development concerning this approach.

4.3 Hybrid Approaches

In addition to purely opportunistic and deterministic routing schemes, techniques which aim to combine the capabilities of both are available.

Wan et al. published the *hybrid multiple copy routing algorithm* (HMCR) in [Wan+17], which combines CGR with a probability-based approach. By default, CGR (thus, path selection) is used. A probabilistic next-hop selection serves as a fall-back solution if the information provided to CGR (the contact plan) is not recent or errors are encountered. Regarding the support of requirements, HMCR has the same properties as CGR, plus support of requirements **/R2/** and **/R3/**. However, HMCR does not allow for planning probabilistic changes in topology and does not estimate link characteristics (requirement **/R4/**) by itself. Also, when switching to opportunistic next-hop selection, no support for calculating end-to-end routes is present. By that, though it seems that HMCR supports most requirements, it does not support them at the same time.

Another approach to combining deterministic with opportunistic DTN routing was introduced by Burleigh et al. in [Bur+16]. *Opportunistic CGR* (O-CGR) modifies the CGR graph structure by adding a *confidence* metric to each contact. New contacts are introduced based on a stored *contact log*, i.e., the history of contact observations. For calculating the confidence, the mean and standard deviation of the contact duration and inter-contact time are computed. Furthermore, the dynamic discovery of new nodes is enabled. Using the modified contact graph, route computation is performed as usual in CGR. Bundles may be scheduled for multiple contacts based on the confidence metric, such that a specific confidence threshold is reached. O-CGR supports requirements **/R2/** and **/R3/** in addition to those supported by CGR. However, it does not take into account link characteristics (requirement **/R4/**) or a deterministic mobility pattern (requirement **/R5/**) when introducing opportunistic contacts. Thus, for these requirements, only partial support (via an appropriate contact plan) is present.

4.4 Prediction of Communication Intervals

The prediction of communication intervals is essential for many LEO satellite applications, including networks that feature continuous connectivity, e.g., to determine the appropriate satellite constellation in the design phase or predict hand-over times when the system is in operation.

In [GW09], Gilmore and Wolhuter present a model for deriving visibility and communications time windows with a LEO satellite based on orbital trajectory data. A threshold is set to infer the contact duration from a maximum distance between satellite and ground station. A similar approach using a minimum elevation (that directly corresponds to a maximum distance) is leveraged in [WF16].

Papaetrou and Pavlidou show in [PP05] that the elevation angle can also be derived from measurements of the Doppler shift. Based on that, they apply a minimum elevation threshold to determine the time at which a hand-over to another satellite has to take place.

In summary, all of the presented techniques define either a constant minimum elevation or a maximum distance for predicting contact intervals. This implies that the threshold is an important tuning parameter and has to be chosen carefully considering the involved communication systems.

The prediction approaches from this group provide solutions for estimating the link availability time using a fixed threshold. Thus, they only support requirement **/R4/** partially, as variations in link quality during the contact are not addressed. Deterministic satellite trajectories are the underlying data source in all cases. Thus, requirement **/R5/** is natively supported.

4.5 RRND Dynamic Discovery Component

The *Ring Road Neighbor Discovery* (RRND) approach that is described in [FW15a] and [WF16] not only allows for predicting contact intervals but also provides a basic estimation of contact probabilities. For that purpose, it leverages the history of contact observations in concert with an estimation of the accuracy of available data describing the node characteristics. Besides that, it is highly optimized for low-power hardware: A bit-field is used to store an indication of whether a contact was discovered for each available prediction. The ratio of successful observations and, by that, an estimate of the contact probability is derived from the hamming weight of that bit field. The chosen approach is very minimal as it only stores a binary, fixed-length, record of contact occurrence.

RRND supports scheduled contacts (requirement **/R1/**), probabilistic contacts (requirement **/R2/**), and the dynamic integration of nodes (requirement **/R3/**). Moreover, the system employs satellite trajectory information (requirement **/R5/**). However, link characteristics (requirement **/R4/**) are only taken into account in a very basic manner, and no transitive information (requirement **/R6/**) is used.

4.6 Summary and Comparison

In table 4.1, the support of individual functional requirements (as defined in section 3.5) by the delineated approaches is summarized. The table additionally indicates the support of path selection and differentiates between full, partial, and no support of every feature. The corresponding reasoning is provided in the referenced subsections.

Approach	Sec.	/R1/	/R2/	/R3/	/R4/	/R5/	/R6/	PS
Non-node-selective	4.1.1		●	●				
Probability-based opp.	4.1.2		●	●		○	●	
Time-based opp.	4.1.3		●	●			●	
Mobility-based opp.	4.1.4		●	●		●		
Social-based opp.	4.1.5		●	●			●	
Congestion-aware opp.	4.1.6		●	●	○			
MDDPC, CAR	4.1.7		●	●	○	○	●	
Dudukovich et al.	4.1.7		●	●	○	○		
CGR / CGR-based	4.2.1	●		○	○	○	●	●
HotSel, SatSel, ColdSel	4.2.2	●			○	●		○
MARVIN	4.2.3	●			○	●	●	●
HMCR	4.3	●	●	●	○	○	●	○
Opportunistic CGR	4.3	●	●	●	○	○	●	●
Interval Prediction	4.4				○	●		
RRND	4.5	●	●	●	○	●		

Table 4.1: Feature matrix: Support of functional requirements and path selection;
● = fully supported, ○ = partially supported, **PS** = path selection;
gray columns only have to be supported by path selection approaches

The table clearly indicates that none of the delineated routing approaches supports all given requirements. Especially regarding the estimation of contact-specific link characteristics (requirement **/R4/**), only partial solutions exist.

5 Contact Prediction Framework

Based on the overview given in the previous chapter, it is evident that no state-of-the-art solution is available for scheduled routing supporting probabilistic contacts and link characteristics. Hence, a framework for dynamic contact prediction that takes these peculiarities of the use case into account is introduced in this chapter. A component instantiating this framework can be combined with a deterministic routing algorithm, to form a solution for prediction-enhanced proactive routing. The concepts necessary to enable such a combination are described in chapter 6.

Intentionally, a subdivision of the prediction concept into two parts is performed. While the first part represents a generic contact prediction *framework*, the second part (section 5.8) *instantiates* this framework for (i.e., applies it to) the Ring Road use case. The framework provides the necessary interface for this instantiation as a set of functions defining core parts of the contact prediction algorithms. This approach aims to facilitate a straight-forward transfer of the solutions introduced in this thesis to further use cases.

The remainder of this chapter is structured as follows. First, section 5.1 provides a brief architectural overview of the prediction framework. In sections 5.2 and 5.3, the applied probabilistic models for contact observations as well as contact predictions are introduced. Following that, the leveraged node description in the form of metrics is outlined in section 5.4. Afterward, the process of inferring metrics from contact observations is presented in section 5.5. The individual steps allowing contact predictions to be generated are described in detail in section 5.6.

The rest of the chapter focuses on the application of the framework. Section 5.7 provides a guideline for instantiating the framework to derive a prediction component. This guideline is applied in section 5.8 to Ring Road networks, resulting in a prediction component tailored to this use case. Finally, section 5.9 discusses the derived instance with a specific focus on constraints of the generic prediction framework.

5.1 Overview

An overview of the contact prediction approach is displayed in figure 5.1. All components addressed in this chapter are highlighted. The core results generated by the prediction component are *contact predictions* (see also subsection 2.1.1) that define the expected probabilistic properties of future contacts. This information can be used to perform *path selection* in a suitable routing algorithm.

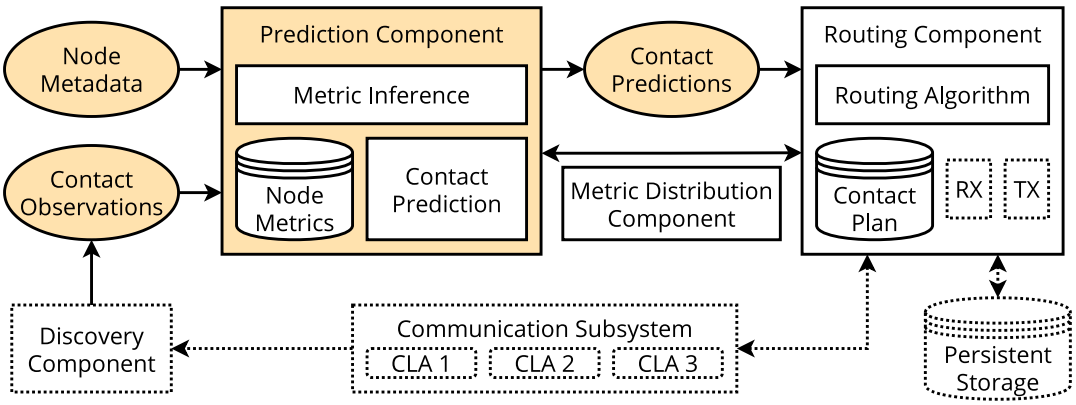

Figure 5.1: Components and context of the contact prediction approach

The prediction component requires past *contact observations* to be supplied, the set of which is called the *contact history*. In addition to that, *node metadata* have to be provided. These contain information about the locations or trajectories of nodes (e.g., GPS coordinates or orbital elements) and can be obtained from several sources. A central configuration can be imagined as well as a distribution of such data via discovery beacons. Internally, the prediction component makes use of *node metrics*, which represent the observed characteristics of other network nodes. These metrics may be distributed, allowing nodes throughout the network to derive predictions.

Figure 5.2 shows the overall contact prediction process that can be divided into two core *phases*. After contact observations have been collected, e.g., by a discovery component such as introduced in [WF16], the *metric inference* phase is executed. In this phase, the properties of available neighbors are inferred and modeled as tuples of numerical values called *node metrics*. Previous predictions are leveraged in a feedback loop to improve the accuracy iteratively. In the second phase, *contact prediction* is performed, which yields a *contact plan* that provides topological information to the routing component.

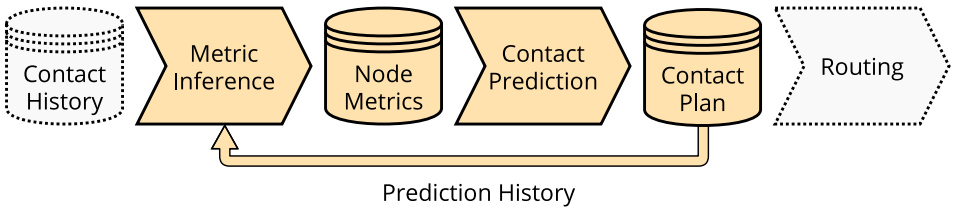

Figure 5.2: Phases and intermediary data of the overall process

Both phases of the contact prediction process can be performed asynchronously, i.e., no real-time operation is required. Stored node metrics only have to be available in case predictions should be generated. Therefore, the prediction concept can be used in a centralized fashion: Nodes with limited computing resources may forward contact observations to a central entity and have the prediction process executed there. This way, compute-intensive tasks can be offloaded to powerful nodes, and external auditing and management of the contact prediction process become possible, which might be desirable in some deployment scenarios. However, such an extension is left to future work as this chapter focuses on a node-local prediction concept. For an extended discussion of a deterministic routing scheme with central topological knowledge, the reader is referred to [FWB17].

5.2 Contact Observation Model

The model for representing contact observations consists of the following data:

1. **Unique identifiers of nodes**. The nodes between which a contact has been observed need to be identified unambiguously. By that, collected measurements can be associated with the correct neighbor and further information such as trajectory data or measurements collected by other nodes can be obtained from external sources. The pair of identifiers for the sending and receiving node is recorded as a tuple (N_s, N_r) for each contact.

2. **Observed connection interval**. The time interval during which a link was observed (denoted $[t_{start}, t_{end}]$) has to be recorded. This interval can be compared to an existing contact prediction to assess its accuracy and, by that, used to update available knowledge. It should be noted that several individual connection intervals may be observed for a single predicted contact, as unanticipated interruptions may always occur. During information acquisition, they cannot be differentiated from distinct contacts and, thus, each contact observation corresponds to a single connection interval.

3. **Data rate**. For every recorded connection interval, the achieved data rate (denoted $br(t)$) should be supplied to the prediction component. By using this information in concert with the interval duration, the maximum possible transmission volume can be calculated. If the communication system is using a fixed data rate, the measurement of this parameter is not required.

4. **Link quality**. As evident from the example provided in subsection 2.2.1, the main factor influencing the contact volume is the time-varying link quality. Hence, observations need to be complemented with measurements of a link quality indicator. Its exact nature depends on the selected prediction approach. For example, a binary indicator of link presence over time, a measure describing the physical-layer signal-to-noise ratio, or a received signal strength indicator (RSSI) could be used. In the case of a binary indicator, the link quality function can be replaced with a constant while representing individual intervals of connectivity as distinct observations. The link quality over time is denoted $Q_{link}(t)$.

Each contact observation (denoted O) can, thus, be represented as a tuple, as defined by equation 5.1.

$$O = (N_s, N_r, t_{start}, t_{end}, br(t), Q_{link}(t)) \tag{5.1}$$

The set of collected observations represents the *contact history* $\mathbb{O}_{N_s,N_r}$ of a node with a specific neighbor described by the (ordered) pair (N_s, N_r). The set may be limited to a fixed number of elements (by only storing the most recent elements) to meet maximum storage requirements. Though, this is considered an implementation detail. It is expected that the contact history is collected, maintained, and verified by the employed discovery component.

Contact Example represented as Observation

The example contact introduced in subsection 2.2.1 can be represented using a description conforming to the observation model. For that, assumptions regarding the implementation of the component have to be made. It is assumed that the satellite image received in the example corresponds to the link-layer data, the required CNR for link establishment is 30 dB and, thus, a link was available from 275 to 550 seconds into the contact. The data rate of the APT transmission is constant at 4160 bits per second. It is assumed that coding is applied on top of that to provide a reliable link. The assumed coding scheme has a rate of 0.5. Thus, a constant net data rate of 2080 bits per second can be achieved during the time a link is available. The measurement in the example case started at $t_0 =$ 2019-10-23 15:25:24 (UTC) and ended at $t_1 =$ 2019-10-23 15:41:24 (UTC), while data could be decoded from $t_{start} =$ 2019-10-23 15:29:59 (UTC) to $t_{end} =$ 2019-10-23 15:34:34 (UTC). The sender was the *NOAA 19* weather satellite, and the receiver was the *DL4FW* ground station. By that, the observation can be represented using equation 5.2.

$$O_{1132307} = (NOAA19, DL4FW, t_{start}, t_{end}, 2080 \text{ b/s}, 1) \tag{5.2}$$

In this simplified example, the link quality measure is assumed just to represent link presence. Thus, it is set to the constant value 1, and the time interval of the contact observation is equal to the period during which a signal could be decoded.

5.3 Contact Prediction Model

A *contact plan* provides topological information as a list of future contacts. By that, it forms the basis of deterministic DTN routing. More precisely, in the context of this thesis, the elements of the contact plan are *contact predictions*, as defined in subsection 2.1.1. A model representing these contact predictions has to support the characteristics of the use cases to which it is applied to provide all necessary information to the routing component.

In [FW17a], a classification of Ring Road networks has been introduced, specifying the information necessary to describe contacts in Ring Road topologies with different characteristics. Among them, the RRp class is defined for probabilistic Ring Road networks. This class associates a probability value to a contact, corresponding to the expected likelihood of contact occurrence. In [WF19], it has been identified that this description is not always sufficient: A single value cannot represent a time-varying error rate as well as shifts in the contact interval and disruptions on a sub-contact timescale. Thus, an extension is provided by the RRpp class, which relaxes the requirement of a precise contact time interval and additionally considers a sub-contact probability distribution. The model applied in this thesis to represent contact predictions is based on these general ideas.

The following properties describe a contact prediction:

1. **Unique identifiers of nodes**. A contact prediction always needs to be assigned to a pair of uniquely-identified nodes, whereas one of them has to be the transmitting node and the other one the receiving node. This pair of identifiers is recorded as a tuple (N_s, N_r).

2. **Soft time frame**. In line with existing deterministic routing approaches, two timestamps are associated with each contact, describing an interval during which transmission may become possible. These two timestamps do not necessarily match the actual timestamps of the start and end of a communication opportunity. Instead, they define an overall time frame during which one or several communication opportunities may arise. The *soft time frame* is an interval of the form $[t^*_{start}, t^*_{end}]$.

3. **Data rate**. To enable estimations of the contact volume (which defines the amount of data that can be scheduled for the contact), knowledge or an estimation of the time-varying data rate is necessary. The data rate is represented by a function $br(t)$.

4. **Delay**. Links may be subject to non-negligible signal propagation delays. The delay has to be known to estimate the data arrival time at the receiving node. It may vary during the contact and is, thus, represented by a function $\delta(t)$.

5. **Confidence**. The confidence value associated with a contact prediction estimates the likelihood of the given soft time frame containing at least one interval during which data transmission is possible, i.e., that a contact will occur during the predicted time frame. The confidence corresponds to the probability value applied in [FW17a]. The symbol $P_{contact}$ is used in the following, with its value being a real number in the interval $[0, 1]$.

6. **Link presence probability**. As outlined in subsection 2.2.4, episodes of opportunistic disruption may occur even during a contact. The link presence probability, denoted by $P_{link}(t)$, determines the likelihood of a usable radio link being present at a given instant t during the contact.

7. **Time of generation**. Depending on its generation time, a prediction may be based on a different amount of topological knowledge. This instant is referred to as t_0.

These properties can be represented as a tuple C, as defined by equation 5.3.

$$C = (N_s, N_r, t^*_{start}, t^*_{end}, br(t), \delta(t), P_{contact}, P_{link}(t), t_0) \tag{5.3}$$

The tuple C is called the *function-based representation* of a contact prediction. Due to the use of a *soft time frame*, the introduced model effectively reduces the need for precise knowledge about the intervals usable for data transmission, though, still allowing for an estimation of the probability associated with successful data transmission at any time during the contact.

Contact Descriptor

For referring to a predicted contact without knowledge of the specific data rate, delay, and probabilistic properties, it is sufficient to provide a tuple containing the node identifiers as well as the soft time frame and the generation time as defined by equation 5.4.

$$C^+ = (N_s, N_r, t^*_{start}, t^*_{end}, t_0) \tag{5.4}$$

This tuple is called the *descriptor* of the contact prediction in the following. It is a basic, minimal means of uniquely identifying a specific prediction.

Volume-based Representation

In the function-based representation, for obtaining the data rate, the delay, and the link probability, still, a time-dependency is necessary. This time-dependency has several drawbacks:

- It increases the amount of data that has to be stored per contact prediction. Further, should a contact plan distribution approach be applied, the amount of transmitted data is increased as well.
- It is difficult to predict the functions precisely for future contacts, especially when using soft time frames.
- It increases computational demands for leveraging the generated predictions in a routing algorithm.

In order to conquer these drawbacks, an alternative *volume-based representation* is proposed. This representation is based on the expected volume of the contact. As each instant within the predicted interval is associated with a certain probability of successful transmission, the contact volume (denoted by $V^*_{contact}$) becomes a random variable.

The expected value for the contact volume (denoted by $V_{contact}$) can be derived from the above representation using equation 5.5.

$$V_{contact} = \mathbb{E}(V^*_{contact}) = \int_{t^*_{start}}^{t^*_{end}} P_{link}(t) \cdot br(t) \cdot dt \tag{5.5}$$

Apart from the data rate and link probability, the delay $\delta(t)$ is time-dependent. In the volume-based representation, the delay is replaced by a constant referring to the projected maximum delay during the contact, δ_{max}. For this pessimistic estimation, it is assumed that changes in delay during a contact in the targeted scenarios are generally small, i.e., much smaller than the contact duration. The assumption of a constant delay during contacts is also the basis of the contact definition in the DTN architecture document ([RFC4838], p. 16). Scenarios with non-negligible variations in delay during the contact interval are beyond the scope of this thesis.

Replacing the time-varying data rate, link probability, and delay with constants still allows for indicating whether data will be transmitted or not, when they will arrive latest, and which amount of data can be scheduled at maximum. Additionally, the need for providing a precise, synchronized time source on all nodes is relaxed. The volume-based representation can be derived in a straight-forward manner and forms a tuple C' defined by equation 5.6.

$$C' = (N_s, N_r, t^*_{start}, t^*_{end}, \delta_{max}, P_{contact}, V_{contact}, t_0) \tag{5.6}$$

Due to the mentioned issues concerning the function-based representation (C), this alternative form is used as the primary representation for contact predictions in the following sections.

Contact Example Represented as Prediction

If assuming that an accurate prediction of the example contact introduced in subsection 2.2.1 has been generated, this prediction can be represented as introduced above. The function-based representation is defined by equation 5.7.

$$C_{1132307} = (NOAA19, DL4FW, t^*_{start}, t^*_{end}, 2080 \text{ b/s}, \delta(t), 1.0, P_{link}(t), t_0) \tag{5.7}$$

The soft time frame is given as t^*_{start} = 2019-10-23 15:25:24 (UTC) and t^*_{end} = 2019-10-23 15:41:24 (UTC), with $t_0 < t^*_{start}$. The time-dependent link probability is defined by equation 5.8 (timestamps in UTC).

$$P_{link}(t) = \begin{cases} 0, & \text{if } t < \text{2019-10-23 15:29:59} \\ 1, & \text{if } t \geq \text{2019-10-23 15:29:59 and } t \leq \text{ 2019-10-23 15:34:34} \\ 0, & \text{if } t > \text{2019-10-23 15:34:34} \end{cases} \tag{5.8}$$

The soft time frame was estimated pessimistically using the time interval during which the satellite has been observed above the horizon of the ground station, i.e., at an elevation greater than 0 °. The delay $\delta(t)$ during the interval $[t^*_{start}, t^*_{end}]$ varies between 2.835 ms and 11.580 ms and is a function of the range between satellite and ground station.

For the function-based representation, precise timestamps have to be available to define $P_{link}(t)$. The fact that precise time synchronization is problematic in DTN has been discussed, e.g., by Wood et al. in [WEH09]. Thus, it is more practical to use the volume-based representation, provided for the specific example contact by equation 5.9, whereas t^*_{start} and t^*_{end} have the same values as defined above.

$$C'_{1132307} = (NOAA19, DL4FW, t^*_{start}, t^*_{end}, 11.580\ \text{ms}, 1.0, 572\ \text{kb}, t_0) \tag{5.9}$$

5.4 Node Metrics

The prediction component infers *node metrics* that represent relevant characteristics of network nodes. These metrics are tuples provided as parameters to the prediction functions and, thus, a mathematical representation of initial data for generating contact plans. All symbols and functions discussed here are either associated with a single node or with an ordered pair (i.e., a 2-tuple) of nodes. Thus, in the latter case, for every two nodes, two distinct sets of metrics exist.

If the source of metrics is local (such as the approach described in section 5.5), only metrics for the local node as well as for tuples (N_s, N_r) are available, where either N_s or N_r is the local node. In case contacts are unidirectional or the observation process is limited to received signals, only metrics with N_r being the local node can be collected. Metrics for any other node tuple have to be either configured or distributed.

To enable predictions according to the model introduced in section 5.3, the following types of metrics have to be available:

1. **Location metrics**. To determine soft time frames as well as the signal propagation delay, a description of the time-varying node locations has to be available. It is expected that, for each node N, a *location function* $\vec{L_N}(M_{L,N}, t)$ is known using the current tuple $M_{L,N}$ of location metrics.

2. **Contact confidence metrics**. These metrics allow for deriving the confidence associated with future contact predictions. By that, they estimate how likely it is for a contact to occur the next time a given node comes in range. These metrics are represented as a tuple denoted by M_{P,N_s,N_r}.

3. **Contact volume metrics**. For predicting the contact volume, an additional set of metrics is necessary. These metrics describe the constant aspects of volume characteristics between two nodes. The symbol for the tuple of contact volume metrics is M_{V,N_s,N_r}.

In a technical realization, the nature of collected node metrics depends on the chosen prediction approaches, which themselves depend on the characteristics of the specific scenario. Nevertheless, node metrics always have to fulfill the following set of requirements to be effectively leveraged.

/M1/ No or constant external dependencies

Node metrics shall not depend on variable external data except the values of other node metrics. Without this condition being met, contact prediction solely by the set of node metrics would not be possible. Constant parameters that can be configured in advance (such as the type of trajectory or parameters of the communication system) are not subject to this constraint.

/M2/ Convergency

Node metrics shall converge toward single, static values that change over time scales much larger than the duration of individual contacts. As node metrics may be distributed, they should not vary significantly on a time scale comparable to the maximum amount of time required for distribution.

/M3/ Compact representation

Node metrics shall consist of numerical values and not be associated with large amounts of auxiliary data to ensure storage and transmission efficiency.

/M4/ Serializability

It must be possible to obtain a representation of the current set of node metrics that can be distributed over the network.

5.5 Metric Inference

The first processing phase performed by the prediction component is called *metric inference*. The goal of this phase is to derive *node metrics* from *contact observations*. The metric inference phase can be divided into four steps that are shown in figure 5.3. While the first two steps build upon each other, after they have been executed, confidence and volume metric inference may be performed in parallel.

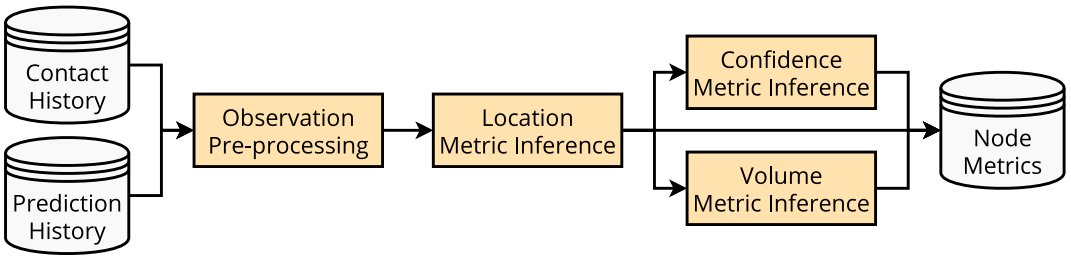

Figure 5.3: Metric inference process

In a first step, a *pre-processed contact history*, denoted $\mathbb{O}'_{N_s,N_r}$, is derived from the contact history $\mathbb{O}_{N_s,N_r}$. All subsequent steps are defined recursively, i.e., they take into account the current values of node metrics to compute new values. Due to this property, the function definitions can be highly flexible.

The metric inference phase may be executed right after every update to the contact history. Though, as mentioned in section 5.1, this is not required; it can also be performed asynchronously during idle time or even on another node.

5.5.1 Observation Pre-processing

The metric inference phase is executed once per predicted soft time frame. As mentioned in section 5.2, several contact observations may be related to a single contact prediction, but collected as distinct records due to link interruptions. Thus, before metrics can be inferred, available contact observations have to be matched to previous contact predictions. The set of previous predictions is called the *prediction history* and is provided to the metric inference phase via a feedback loop mechanism that is displayed in figure 5.2. If the interval of a contact observation is a subset of the soft time frame of a contact prediction, it is added to a list associated with that prediction. Due to the nature of the soft time frames (see section 5.3), an observed interval should always be fully contained in a single soft time frame.

Hence, given a soft time frame $[t^*_{start}, t^*_{end}]$ and a set of observations $\mathbb{O}^+_{N_s,N_r} = O_0 ... O_n$ with $n, i \in \mathbb{N}$, $i \leq n$, $O_i = (N_s, N_r, t_{start,i}, t_{end,i}, br_i(t), Q_{link,i}(t))$, $t_{start,i} \geq t^*_{start}$, and $t_{end,i} \leq t^*_{end}$, the resulting processed observation is obtained via equation 5.10.

$$O'_{N_s,N_r} = (N_s, N_r, t^*_{start}, t^*_{end}, br'(t), Q'_{link}(t)) \tag{5.10}$$

The time-varying data rate $br'(t)$ and link quality $Q'_{link}(t)$ for the processed observation are derived using equations 5.11 and 5.12.

$$br'(t) = \begin{cases} br_i(t), & \text{if } t \geq t_{start,i} \text{ and } t \leq t_{end,i} \text{ for all } i \leq n \\ 0, & \text{otherwise} \end{cases} \tag{5.11}$$

$$Q'_{link}(t) = \begin{cases} Q_{link,i}(t), & \text{if } t \geq t_{start,i} \text{ and } t \leq t_{end,i} \text{ for all } i \leq n \\ 0, & \text{otherwise} \end{cases} \tag{5.12}$$

If a predicted soft time frame has elapsed, but no observation could be assigned to it, still, a processed observation is generated using an empty set for $\mathbb{O}^+_{N_s,N_r}$. This way, the feedback mechanism also applies to contacts that have not been observed.

An important peculiarity of this overall approach is that, upon initial discovery of a node, it might not be possible to derive contact predictions, because node metrics are not available initially and, thus, no soft time frames can be predicted. This fact has to be considered during instantiation of the framework; if no metrics are present, default values should be assumed to generate initial predictions.

5.5.2 Location Metric Inference

For the inference of location metrics it is assumed that necessary trajectory and position data can be either configured statically (e.g., in the case of satellites), or can be broadcasted via discovery beacons (e.g., GPS-equipped nodes at fixed locations, nodes that broadcast their trajectory function). If the latter is not possible, for nodes at fixed locations, a technique such as delineated in [WF16] may be used.

Overall, the location metric inference step can be defined as a function of the location metrics that are available for the considered node N, a set of configured or received parameters concerning the node location p_N (that may be empty), and the pre-processed contact history $\mathbb{O}'_{N_s,N_r}$, whereas $N = N_r$. The application of this function is shown in equation 5.13.

$$M_{L,N,i+1} = M'_L(M_{L,N,i}, \mathbb{O}'_{N_s,N_r}, p_N) \tag{5.13}$$

The result of the location metric inference step is the updated location metrics tuple $M_{L,N,i+1}$ as introduced in section 5.4, i.e., the parameters of a mathematical description concerning the trajectory (or static position) of the node N.

5.5.3 Confidence Metric Inference

For deriving confidence metrics, a function is defined that takes the current confidence metrics, the set of pre-processed observations, and the location metrics for the sender and the receiver as inputs. By that, it can also take node locations into account, based on results provided by the previous step. The confidence metrics are updated using this function according to equation 5.14.

$$M_{P,N_s,N_r,i+1} = M'_P(M_{P,N_s,N_r,i}, \mathbb{O}'_{N_s,N_r}, M_{L,N_s}, M_{L,N_r}) \tag{5.14}$$

5.5.4 Volume Metric Inference

The inference of volume metrics is performed using a function defined in the same manner as the one for confidence metric inference. The metrics are updated according to equation 5.15.

$$M_{V,N_s,N_r,i+1} = M'_V(M_{V,N_s,N_r,i}, \mathbb{O}'_{N_s,N_r}, M_{L,N_s}, M_{L,N_r}) \tag{5.15}$$

5.6 Contact Prediction

Based on the aggregated node metrics, contact predictions are generated. These predictions use the volume-based representation introduced in section 5.3.

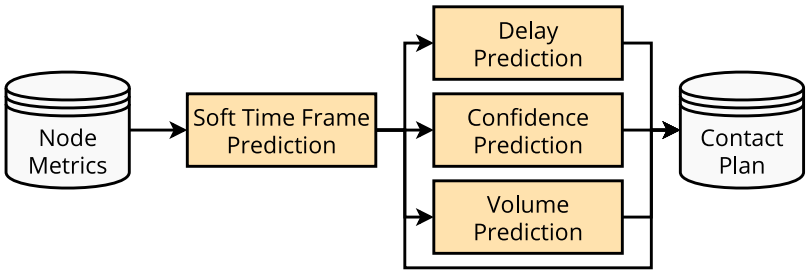

Figure 5.4: Contact prediction process

Overall, the prediction phase can be divided into four steps. Similarly to calculating location metrics, the soft time frame has to be predicted first, as it is an input to the following three prediction steps. The latter, however, can be executed in parallel. An overview of the steps performed in the prediction phase is given in figure 5.4. It is expected that predictions either occur within a fixed interval T_{pred} or are executed immediately when a new pre-processed observation becomes available. In the latter case, T_{pred} is set to zero.

5.6.1 Soft Time Frame Prediction

The soft time frame is predicted using the available location metrics. This step is based on the concept which has been discussed in [WF16], but is generalized here to be applicable to further scenarios with predictable node mobility.

For each pair of nodes (N_s, N_r), a *visibility function* r'_{vis} is defined depending on the available location metrics. This function calculates the time-dependent *range* $r_{vis,N_s,N_r}(t)$ between the involved nodes, as defined by equation 5.16.

$$r_{vis,N_s,N_r}(t) = r'_{vis}(t, N_s, N_r, M_{L,N_s}, M_{L,N_r}) \tag{5.16}$$

A positive result of this function indicates that the *range* of the involved nodes is below the maximum range of their communication systems, i.e., the nodes are *in range* of each other. It should be noted that *range* in this context does not always mean distance: For example, in the case of LEO satellites, a common way to determine whether a satellite is in range of a ground station is to compare the elevation over the horizon to a threshold (see also section 4.4). The maximum range parameter should be set in a pessimistic manner, such that all possible periods of connection establishment are subsets of the soft time frame.

This way, the set of soft time frames consists of all continuous intervals $[t^*_{start,i}, t^*_{end,i}]$ for which $r_{vis,N_s,N_r}(t) \geq 0$ for all $t \in [t^*_{start,i}, t^*_{end,i}]$.

5.6.2 Delay Prediction

After predicting the soft time frame, an estimation for the delay can be derived similarly. The physical distance between the nodes can be calculated based on the location functions. By that, the maximum distance r_{max} during each soft time frame can be inferred using equation 5.17.

$$r_{max}(C^+) = max_{t=t^*_{start}}^{t^*_{end}} |\vec{L}_{N_s}(M_{L,N_s}, t) - \vec{L}_{N_r}(M_{L,N_r}, t)| \tag{5.17}$$

An estimate for the maximum delay can be derived by calculating the maximum line-of-sight delay via the signal propagation speed v_s using equation 5.18.

$$\delta_{max}(C^+) = \frac{r_{max}(C^+)}{v_s} + \delta_0 \tag{5.18}$$

In many use cases employing radio links, the speed of light c may be an appropriate estimate for v_s.

A constant margin δ_0 can be added to the maximum line-of-sight delay to adjust the estimate for delays caused by the transceiver systems and the convergence layer, plus additional delays expected in the given use case, e.g., due to non-line-of-sight propagation.

5.6.3 Confidence Prediction

The contact confidence is predicted using a function based on the stored confidence metrics. This function is applied as shown in equation 5.19.

$$P_{contact}(C^+) = P'_{contact}(C^+, M_{P,N_s,N_r}) \tag{5.19}$$

5.6.4 Volume Prediction

Likewise, contact volume prediction employs a function of the volume metrics for the node tuple, as indicated by equation 5.20.

$$V_{contact}(C^+) = V'_{contact}(C^+, M_{V,N_s,N_r}) \tag{5.20}$$

5.7 Instantiation Guideline

Because the introduced prediction framework is based on a set of functions, it is highly flexible and can be adapted to a range of scenarios with different characteristics. This section provides a guideline for deriving an instance of the framework, which can be used to develop a technical realization. Table 5.1 gives an overview of functions that have to be defined during instantiation.

Function	Section	Description
$\vec{L}_N(M_{L,N}, t)$	5.4	computes the location of a node N at a given time t
$M'_L(M_{L,N,i}, \mathbb{O}'_{N_s,N_r}, p_N)$	5.5.2	updates the node location metrics
$M'_P(M_{P,N_s,N_r,i}, \mathbb{O}'_{N_s,N_r}, M_{L,N_s}, M_{L,N_r})$	5.5.3	updates the contact confidence metrics
$M'_V(M_{V,N_s,N_r,i}, \mathbb{O}'_{N_s,N_r}, M_{L,N_s}, M_{L,N_r})$	5.5.4	updates the contact volume metrics
$r'_{vis}(t, N_s, N_r, M_{L,N_s}, M_{L,N_r})$	5.6.1	computes the visibility between two nodes at a given time t
$P'_{contact}(C^+, M_{P,N_s,N_r})$	5.6.3	computes the confidence for a given prediction C^+
$V'_{contact}(C^+, M_{V,N_s,N_r})$	5.6.4	computes the volume for a given prediction C^+

Table 5.1: Functions to be defined during instantiation

By providing the function definitions, the framework is tailored to a specific use case. This instantiation can be performed as a three-step process:

1. **Analyze the use case**. In the first step, the use case has to be analyzed to identify the sources of metric information.

2. **Define node metrics**. Secondly, the node metric tuples have to be defined, i.e., it needs to be determined which parameters are part of them and which order is applied in their representation.

3. **Define functions**. In the last step, the metric update functions as well as the prediction functions have to be defined.

An instantiation for the Ring Road use case is presented in the next section and provides a straight-forward example for the adaptation to further use cases.

5.8 Instantiation for Ring Road Networks

By the instantiation of the prediction framework for Ring Road networks provided in this section, a complete algorithmic description to obtain contact predictions from contact observations becomes possible. The following descriptions are structured by the parameter that is predicted, i.e., the metric inference and prediction approaches for the soft time frame, the contact confidence, and the contact volume are described in separate sections.

It should be noted that no adaptation is necessary for the delay. This parameter is derived from the node locations as delineated in subsection 5.6.2. Additionally, it is assumed that in the targeted use case, during a contact between two nodes, only one communication link is active per direction, i.e., two nodes do not use multiple communication systems with each other at the same time.

5.8.1 Soft Time Frame

For allowing time frame predictions, first, the location functions have to be defined for all types of nodes that are involved. The inference and validation of node locations in a Ring Road network have been discussed in detail in [WF16].

In the case of satellites, the location function is based on the orbital elements that are stored as node metrics. These data can, e.g., be uploaded by the mission control center operating the satellite network. The satellites themselves may broadcast their orbital elements with discovery beacon messages, such that every ground station can obtain and store them on the first contact. Thus, assuming an orbital propagator (such as SGP4 [Val+06] or a high-precision orbital propagator based on differential equations) is available as a function propagate($M_{L,S}, t$), the location of a satellite node S can be calculated using equation 5.21.

$$\vec{L_S}(M_{L,S}, t) = \text{propagate}(M_{L,S}, t) \tag{5.21}$$

In the case of ground stations (G), the constant location vector itself is stored as the metric value, as shown in equation 5.22.

$$\vec{L_G}(M_{L,G}, t) = \vec{L_G}(M_{L,G}) = M_{L,G} \tag{5.22}$$

Based on the locations of a satellite and a ground station of which either may be the sending and the other the receiving node, the visibility function can be defined using the elevation of the satellite $\theta_{N_s,N_r}(t)$, as shown in equation 5.23. The threshold θ_{min} is set depending on the used communication systems, such that no link can be expected in case the satellite is at a lower elevation.

$$r'_{vis}(t, N_s, N_r, M_{L,N_s}, M_{L,N_r}) = \theta_{N_s,N_r}(t) - \theta_{min} \tag{5.23}$$

The elevation θ is a function of the location vectors, as defined by equation 5.24.

$$\theta_{N_s,N_r}(t) = \theta(\vec{L_S}(M_{L,S}, t), \vec{L_G}(M_{L,G}, t)) \tag{5.24}$$

In the context of equation 5.24, $S = N_s$ and $G = N_r$ if N_s is a satellite and N_r is a ground station, and $S = N_r$ and $G = N_s$ if N_r is a satellite and N_s is a ground station. The elevation can be calculated using trigonometric laws as outlined in [WF16]. An example formula for constant location vectors $\vec{L_S}$ and $\vec{L_G}$ is given by equation 5.25.

$$\theta(\vec{L_S}, \vec{L_G}) = \arcsin \frac{|\vec{L_S}|^2 - |\vec{L_G}|^2 - |\vec{L_S} - \vec{L_G}|^2}{2 \cdot |\vec{L_G}| \cdot |\vec{L_S} - \vec{L_G}|} \tag{5.25}$$

Similarly, the azimuth α can be calculated from the location vectors. The function definition is provided by equation 5.26. The azimuth defines the horizontal angular direction in which the satellite is observed from the perspective of the ground station and is derived using trigonometric laws as well.

$$\alpha_{N_s,N_r}(t) = \alpha(\vec{L_S}(M_{L,S}, t), \vec{L_G}(M_{L,G}, t)) \tag{5.26}$$

As it is assumed that the values of location metrics are provided either by static configuration, by distribution, or via broadcasted discovery beacons, the location metric update function does not depend on the set of contact observations. Its definition is provided by equation 5.27.

$$M'_L(M_{L,N,i}, \mathbb{O}'_{N_s,N_r}, p_N) = M_{L,N,i} \tag{5.27}$$

5.8.2 Confidence

The applied confidence metric is called *node reliability*. This value describes the estimated likelihood that a node will be available during the next predicted contact. Hence, it reflects indeterministic patterns of contact occurrence with a given node, while random variations during contacts are taken into account as part of the predicted volume.

It is assumed that, in Ring Road networks, the probability of contact occurrence varies much more slowly than new contacts with the affected nodes are observed. Thus, the node reliability metric can converge toward a constant value representing the average contact probability. When a change occurs (e.g., a node breaks down), the metric can again converge (in the case of node downtime, to zero).

For estimating the node reliability metric, two techniques are proposed that have been evaluated for a Ring Road topology by the author in [Wal17]: A ratio-based inference and an exponential averaging method. Example code for both metric inference algorithms is provided in appendix A.

Metric Inference using Ratio

The first technique for calculating a node reliability value is to use the ratio of the count of non-empty processed contact observations divided by the count of contact predictions within a specific interval. On that basis, the confidence metric update function is defined by equation 5.28, with $\mathbb{O}''_{N_s,N_r} \subseteq \mathbb{O}'_{N_s,N_r}$ being the set of the $n_{obs,P}$ most recent processed observations with cardinality $|\mathbb{O}''_{N_s,N_r}| \leq n_{obs,P}$.

$$M'_P(M_{P,N_s,N_r,i}, \mathbb{O}'_{N_s,N_r}, M_{L,N_s}, M_{L,N_r}) = P_{N_s,N_r}(\mathbb{O}''_{N_s,N_r}) \tag{5.28}$$

The node reliability P_{N_s,N_r} is calculated using the mentioned ratio as defined by equation 5.29.

$$P_{N_s,N_r}(\mathbb{O}''_{N_s,N_r}) = \frac{|\{O'' \,|\, O'' \in \mathbb{O}''_{N_s,N_r} \wedge \int_{t^*_{start}}^{t^*_{end}} br'(\tau) \cdot d\tau > 0 \wedge \int_{t^*_{start}}^{t^*_{end}} Q'_{link}(\tau) \cdot d\tau > 0\}|}{|\{O'' \,|\, O'' \in \mathbb{O}''_{N_s,N_r}\}|} \tag{5.29}$$

It should be noted that the set of processed contact observations $\mathbb{O}''_{N_s,N_r}$ also contains elements for which no contact has been observed or no data could be transmitted due to insufficient signal quality. In these cases, $\int_{t^*_{start}}^{t^*_{end}} br'(\tau) \cdot d\tau = 0$ or $\int_{t^*_{start}}^{t^*_{end}} Q'_{link}(\tau) \cdot d\tau = 0$.

Metric Inference using Exponential Weighted Moving Average

In [Wal17], a second technique for confidence estimation has been evaluated. Compared to using a ratio, an exponential average is applied iteratively, allowing faster reactions to changes in the effective probability of contact occurrence. The corresponding confidence metric update function is defined by equation 5.30.

$$M'_P(M_{P,N_s,N_r,i}, \mathbb{O}'_{N_s,N_r}, M_{L,N_s}, M_{L,N_r}) = P_{N_s,N_r,i+1}(P_{N_s,N_r,i}, \mathbb{O}'_{N_s,N_r}) \tag{5.30}$$

In that case, the node reliability value is defined recursively using equation 5.31.

$$P_{N_s,N_r,i+1}(P_{N_s,N_r,i}, \mathbb{O}'_{N_s,N_r}) = w_c \cdot \text{last\_observed}(\mathbb{O}'_{N_s,N_r}) + (1 - w_c) \cdot P_{N_s,N_r,i} \tag{5.31}$$

For a set of processed contact observations $\mathbb{O}'$, the function last_observed($\mathbb{O}'$) yields 1 if the latest observation in the set is associated with at least one recorded contact interval (i.e., the integrals over the observed data rate and link quality for the last processed observation are not equal to zero), and 0 otherwise.

By the parameter w_c (the *confidence observation weight*), the weight put on new observations can be configured. Thus, the speed of convergence and the sensitivity to fluctuations can be controlled.

Prediction

For the contact confidence, the prediction approach is straight-forward: As the stored node reliability metric already indicates the estimated probability of the next contact, the up-to-date value of this metric is used during the prediction step as the contact confidence value. The corresponding function definition is provided by equation 5.32.

$$P'_{contact}(C^+, M_{P,N_s,N_r}) = M_{P,N_s,N_r} = P_{N_s,N_r} \tag{5.32}$$

5.8.3 Volume

For predicting the contact volume, the concept evaluated in [WF18] is taken as a basis. As discussed in subsection 2.2.2, the measured signal quality throughout a contact is influenced by a variety of deterministic and random factors.

Two techniques have been developed to predict contact volumes based on available observations. The first of them uses a two-state model to represent the contact interval and does not take into account the azimuth-dependency of the fading loss (see subsection 2.2.2). The second technique extends this model by an additional state to also consider fading at low elevation angles.

Two-state Volume Prediction

For the two-state prediction technique, it is assumed that during a contact the communication link may be either established and stable (the *good state*) or unavailable (the *off state*). It is further assumed that the link becomes available and breaks at an elevation angle that is approximately constant over subsequent contacts. Contact volume prediction is based on the inference of this elevation threshold.

The approach represents an adaptation of the concept in [WF18]. While the latter employs a threshold as well, it is based on the estimated signal loss. Additional characteristics that could be taken into account by the concept in [WF18] (e.g., the frequency) are considered constant for any pair of nodes in this instantiation.

The two-state volume metrics tuple is defined by equation 5.33.

$$M_{V,N_s,N_r} = (\theta_{good}, q_{good}) \tag{5.33}$$

The tuple is composed of the threshold elevation angle for the *good state* θ_{good} and an associated link quality estimation q_{good}. In order to calculate these metrics, the following steps are performed for every of the $n_{obs,V}$ most recent processed observations O''. In that context, $O'' \in \mathbb{O}''_{N_s,N_r} \subseteq \mathbb{O}'_{N_s,N_r}$ with $|\mathbb{O}''_{N_s,N_r}| \leq n_{obs,V}$.

Firstly, the signal quality $Q'_{link}(t)$ is sampled over the observation interval of O'', yielding a series $Q^*_{link} = (q_{link,0}, ..., q_{link,n_s-1})$ with n_s being the number of samples, i.e., the contact duration divided by the sample duration $T_{sample,V}$ and rounded down to the next integer.

Secondly, the samples are processed to derive a *volume achievement ratio* $r_{V,s,i}$ for each sampling interval $[t_{s,0,i}, t_{s,1,i}] \subseteq [t^*_{start}, t^*_{end}]$. This ratio is calculated using equation 5.34, dividing the achieved volume of the sampling interval by the volume achievable with the maximum data rate of the communication system.

$$r_{V,s,i} = \frac{V_{s,observed,i}}{V_{s,max,i}} \tag{5.34}$$

The maximum sample volume $V_{s,max,i}$ is defined by equation 5.35, with br_{max} being the maximum data rate of the communication system.

$$V_{s,max,i} = (t_{s,1,i} - t_{s,0,i}) \cdot br_{max} \tag{5.35}$$

How the *achieved sample volume* $V_{s,observed,i}$ is calculated is dependent on the technical realization. For example, if the residual bit error rate on the convergence layer is available as signal quality metric (i.e., $q_{link,i} = BER_{CL,i}$ for $q_{link,i} \in Q^*_{link}$), the CLA detects all errors and performs ideal re-transmission, and the data rate of the link is constant, the achieved volume can be approximated using the convergence-layer packet size $l_{packet,CL}$, as shown in equation 5.36.

$$V_{s,observed,i} = V_{s,max,i} \cdot (1 - BER_{CL,i})^{l_{packet,CL}} \tag{5.36}$$

Thirdly, the start and end of the *good* interval $[t_{good,0}, t_{good,1}] \subseteq [t^*_{start}, t^*_{end}]$ are derived for the processed observation O'' using equations 5.37 and 5.38. R_V is the series of volume achievement ratios with $R_V = (r_{V,s,0}, ..., r_{V,s,n_s-1})$. A configurable threshold $r_{V,s,min,good}$ is employed in the calculation, defining the *minimum volume achievement ratio* for the *good state*.

$$t_{good,0} = t_{s,0,a} \text{ for } a = min\{i \,|\, r_{V,s,i} \in R_V \wedge r_{V,s,i} \geq r_{V,s,min,good}\} \tag{5.37}$$
$$t_{good,1} = t_{s,1,b} \text{ for } b = max\{i \,|\, r_{V,s,i} \in R_V \wedge r_{V,s,i} \geq r_{V,s,min,good}\} \tag{5.38}$$

Especially if the sampling interval is short, an implementation may choose to extend the last condition such that multiple subsequent elements $r_{V,s,i} \in R_V$ have to be greater than $r_{V,s,min,good}$ in both equations.

Fourthly, two elevation samples $\theta_{good,0}$ and $\theta_{good,1}$ are calculated for each observation from the timestamps as defined by equations 5.39 and 5.40. For calculating the elevation angles $\theta_{N_s,N_r}(t)$, equation 5.24 is applied.

$$\theta_{good,0}(O'') = \theta_{N_s,N_r}(t_{good,0}) \tag{5.39}$$
$$\theta_{good,1}(O'') = \theta_{N_s,N_r}(t_{good,1}) \tag{5.40}$$

Fifthly, in addition to the elevation thresholds, an *average link quality indicator* q_{good} is derived for the determined interval. This indicator is calculated as a volume achievement ratio for the complete *good* interval as defined by equation 5.41, with a and b being set as defined by equations 5.37 and 5.38.

$$q_{good}(O'') = r_{V,good} = \frac{\sum_{i=a}^{b} V_{s,observed,i}}{\sum_{i=a}^{b} V_{s,max,i}} \tag{5.41}$$

As mentioned, the above steps are performed for every of the $n_{obs,V}$ most recent observations. The metric values are ultimately derived by obtaining the median of the existing samples using equations 5.42 and 5.43.

$$\theta_{good} = median(\{\theta_{good,0}(O'') \,|\, O'' \in \mathbb{O}''_{N_s,N_r}\} \cup \{\theta_{good,1}(O'') \,|\, O'' \in \mathbb{O}''_{N_s,N_r}\}) \tag{5.42}$$

$$q_{good} = median(\{q_{good}(O'') \,|\, O'' \in \mathbb{O}''_{N_s,N_r}\}) \tag{5.43}$$

The function updating the volume metrics is defined by equation 5.44.

$$M'_V(M_{V,N_s,N_r,i}, \mathbb{O}'_{N_s,N_r}, M_{L,N_s}, M_{L,N_r}) = (\theta_{good}, q_{good}) \tag{5.44}$$

Volume prediction is performed based on a *good* interval $[t^*_{good,0}, t^*_{good,1}]$ calculated for the upcoming contact C^+ using equations 5.45 and 5.46.

$$t^*_{good,0} = min\{t \,|\, t^*_{start} \leq t \leq t^*_{end} \wedge \theta_{N_s,N_r}(t) \geq \theta_{good}\} \tag{5.45}$$
$$t^*_{good,1} = max\{t \,|\, t^*_{start} \leq t \leq t^*_{end} \wedge \theta_{N_s,N_r}(t) \geq \theta_{good}\} \tag{5.46}$$

Afterward, the volume prediction is derived as the product of the *good* interval duration, the maximum data rate of the communication system br_{max}, and the link quality estimation for the *good state* q_{good}, via equation 5.47.

$$V'_{contact}(C^+, M_{V,N_s,N_r}) = T^*_{good} \cdot br_{max} \cdot q_{good} \tag{5.47}$$

The duration of the *good* interval is determined using equation 5.48.

$$T^*_{good} = t^*_{good,1} - t^*_{good,0} \tag{5.48}$$

A code sample for the two-state prediction algorithm is provided in appendix A.

Three-state Volume Prediction

The second volume prediction technique supports an estimation of the fading amplitude at low elevation angles using an additional state of the link. This state is termed the *bad state*. It is assumed that when a link is in *bad state*, it allows for data transmission, though, possibly facing a higher error rate. This differentiation is inspired by the LEO satellite channel model and analysis published by Lopez-Salamanca et al. [LS+19].

The volume metrics tuple for the three-state technique is defined by equation 5.49.

$$M_{V,N_s,N_r} = (\theta_{good}, q_{good}, \Theta_{bad}, q_{bad}) \tag{5.49}$$

In extension to the *good* elevation threshold and quality estimation, a quality estimation q_{bad} as well as a series of elevation thresholds Θ_{bad} are added for the *bad state*. The latter is defined by equation 5.50.

$$\Theta_{bad} = (\theta_{bad,\alpha_0,\alpha_1}, ...\theta_{bad,\alpha_{n_{az}-1},\alpha_{n_{az}}}) \tag{5.50}$$

Each elevation threshold for the *bad state* is connected to a given range in the azimuth angle between satellite and ground station, with: $[\alpha_i, \alpha_{i+1}) \subseteq [0\,°, 360\,°)$ and $\alpha_i < \alpha_{i+1}$ for all $0 \leq i < n_{az}$.

By that, different thresholds can be recorded for different azimuth angles. The granularity can be chosen by the implementation, considering the trade-off between a high resolution for the azimuth angle and the resulting amount of data that has to be recorded and stored as metrics.

The first two elements in the metrics tuple are calculated in the same manner as in the two-state metric inference process, with one exception: If the elevation threshold for the *good state* θ_{good} is smaller than a configurable parameter $\theta_{good,min}$, it is set to that value. Thus, $\theta_{good,min}$ is a lower bound for the threshold.

The *bad state* values are determined as follows. For every of the $n_{obs,V}$ most recent observations, a *bad* interval $[t_{bad,0}, t_{bad,1}] \subseteq [t^*_{start}, t^*_{end}]$ is derived in the same manner as defined by equations 5.37 and 5.38 for the *good* interval. In that case, a configurable threshold $r_{V,s,min,bad} \leq r_{V,s,min,good}$ is employed as the *minimum volume achievement ratio*.

Using the timestamps $t_{bad,0}$ and $t_{bad,1}$, two elevation thresholds $\theta_{bad,0}(O'')$ and $\theta_{bad,1}(O'')$ are derived in the same manner as defined by equations 5.39 and 5.40 for the *good* case. Furthermore, two azimuth values $\alpha_{bad,0}$ and $\alpha_{bad,1}$ are calculated from the *bad* interval as defined by equations 5.51 and 5.52. (See equation 5.26 for the definition of $\alpha_{N_s,N_r}(t)$.)

$$\alpha_{bad,0}(O'') = \alpha_{N_s,N_r}(t_{bad,0}) \tag{5.51}$$
$$\alpha_{bad,1}(O'') = \alpha_{N_s,N_r}(t_{bad,1}) \tag{5.52}$$

For the start and end times of the considered observations, $2 \cdot n_{az}$ sets of elevation samples are derived using equations 5.53 and 5.54. The sets containing start elevations are termed $\Theta'_{bad,\alpha_i,\alpha_{i+1},0}$, whereas the sets containing end elevations are termed $\Theta'_{bad,\alpha_i,\alpha_{i+1},1}$.

$$\Theta'_{bad,\alpha_i,\alpha_{i+1},0} = \{\theta_{bad,0}(O'') \,|\, O'' \in \mathbb{O}''_{N_s,N_r} \wedge \alpha_i \leq \alpha_{bad,0}(O'') < \alpha_{i+1}\} \tag{5.53}$$
$$\Theta'_{bad,\alpha_i,\alpha_{i+1},1} = \{\theta_{bad,1}(O'') \,|\, O'' \in \mathbb{O}''_{N_s,N_r} \wedge \alpha_i \leq \alpha_{bad,1}(O'') < \alpha_{i+1}\} \tag{5.54}$$

Based on these sets, n_{az} elevation thresholds for the *bad state* are calculated with equation 5.55.

$$\theta_{bad,\alpha_i,\alpha_{i+1}} = median(\Theta'_{bad,\alpha_i,\alpha_{i+1},0} \cup \Theta'_{bad,\alpha_i,\alpha_{i+1},1}) \tag{5.55}$$

The average link quality indicator q_{bad} is derived for the *bad* interval in the same manner as q_{good} for the *good* interval using equation 5.43.

For the metric update function, a definition is given by equation 5.56.

$$M'_V(M_{V,N_s,N_r,i}, \mathbb{O}'_{N_s,N_r}, M_{L,N_s}, M_{L,N_r}) = (\theta_{good}, q_{good}, \Theta_{bad}, q_{bad}) \tag{5.56}$$

The three-state volume prediction process is based on the two-state variant.

Firstly, the *good* interval $[t^*_{good,0}, t^*_{good,1}]$ for the upcoming contact C^+ is calculated in the same manner as done with the two-state technique.

Secondly, the azimuth values $\alpha_{N_s,N_r}(t^*_{start})$ and $\alpha_{N_s,N_r}(t^*_{end})$ are used to look up the applicable elevation thresholds $\theta^*_{bad,start}$ and $\theta^*_{bad,end}$ from the set Θ_{bad}, as defined by equations 5.57 and 5.58.

$$\theta^*_{bad,start} = \theta_{bad,\alpha_i,\alpha_{i+1}} \text{ with } \alpha_i \leq \alpha_{N_s,N_r}(t^*_{start}) < \alpha_{i+1} \wedge 0 \leq i < n_{az} \tag{5.57}$$
$$\theta^*_{bad,end} = \theta_{bad,\alpha_i,\alpha_{i+1}} \text{ with } \alpha_i \leq \alpha_{N_s,N_r}(t^*_{end}) < \alpha_{i+1} \wedge 0 \leq i < n_{az} \tag{5.58}$$

Thirdly, the *bad* interval $[t^*_{bad,0}, t^*_{bad,1}]$ is derived by using the thresholds $\theta^*_{bad,start}$ and $\theta^*_{bad,end}$ in equations 5.45 and 5.46 instead of θ_{good}. This leads to equations 5.59 and 5.60.

$$t^*_{bad,0} = min\{t \mid t^*_{start} \leq t \leq t^*_{end} \wedge \theta_{N_s,N_r}(t) \geq \theta_{bad,start}\} \tag{5.59}$$

$$t^*_{bad,1} = max\{t \mid t^*_{start} \leq t \leq t^*_{end} \wedge \theta_{N_s,N_r}(t) \geq \theta_{bad,end}\} \tag{5.60}$$

Fourthly, the contact volume prediction is calculated by adding the volume of the *good* interval to that of the surrounding parts of the *bad* interval, as defined by equation 5.61.

$$V'_{contact}(C^+, M_{V,N_s,N_r}) = (T^*_{good} \cdot q_{good} + (T^*_{bad} - T^*_{good}) \cdot q_{bad}) \cdot br_{max} \tag{5.61}$$

The duration of the *bad* interval is determined using equation 5.62.

$$T^*_{bad} = t^*_{bad,1} - t^*_{bad,0} \tag{5.62}$$

5.9 Discussion of Applicability to Further Use Cases

Intentionally, the introduced prediction framework is highly flexible and modular. The following main features are part of its design:

- **Straight-forward, modular definition of functions**. All core algorithms are based on modular functions that can be defined and replaced to adapt the framework to a given scenario.
- **Asynchronous computations**. All necessary computations can be performed asynchronously, i.e., no real-time operation is required. This way, observations may be forwarded to a central, powerful node to offload computations or build a *spot of maximum knowledge* as discussed in [FWB17].

Though, the framework requires a set of properties from the network to which it is applied. These are summarized in the following list to support considerations of transferring the presented concepts to further use cases.

1. **Deterministic mobility with known movement model**. Network nodes have to be subject to deterministic mobility, and a model to calculate their locations has to be available.
2. **Location-dependent contacts**. The occurrence and time interval of contacts has to depend on the location of the involved nodes.

3. **Small variation in delay**. Compared to the contact periods, the variations in propagation delay have to be small.

4. **Availability of a trusted entity**. At least one trusted entity has to be present in the network and supply authenticated initial information to the nodes. In most scenarios, such an entity should be available at some point. In the Ring Road use case, this role could be served by a mission control center of the satellite operator.

5. **Discovery mechanism for nodes and link intervals**. A discovery component is required that is able to collect necessary contact observations, including the node identifiers. Furthermore, even if no payload data are transmitted between two nodes, the discovery component should be able to observe time frames of possible data transmission and report them to the prediction component.

6. **Availability of link information**. At least the (potentially time-varying) effective data rate has to be available for each link associated with a contact observation to determine its volume.

7. **Slow changes in contact probability**. In the targeted scenarios, the probability of contact occurrence has to change slowly, i.e., with a period much larger than the inter-contact time, allowing the confidence metrics to converge toward constant values between changes.

6 Leveraging Predicted Contacts for Routing

To effectively leverage a prediction component that has been derived from the framework introduced in the previous chapter, it has to be combined with a routing component. From the set of routing techniques present in the state of the art as outlined in chapter 4, only deterministic path selection algorithms support the scheduling of data for specific contacts. Though, the available candidates do not support all properties of the applied contact description and, thus, have to be extended.

This chapter introduces such an extension and, by that, provides a solution for prediction-based routing. First, section 6.1 gives an overview of the system architecture, its components, and the data flow between them. Section 6.2 describes the metric distribution concept that is required to synchronize the information necessary to generate contact plans. Subsequently, section 6.3 introduces an extended probabilistic routing approach that makes use of the full contact description provided by the prediction framework. Section 6.4, finally, summarizes and discusses noteworthy features of the overall system.

6.1 Overview

In figure 6.1, the overall concept for prediction-based bundle routing is depicted. The highlighted components are addressed in this chapter.

With a configurable period of T_{update}, the *routing component* requests a new *contact plan* from the *prediction component*, which generates it based on the stored *node metrics*. The period may be set to zero to force updates immediately when new node metrics become available. However, as changes to the contact plan invoke the re-generation of routes, buffering can reduce the computational impact.

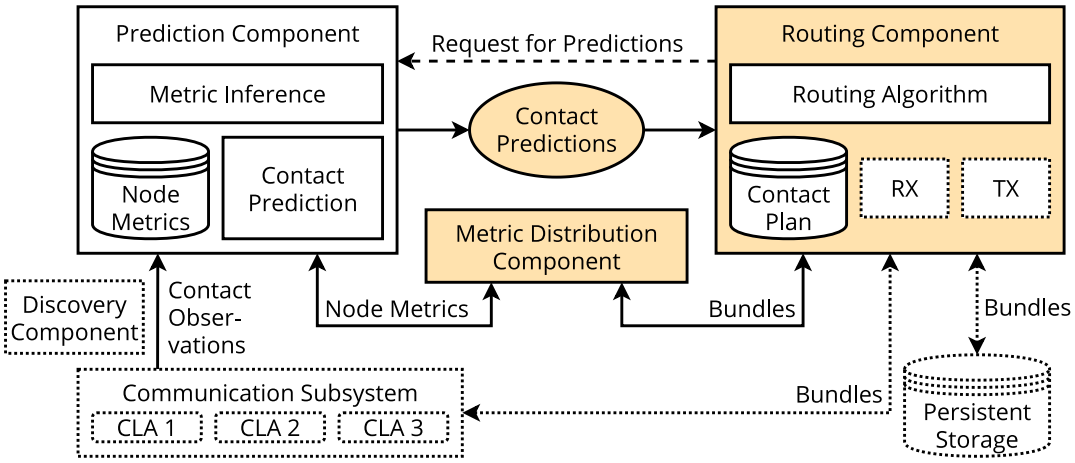

Figure 6.1: Components and context of the prediction-based routing approach

After computing a route for a bundle based on the contact plan, it is scheduled in a transmission queue. Then, forwarding is performed by removing one bundle after another from the queue and handing it over to a CLA for transmission. In order to enable contact predictions for non-neighboring nodes, a *metric distribution component* is added, exchanging node metrics inferred by the prediction component with other network nodes.

6.2 Distribution of Node Metrics

Besides contact prediction itself, the distribution of necessary information is a core feature required to enable proactive routing in a network topology that has to be either fully or partially discovered by the participating nodes. State-of-the-art deterministic routing using CGR assumes the advance distribution or configuration of a *contact plan* (see subsection 4.2.1). In a predictable topology that is entirely under the control of a single entity, providing contact plans to all nodes is a straightforward and controllable way to allow deterministic routing. In the topologies considered in this thesis, however, generating a full contact plan in advance at a central location is not possible. The set of active nodes, as well as contact characteristics such as the contact probability between two nodes, may be subject to variations over time.

The probabilistic prediction framework introduced in the previous chapter facilitates the inference of these factors from the point of view of an individual node in the form of *node metrics* (see section 5.4). However, node-local predictions alone only provide a fraction of the information required to determine optimal paths through the network.

Consequently, the distribution of either the contact plan or the underlying node metrics enabling its generation is necessary. Four core advantages of distributing node metrics over contact plans are expected, of which an overview is given in the following list.

1. **Extended validity duration**. As node metrics are designed to converge to constants representing the network topology, a set of node metrics is expected to be applicable for deriving contacts over extended periods of time.

2. **Compact representation**. Properties of future contacts can be represented in a much more compact form compared to a contact plan.

3. **Simplified merging**. As node metrics converge to constants in case the topology does not change, when receiving multiple sets of node metrics describing the same node relations, it is much easier to detect changes and merge the values to derive an updated representation.

4. **Collaborative knowledge aggregation**. Several nodes that share a neighbor (e.g., satellites that encounter the same ground station) may combine the information available about it based on the node metrics distributed by others. Such an aggregation of knowledge can make it possible to achieve faster convergence and more precise predictions, as shown in [Wal17].

Besides by the nodes participating in data forwarding, node metrics may also be generated by central nodes that possess up-to-date knowledge of parts of the topology. In a Ring Road network, for example, the mission control center may broadcast node metrics describing the characteristics of the deployed satellites as well as of hot spots operated by the network provider.

6.2.1 Information Exchange

The applied process is shown in figure 6.2. As soon as a new prediction becomes available for a given neighbor, a node serializes and encapsulates the information contained in the node metrics storage for this neighbor into a *metric update* message. For reducing the transmission overhead, these messages may be buffered for a maximum duration defined as T_{dist}, and multiple messages can be aggregated into a single bundle.

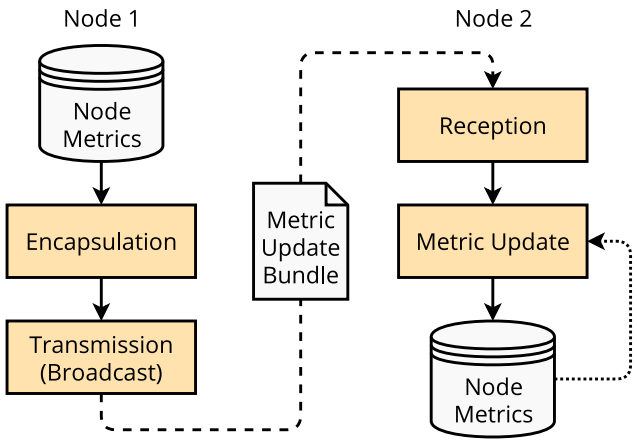

Figure 6.2: Metric distribution process

The resulting bundles are distributed via the routing component to all available neighbors in a multicast manner. At the start of each contact, two nodes exchange all metric update bundles which they have not already exchanged in the past. This implies that a node remembers to which neighbor it has sent a given metric update bundle.

When new metric update bundles are received, the node decodes the contained messages and performs a *metric update* step, as described in subsection 6.2.2.

The exchanged metric update messages have to contain at least the following information for each set of nodes they describe:

1. **Tuple of node identifiers**. Each distributed set of metrics is associated with a tuple of unique node identifiers.

2. **Node metrics**. The current sets of node metrics are contained in the message in a serialized format.

3. **Update timestamp**. The timestamp associated with the last update of the contained data allows for determining their timeliness.

It should be noted that metric distribution consumes transmission bandwidth otherwise usable for payload data exchange. Therefore, there is a trade-off between distributing node metrics more often and, thus, inducing a higher load on the network and accepting a less up-to-date description of the topology at individual nodes while causing less traffic.

6.2.2 Metric Update

To update locally-available node metrics based on a received metric update message, first, the contained tuple of identifiers is checked. If the local node is part of it and has inferred metrics for the node relation contained within the message, the message is not processed further as locally-discovered metrics take precedence.

In some cases, this behavior may be adjusted: The metrics describing specific node relations can enable the inference of the characteristics of others. For example, if all satellites which regularly connect to a specific cold spot report a reduced probability of contacts to this station, another satellite may also observe such degradation in availability. This way, the inference of node metrics from values reported by transitive neighbors is possible, as shown in [Wal17]. However, these concepts are considered subject to future work as further analyses on their performance will be necessary to leverage them effectively.

Following the filtering for applicable node tuples, the timestamp associated with the metric update message is compared to the last update of the metrics stored locally for the given tuple of nodes. If the timestamp is more recent or no metrics have been stored yet for the tuple of nodes, the local values are overwritten, i.e., the received and de-serialized node metrics are *assigned* to the local data base. If the node tuple is not yet known, it is created using the received message.

6.2.3 Application to Fully-deterministic Networks

Even in case the network topology is fully-deterministic and a central entity configures all contacts in advance, this concept may be leveraged to reduce the amount of data that has to be transferred. The central node that would otherwise provide a contact plan infers values for the node metrics, which are then broadcasted. In that case, only part of the prediction component has to be implemented on individual nodes, as the discovery and metric inference phases are not executed.

6.3 Probability-enhanced Contact Graph Routing

A state-of-the-art routing algorithm was extended to leverage the probabilistic contact predictions.

6.3.1 Conceptual Basis

CGR, as specified in the CCSDS standard document *Schedule-Aware Bundle Routing* (SABR) [CCSDS19], as well as the *Opportunistic CGR* (O-CGR) extension (introduced in [Bur+16]), are used as a basis for the routing concept.

The CGR algorithm can make use of the volume-based contact representation C' (see subsection 5.3) as follows. The expected *mean data transmission rate* (see [CCSDS19], 2.3.1) is computed by dividing the predicted contact volume by the duration of the soft time frame. The soft time frame itself is used as the contact interval for CGR. Additionally, CGR's requirement of all nodes being identified by a *node number* (see [CCSDS19], 2.2) is relaxed to requiring an arbitrary but unique identifier that can be derived for each node.

CGR already takes into account most characteristics describing a contact prediction, except the confidence. As discussed in section 4.2, deterministic routing algorithms do not assume that contacts may have a probability of occurrence below 1.0. An extension to CGR that adds a contact confidence value (O-CGR) was published by Burleigh et al. in [Bur+16]. Though, this approach assumes a reduced confidence only for opportunistic contacts. A concept to extend this work to support contact predictions as applied in this thesis is presented in subsection 6.3.3.

6.3.2 Contact Plan Prediction

To determine the list of routes, CGR requires a *contact plan* from which a *contact graph* is derived. Usually, this contact plan must be provided to the nodes in advance. When a prediction component is used, however, the contact plan has to be generated locally. Furthermore, in case any node metrics change, all contact predictions available for the affected node tuple have to be re-generated to reflect the change.

For reducing the overhead of predicting an extensive number of unnecessary contacts, contact plan prediction is limited as follows. A *maximum look-ahead duration* $T_{lookahead,max}$ is configured that defines the maximum period for which contacts are predicted in advance. Every time node metrics are updated, the prediction process is executed, inferring all upcoming contacts with a start time t^*_{start} not later than the current time advanced by $T_{lookahead,max}$. In that context, it must be ensured that updates happen periodically to enforce the constant availability of a sufficiently-large contact plan. Therefore, a maximum update period $T_{update,max}$ with $T_{update,max} < T_{lookahead,max}$ is defined, with which new contact predictions are added to the contact plan. These updates are performed even if the node metrics were not updated in the meantime.

It is important to note that every update to the contact plan forces the node to discard all computed routes (see [CCSDS19], 3.2.3.1). As new node metrics might be received at a comparably high rate, to limit the computational impact of the resulting contact plan updates, two measures are taken:

1. Metric updates are aggregated and applied only with a certain minimum period $T_{update,min}$ with $T_{update,min} \leq T_{update,max}$. If this period is set to zero, metric updates are applied instantly. However, if a non-zero value is provided, contact plan updates are scheduled with this period and are only executed if new metrics were received or $T_{update,max}$ has elapsed as well.

2. When a contact plan update occurs, bundles that have already been scheduled for transmission are only re-routed if contacts on their planned route are removed. For that purpose, the planned route is stored for each bundle. An implementation may choose to also re-route bundles if they do not continue to fit into the updated contact volumes on the route. However, such an extension is not evaluated in this thesis.

6.3.3 Probabilistic Extensions

Two extensions are added to CGR to consider the contact confidence in routing decisions. A performance analysis of an initial version of these extensions can be found in [WF19]. The referenced version, however, requires a time-varying error probability which is not present in the contact model considered here (see section 5.3). The current extensions are described conceptually in the following, whereas code samples of an implementation can be found in appendix A.

Modified Route Computation

As outlined in subsection 4.2.1, CGR generates a list of routes for all possible destinations in advance, using the contact graph. For every new route, all previously-generated routes shall be excluded (see [CCSDS19], 3.2.6.10). By default, the CGR algorithm only takes the *earliest arrival time* into account. The following modification to this behavior is proposed.

As with the approach taken by O-CGR [Bur+16], every route is tagged with a *route confidence* P_{route} that is calculated from the individual contact confidence values. Given $n_{contacts}$ being the number of contacts on the route and $P_{contact,i}$ being the confidence of the i-th contact, the route confidence is derived using the product of contact confidence values, as shown by equation 6.1.

$$P_{route} = \prod_{i=1}^{n_{contacts}} P_{contact,i} \tag{6.1}$$

Compared to O-CGR, however, the route confidence is not only used for replicating bundles: As introduced in [WF19], a threshold for filtering routes is applied, termed *minimum route confidence* ($P_{route,min}$). Assuming a route with index j has a route confidence of $P_{route,j}$, it is only considered if $P_{route,j} \geq P_{route,min}$.

For computing alternative routes, a *limiting contact* is determined[1] as described for the CGR algorithm in [Fra+17]. This contact is then disabled during all following route searches, e.g., by setting the weights of all edges leading to the associated graph vertex to infinite.

The limiting contact is determined by identifying the (as estimated) least-probable contact as proposed and evaluated in [WF19], with two modifications:

1. The contact confidence is used instead of a time-varying packet error rate, as variations in time cannot be predicted precisely.

2. In case the minimum route confidence is achieved by a given route, the earliest-ending contact is set as limiting contact.

Therefore, if routes do not achieve the minimum route confidence, the contact with the lowest confidence is disabled for following route searches. If two contacts have the same (lowest) confidence value, the earliest-ending of them is disabled.

This approach has three effects:

- The route list generation ensures a minimum confidence per route to reduce the number of low-probability routes and, by that, the number of bundle replicas created for achieving a target confidence value.

- For purely-deterministic contacts (i.e., contact and route confidence being 1.0), the approach is equivalent to the behavior of the unmodified CGR algorithm.

- The order of returned routes (by arrival time) is retained. This way, routes with low arrival times are attempted first.

[1] It should be noted that this approach fully conforms to the SABR specification (see [CCSDS19], 3.2.6.10). The use of Yen's algorithm is provided as an example for an implementation-dependent solution. In [Fra+18], further valid options for determining a limiting contact are compared. In the context of this thesis, a comparison of three options was performed and is documented in section B.3 of the appendices.

Selection of Multiple Routes

In order to increase the chance of (timely) data delivery, bundles are replicated over a set of routes. This replication is based on the mechanism assumed by O-CGR [Bur+16] integrated into the CGR algorithm specified in [CCSDS19]. As in O-CGR, a configurable *minimum total delivery confidence* threshold ($P_{bundle,min}$) is set, defining the minimum confidence the routing algorithm aims to achieve for delivering a given bundle.

From the list of candidate routes (see [CCSDS19], 3.2.6), the first n_{routes} routes are selected for which the *total delivery confidence* of a bundle P_{bundle} exceeds the configured minimum threshold. The total delivery confidence is calculated using equation 6.2.

$$P_{bundle} = 1 - \prod_{j=1}^{n_{routes}} (1 - P_{route,j}) \tag{6.2}$$

It should be noted that routes may overlap at a number of intermediate nodes and, thus, the resulting value does not always reflect the real (theoretical) probability of bundle delivery via the given set of routes, even under the assumption that the exact mathematical probability of occurrence is known for each contact. For that, a much more extensive analysis of individual routes would be necessary.

However, in [WF19], it has been shown that the introduced estimation of the total delivery confidence can already lead to a considerable improvement regarding delivery probability in Ring Road networks with unreliable links. Additionally, the minimum total delivery confidence can be tuned to account for possible inaccuracies in probability estimations. Thus, an extended analysis of the total delivery confidence is left as a subject for future work.

In order to limit the maximum number of created bundle replicas, two additional mechanisms not present in O-CGR are employed in the route selection process:

1. A maximum number of replicas $n_{routes,max}$ is configured, such that only up to $n_{routes} \leq n_{routes,max}$ routes are selected for a bundle.

2. Specific handling is employed for bundles that have already been forwarded. As noted in the SABR specification (see [CCSDS19], 3.2.8.1.4), nodes may discard bundles that have already been forwarded in the past. By that, routing loops can be prevented. Though, by not re-forwarding such bundles, data may be lost. This peculiarity is especially relevant in networks for which the predicted time-varying topology may be inaccurate, i.e., the cases considered in this thesis. A compromise is achieved by limiting n_{routes} to 1 for all bundles that have been forwarded previously. Consequently, these bundles are still sent on one route, but exponential replication of looping bundles is prevented. The behavior in the case of contacts with a confidence value of 1.0 is still identical to the unmodified CGR algorithm.

6.4 Discussion

The concept introduced in this chapter enables prediction-based routing, while still allowing the same flexibility and possibility of control as provided by the unmodified CGR algorithm.

Especially the following features are considered noteworthy:

- Due to having routing behavior that is identical to CGR in the case of fully-deterministic contacts, the routing algorithm is a drop-in replacement for CGR, supporting further networks.
- The metric-based contact prediction approach allows nodes to distribute sets of static values instead of contact plans, effectively reducing the amount of data to be transmitted.
- The probabilistic route selection technique is optimized for creating a low number of replicas due to higher-probability routes being preferred.

An application example, making use of both a prediction component and a proactive routing component in a *Bundle Protocol Agent* (BPA), is shown in figure 6.3.

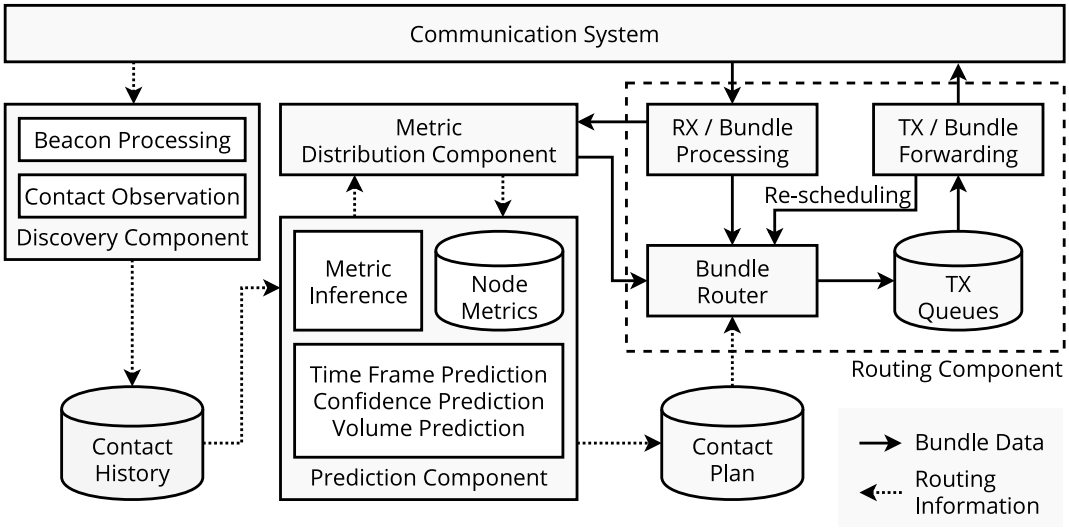

Figure 6.3: Bundle Protocol Agent integration example

7 Evaluation

This chapter documents the evaluation performed for the concepts introduced in chapters 5 and 6 to show their suitability for the Ring Road use case and assess the fulfillment of the defined requirements.

The chapter is structured as follows. Initially, section 7.1 provides an overview of the evaluation methodology. Afterward, the technical realization is described in section 7.2. The conducted validations of the overall toolchain are outlined in section 7.3. Following that, section 7.4 defines realistic test scenarios.

In the second part of the chapter, individual test cases are described, and the results of each test are discussed. The accuracy of contact predictions is assessed in section 7.5, and the performance impact of metric distribution is analyzed in section 7.6. Tests of routing performance aspects are documented in section 7.7, and the implications of prediction-based routing on computational and power demands are discussed in section 7.8. Each test case is introduced with a short summary, which identifies the associated requirements and core parameters.

The chapter concludes with an overview of further findings in section 7.9 and a summary concerning the fulfilled requirements in section 7.10.

7.1 Methodology

A systematic six-step evaluation process was applied. This process is sketched in figure 7.1 and takes into account recommendations by Kuppusamy et al. [Kup+19].

Figure 7.1: Evaluation process

The following steps were performed:

1. **Implement** the concepts which should be evaluated, plus any technique needed for comparison.

2. **Verify** the functions of the implementation. This step includes validations of implemented state-of-the-art algorithms.

3. **Define scenarios**. Thirdly, test scenarios that exhibit the identified characteristics of the use case are defined using realistic data.

4. **Define test cases** to validate the identified requirements. In this step, suitable scenarios as well as algorithms and parameters are selected.

5. **Test** the implementation. The test cases are executed, and relevant performance data are collected to be analyzed in the next step.

6. **Analyze** results. Finally, the data records from each test case are used to analyze the evaluation results and derive findings.

It should be noted that steps 4 to 6 are described in a combined fashion in sections 7.5 to 7.8, to provide a comprehensive overview of the individual test cases.

7.2 Implementation

The described concepts were implemented and embedded into a modular evaluation setup. Earlier versions of the involved tools have already been used in [FW17a], [Wal17], [WF18], and [WF19]. Based on these building blocks, an overall toolchain was developed that incorporates data acquisition, scenario generation, traffic generation, contact prediction, metric distribution, routing, simulation, and statistical analysis. Its components are depicted in figure 7.2.

The toolchain was implemented in the Python 3.7 programming language. A central idea behind this technical decision is the accessibility for possible further developments. Furthermore, the toolchain consists of small individual tools, interacting on the basis of JSON files, to provide flexibility concerning possible extensions and make intermediate results persistently accessible. Hence, the results of every evaluation step can be archived and used for reproducing the process later.

The data acquisition process is based on measurements derived from satellite observations. Using these measurements, characteristics of LEO satellite links can be simulated in a much more realistic manner than assuming and applying random models. Contact observations are assumed to be provided in the form of an audio file containing the received base-band signal, plus a JSON file with metadata. The developed implementation supports the processing of NOAA weather satellite observations (such as the one depicted in the example in subsection 2.2.1) but could be easily extended to support additional satellites. The individual processing steps are delineated in the following.

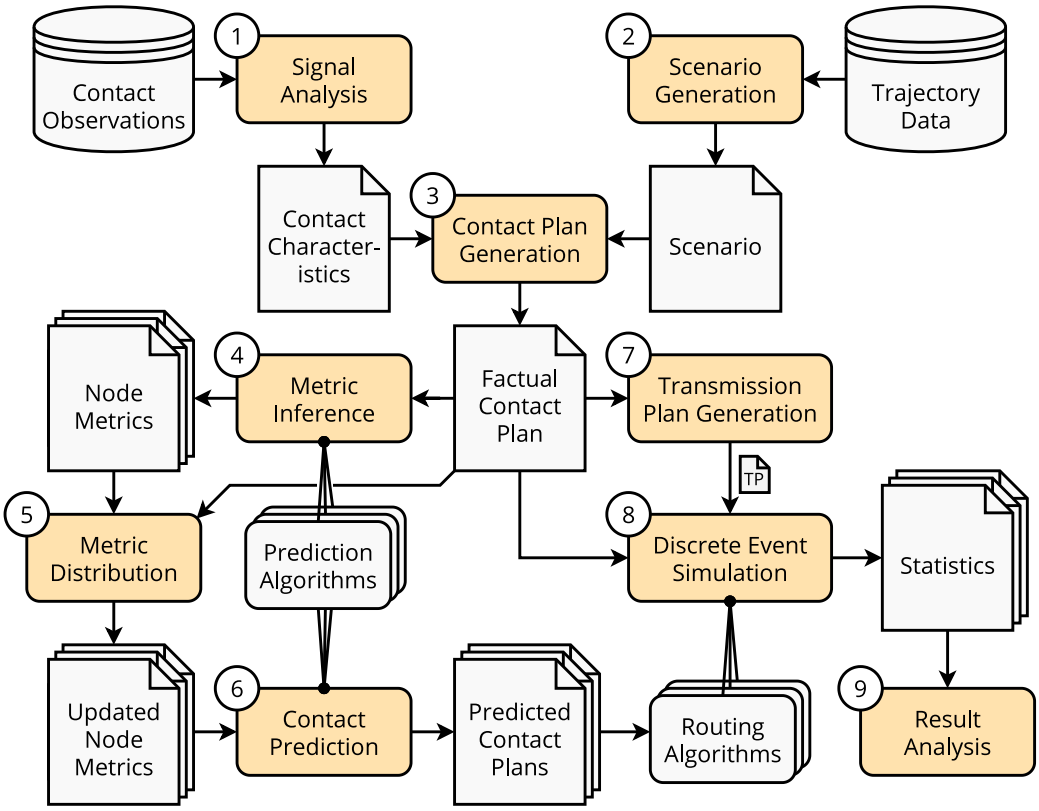

Figure 7.2: Evaluation toolchain and data flow (extension of figure 2 in [WF19])

In a first step (1), the signal file for each observation is decoded. Predictable parts of the signal (e.g., synchronization patterns) are extracted. These are used to derive an estimate for the error rate before coding using the kernel-based estimation technique that has been leveraged in [WF19]. Based on configurable coding characteristics, a residual error rate is calculated. This error rate is considered bi-directionally, as observations of the return channel, i.e., as observed by the satellites, are not commonly available. Though, assuming the same communication systems are used for uplink and downlink, the considered variable influences (see subsection 2.2.2) are symmetrical. Thus, this limitation is deemed acceptable.

The second step (2) generates a custom Ring Road scenario. This scenario describes the satellite orbits, locations of ground stations, and an overall time interval. It can be a subset of the observation data or be fully-independent with an arbitrary size. The latter case is particularly important as only a limited number of satellites can be reliably analyzed for signal quality over the complete duration of every observed contact.

In the third step (3), a *factual contact plan* is generated. This plan indicates contacts in the form they will be simulated, including the interval, disruptions, error rate, and data rate. It adheres to the observation model described in section 5.2. No probability of occurrence is associated with these contacts as they represent the state observed by nodes during the simulation. The contact interval is determined by setting it to the time frame of line-of-sight visibility. Afterward, an observation is selected as follows. If the scenario is a subset of the observation data, the corresponding characteristics are assigned to the factual contacts.

If, however, a generated scenario is used, a fixed mapping to a tuple of nodes from the observation data is built in a round-robin manner. Then, a random observation is selected for the given node tuple. The characteristics of this observation are assigned to the new contact.

Step (4) infers node metrics (see sections 5.4 and 5.5) for each pair of nodes at the respective receiver.

Afterward, metric distribution is performed in step (5), as described in subsection 6.2.1. This step already generates a report containing relevant statistics suitable for evaluating distribution characteristics plus the adjustment of factual contact plans to accommodate for the metric distribution overhead.

In the sixth (6) step, *predicted contact plans* are derived according to the representation introduced in section 5.3 (see also section 5.6). A predicted contact plan contains the topological knowledge of a single network node at a given instant.

In step (7), a *transmission plan* is generated that specifies every bundle injected into the network. The traffic pattern can be configured via command line parameters.

Step (8) invokes an event-based simulation engine. In every simulation run, a specific routing algorithm is applied to the scenario defined by the input data. Afterward, a file with collected statistics is stored. This record is used in the last (9) step to generate a statistical summary and plots for analysis.

7.2.1 Contact Prediction

Contact soft time frames are predicted using the SGP-4 orbital propagation algorithm and a numeric root-finding approach. This implementation was adapted from the one documented in [WF16]. The prediction of contact confidences and volumes, however, was implemented from scratch as a modular library. This library provides functions performing the following actions via a unified interface:

1. **Metric inference**. This function uses a processed contact observation to perform the metric inference step as described conceptually in section 5.5.

2. **Prediction**. The second function leverages the node metrics stored within the class instance to derive a prediction of either the contact confidence or the contact volume (depending on the class).

3. **Metric export**. The stored node metrics can be exported using this function to a serializable format for facilitating metric distribution.

4. **Metric import**. At a node receiving distributed metric bundles, this method is leveraged to update locally-stored node metrics.

Individual classes to be used can be selected flexibly by parameters to the metric inference and prediction tools (steps 4 and 6), which invoke the respective library functions.

7.2.2 Simulation Engine and Routing Algorithms

As the simulation engine, *aiodtnsim* was used. This open-source, event-based simulation framework has been developed by the author and had already been applied successfully in several analyses, including one commercial project. The simulator employs the *asyncio* Python module for cooperative multitasking. This technological basis enables a developer to attach DTN protocol implementations for realistic real-time simulations transparently. The state of each simulation run is monitored by recording events such as *bundle injected* or *bundle delivered* and providing a summary in a JSON file.

Reasoning for a Custom Simulation Framework

The authors of [Kup+19] recommend the use of a state-of-the-art DTN simulation engine rather than a custom implementation. In early stages of development, it was considered to use the extensive simulation toolchain presented in [Fel+17] that is based on the ONE simulator [KOK09]. However, though having a set of advantages, including a large number of readily-available routing algorithms, this initial plan was discarded because of several limitations:

- The ONE is designed for terrestrial networks and two-dimensional movement. A custom integration of satellite orbits was added as a movement model (see [Fel+17]), but requires the projection of orbital trajectories.
- No sub-contact disruptions based on realistic data could be modeled. The existing satellite extension to the ONE determines all contacts purely based on range. It would not make sense to evaluate a volume prediction technique with contact intervals and volumes precisely derived from the range.
- No delay support is present. In the ONE simulator, messages are received instantly after the transmission process concludes, and nodes access each other's buffers without any delay. Though in the evaluated Ring Road use case delays are generally small, it was a goal to be able to simulate scenarios with extended delays as well.
- The ONE simulator is time-based. In a fixed interval, it executes an *update* function for every object, e.g., nodes, their interfaces, routers, and connections. This creates a comparably large overhead.
- All available implementations of CGR for the ONE that are known to the author at the time of writing ([Fel+17], [Ber+17]) were based on older versions, i.e., not on the algorithm standardized in [CCSDS19]. Thus, significant modifications would have been required to the existing source code.

Thus, it was decided to develop own tooling for DTN simulations. A core property of aiodtnsim is that the engine is entirely deterministic, whereas this determinism is not based upon a fixed random seed as, e.g., in the ONE. All contacts as well as bundle injection events are defined via input files.

Link Representation

For simulating realistic impairments during a contact, bundles are transmitted via a simulated link-layer protocol (see also figure 2.1). It is assumed that the corresponding convergence layer adapter splits bundles into packets with maximum size $l_{packet,CL}$ and automatically re-transmits them in case of bit errors in individual packets. For that purpose, in addition to the packet size, samples of the time-varying residual bit error rate can be provided to the simulation engine. The implementation applies a reduced transmission rate over time to exhibit *ideal ARQ-Selective Repeat* performance, as described in [LS+19] (p. 7, equation 16).

Implemented Routing Approaches

The CGR algorithm specified in the CCSDS blue book [CCSDS19] (SABR) was implemented in aiodtnsim. Two different options for determining a list of shortest paths (see [CCSDS19], 3.2.6.10) were implemented in the process, the use of Yen's algorithm plus the exclusion of contacts by deriving a *limiting contact* (see [Fra+17]). A comparison of both can be found in section B.3 of the appendices.

In addition to the CGR algorithm, an alternative route search procedure was implemented. This implementation is based on the one applied in the evaluation of [FW17a] and [WF18] (see linked materials) and uses network nodes as graph vertices instead of contacts[1]. An independent implementation of the same idea has been published in the meantime, which is called *SP-CGR*. [DeJ19] The reader is referred to [DeJ19] for an overview of the general idea. It was discovered that significant performance gains are possible in the Python implementation by using this alternative route search procedure. As tests showed that routes match those computed by state-of-the-art alternative implementations (see section B.2), it was used as the primary route search procedure. Simplified Python code for the core route search algorithm is provided in appendix A.

The CGR routines were further extended with the probabilistic concepts described in section 6.3. This variant of the implementation is called *CGR-Prob*. In addition to CGR and *CGR-Prob*, the Epidemic Routing [VB00] and Spray and Wait [SPR05] routing algorithms were implemented.

7.3 Validation of Implemented Approaches

An extensive unit test suite covers the core routines handling contact plans and time-varying graphs to validate the correct functionality of the implementation. This test suite is executed by a continuous integration system (GitLab CI), ensuring timely testing of every change. For the aiodtnsim simulator, continuous integration is employed as well, invoking static analysis tools and an integration test.

[1] It should be noted that the calculated routes still consist of contacts.

The link representation in aiodtnsim was validated using a Monte-Carlo simulation. For that purpose, a second implementation was realized, which randomly introduces bit errors and performs selective re-transmission of all faulty packets.

The performance of the Epidemic Routing and Spray and Wait implementations was compared to results generated using the ONE simulator. For that purpose, the original scenario 2 (without ISL) from [FW17b] was converted to a factual contact plan that can be provided to aiodtnsim. The results of this validation step are described in further detail in section B.1.

In the course, a set of issues in the ONE concerning a realistic Ring Road simulation were discovered[2]:

- Connections in ONE do not separate uplink and downlink. As long as a transfer in one direction occurs, the reverse link cannot be utilized.
- Concurrent transfers are not possible, i.e., a node can only transfer to a single neighbor at the same time. By that, no multiple access scheme that allows concurrent transfers can be simulated.
- Nodes in ONE access the buffers of their neighbors without delay, e.g., to check whether a given neighbor already has a specific message. This behavior is expected by routing algorithms; for example, in Epidemic Routing it replaces the exchange of the *summary vectors*.

Following the validation of the opportunistic routing algorithms as well as the simulator core, the CGR implementation was validated in three steps:

1. The routes computed for nine destination nodes in a randomly generated ten-node Ring Road scenario were compared to the routes computed by a preview version (*preview3*) of ION 4.0.0 provided by the JPL.
2. The selected alternative routes were compared to the *pycgr* Python implementation of CGR[3] in example scenarios, showing that the alternative routes computed using Yen's algorithm as well as using a limiting contact selected by the earliest end time match.
3. A test comparing the data delivery performance to the *pydtnsim*[4] implementation of CGR was conducted. This implementation has been derived from the ION 3.5 pseudocode documented in [Fra+17]. It was found that both implementations exhibit comparable performance. In most cases, minor improvements were observed for the SABR-based implementation in aiodtnsim when compared to pydtnsim. The comparison results can be found in section B.2 of the appendices.

[2] Code changes addressing the first two issues have been proposed to the developers of the ONE simulation environment.
[3] Available via: `https://bitbucket.org/juanfraire/pycgr/src/master/` (accessed: 7 May 2020)
[4] pydtnsim is open-source software available via: `https://github.com/ducktec/pydtnsim` (accessed: 7 May 2020)

7.4 Simulation Scenarios

The used test scenarios were defined based on satellite observations from the SatNOGS network (see [Whi+18]), including observations from a satellite ground station created by the author. Subsection 2.2.1 provides an example observation obtained with the latter, together with relevant technical parameters.

7.4.1 Satellite Observations

The selected ground stations are listed in table 7.1 with their properties according to the information recorded in the SatNOGS network. A graphical overview of the involved ground stations and satellites is given in figure 7.3.

ID	Station	Country	Lat. (°)	Lon. (°)	Alt. (m)	Observations
2	KB9JHU	USA	39.236	-86.305	280	59
77	N5CNB-VHF	USA	29.855	-96.535	6	99
147	F6KKR	France	48.635	1.829	200	235
579	Um Alaish 4	Kuwait	29.104	48.125	20	44
1045	DL4FW	Germany	51.175	14.174	224	41

Table 7.1: List of selected ground stations

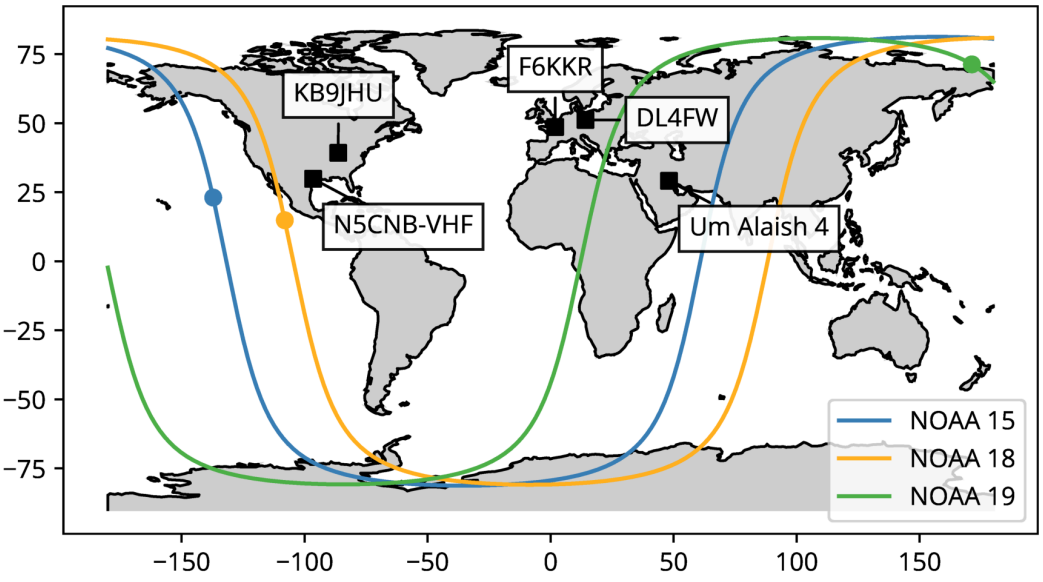

Figure 7.3: Overview of selected ground stations and satellites

The ground stations were chosen based on three criteria. Firstly, the number of observations recorded for the NOAA weather satellites in the selected period was an important factor. Only stations with more than 40 observations over three weeks were considered.

Secondly, the SatNOGS client allows for configuring a minimum elevation angle to start and end observations. This parameter restricts the collected amount of data. Therefore, only stations setting it such that the complete interval of a detectable signal is recorded were selected. The concrete value to achieve this depends on the hardware, varying approximately between 0 ° and 15 °.

Thirdly, a goal was to choose ground stations with different antenna and receiver characteristics. Ground station 2 (*KB9JHU*) has a steerable antenna, while the others use fixed antennas. The geographical distribution results from most SatNOGS ground stations being located either in Europe or in the United States of America[5]. Unfortunately, all examined candidates in the southern hemisphere either provided too few observations or had outages in the considered period.

For stations 2 to 579, observations created during the three weeks between 7 January 2020 00:00:00 UTC and 28 January 2020 23:59:59 UTC were downloaded from the SatNOGS network. Additionally, observations recorded by the ground station of the author (ID 1045, *DL4FW*) in the period between 22 October 2019 00:00:00 UTC and 30 October 2019 23:59:59 UTC were added to the data set.

The observation data were analyzed to derive the residual error rate present in the decoded signal. For the link-layer coding, a Reed-Solomon (RS) code with parameters (255,223) and a correction capability of 16 bits was assumed. These parameters are equal to the E=16 RS code recommended by the CCSDS in [CCSDS17] (section 4). Samples for the time-dependent residual error rate were derived with a period of 10 seconds. The data rates of a state-of-the-art S-band transceiver as installed in the CubeSat *OPS-SAT* of the European Space Agency [EM14] were assumed, i.e., 256 kbit/s on the uplink and 1 Mbit/s on the downlink.

7.4.2 Derived Scenarios

Based on these data, more extensive scenarios were defined, listed in table 7.2.

Scenario	Sat.	GS	Duration	Contacts	Description
S1	9	10	10 days	10348	Walker-delta constellation, 3 planes at 98 ° inclination
S1-p3b	9	10	10 days	7964	S1 with LEO-11, LEO-12, and LEO-13 breaking down after 72, 73, and 74 hours
S1-p5i	9	10	10 days	8276	S1 with GS-01 to GS-05 switching between *active* and *inactive* every 12 hours
S2	6	15	10 days	11428	3 satellites at 98 ° and 3 satellites at 45 ° inclination

Table 7.2: Overview of test scenarios

[5] See map on: `https://network.satnogs.org/` (accessed: 7 May 2020)

Scenario S1 (displayed in figure 7.4) consists of a Walker-delta satellite constellation with three planes at 98 ° inclination and three satellites per plane. The Walker-delta formation is common for the deployment of LEO constellations and has been analyzed in the DTN context, e.g., in [Fra+17].

For evaluating the probabilistic concepts, two scenarios were derived from S1, one (S1-p3b) with satellites breaking down permanently after a fixed time span (e.g., due to the collision with debris along their orbit), and one (S1-p5i) with five ground stations being only active, i.e., transmitting and receiving bundles, every 12 hours (e.g., because of a solar power source).

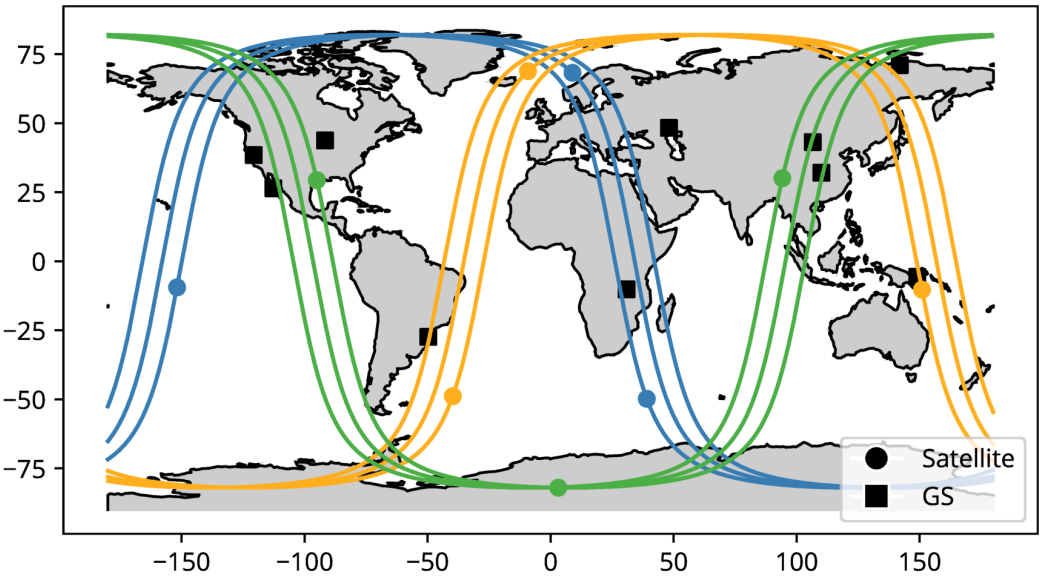

Figure 7.4: Overview of scenario S1

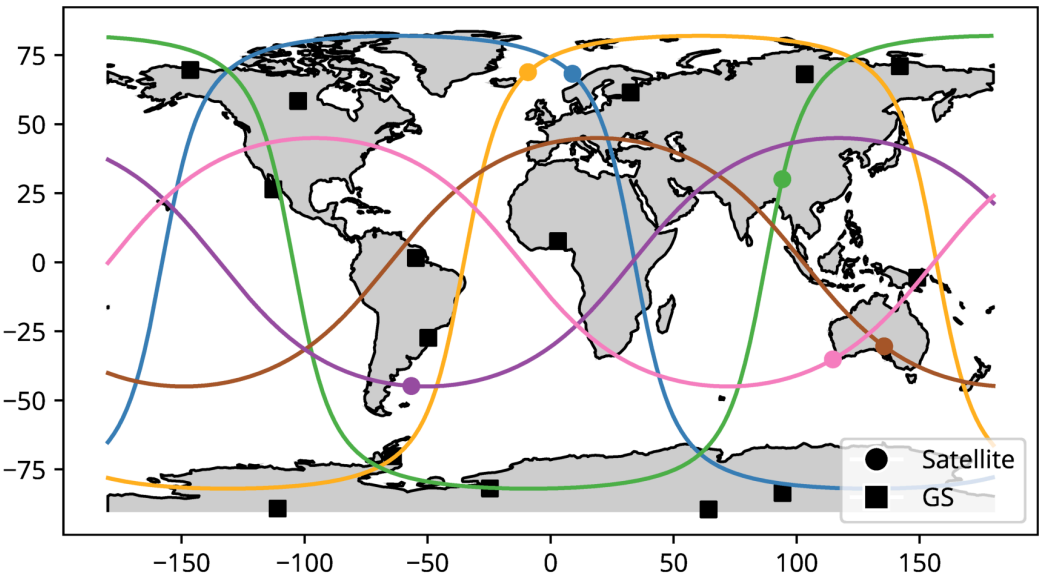

Figure 7.5: Overview of scenario S2

Scenario S2 (shown in figure 7.5) assumes six independent satellites, of which half have an inclination of 98 ° and the others have an inclination of 45 °. This scenario represents an opportunistic deployment strategy of low-cost satellites: For many small satellite missions, the final orbit depends vastly on the orbital parameters of the primary mission, as they are secondary payloads (see, e.g., [CSH15]).

In both scenarios, the ground stations were distributed randomly over the Earth's landmasses.

7.4.3 Prediction Parameters

The prediction framework introduced in chapter 5 requires several parameters to be set in an implementation. These parameters and their configured values are listed in table 7.3. The values were selected based on initial tests of individual parts of the implementation.

Symbol	Section	Description	Value
T_{pred}	5.6	prediction generation interval	12 h
θ_{min}	5.8.1	minimum elevation threshold	0 °
v_s	5.6.2	signal propagation speed	c
δ_0	5.6.2	additional delay margin	0
$n_{obs,P}$	5.8.2	confidence: max. observation count	10
$n_{obs,V}$	5.8.3	volume: max. observation count	10
$l_{packet,CL}$	5.8.3	volume: convergence-layer packet size	1500 B
$T_{sample,V}$	5.8.3	volume: sample duration	30 s
$r_{V,s,min,good}$	5.8.3	volume: min. ratio for good state	0.5
$r_{V,s,min,bad}$	5.8.3	volume (three-state): min. ratio for bad state	0.25
$\theta_{good,min}$	5.8.3	volume (three-state): min. elev. for good state	20 °
n_{az}	5.8.3	volume (three-state): azimuth bins	8 (45 °)

Table 7.3: Prediction parameters

7.5 Evaluation of Contact Prediction Accuracy

The accuracy of contact predictions over time was assessed using the first set of test cases. In these initial evaluation steps, no routing simulation was applied; only the accuracy of the predictions generated by each node for its neighbors was analyzed. All plots provided in this section show the average values and standard deviations of samples collected in discrete time intervals with a duration of 12 hours.

7.5.1 Confidence Prediction

/E1/ Confidence prediction accuracy

Goal: Verify accuracy and convergency of confidence predictions over time.

Requirements concerned: /R2/

Approaches compared: Ratio-based estimation with window size of 10 and exponential weighted moving average with $w_c = 0.1$ and $w_c = 0.3$ (see 5.8.2)

Scenarios compared: S1-p3b (total node failure), S1-p5i (intermittent availability)

Confidence prediction was evaluated for the two probabilistic variants of S1 (S1-p3b and S1-p5i). In figure 7.6, the contact confidence values predicted by ground stations for one of the failing satellites in S1-p3b are plotted over time.

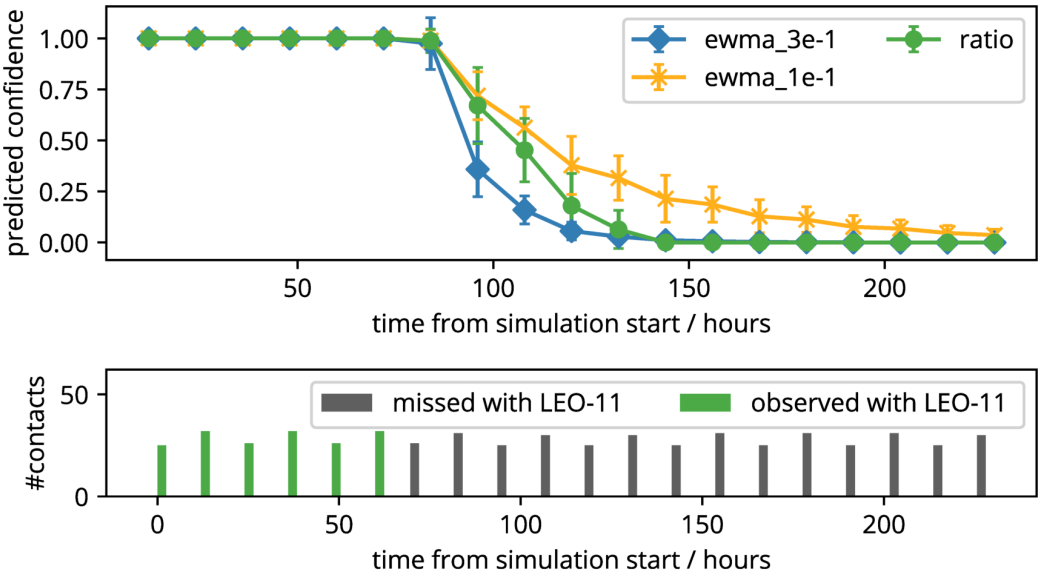

Figure 7.6: /E1/: Confidence prediction accuracy over time, scenario S1-p3b

By inspecting the figure, it can be seen that the ratio-based technique converges toward zero after the breakdown of node LEO-11 within approximately three days. The exponential moving average with weight 0.3 converges faster, whereas the one with weight 0.1 converges slower than the ratio-based estimation.

In figure 7.7, the same plots are shown for the confidence predicted by satellites for one of the intermittently-available ground stations in scenario S1-p5i. The exponential averaging technique with weight 0.3 shows faster reactions to variations in the number of failed contacts. This way, the observed oscillating behavior as well as the large standard deviation can be explained.

Overall, it can be seen that probabilistic changes in the network are reflected in the predicted confidence metrics. The ratio-based technique shows more stable results across the analyzed set of scenarios. In summary, confidence prediction can provide a basis for supporting probabilistic contacts (requirement **/R2/**).

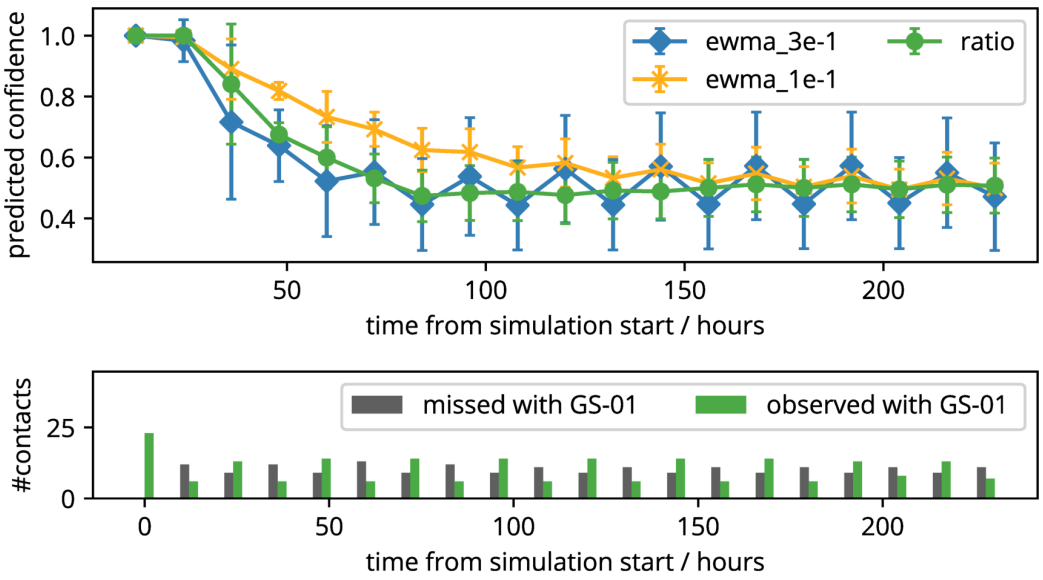

Figure 7.7: /E1/: Confidence prediction accuracy over time, scenario S1-p5i

7.5.2 Volume Prediction

/E2/ Volume prediction accuracy

Goal: Verify accuracy and convergency of volume predictions over time.

Requirements concerned: /R3/, /R4/, /R5/

Approaches compared: Minimum elevation of 10 ° (*minelev-10deg*), minimum elevation of 20 ° (*minelev-20deg*), two-state (*twostate*), and three-state (*threestate*) prediction techniques (see 5.8.3)

Node types compared: Nodes with high minimum elevation (based on ground station 579), nodes with low minimum elevation (based on ground station 2)

Figure 7.8 shows a plot of the relative accuracy of the predicted contact volume. This parameter represents the difference between the predicted and measured contact volume, normalized by the measured volume. In this first plot, the accuracy is depicted for a ground station with a high minimum elevation (derived from observations of station 579). Predictions of both the two-state and the three-state techniques converge toward the simulated value within 24 to 48 hours, whereas the minimum elevation-based predictions are offset by a huge margin. The latter can be explained by the fixed directional antenna used by the ground station.

Figure 7.9 depicts the same plot for a ground station with a steerable antenna (derived from observations of station 2). In the latter case, the three-state technique yields slightly improved predictions on average, which is assumed to be caused by considering the azimuth-dependency. The average difference compared to the predictions of the two-state technique is, however, still very small. For this specific station, assuming a minimum elevation of 10 ° for predictions also leads to low average differences.

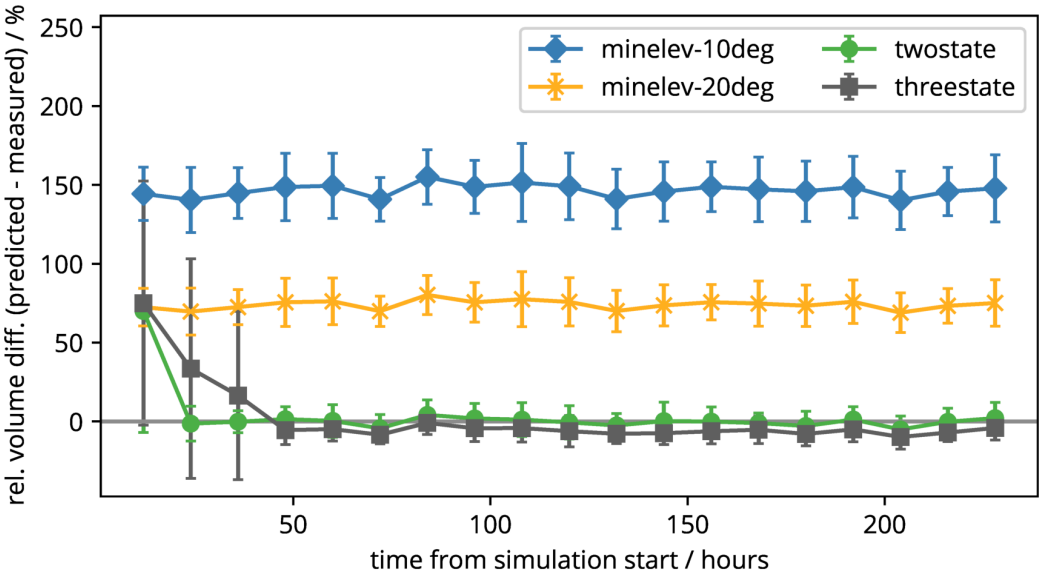

Figure 7.8: /E2/: Relative volume prediction accuracy over time for ground stations with high minimum elevation, scenario S1 (normalized by measured volume)

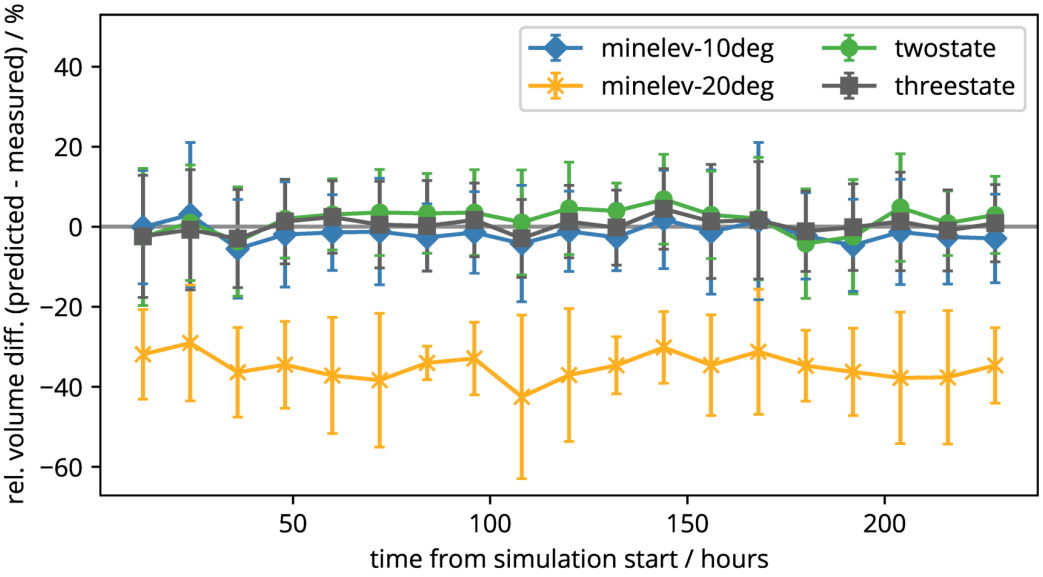

Figure 7.9: /E2/: Relative volume prediction accuracy over time for ground stations with low minimum elevation, scenario S1 (normalized by measured volume)

In summary, the volume prediction techniques allow for an automatic adaptation to different link characteristics (requirement **/R4/**). This property enables the dynamic integration of new nodes with heterogeneous hardware (requirement **/R3/**). The deterministic mobility patterns are considered by all applied estimation and prediction methods (requirement **/R5/**).

7.6 Evaluation of Metric Distribution

/E3/ Metric distribution analysis

Goal: Analyze distribution delay as well as use of network resources by metric distribution.

Requirements concerned: /R6/, /R8/

In the third test, metric distribution was applied to evaluate the associated delays and the resulting overhead. The expiration time of metric bundles was set to 12 hours, and their size was derived by serializing node metrics in Concise Binary Object Representation (CBOR) [RFC7049] format. The latter is also used as the primary serialization format in the Bundle Protocol version 7 draft [BFB20]. The test was performed for different volume prediction approaches: While using a minimum elevation does not need metric distribution, the two-state and three-state techniques apply metric tuples of different size. All confidence prediction approaches distribute information of the same size.

It was found that the delivery delays of metric bundles varied only insignificantly in the analyzed cases with their average value ranging between 5669.4 s and 5754.3 s and a standard deviation between 3147.5 s and 3464.1 s. In figure 7.10, the relative contact time utilization by metric bundles is depicted, which does not exceed 1 % for 90 % of contacts. The buffer utilization was measured as well and is depicted in section C.1 of the appendices. The highest maximum buffer utilization was observed for the three-state technique in S2, with a value of 461.47 KiB. Consequently, the test results show that nodes can apply transitive information timely after their creation (requirement **/R6/**) and metric distribution causes only minor overhead (requirement **/R8/**).

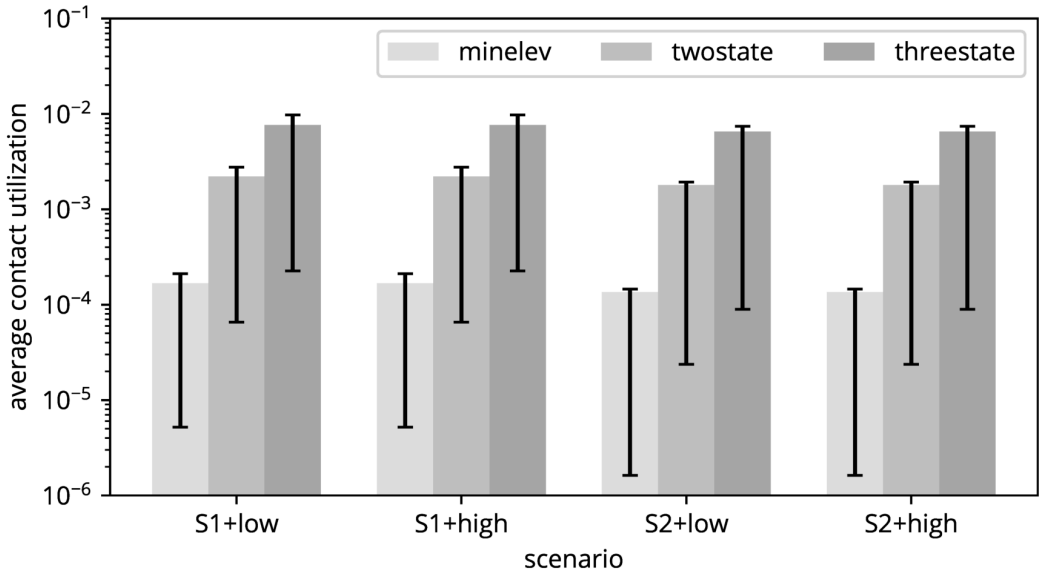

Figure 7.10: /E3/: Average contact utilization by metric bundles (occupied ratio of contact time; logarithmic scale, with 80 % intervals)

7.7 Evaluation of Routing Performance

The tests presented in this section compare the data delivery performance of prediction-based routing with state-of-the-art opportunistic techniques. For that comparison, the Epidemic Routing and Spray and Wait algorithms are leveraged, which have been shown applicable to Ring Road networks in [FW17b]. All plots in this section depict average values as well as the 95 % intervals.

Before discussing specific test setups and results, general expectations of the results can be summarized as follows. As identified in [FW17a], huge variations in the end-to-end delays have to be expected due to the sparse topology of a Ring Road network. Thus, bundle delivery delays from a few seconds up to several hours can be observed regularly.

Moreover, some tests (**/E4/** and **/E6/**) use the CGR algorithm provided with the observed contact volumes for comparison. It can be expected that this allows for increased performance compared to the prediction approaches. However, in a realistic Ring Road network, this advance knowledge cannot be expected.

7.7.1 Traffic and Buffer Configurations

For evaluating the routing performance in a specific scenario, bundle transmissions have to be simulated. The bundle generation has to assume a particular traffic pattern. Additionally, assumptions regarding the available buffer space of nodes and the lifetime of bundles have to be made.

It is expected that, due to its overall capacity and the inherent end-to-end delays, a Ring Road network is mostly suitable for the exchange of messages, machine-to-machine communication, and the transmission of small application payload data between ground stations. Two traffic load configurations (*low* and *high*) were investigated that reflect such an application. In the *high* configuration, a bundle size of 2 MiB (2^{24} bit) was assumed. In the *low* configuration, a bundle size of 200 KiB ($200 \cdot 2^{13}$ bit) was used.

Bundles are generated by random ground stations with another (different) ground station being set as the destination node, in an equally-distributed random interval. The arithmetic mean of the generation interval is computed for each scenario based on the minimum ratio of contact time occupied when assuming all bundles are transmitted via two hops. In the *high* configuration, this ratio was set to 50 %, whereas, in the *low* configuration, a value of 2 % was assumed.

The standard deviation of bundle generation times was set to one fourth of the configured average period. After 80 % of the scenario duration has elapsed, the bundle generation stops to provide sufficient time for delivering remaining bundles.

The bundle lifetime was set to 10 days (i.e., the scenario duration) to prevent it from influencing the measured delivery probability and delay figures.

Concerning node buffers, a size of 2 GiB (2^{34} bit) was assumed. Given a maximum data rate of 1 Mbit/s for the downlink, 35.79 minutes of continuous data reception is needed to exhaust a buffer of that size. The average contact volumes for the two test scenarios were calculated for comparison: Scenarios S1 and S2 have average contact volumes of 9.57 MiB and 10.59 MiB, respectively.

7.7.2 Further Parameters

The distribution procedures as well as the applied routing algorithms require the definition of additional parameters, listed in table 7.4. The individual values were chosen either based on recommendations by the respective authors or according to preliminary studies of the use case. For example, the number of replicas for Spray and Wait was set to 6 as tests documented in [FW17b] had shown promising results in Ring Road networks of similar size.

Description	Defined in	Value
Contact plan update period (T_{update})	section 6.1	12 h
Distribution buffering period (T_{dist})	subsection 6.2.1	0
Max. look-ahead duration for graph updates ($T_{lookahead,max}$)	subsection 6.3.2	10 d
Min. contact plan update period ($T_{update,min}$)	subsection 6.3.2	12 h
Max. contact plan update period ($T_{update,max}$)	subsection 6.3.2	12 h
CGR-Prob: max. number of replicas ($n_{routes,max}$)	subsection 6.3.3	6
Epidemic, max. hop count	[VB00]	6
Epidemic, size of summary vector entry	[VB00]	32 bit
Spray and Wait, type	[SPR05]	binary
Spray and Wait, number of copies	[SPR05]	6

Table 7.4: Routing parameters

For every test case and algorithm, 50 simulation runs were performed. Five concurrent incoming and outgoing bundle transmissions were allowed in routing simulations.

The approach to determine alternative routes in CGR was set to use a limiting contact based on the earliest end time and the route confidence, as documented in subsection 6.3.3. A comparison with Yen's algorithm for a small scenario can be found in section B.3 of the appendices. The tests showed that the implementation leveraging Yen's algorithm was unsuitable for larger scenarios due to a massive increase in simulation run-time.

7.7.3 Soft Time Frames

/E4/ Soft time frame applicability

Goal: Evaluate the impact of soft time frames on routing performance.

Requirements concerned: /R1/, /R3/

Approaches compared: CGR with accurate time frames (*CGR-factual*), CGR with soft time frames (*CGR-soft*), Epidemic Routing, Spray and Wait

In this test case, the basic assumption that soft time frames do not negatively impact data delivery performance was investigated. For that, CGR with accurate contact intervals was compared to CGR with soft time frames, as defined in subsection 5.8.1. In both cases, accurate contact volumes were provided. In addition to the CGR variants, in this test, the Epidemic Routing and Spray and Wait algorithms were simulated for comparison.

The resulting delivery probabilities are depicted in figure 7.11, which indicates no significant differences between *CGR-factual* and *CGR-soft*.

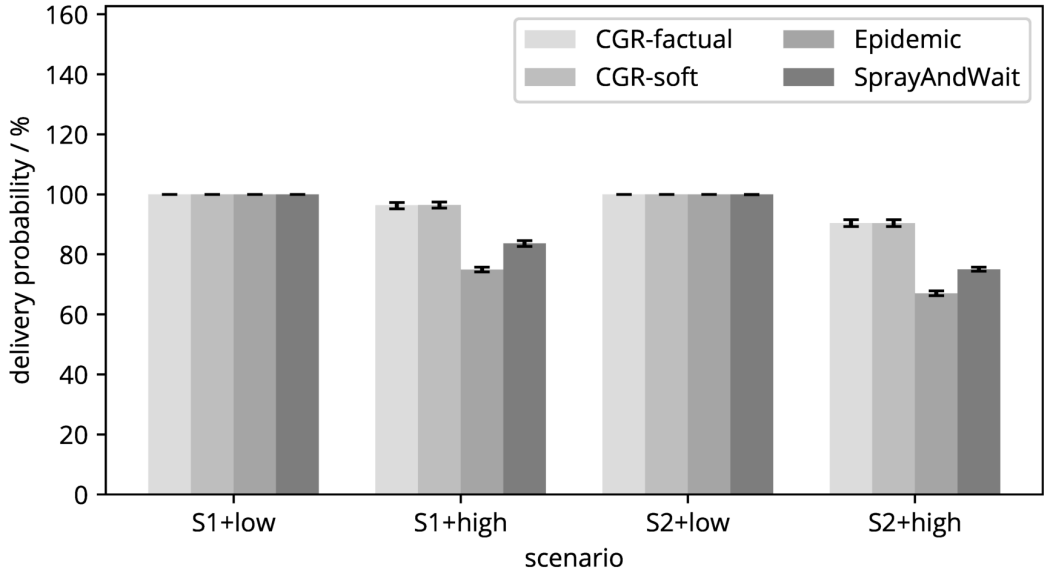

Figure 7.11: /E4/: Bundle delivery probability with soft time frames

Concerning end-to-end delays, buffer utilization, and the use of contact time, no significant differences could be observed either. A small increase in the number of bundles re-scheduled was reported under *low* load (depicted in section C.2.1 of the appendices), for which late arrival of bundles (after the end of the physical contact but before the end of the soft time frame) is assumed a cause. Overall, only minor performance differences were found in the analyzed scenarios.

Therefore, soft time frames can be applied without major drawbacks, making it possible to support networks with dynamically-integrated nodes (requirement **/R3/**). As CGR is used for routing, native support of scheduled contacts (requirement **/R1/**) is present as well.

7.7.4 Confidence Prediction

/E5/ Routing performance with confidence prediction

Goal: Evaluate the performance characteristics of the probabilistic CGR extension.

Requirements concerned: /R2/, /R7/, /R9/

Approaches compared: CGR, CGR with probabilistic extensions (*CGR-Prob-RC-TC*), Epidemic Routing, Spray and Wait

In this test case, the performance of different routing approaches in networks with unreliable nodes (S1-p5i and S1-p3b, as defined in section 7.4) was analyzed. Concerning the traffic pattern, in both scenarios, unreliable nodes were excluded from being bundle sources or sinks.

The probabilistic CGR extension was evaluated with two different settings, a minimum route confidence (RC) of 30 % and a minimum total delivery confidence (TC) of 30 % (*CGR-Prob-30-30*), as well as a minimum route confidence of 10 % and a minimum total delivery confidence of 80 % (*CGR-Prob-10-80*). Whereas the former configuration only filters routes with a confidence lower than 30 %, the latter configuration replicates bundles to achieve the targeted total delivery confidence of 80 %. For all CGR-based approaches, contact volumes based on a minimum elevation of 10 ° were assumed. As confidence prediction strategy, the ratio-based technique with a window size $n_{obs,P}$ of 10 contacts (as evaluated in subsection 7.5.1) was used.

First, the *low* traffic setting was simulated. It was found that all routers achieved a delivery probability of 100 % in that case. In figure 7.12, the resulting delivery delays are shown. As expected, if no probabilistic contacts are introduced, the performance of *CGR-Prob* is equivalent to that of unmodified CGR.

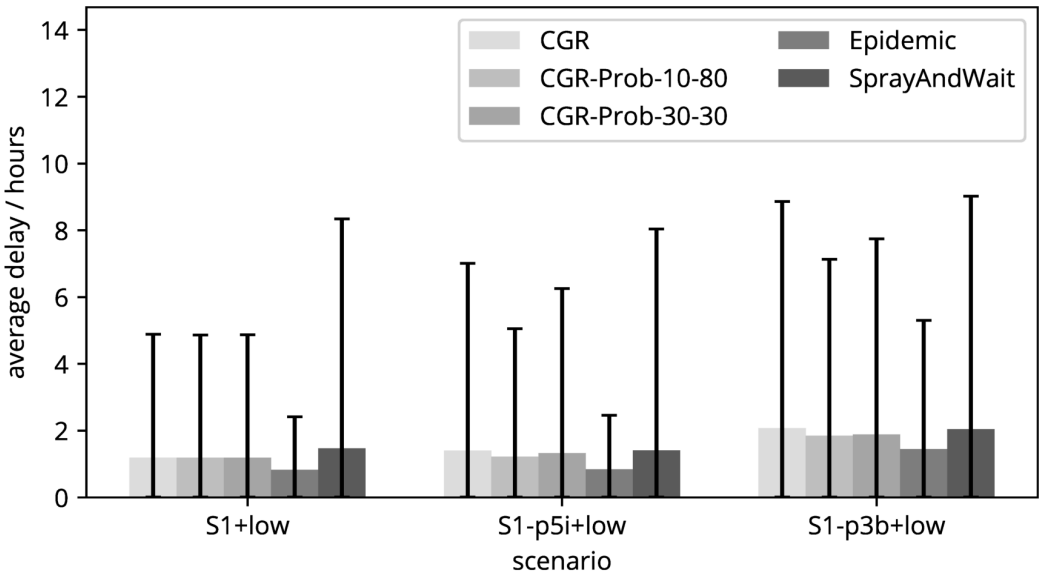

Figure 7.12: /E5/: Average bundle delivery delay with confidence prediction, low traffic load

Concerning average delays, the CGR-based approaches perform comparable to Spray and Wait in the probabilistic scenarios. Filtering unreliable routes leads to slight decreases in the 97.5th-percentile delays, and the use of replication in CGR reduces delays further. However, the lowest delays can be achieved by Epidemic Routing. In that case, an increased number of bundle transmissions could be observed (see figure C.6 in the appendices) for which, under *low* load, sufficient network resources (buffer space and contact time) are still available.

Taking into account bundle re-scheduling, i.e., the number of additional invocations of the CGR algorithm for a bundle[6], probabilistic route exclusion (*CGR-Prob-30-30*) shows advantages over the unmodified CGR router in *low*-traffic scenarios. A plot is provided in figure 7.13. The shown parameter represents the number of cases in which bundles were scheduled more than once on a given node, normalized by the sum of nodes via which the bundles were transmitted.

It can further be seen that using replication (*CGR-Prob-10-80*) significantly increases the number of re-scheduled bundles in scenario S1-p5i, which is explained by the number of additional bundles that are introduced. In scenario S1-p3b, the filtering of low-confidence routes that is also performed by *CGR-Prob-10-80* has a more significant effect on the number of re-scheduled bundles.

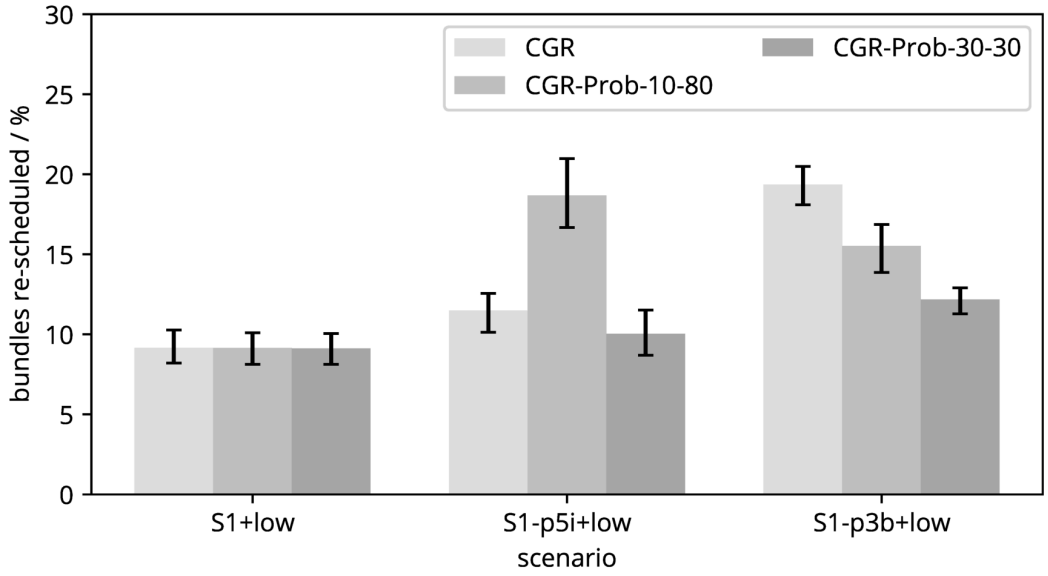

Figure 7.13: /E5/: Ratio of bundles re-scheduled with confidence prediction, low traffic load

The *high* load configuration was analyzed in the same manner. The bundle delivery probabilities are shown in figure 7.14. By filtering routes with low confidence (*CGR-Prob-30-30*), minor increases in the average delivery probability are possible in the S1-p3b scenario. All routers that replicate bundles exhibit a lower delivery probability than unmodified CGR. This observation can be explained by the additional load, leading to bundle dropping.

[6] Re-scheduling is performed for all bundles that were not sent during the contact for which their transmission was initially planned.

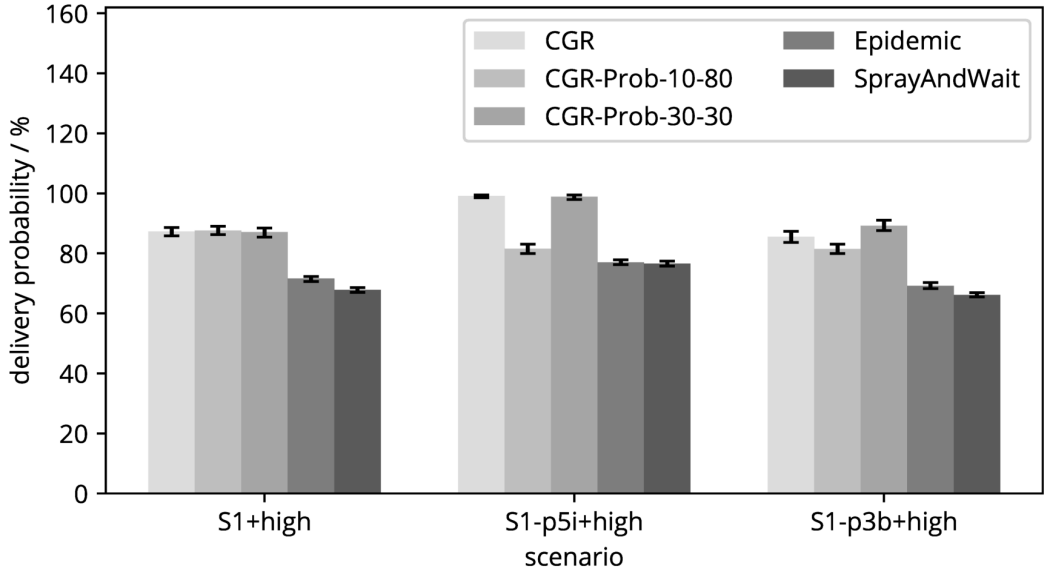

Figure 7.14: /E5/: Bundle delivery probability with confidence prediction, high traffic load

It further appears that, overall, a higher delivery probability can be achieved using CGR-based routing in the scenario with intermittent availability (S1-p5i) compared to the non-intermittent scenario (S1). This aspect, however, can be explained by the increased number of bundles injected in S1, which has a higher total contact volume (see subsection 7.7.1 for a description of the traffic parameters).

Additional plots, including the average end-to-end delay and the ratio of bundles re-scheduled under *high* load, can be found in section C.2.2 of the appendices. Based on the test results, it is concluded that in a network with reliable hop-by-hop links, e.g., due to the use of an ARQ scheme, replication in CGR does not provide advantages in most cases. Minor delay reductions come at the cost of much larger increases in traffic load. Similar issues with Opportunistic CGR in networks with random mobility were already discussed by Net and Burleigh in [NB18]. The results of this test case show that such issues may even be present with an extended level of probabilistic contact information.

Nevertheless, as indicated in [WF19], if links are unreliable and bundle loss is possible on a hop-by-hop basis (e.g., due to signal propagation delays preventing the use of reliable ARQ), replication can drastically increase the bundle delivery probability. This way, a CGR-based probabilistic router can outperform opportunistic approaches. In the evaluated Ring Road scenarios, another feature of *CGR-Prob* has shown advantages: The exclusion of routes with a low confidence value allows for reducing bundle re-scheduling significantly, especially under low network load.

Regarding the requirements defined in section 3.5, support of probabilistic contact occurrence (requirement **/R2/**) is present in all of the analyzed routing approaches. A low replication overhead (requirement **/R9/**) is achieved as replication can be fully disabled, while allowing for improved performance compared to opportunistic routing in many cases. Due to a reduction in the number of bundle transmissions, it is possible to reduce computational demands (requirement **/R7/**) as well.

7.7.5 Volume Prediction

/E6/ Routing performance with volume prediction

Goal: Evaluate the performance characteristics of CGR using different volume prediction approaches.

Requirements concerned: /R3/, /R4/, /R5/, /R7/

Approaches compared: CGR with measured volume (*CGR-factual*), CGR with volume based on minimum elevation (*CGR-minelev*; with $\theta_{min} = 5\,°$, $10\,°$, or $20\,°$), CGR with two-state (*CGR-twostate*), and three-state (*CGR-threestate*) prediction (see subsection 5.8.3), Epidemic Routing, Spray and Wait

Figure 7.15 depicts the bundle delivery probabilities observed when simulating different volume prediction techniques. For all CGR-based routers, the probabilities are comparable and exceed those of opportunistic approaches under *high* load. When using *CGR-minelev* with a high minimum elevation threshold of 20 °, the delivery probability may be reduced moderately. This reduction can be explained by a resulting under-estimation of the volume for several contacts, leading to otherwise viable routes being discarded.

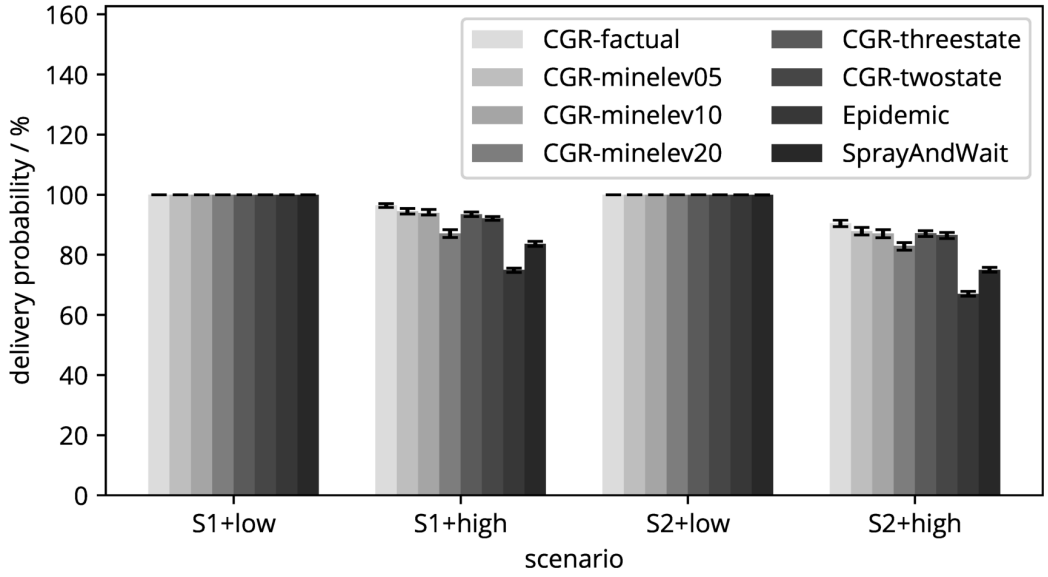

Figure 7.15: /E6/: Bundle delivery probability with volume prediction

Under *high* network load, a significant influence of volume prediction on the average end-to-end delay could be observed. A plot of the average delay under *high* load is provided in figure 7.16. The same plot for *low* load, indicating smaller differences, can be found in figure C.9 in the appendices.

Figure 7.16 shows that the average and 97.5th-percentile end-to-end delays of *CGR-twostate* and *CGR-threestate* are significantly lower than those of the *CGR-minelev* variants in S1 and comparable to them in S2. The reference approach (*CGR-factual*) always achieves the lowest delays, except under *low* load, where it is slightly outperformed by Epidemic Routing (see also figure 7.12).

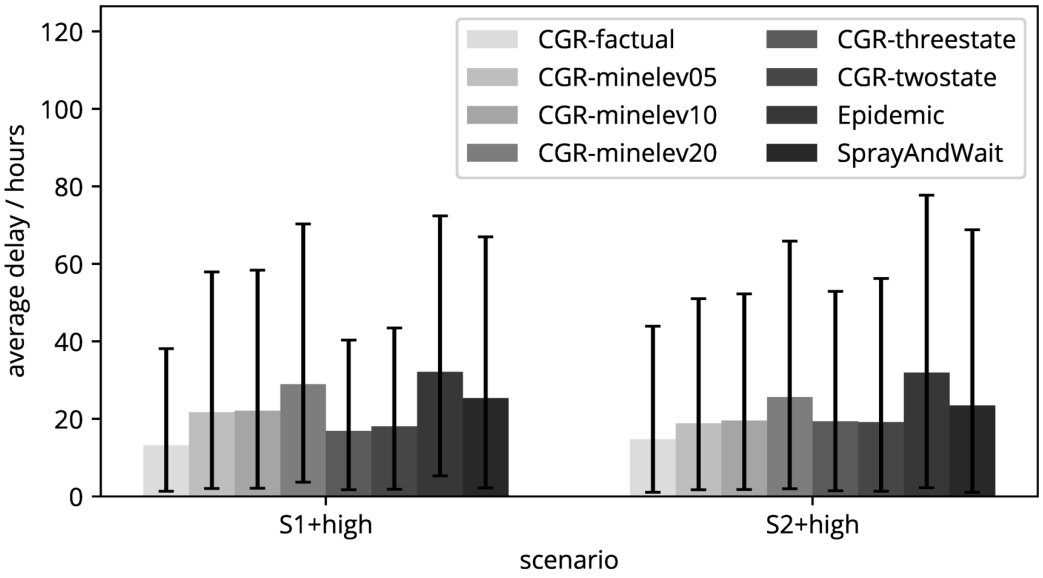

Figure 7.16: /E6/: Average bundle delivery delay with volume prediction, high traffic load

The core advantage of using more accurate volumes in CGR, however, is shown in figure 7.17, which depicts the ratio of bundles re-scheduled. (See subsection 7.7.4 for an explanation of this parameter.)

Given an improved prediction of the contact volume, significantly fewer bundles have to be re-scheduled. This implies an overall reduction in computational demands due to fewer executions of the expensive routing algorithm: Even though CGR caches available routes, the list has to be filtered to determine candidate routes for each bundle (see section 4.2.1), which has non-negligible overhead.

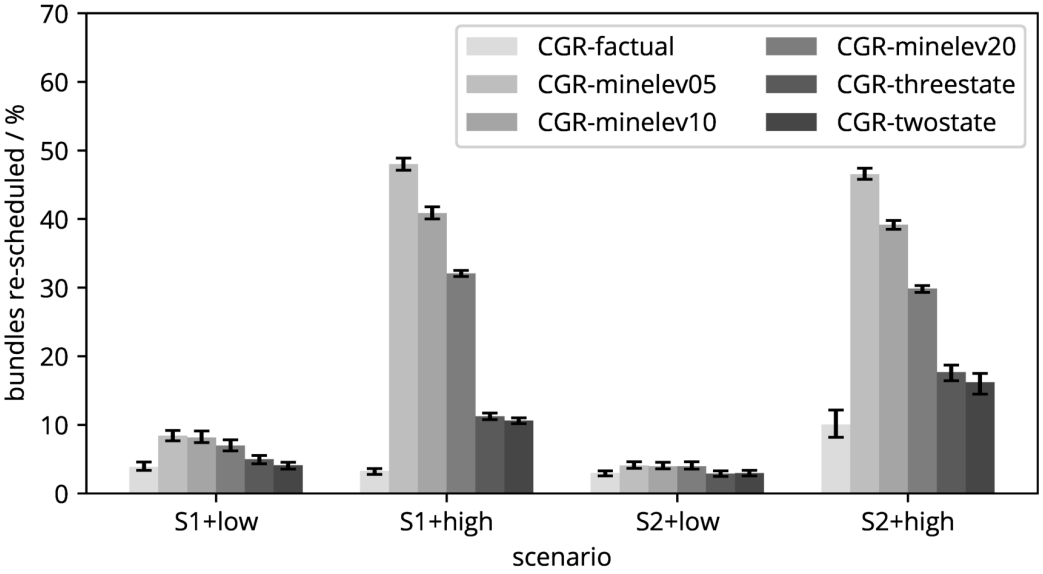

Figure 7.17: /E6/: Ratio of bundles re-scheduled with volume prediction

Figure 7.18 shows the mean buffer utilization over time in scenario S1 under *high* load. *CGR-factual* achieves the lowest overall buffer utilization. At the same time, *CGR-threestate* exhibits a mean buffer utilization comparable to *CGR-twostate*, which is slightly reduced until bundle generation stops.

More importantly, under both prediction techniques, CGR frees buffer space much faster toward the end of the simulation run than the *CGR-minelev* variants. This improvement can be explained by decreased delays resulting from reduced re-scheduling as, by their timely transmission, bundles consume buffer space for shorter periods of time. As can be expected, the buffer utilization for both opportunistic routers grows rapidly and is in total much more extensive than that of CGR-based routing.

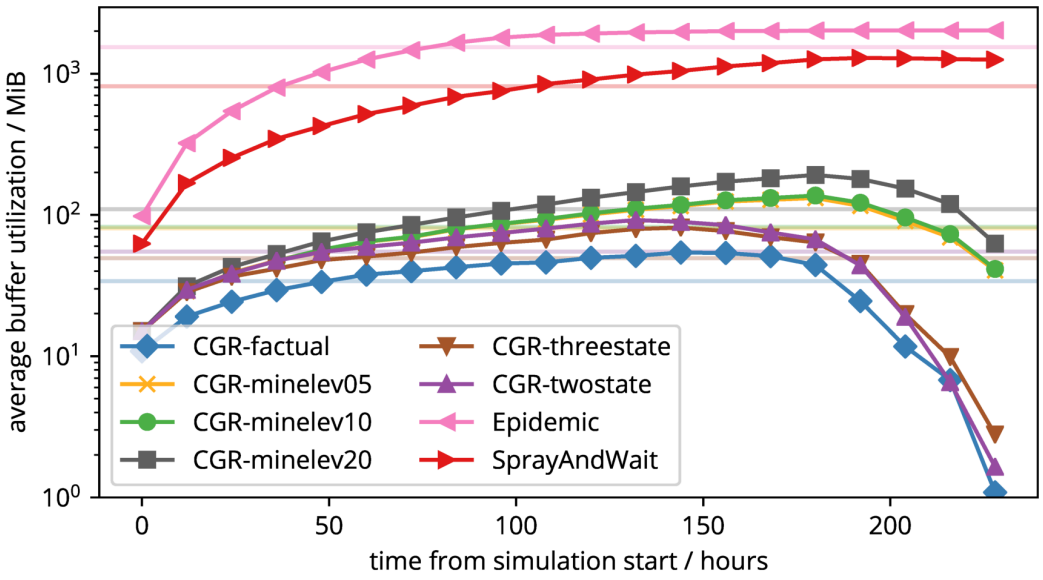

Figure 7.18: /E6/: Average buffer utilization over time with volume prediction, scenario S1, high traffic load (logarithmic scale)

Additional plots depicting results of this test case can be found in subsection C.2.3 of the appendices.

It is concluded that the proposed volume prediction techniques provide significant advantages over estimations based on an elevation threshold, especially due to reductions in re-scheduling overhead. Compared to *CGR-twostate*, however, no substantial improvements could be observed when using *CGR-threestate* in the analyzed scenarios.

Concerning the stated requirements, the dynamic integration of nodes (requirement **/R3/**) is supported by the possibility to infer contact volumes dynamically. Link characteristics (requirement **/R4/**), as well as the deterministic mobility pattern (requirement **/R5/**), are taken into account to derive the volume with greater accuracy. By that, processing demands can be reduced due to fewer executions of the routing algorithm (requirement **/R7/**).

7.8 Discussion of Computational and Power Demands

/E7/ Computational and power demands

Goal: Evaluate the impact of using prediction-based deterministic routing on computational demands and the available power budget.

Requirements concerned: /R7/

In a small communication satellite, the power consumption of two subsystems can be influenced by the applied routing algorithm. First, the load on the onboard processing unit(s) that perform, e.g., decoding, routing, and encoding of bundles, depends on the computational requirements of the routing algorithm and the count of bundles passing through the system. Second, the radio transceiver hardware consumes power depending on its utilization. The consumption is especially high during data transmission.

By the DINET experiment aboard the Deep Impact spacecraft [Wya+09], the ION software with CGR was successfully flight-tested. This spacecraft's main processor is of type Rad 750 (see [NAS05], p. 22), a PowerPC processor running at a maximum frequency of 133 MHz and consuming approximately 5 W. [Sys02] For comparison, modern CubeSats such as the OPS-SAT have much more capable processors, partially due to reduced requirements, e.g., concerning radiation safety. The OPS-SAT processing platform contains a dual-core ARM Cortex-A9 processor running at up to 800 MHz as well as an FPGA. [EM14]

However, although for CubeSat applications more capable hardware is available for some time, one central difference to the DINET validation setup cannot be neglected: The number of nodes and, consequently, the number of contacts in a Ring Road deployment will be much larger. Thus, computational demands by the CGR algorithm will be increased as well. The scalability of the CGR routing algorithm has been discussed, e.g., by Wang et al. in [Wan+16]. As the networks grow in size, route computations become more and more expensive due to the enlarged contact graphs.

One proposal to mitigate this issue is a separation of the network into distinct routing domains, also called *regions* (see, e.g., Wang et al. [Wan+16], p. 51, and Madoery et al. [Mad+18]). Such a divide-and-conquer approach fits exceptionally well to a Ring Road network, as networks connected to different ground stations can be assigned to different regions. The satellites might also be assigned to different regions based on the orbital plane, such as suggested by Madoery et al. in [Mad+18]. However, the concrete architecture and specifics of such an approach are beyond the scope of this thesis.

The design of the introduced prediction framework further allows asynchronous execution, even on different nodes. Hence, predictions could also be offloaded to a central entity in the network. Combined with that, a technique such as Spot-of-Maximum-Knowledge (SMK) routing [FWB17], can be applied. This way, it can be ensured that the available computational power aboard the satellites does not become a major bottleneck for the overall communication network.

The second factor significant for onboard power consumption is the count of transmissions that have to be performed. A higher number of bundle transmissions not only increases computational demands for decoding and encoding but, first and foremost, increases power demands by the radio transmitter.

According to [Kam+14], the power consumption of a state-of-the-art nanosatellite S-band radio transmitter, when active, is about 5 W. This matches the power consumption estimated for the OPS-SAT, as documented by Issler et al. in [Iss+14]. In the latter case, a maximum power demand of 5 to 9 W was expected, depending on the configured RF output power. However, this amount of power is only consumed when transmission takes place. Thus, a significant reduction in power demand can be achieved by reducing the count of transmissions.

The total number of transmissions that occurred during the tests performed for **/E6/** is depicted in figure 7.19.

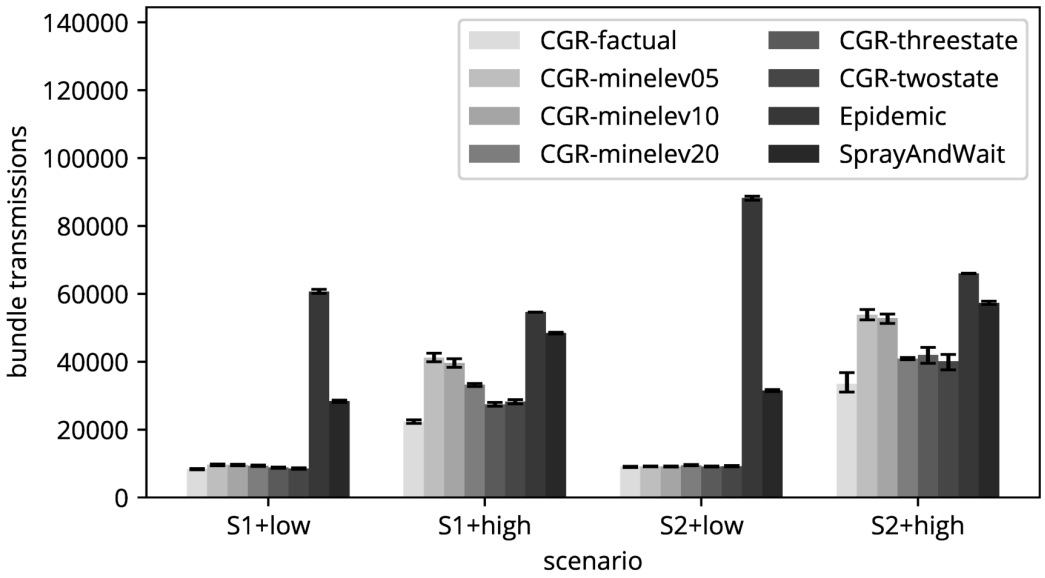

Figure 7.19: /E7/: Total number of transmissions performed in test runs with volume prediction (see also /E6/)

It can be expected that the significantly reduced number of transmissions indicated for CGR-based routing provides essential benefits compared to opportunistic routing, as transmitters can be turned off more frequently. Using the advanced contact prediction techniques introduced in subsections 7.7.4 and 7.7.5 can further decrease computational demands due to fewer re-scheduled bundles and less overall transmissions.

In summary, this discussion shows that an efficient use of the available energy (requirement **/R7/**) is possible by using prediction-based proactive routing.

7.9 Further Findings

While developing the toolchain and performing the described evaluation steps, a set of additional findings were discovered that, although not directly relevant for assessing the fulfillment of the requirements, might be of interest for future research:

1. Contact Graph Routing can operate with incomplete and imprecise topological knowledge and still provide satisfactory routing performance. In the analyzed use case, failing contacts as well as imprecise contact time frames and data rates often only lead to a significant increase in the number of re-scheduled bundles. This, albeit inducing non-negligible computational overhead, often does not drastically increase delivery delays or decrease the delivery probability. Thus, accepting minor deficiencies in the contact plans instead of trying to resolve them as fast as possible can be a viable approach to minimizing the overhead of used estimation methods as well as the distribution of information.

2. An over-estimation of the contact volume on average may be advantageous in networks where predictions are not entirely accurate. By attempting to transmit more bundles than theoretically possible (according to the predicted volume), the factual contact volumes can often be used to a greater extent and, thus, more bundles can be delivered via individual contacts.

3. The replication of bundles can quickly exhaust network resources and should be avoided if possible. If it is not possible to avoid replication altogether, it should be limited to a minimum number of copies to achieve reliable delivery.

4. In contrast to CGR-based routing, opportunistic approaches are very susceptible to reductions in buffer size and the overall buffer management (i.e., bundle dropping) strategy. The settings used in this evaluation were chosen such that sufficient buffer space was available and no negative impact on the performance of opportunistic routing algorithms could be observed in preliminary tests.

5. If nodes themselves are unreliable (i.e., drop bundles unexpectedly), a controlled replication scheme such as Spray and Wait often outperforms CGR-based routing. Hence, in these scenarios, it might be a viable option to integrate a Spray-and-Wait-like replication strategy into a scheduled routing approach.

6. It is essential to validate the whole technological stack employed for evaluation. Validations have to be performed for own implementations as well as the used test data. Furthermore, this especially includes software provided by third parties. State-of-the-art implementations, even if used by a large number of researchers, do not necessarily contain correct and realistic implementations of claimed features.

7.10 Summary

Based on the presented test results, the fulfillment of the defined requirements (see section 3.5) can be assessed. An overview of which test cases are related to which overall requirements is given in table 7.5 and described below.

Test	Sec.	/R1/	/R2/	/R3/	/R4/	/R5/	/R6/	/R7/	/R8/	/R9/
/E1/	7.5.1		•							
/E2/	7.5.2			•	•	•				
/E3/	7.6						•		•	
/E4/	7.7.3	•		•						
/E5/	7.7.4		•					•		•
/E6/	7.7.5			•	•	•		•		
/E7/	7.8							•		

Table 7.5: Mapping of requirements to test cases

1. By test **/E1/**, it was shown that the developed concept can be applied to estimate contact probabilities and changes in the reliability of nodes. This enables the prediction component to support probabilistic contacts (requirement **/R2/**).

2. In **/E2/**, the volume prediction accuracy was analyzed. By the introduced prediction techniques, higher accuracy can be achieved, and support of different communication system and link characteristics (requirement **/R4/**), dependent on the deterministic node mobility pattern (requirement **/R5/**), is provided. The latter is especially relevant for dynamically integrating nodes with heterogeneous hardware characteristics (requirement **/R3/**).

3. The metric distribution test **/E3/** investigated the ability to exchange routing information (requirement **/R6/**) among nodes in a timely manner. It was shown that distributing node metrics introduces an almost negligible overhead compared to the overall network resources (requirement **/R8/**).

4. The use of predicted soft time frames was compared to the provisioning of accurate contact intervals in **/E4/**. The results show that soft time frames are supported without significant drawbacks. This is necessary for the dynamic integration of nodes (requirement **/R3/**). Furthermore, the test shows that scheduled contacts can be leveraged; thus, requirement **/R1/** is fulfilled.

5. Test **/E5/** analyzed the impact of using a probabilistically-extended variant of CGR based on contact confidence predictions. From the test results, it can be concluded that the overall concept is applicable to probabilistic contacts (requirement **/R2/**). Moreover, it was shown that the replication overhead can be kept low (requirement **/R9/**), which leads to reduced computational demands (requirement **/R7/**).

6. In **/E6/**, an analysis of the impact on data delivery performance resulting from different approaches to contact volume prediction was conducted. It was found that an improved volume prediction taking into account the link characteristics (requirement **/R4/**) and mobility patterns of nodes (requirement **/R5/**) can especially reduce computational overhead caused by bundle re-scheduling (requirement **/R7/**).

7. The last test case (**/E7/**) assessed the overall computational and power demands of the developed concepts. The discussion shows that leveraging prediction-based proactive routing can enable efficient use of the available energy (requirement **/R7/**).

Overall, as stated in section 3.5, the fulfillment of the requirements provides a foundation for assessing the core research theses as well as the underlying research question.

8 Conclusion and Outlook

8.1 Conclusion

In section 1.3, the following research question is stated:

> *Can routing based on the prediction of contacts improve the data delivery performance in disruption-tolerant satellite networks?*

From the validation of requirements in section 7.10, an answer to this question can be derived via the assessment of its decomposed form, i.e., the three research theses introduced along with it. The assessment for each research thesis is summarized below.

/T1/ Leveraging the observed quality of the communication channel during contacts in concert with the underlying deterministic mobility pattern improves the accuracy of contact predictions.

Assessment: The developed prediction framework considers the quality of the communication channel as well as the deterministic mobility pattern of involved nodes. In subsection 7.5.2, it is shown that this allows for achieving higher accuracy than by comparable strategies, which, e.g., only take into account node mobility.

/T2/ By distributing a numerical description of node characteristics throughout the network, future communication opportunities can be inferred by all nodes, enabling the decentralized calculation of end-to-end paths.

Assessment: The presented prediction-based routing approach distributes *node metrics* to inform other nodes of observed topological characteristics while causing only minor overhead. This way, the same set of future communication opportunities (in the form of a contact plan) can be derived by all nodes in the network.

/T3/ The use of accurate contact predictions in conjunction with a routing approach that plans end-to-end paths based on future communication opportunities leads to improved data delivery performance.

Assessment: By using contact predictions as a basis for route computations, the resulting system can leverage end-to-end paths to schedule bundles proactively. In section 7.7, it is shown that, compared to the evaluated opportunistic techniques, an improved data delivery performance can be achieved with scheduled routing.

In summary, firstly, the fulfillment of requirements confirms the applicability of the presented prediction-based routing framework to the selected use case. Secondly, it was shown by evaluation steps **/E1/** and **/E2/** that the introduced prediction approaches increase accuracy regarding the contact volume when compared to widely-used alternatives and allow for estimating the probability of contact occurrence for all network nodes. Thirdly, evaluation steps **/E4/** to **/E6/** indicated that scheduled routing (i.e., CGR) outperforms the analyzed opportunistic techniques.

Hence, an affirmative answer to the core question can be derived. This thesis has shown that the use of a proactive, prediction-based routing approach can improve data delivery performance in a disruption-tolerant satellite network.

Besides the described contact prediction and routing concepts, this thesis makes further contributions, which can be summarized as follows:

1. **Metric-based alternative to contact plan distribution**. To the best of the author's knowledge, the description of node characteristics in the compressed form of *node metrics* (see section 5.4) has not yet been proposed. This representation, which converges toward values that are constant over time, may drastically reduce the amount of distributed information compared to the exchange of full contact plans.

2. **Approach to derive realistic test data from satellite observations**. The developed evaluation setup employs realistic test data that are generated from satellite observations. These observations can be obtained from a world-wide network of satellite ground stations.

3. **Simulation toolchain**. A novel event-based simulation toolchain was designed and developed that allows evaluating opportunistic as well as deterministic routing protocols. By using Python as the technological basis, it enables rapid prototyping of new routing techniques. The simulator is provided as open source software to the research community.

4. **First analysis of CGR with inaccurate contact volumes**. In past literature, CGR has been studied either in fully-deterministic networks, in networks with random outages (i.e., node or contact failures), or, with the Opportunistic CGR extension, in networks with random node mobility. To the best of the author's knowledge, routing under inaccurate contact volumes or data rates has not yet been analyzed.

8.2 Future Work

On the path to a productive application of the presented techniques, further issues need to be addressed. These issues include the management of congestion when applying deterministic routing. Choosing the fastest routes for all bundles on all nodes in the network can lead to increased load and possibly reduced performance on the most viable paths. Mechanisms known from TCP/IP networks such as the detection of packet dropping or an explicit congestion notification (ECN) [RFC3168] cannot be applied to a DTN as the feedback may reach the responsible nodes too late. Thus, further research is needed for proactive solutions to this problem.

A second problem is the scalability of CGR, i.e., the support of networks with an extended number of nodes (see section 7.8). Though currently being investigated by several authors, there is no unified solution yet.

Thirdly, the support of networks with less-predictable contacts by CGR still needs further research. As shown in related work ([Bur+16], [NB18]) as well as in subsection 7.7.4, replication based on CGR can lead to significant performance penalties. The combination of CGR with opportunistic routing techniques (e.g., an exchange of known bundle identifiers or specific replication limits) may provide a solution to this issue. Apart from that, it should be considered to improve the computation of an aggregate delivery probability over several routes by taking intersections on the routes into account, as mentioned in subsection 6.3.3.

Concerning the proposed prediction concepts, some extensions that are considered viable topics for future work are listed below.

- Predictions may be further enhanced by using distributed information for providing feedback to update local neighborhood information, as indicated in subsection 6.2.2. A preliminary analysis of such an approach for the estimation of contact probabilities has been presented in [Wal17].

- An evaluation of the prediction-based routing concept in more sparse networks with less frequent contacts or increased propagation delays needs to be performed, e.g., to assess the suitability for deep-space systems.

- The applicability of the prediction framework, especially regarding volume prediction, to adaptive data rate transceiver systems has to be evaluated.

Appendices

A Core Algorithm Code Samples

The following listings provide simplified Python code for core algorithms applied in this thesis. Intentionally, no pseudocode is provided, but simplified versions of the Python code used in the evaluation toolchain. By that, the code is still straight-forward to understand but can also be executed without modifications.

In listing A.1, code for the ratio-based confidence metric inference is provided. The algorithm uses a list of predictions that were generated for contacts whose time frames have elapsed already. A function *get_observation* is invoked, which either returns a valid processed contact observation for the given time frame or *None* if no such observation is found.

Listing A.1: Simplified Python code for ratio-based confidence metric inference (see subsection 5.8.2)

```
# Inputs: List of available contact predictions (ordered),
#         Current timestamp
# Output: Current estimation of confidence (= confidence metric)
# Constants: Minimum number of contacts: MIN_CONTACTS,
#            Default value for contact confidence: DEFAULT_CONFIDENCE
def confidence_ratio(predicted_contacts, current_time):
    wnd = list()

    for c in predicted_contacts:
        if c.end_time > current_time:
            break
        if get_observation(c.start_time, c.end_time) is not None:
            wnd.append(1)
        else:
            wnd.append(0)

    if len(wnd) < MIN_CONTACTS:
        return DEFAULT_CONFIDENCE
    else:
        # Ratio of #observed divided by #predicted.
        return sum(wnd) / len(wnd)
```

Listing A.2 provides code for the exponential weighted moving average confidence metric inference, which is an alternative to the ratio-based technique. The available list of predictions is processed in a similar manner, but the confidence value is calculated iteratively in this case.

Listing A.2: Simplified Python code for EWMA-based confidence metric inference (see subsection 5.8.2)

```
# Inputs: List of available contact predictions (ordered),
#         Current timestamp
# Output: Current estimation of confidence (= confidence metric)
# Constants: Confidence observation weight: W_OBS,
#            Minimum number of contacts: MIN_CONTACTS,
#            Default value for contact confidence: DEFAULT_CONFIDENCE
def confidence_ewma(predicted_contacts, current_time):
    p_cur = DEFAULT_CONFIDENCE
    counter = 0

    for c in predicted_contacts:
        if c.end_time > current_time:
            break
        if get_observation(c.start_time, c.end_time) is not None:
            x_i = 1
        else:
            x_i = 0
        p_cur = W_OBS * x_i + (1 - W_OBS) * p_cur
        counter += 1

    if counter < MIN_CONTACTS:
        return DEFAULT_CONFIDENCE
    else:
        return p_cur
```

In the following listing A.3, simplified Python code for the two-state volume metric inference is provided. This algorithm infers an elevation threshold and a link quality estimation from a list of processed contact observations. For that purpose, it obtains the maximum data rate achievable by the communication systems of the given transmitting and receiving node (function *get_br_max*, line 7). For each observed contact, the volume associated with individual signal quality samples is calculated using a function *get_sample_volumes* (line 15) leveraging equation 5.36. The resulting list of volumes is analyzed to get the indices for the start and the end of the *good* interval (lines 19–23, see also equations 5.37 and 5.38). The elevation values at the instants defined by these indices are added to the list of elevation thresholds. As a final processing step of the contact observation, the average link quality indicator is determined by calculating the ratio of the measured volume during the *good* interval divided by the maximum achievable volume during that interval (lines 42–49, see also equation 5.41). The median values of the lists of elevation thresholds and link quality indicators are returned and represent the tuple of volume metrics.

Code for the metric inference step of the three-state volume prediction technique is not provided here, as its operation is very similar, except that an additional threshold is employed for the *bad* state, and the associated list of elevation thresholds is split into multiple lists (one for each azimuth interval).

Listing A.3: Simplified Python code for two-state volume metric inference (see subsection 5.8.3)

```python
# Inputs: List of processed contact observations,
#         Transmitting node, Receiving node
# Outputs: Elevation threshold and quality estimation for good state
# Constants: Volume sample duration: SAMPLE_DURATION,
#            Minimum volume achievement ratio: MIN_GOOD_VA_RATIO
def volume_inf_twostate(observations, tx_node, rx_node):
    br_max = get_br_max(tx_node, rx_node)
    max_sample_volume = br_max * SAMPLE_DURATION

    good_elevations = list()
    good_q = list()

    for obs in observations:
        # Get a list of volumes calculated for the signal quality samples.
        sample_vols = get_sample_volumes(obs.q_samples, max_sample_volume)

        try:
            # Get sample indices at the start and the end of the interval.
            good_start, good_end = get_va_interval(
                sample_vols,
                max_sample_volume,
                MIN_GOOD_VA_RATIO,
            )
        except ValueError:
            # No "good" interval found, skip this observation.
            continue

        # Get the elevation at the start and end of the "good" interval.
        good_start_elev = get_elevation_at(
            tx_node,
            rx_node,
            obs.start_time + good_start * SAMPLE_DURATION,
        )
        good_end_elev = get_elevation_at(
            tx_node,
            rx_node,
            obs.start_time + good_end * SAMPLE_DURATION,
        )
        good_elevations.append(good_start_elev)
        good_elevations.append(good_end_elev)

        good_int_volume = 0
        good_max_volume = 0
        for i, volume in enumerate(sample_vols):
            # Only include volume samples within the "good" interval.
            if i >= good_start and i <= good_end:
                good_int_volume += volume
                good_max_volume += max_sample_volume
        good_q.append(good_int_volume / good_max_volume)

    return median(good_elevations), median(good_q)
```

Listing A.4 provides simplified code for performing two-state volume prediction. This code employs a function *get_interval_for_elevation*, which returns a start and end timestamp for an interval during the provided soft time frame, for which the elevation angle between the two nodes is always greater than the given minimum elevation.

Listing A.4: Simplified Python code for two-state volume prediction (see subsection 5.8.3)

```
# Inputs: Transmitting node, Receiving node, Start and end of soft time
#         frame, Elevation threshold and quality estimation for good state
# Output: Predicted volume for contact in given soft time frame
def volume_pred_twostate(tx_node, rx_node, soft_start, soft_end, theta, q):
    br_max = get_br_max(tx_node, rx_node)

    # Get the start and end time of an interval that is part of the soft
    # time frame and during which the elevation is >= min_elevation.
    t_good_0, t_good_1 = get_interval_for_elevation(
        tx_node,
        rx_node,
        soft_start,
        soft_end,
        min_elevation=theta,
    )

    return (t_good_1 - t_good_0) * br_max * q
```

Listing A.5 provides a function for computing the shortest paths from a source node (*source*) to all other reachable nodes, given a graph based on nodes as vertices as mentioned in subsection 7.2.2. The shown *tvdijkstra* function is based on Dijkstra's shortest path algorithm that has been modified to comply with the SABR specification [CCSDS19] and allow the exclusion of nodes and contacts from the search. The code is heavily simplified, e.g., a list is used instead of a priority queue, nodes may be analyzed twice, and further properties of the determined routes are not stored. Based on this function, algorithms to derive a list of alternative routes can be implemented. These include Yen's k-shortest-path algorithm as well as techniques based on a *limiting contact* as used in the case of *CGR-Prob* depicted later in this appendix (see also section 6.3).

In the following code, the *graph* object contains a dictionary (hash map) of *vertices*, which maps unique identifiers of network nodes to a list of unique identifiers of reachable neighbors, i.e., nodes which are reachable via any single contact from the given node. The node identifiers may be, e.g., EID or node numbers. The only requirement is that they are immutable and can be used as dictionary keys. Additionally, the graph contains a dictionary of *edges* that associates an ordered pair (2-tuple) of node identifiers to a list of contact objects, which are ordered by start time. The contact objects have to be immutable as well and shall provide at least the sending and receiving node identifiers (*tx_node* and *rx_node*), the start and end time (*start_time* and *end_time*), the average data rate (*bit_rate*), the expected maximum delay (*delay*, equivalent to the range in light seconds), and the contact confidence (*confidence*). The arguments *excluded_nodes* and *excluded_cts* are sets of node identifiers and contact objects that need to be excluded from the search, e.g., to find another applicable route.

Listing A.5: Simplified Python code for route search over node graph (see subsection 7.2.2)

```
# Inputs: Node graph, Source node ID for search, Arrival time at source,
#         List of excluded node IDs, List of excluded contact objects
# Outputs: Dict mapping node ID to earliest arrival time,
#          Dict mapping node ID to contact for earliest arrival
# Constants: One-way light time margin of mission: OWLT_MARGIN
def tvdijkstra(graph, source, arrival_time, excluded_nodes, excluded_cts):
    # A mapping of node identifier to earliest arrival time.
    distance = {key: float("inf") for key in graph.vertices}
    distance[source] = arrival_time
    # A mapping of node identifier to contact for earliest arrival.
    taken_contact = {key: None for key in graph.vertices}
    # The queue is used to access nodes ordered by distance.
    queue = [(arrival_time, source)]
    while queue:
        cur_dist, cur_node = queue.pop(0)
        # Neighbors: All nodes to which any contact exists from cur_node.
        for neigh_node in graph.vertices[cur_node]:
            for contact in graph.edges[(cur_node, neigh_node)]:
                # If contact excluded or bundle arrives too late, skip.
                if contact in excluded_cts or contact.end_time < cur_dist:
                    continue
                # SABR 3.2.4.1.1: Earliest Transmission Time
                ett = max(contact.start_time, cur_dist)
                # SABR 3.2.4.1.2: Earliest Arrival Time
                next_eat = ett + contact.delay + OWLT_MARGIN
                # If the new distance is smaller, update and add to queue.
                if next_eat < distance[neigh_node]:
                    distance[neigh_node] = next_eat
                    taken_contact[neigh_node] = contact
                    if neigh_node not in excluded_nodes:
                        queue.append((next_eat, neigh_node))
                        queue.sort()
                    # The contacts are ordered.
                    break
    return distance, taken_contact
```

Listing A.6 provides a code sample for obtaining the list of contacts taken to reach a specific destination node given the *taken_contact* result of *tvdijkstra* and a destination node identifier.

Listing A.6: Simplified Python code for obtaining the route as list of contacts

```
# Inputs: Dict mapping node ID to contact for earliest arrival,
#         Destination node ID
# Output: List of contacts from source to destination
def get_route(taken_contact, destination):
    cur_node = destination
    contact_path = list()
    while taken_contact[cur_node] is not None:
        # Add contacts to the path in reverse order.
        contact_path.append(taken_contact[cur_node])
        cur_node = taken_contact[cur_node].tx_node
    return list(reversed(contact_path))
```

In listing A.7, a code sample to generate a list of routes is provided, implementing the probabilistic extension introduced in section 6.3.3. The parameter *source* represents the identifier of the current node, and *start_time* is the current time at the start of the search.

Listing A.7: Simplified Python code for probabilistic route computation (see subsection 6.3.3)

```python
# Inputs: Node graph, Source node ID, Destination node ID,
#         Arrival time at source, List of excluded node IDs
# Output: List of routes (lists of contacts to reach destination)
# Constants: Minimum route confidence: P_ROUTE_MIN
def route_gen(graph, source, destination, start_time, excluded_nodes):
    routes = list()
    excluded_contacts = set()
    while True:
        _, taken_contact = tvdijkstra(
            graph,
            source,
            start_time,
            excluded_nodes,
            excluded_contacts,
        )

        contact_list = get_route(taken_contact, destination)
        if not contact_list:
            break  # Nothing found, terminate.

        # Find earliest-ending contact.
        route_end_time = float("inf")
        time_limiting_contact = None
        for contact in contact_list:
            if contact.end_time < route_end_time:
                route_end_time = contact.end_time
                time_limiting_contact = contact

        # Find least-probable contact.
        route_min_confidence = 1.0
        conf_limiting_contact = None
        for contact in contact_list:
            if contact.probability < route_min_confidence:
                route_min_confidence = contact.confidence
                conf_limiting_contact = contact

        # Get route confidence.
        route_confidence = 1.0
        for contact in contact_list:
            route_confidence *= contact.confidence

        # Only return the route if Pr >= Pr_min.
        if route_confidence < P_ROUTE_MIN:
            excluded_contacts.add(conf_limiting_contact)
        else:
            excluded_contacts.add(time_limiting_contact)
            routes.append(contact_list)

    return routes
```

Listing A.8 provides simplified code for selecting routes for forwarding a bundle based on the list of candidate routes and the associated confidence values.

Listing A.8: Simplified Python code for probabilistic route selection (see subsection 6.3.3)

```
# Inputs: Node graph, Bundle, Current node ID, Current time,
#         List of excluded node IDs, Maximum number of replicas
# Output: List of routes for bundle
# Constants: Minimum total delivery confidence: P_BUNDLE_MIN
def probcgr(graph, bundle, cur_eid, cur_time, excluded_nodes, n_max):
    # The list is populated according to SABR 3.2.6.
    candidate_routes = get_candidate_routes(
        graph,
        bundle,
        cur_eid,
        cur_time,
        excluded_nodes,
    )

    used_routes = []
    failure_prob = 1.0
    for candidate in candidate_routes:
        used_routes.append(candidate)

        # Get route confidence.
        route_confidence = 1.0
        for contact in candidate.contact_path:
            route_confidence *= contact.confidence

        # Increase total confidence and terminate if sufficient.
        failure_prob *= (1 - route_confidence)
        if failure_prob <= 1 - P_BUNDLE_MIN:
            break
        if len(used_routes) == n_max:
            break

    return used_routes
```

B Validation of Routing Approaches

In this appendix, further validations performed for the implemented routing approaches are documented. All plots show arithmetic means and the 95 % intervals measured for the given parameters and scenarios.

B.1 ONE Validation

As described in section 7.3, the implementations of Epidemic Routing and Spray and Wait were validated against the ONE simulator. Using scenario 2 from [FW17b], five simulation runs were performed with aiodtnsim for each of the following four routing algorithms.

- **aiodtnsim_Epidemic**: The default aiodtnsim implementation of Epidemic Routing, which performs an exchange of *summary vectors* at the start of each contact and limits the maximum hop count as described in [VB00].
- **aiodtnsim_ONE_Epidemic**: A variant of Epidemic Routing, which checks if the other node has a specific bundle using a look-up in its buffer and the list of delivered bundles, without any delay. No summary vectors are exchanged, and the hop count is not limited. This behavior is identical to ONE.
- **aiodtnsim_SprayAndWait**: The default aiodtnsim implementation of Spray and Wait. This algorithm does not check whether the other node already has a specific bundle. It, however, prevents sending a bundle twice during a contact and increases the number of copies if additional bundles are received.
- **aiodtnsim_ONE_SprayAndWait**: A modified Spray and Wait algorithm that accesses the buffer of the encountered neighbor as well as its list of delivered bundles without any delay, to check whether a bundle shall be sent. This behavior is identical to that of the Spray and Wait algorithm in ONE.

In addition to that, four simulation runs were performed using the ONE. Two runs, labeled **U+C**, allowed simultaneous bi-directional (uplink and downlink) and concurrent transmissions (multiple access). In contrast, two further runs used the stock behavior of the ONE, allowing one bundle transmission at a time. The former extension is outlined in section 7.3.

As the ONE simulator uses a fixed random seed, the results of every run are equal. Thus, only a single run was performed per algorithm. The properties of the scenario and the settings used in all simulation runs are summarized in table B.1. A more extensive description of the overall scenario can be found in [FW17b]. It should be noted that the simulation results documented in the referenced paper were obtained using an older version of the ONE toolchain and, thus, show slightly reduced delivery probabilities compared to the following figures.

Parameter	**Value**
Scenario duration	3 days
Satellites	12
Ground stations	9
Bundle lifetime	1 day
Buffer size	1 Gbit
Data rate	250 kbit/s
Minimum elevation	10 °
Bundle generation interval	30 s
Bundle size	1 Mbit
Bundle queue order	random
Bundle source	random GS
Bundle sink	random GS
Epidemic: Hop count limit	6
Spray and Wait: Type	binary
Spray and Wait: Number of copies	6

Table B.1: ONE validation: Scenario and simulation parameters

Figure B.1 shows the delivery probabilities. In the given scenario, the default aiodtnsim implementations of both Epidemic Routing and Spray and Wait allow for slight increases in delivery probability compared to the ONE implementations. The adapted algorithms modeling the ONE behavior gave very similar results compared to ONE if allowing concurrent transmissions and the simultaneous use of the uplink and downlink (U+C). The default behavior of the ONE, allowing only a single transmission at a time, reduces the delivery probability significantly in the case of Epidemic Routing. The reduction can be explained by the reduced contact time available for bundle transmissions and the increased number of transmissions in the case of Epidemic Routing.

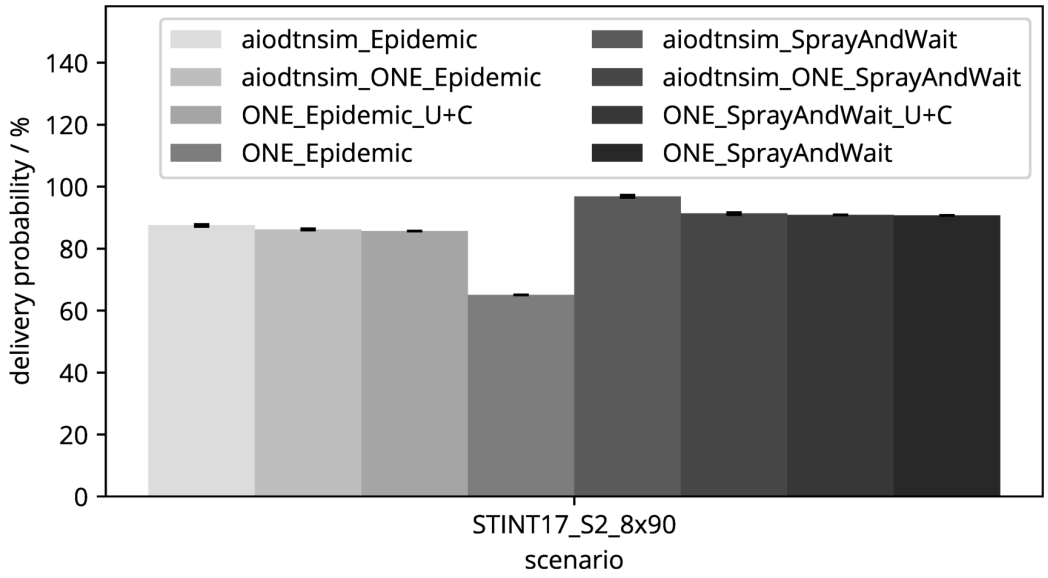

Figure B.1: ONE validation: Bundle delivery probability

The average delivery delays (depicted in figure B.2) have a comparable magnitude for all implementations.

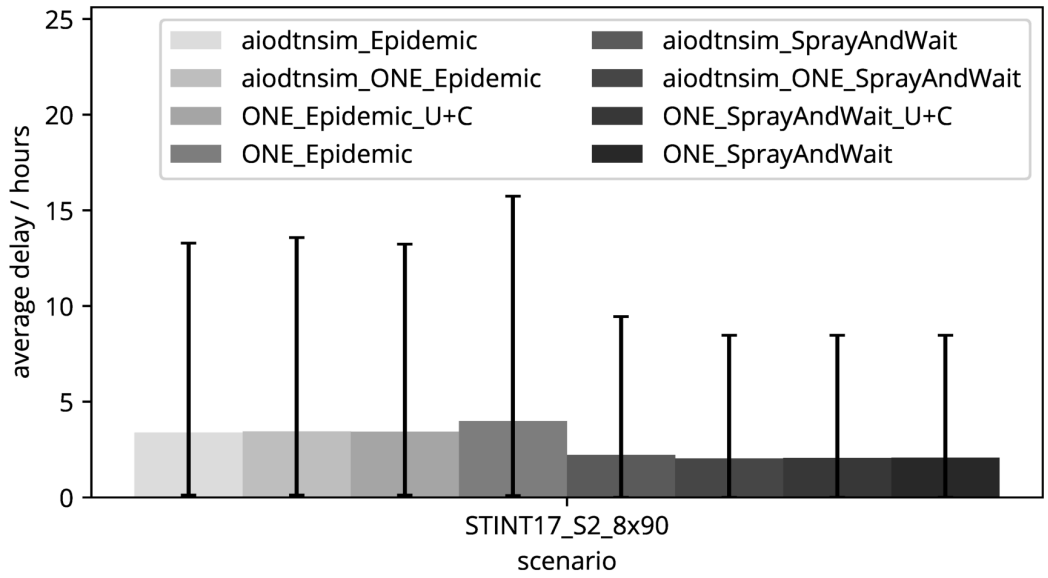

Figure B.2: ONE validation: Average bundle delivery delay

In figure B.3, the average number of transmissions performed for a given bundle (including its replicas) are shown. In the case of ONE, only the average value could be obtained, while aiodtnsim allows for analyzing the parameter for each bundle individually (thus, yields a 95th-percentile interval).

In summary, the test results show that the opportunistic algorithms implemented in aiodtnsim could be compared successfully against another state-of-the-art implementation.

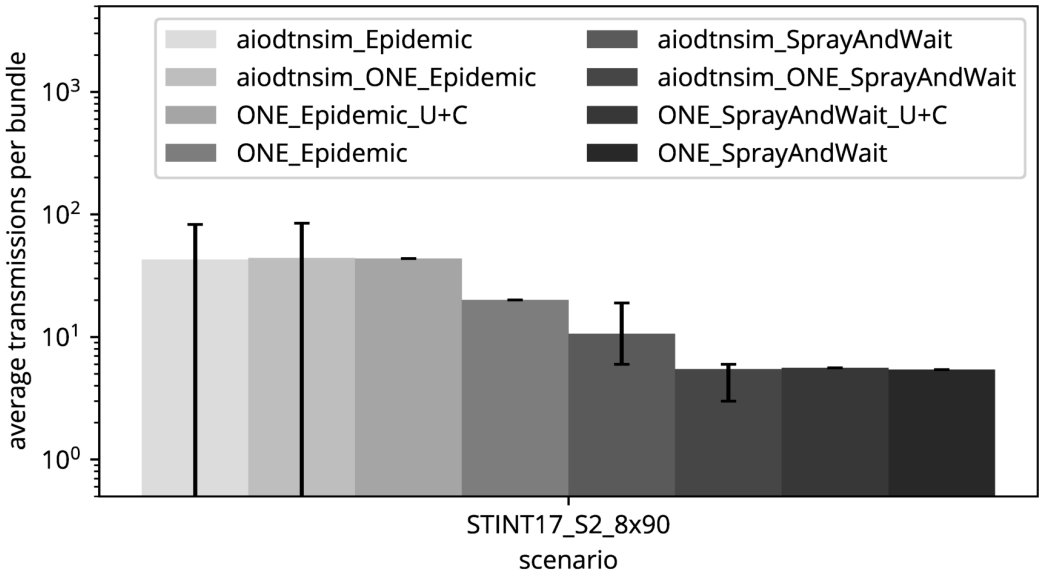

Figure B.3: ONE validation: Total number of transmissions

Several differences could be identified to either result from features described in the original references but not implemented in the ONE (e.g., the summary vector exchange and hop count limit in the case of Epidemic Routing) or from the distinct handling of connections. The latter was addressed by providing patches that add corresponding settings options to the ONE developers, whereas, for modeling the former, adapted algorithms were implemented in aiodtnsim.

The validation of opportunistic algorithms against the ONE doubles as a validation of the statistic evaluation features in the aiodtnsim simulator core.

B.2 CGR Validation

The overall steps taken for validating the CGR implementation are documented in section 7.3. Here, results from the comparison of data delivery performance against *pydtnsim* are depicted. For the simulations, the scenarios and settings described in sections 7.4 and 7.7 were applied.

Figures B.4, B.5, B.6, and B.7 indicate improved performance of the CGR implementation in aiodtnsim compared to pydtnsim in most cases. The delivery probability is slightly increased and delays are significantly reduced under *high* load. Under *low* load, a minor increase in delays could be observed. Figure B.6 indicates a significantly larger total number of transmissions under *high* load when using pydtnsim. As shown in figure B.7, buffers clear faster, and the mean buffer usage is reduced in the case of aiodtnsim. The overall improvements are assumed to result from changes to the CGR algorithm documented in [CCSDS19] compared to that implemented in ION 3.5 (on which pydtnsim is based).

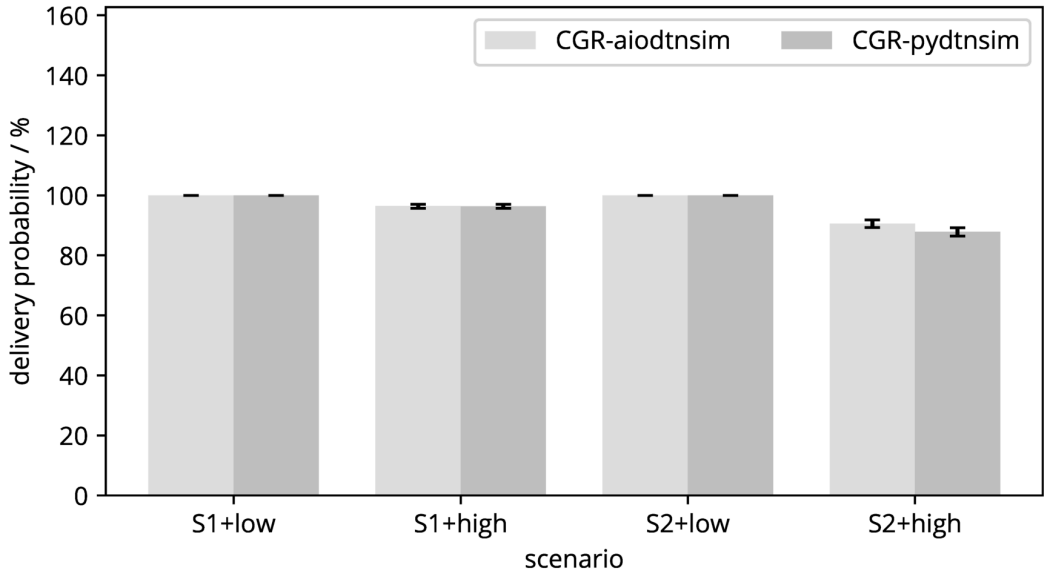

Figure B.4: CGR validation: Bundle delivery probability

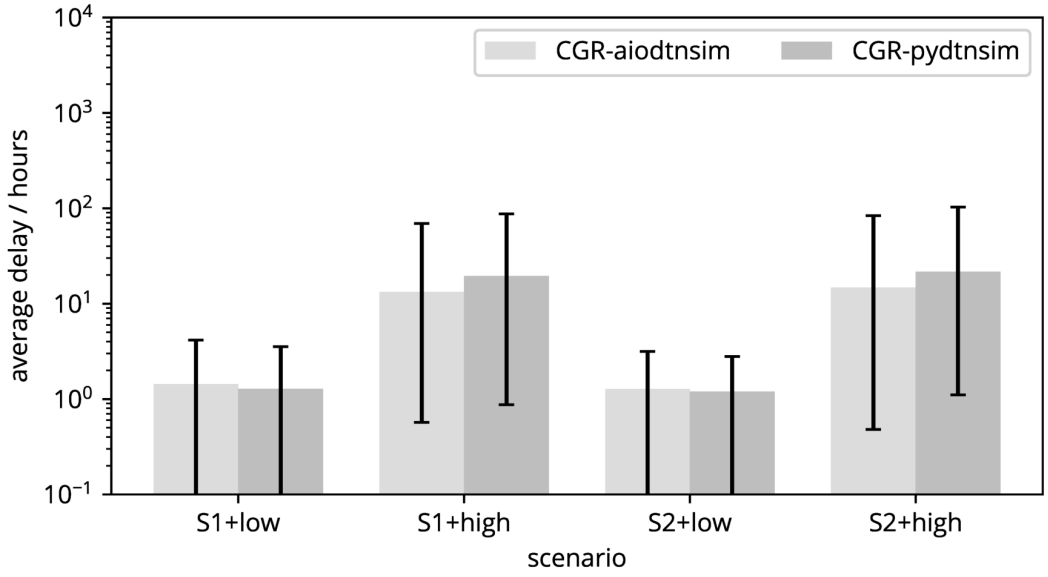

Figure B.5: CGR validation: Average bundle delivery delay (logarithmic scale)

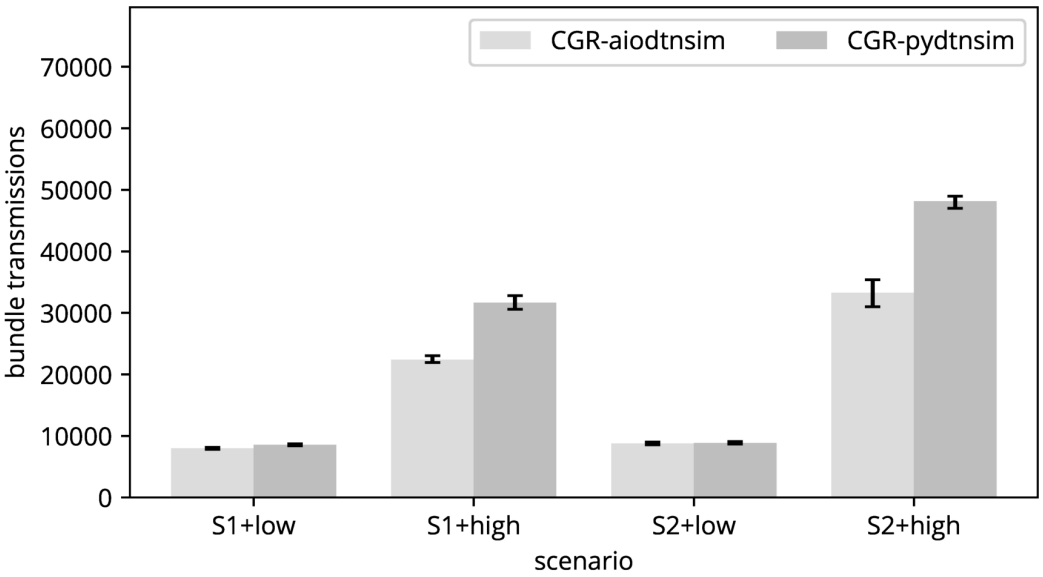

Figure B.6: CGR validation: Total number of transmissions

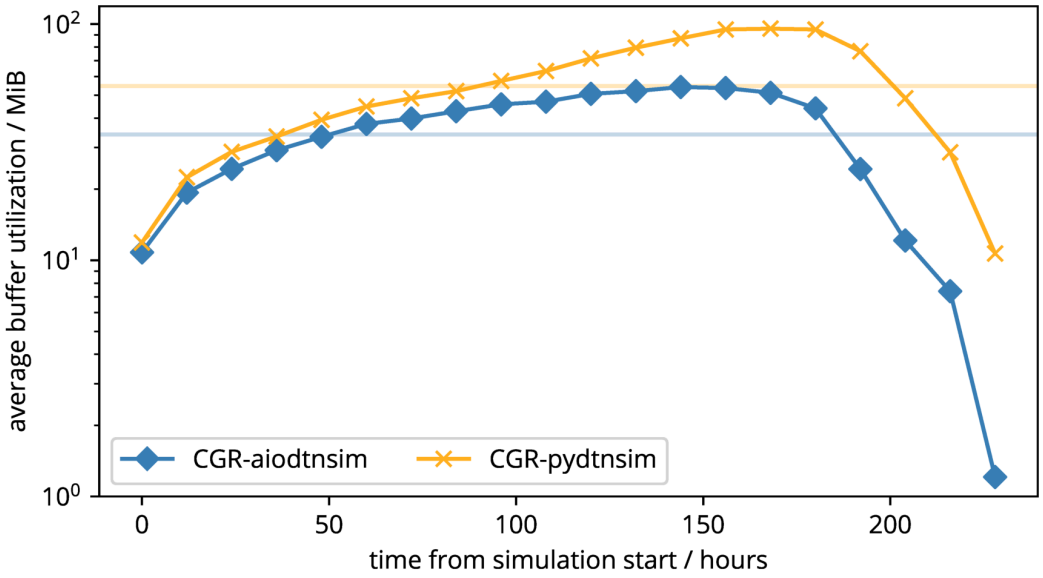

Figure B.7: CGR validation: Average buffer utilization over time, scenario S1, high traffic load

B.3 Alternative Route Determination Techniques

As mentioned in subsections 6.3.3 and 7.2.2, different options for selecting a list of routes exist for CGR and were implemented in the course of the presented evaluation. A set of options is, e.g., provided and discussed by Fraire et al. in [Fra+18].

For use in the simulation study, two basic approaches were considered:

1. Yen's algorithm, a well-known graph algorithm allowing to derive a set of *K* shortest paths for a given graph, and

2. A series of Dijkstra searches, whereas before each search a specific *limiting contact* is excluded.

For the latter, two variants were implemented, one setting the limiting contact to the earliest-*ending* contact, and another one using the contact with the smallest volume (the earliest-*depleted* contact) from the previous route. The maximum number of routes *K* determined by Yen's algorithm was assumed as a configurable parameter and is always provided in the following plots and discussions.

Two scenarios were evaluated with two different traffic configurations. Both scenarios were randomly generated Ring Road networks consisting of the same number of satellites and ground stations. The satellites were selected randomly from the set of CubeSats currently in orbit, and the ground stations were placed randomly on the surface area of the Earth. The scenario duration was set to three days.

The first scenario (*3g3s*) uses three ground stations and three satellites, whereas the second scenario (*5g5s*) consists of five ground stations and five satellites. The contact plan for scenario *3g3s* contains 164 unidirectional contacts, the one for scenario *5g5s* 722 unidirectional contacts.

The traffic load configurations were determined as described in subsection 7.7.1, with one minor difference: For *low* load, the bundle size was set to 2 MiB as in the case of the *high* load configuration. Ten simulation runs were executed for every algorithm in any scenario. No predictions were performed, i.e., all algorithms were provided with accurate contact plans.

The delivery probabilities measured for the comparison are plotted in figure B.8. By inspecting the figure, it becomes clear that a particularly large setting is needed for the parameter *K* of Yen's algorithm to achieve a high delivery probability, especially in the larger scenario under *high* load. This can be explained by the huge number of alternative routes that can be calculated because of the number of contacts and the property of Ring Road networks to offer many alternative paths. In the *5g5s* scenario, even a *K* value of 20000 is not sufficient to deliver all bundles. The approaches using a limiting contact achieve a delivery probability of 100 % in all scenarios. In their case, no limit to the number of alternative routes is applied as the search terminates much faster.

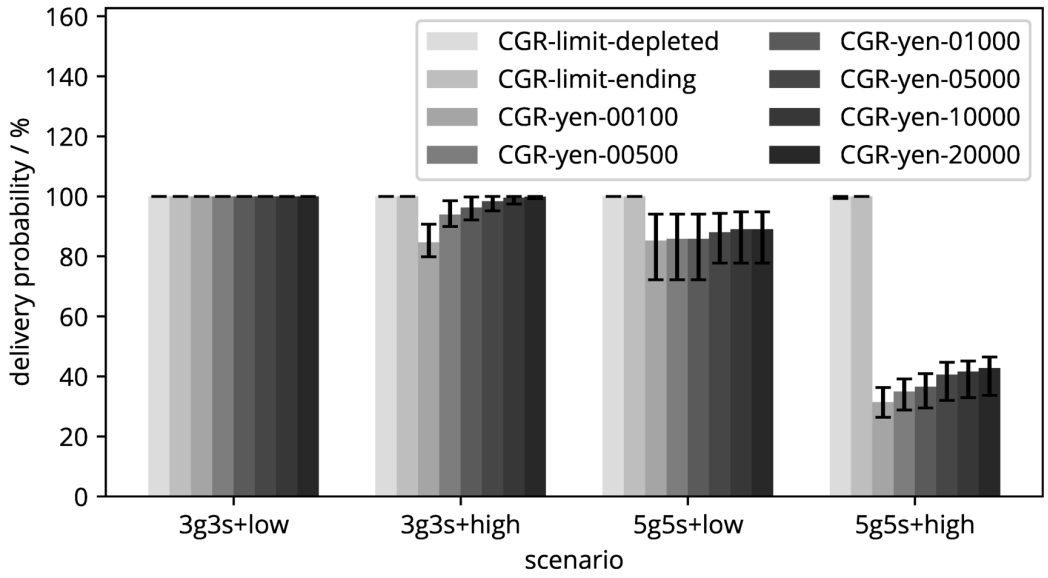

Figure B.8: Route determination test: Bundle delivery probability

The bundle delivery delays shown in figure B.9 are comparable for all CGR instances, as long as the *K* parameter for Yen's algorithm has been set to a sufficiently high value. If this is not the case, only the *K* fastest routes are provided and, when these are fully booked, scheduling fails, i.e., bundles are dropped. Lower delays reported in some cases are explained by dropped bundles not being part of that statistic.

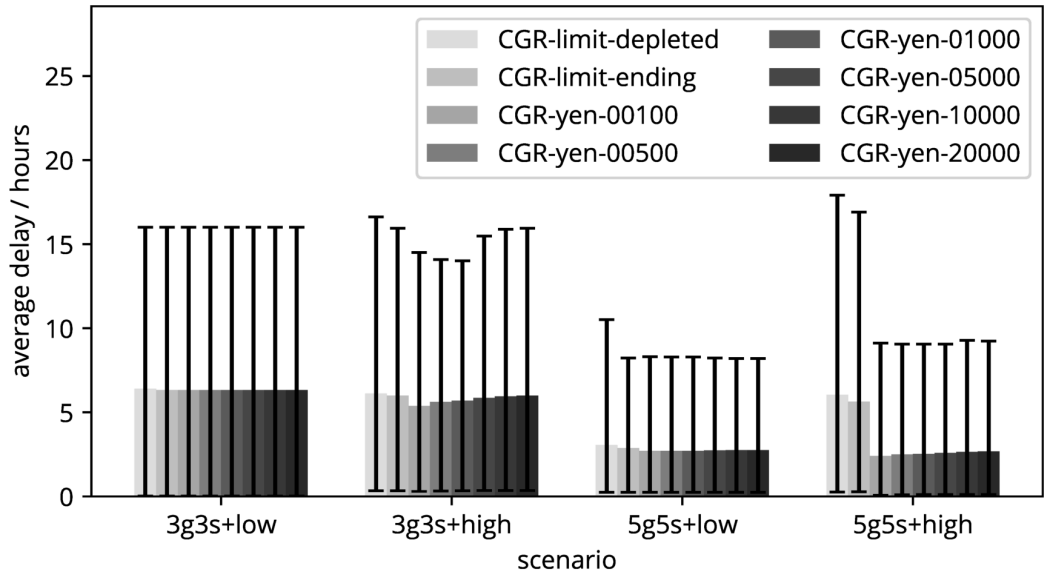

Figure B.9: Route determination test: Average bundle delivery delay

The most important statistic in this comparison, however, is depicted in figure B.10. The run-time, i.e., the duration of a simulator run, is shown with a logarithmic scale. With an increasing parameter *K*, the run-time massively increases, preventing any meaningful application of the CGR implementation with Yen's algorithm to larger scenarios.

The documented issues might be implementation-specific. Though, due to manual tests generating sample routes in scenario *3g3s* using a preview version (*preview3*) of ION 4.0.0, which also uses Yen's algorithm for CGR, it is suspected that this is a general problem with Yen's algorithm, possibly bound to the class of scenario. The generation of sample routes resulted in run-times of several seconds for the ION CGR implementation, whereas a complete simulation run performing over 150 bundle transmissions using *CGR-limit-ending* in the Python-based aiodtnsim implementation terminated in less than one second (scenario *3g3s+high*).

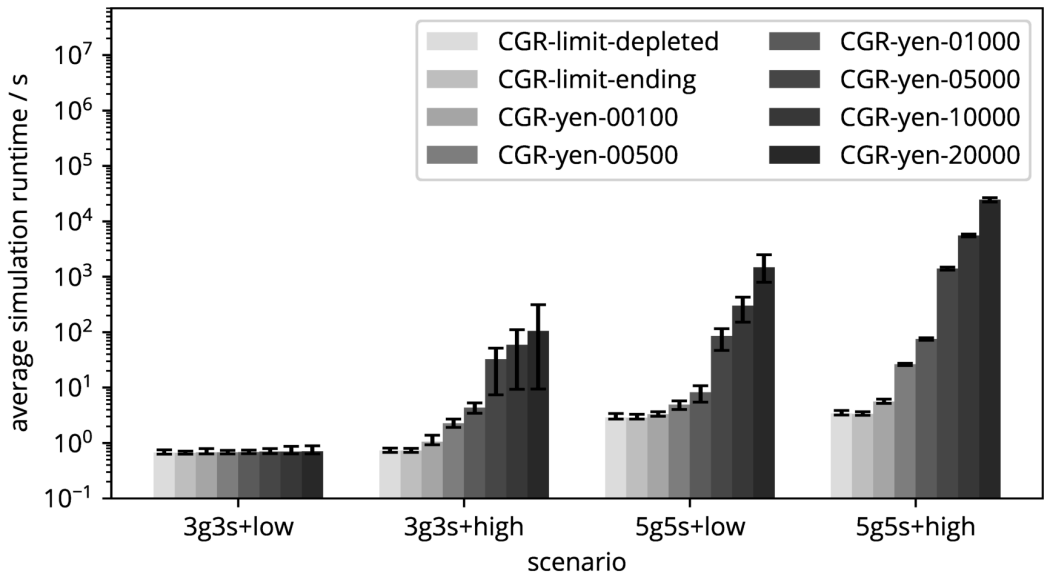

Figure B.10: Route determination test: Average simulation run-time (logarithmic scale)

Due to the extremely increased run-times and non-noticeable benefits regarding delivery probability and delays in the analyzed test scenarios, the limiting contact approach was used in the evaluation, excluding the earliest-ending contact. Compared to excluding the earliest-depleted contact, this configuration showed reduced delays in analyzed example cases.

C Further Performance Results

In this appendix, additional plots of performance results are provided. These plots are not essential for deriving the findings and conclusions of this thesis, but are included as they might be of interest for some readers. The plots show arithmetic means and 95 % intervals if not specified otherwise.

C.1 Metric Distribution Performance

Figure C.1 depicts the node buffer utilization by distributed metric bundles. The overall impact of metric distribution is discussed in section 7.6.

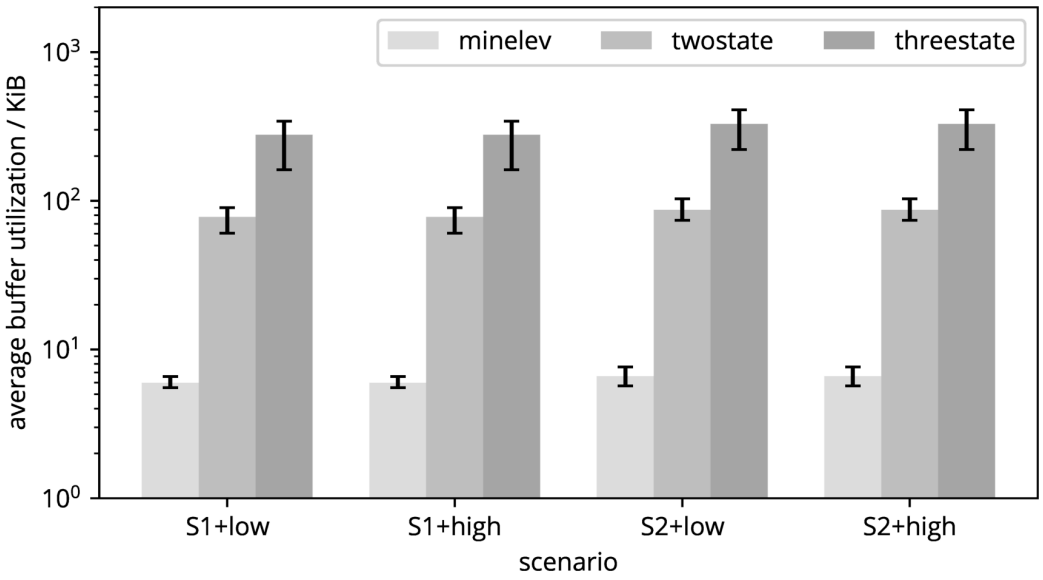

Figure C.1: /E3/: Average buffer utilization by metric bundles (logarithmic scale, with 80 % intervals)

C.2 Routing Performance

C.2.1 Soft Time Frames

In this section, additional parameters measured during the test discussed in section 7.7.3 are depicted. Figure C.2 shows the average end-to-end delay and figure C.3 the ratio of bundles re-scheduled. (See subsection 7.7.4 for an explanation of the latter parameter.)

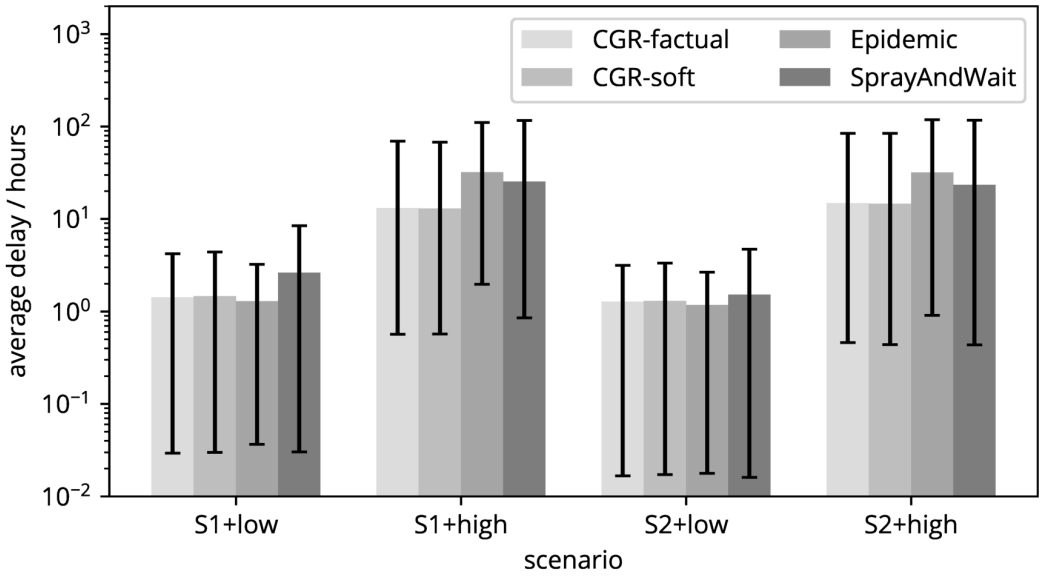

Figure C.2: /E4/: Average bundle delivery delay with soft time frames (logarithmic scale)

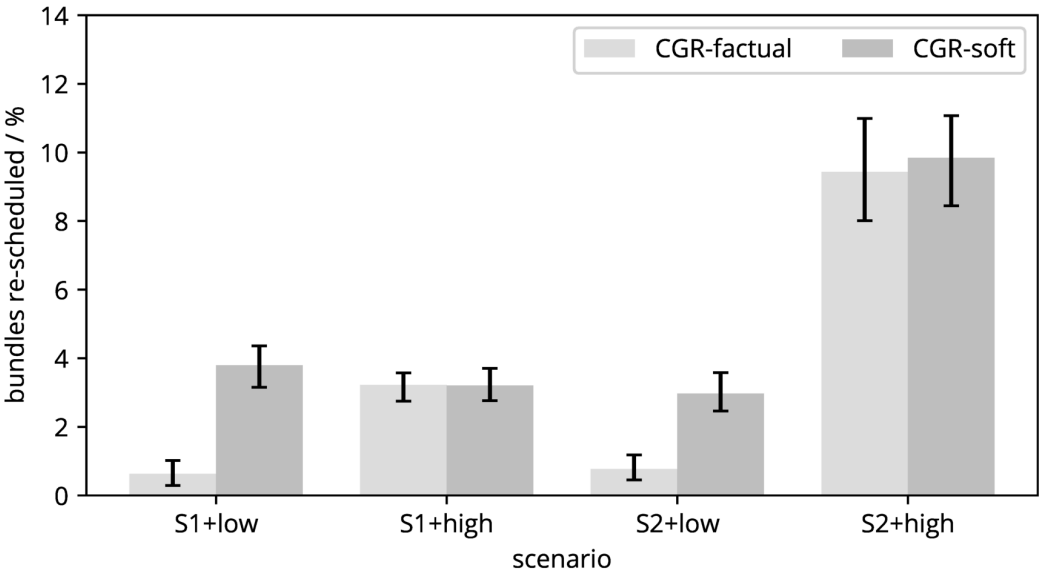

Figure C.3: /E4/: Ratio of bundles re-scheduled with soft time frames

C.2.2 Confidence Prediction

This section provides additional performance results of the evaluation tests documented in subsection 7.7.4. For *high* load, the average end-to-end delay is depicted in figure C.4 and the ratio of bundles re-scheduled is shown in figure C.5. *CGR-Prob* allows for minor reductions in re-scheduled bundles. In the case of *CGR-Prob-10-80* this is, however, assumed a cause of the reduced delivery probability as depicted in figure 7.14.

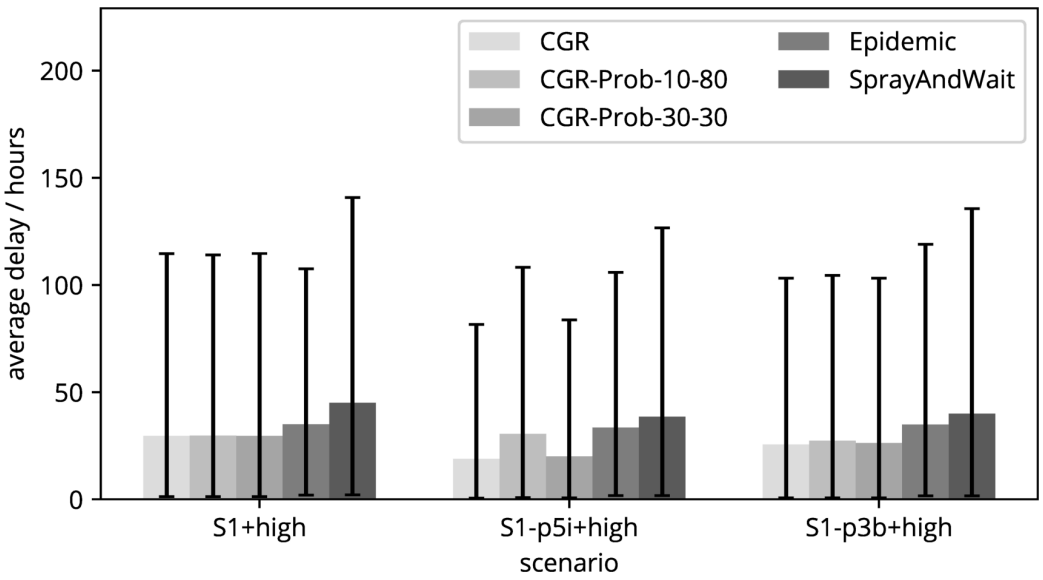

Figure C.4: /E5/: Average bundle delivery delay with confidence prediction, high traffic load

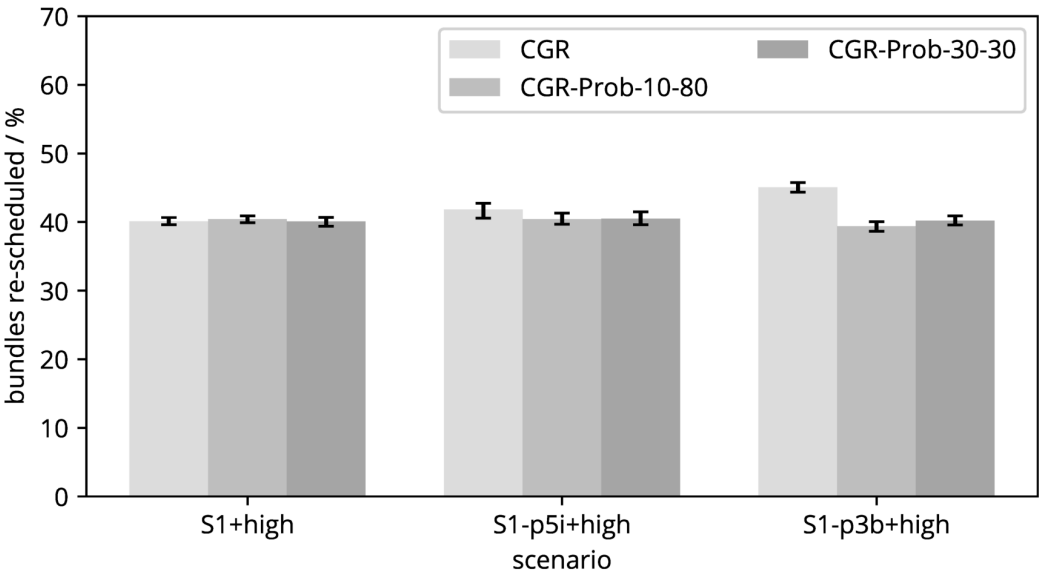

Figure C.5: /E5/: Ratio of bundles re-scheduled with confidence prediction, high traffic load

In figures C.6 and C.7, the total number of transmissions performed during each simulation run is shown. Under *low* load, all CGR instances perform significantly fewer transmissions than the opportunistic routers. Under *high* load, the replicating CGR instance performs a number of transmissions comparable to that of Spray and Wait, whereas the other CGR instances need fewer transmissions. In the scenario with intermittent activity (S1-p5i), the difference is increased, presumably because fewer contacts are available and the estimated confidence values are reduced (but not zero) for extended periods of time. In scenario S1-p3b, the confidence metric for failing nodes quickly drops to zero, leading to reduced replication.

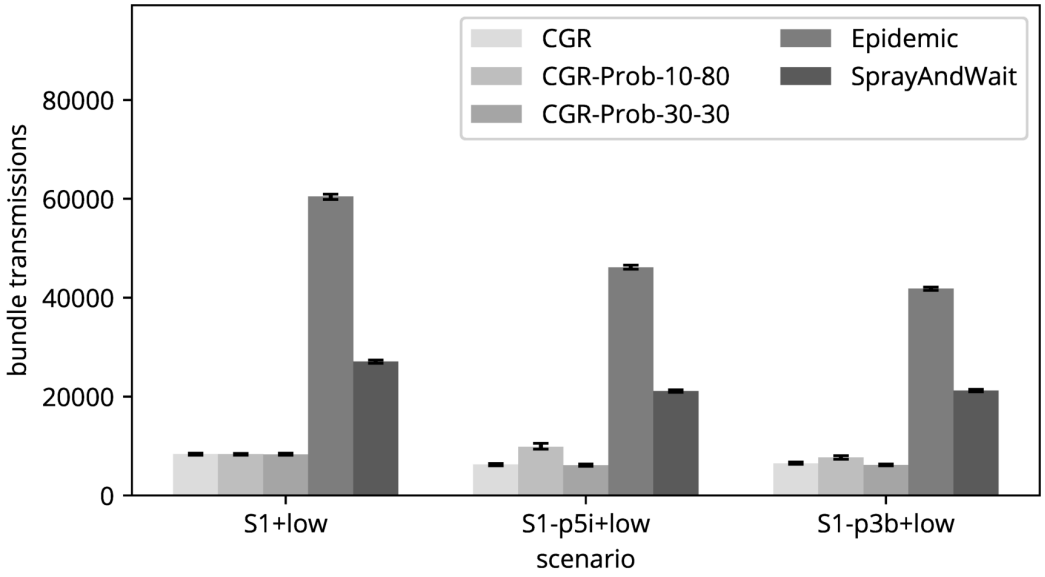

Figure C.6: /E5/: Total number of transmissions with confidence prediction, low traffic load

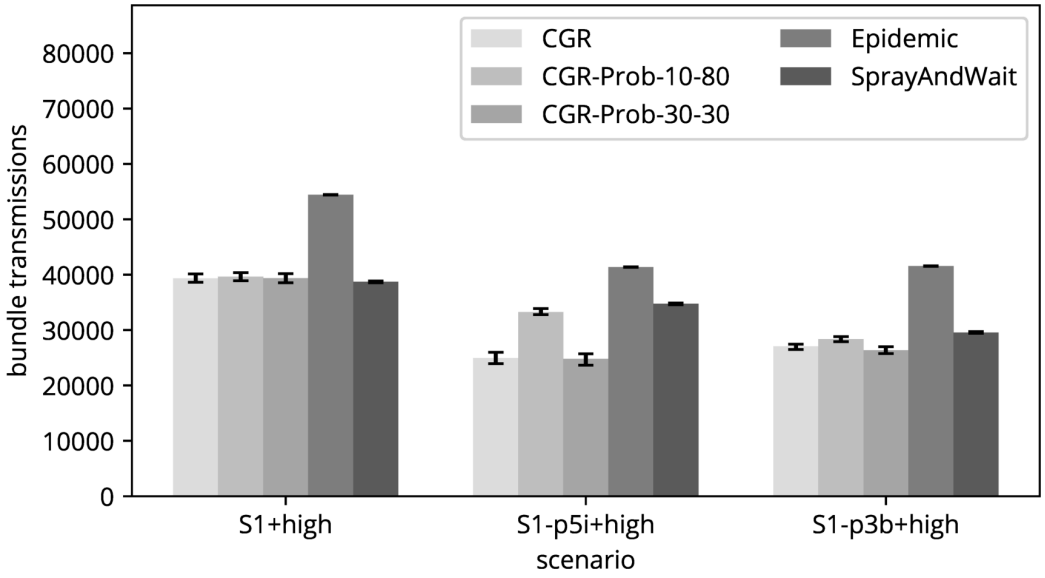

Figure C.7: /E5/: Total number of transmissions with confidence prediction, high traffic load

The average buffer utilization over time in scenario S1-p5i is depicted by figure C.8.

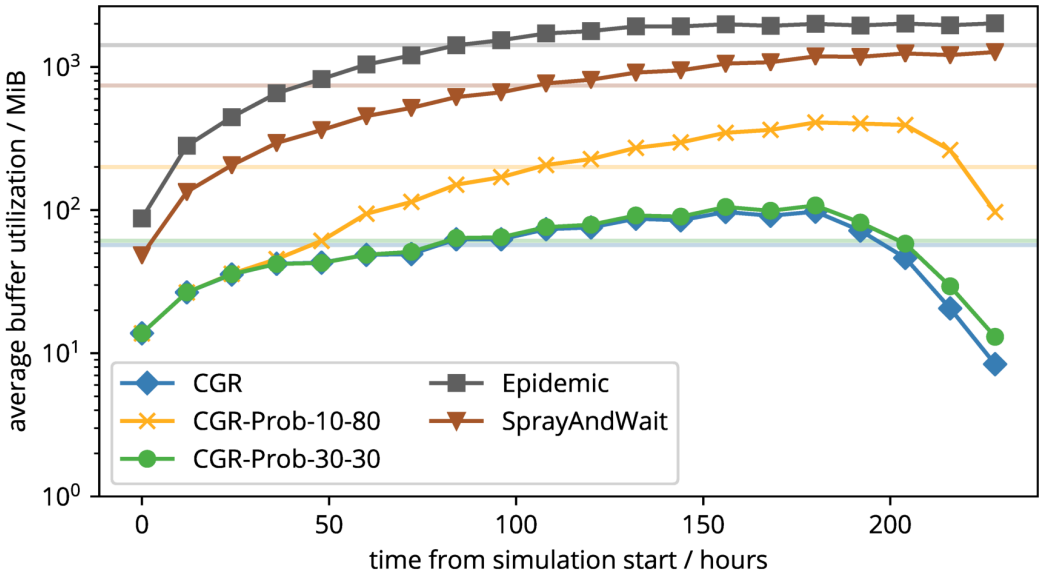

Figure C.8: /E5/: Average buffer utilization over time with confidence prediction, scenario S1-p5i, high traffic load (logarithmic scale)

C.2.3 Volume Prediction

This section provides additional plots for the discussion in subsection 7.7.5. Figure C.9 shows the bundle delivery delay when applying different volume prediction techniques under *low* load.

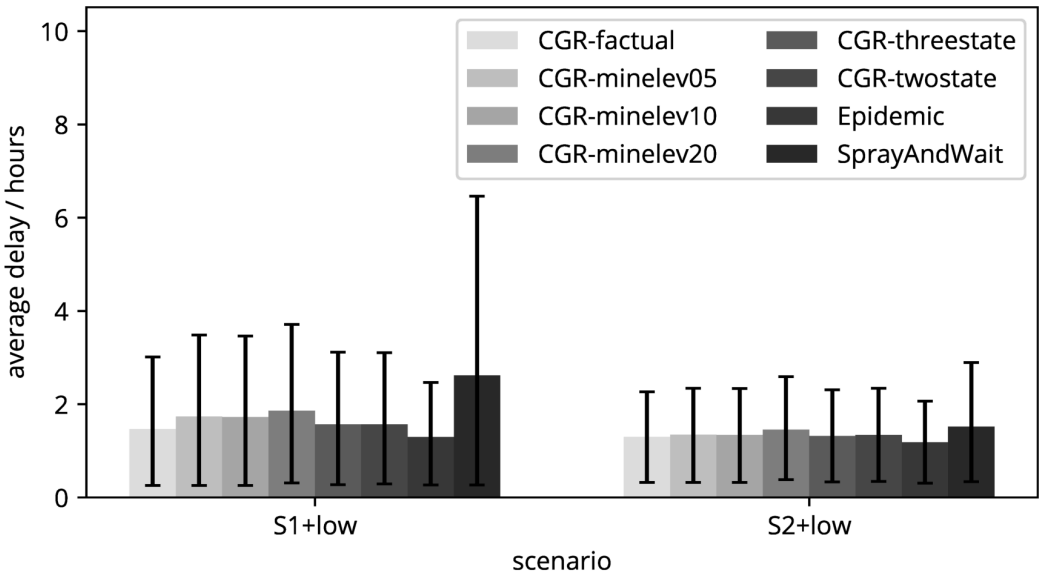

Figure C.9: /E6/: Average bundle delivery delay with volume prediction, low traffic load

In figure C.10, the average use of contact time for data forwarding by the various approaches is shown.

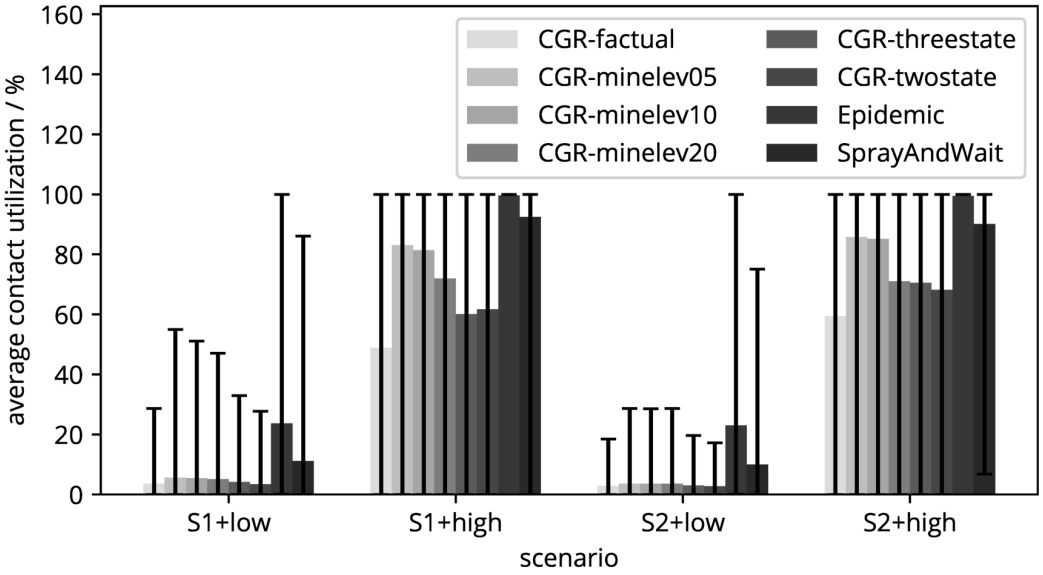

Figure C.10: /E6/: Average contact utilization with volume prediction

In figure C.11, the same parameter is plotted over the simulation time for scenario S1 and *high* traffic load.

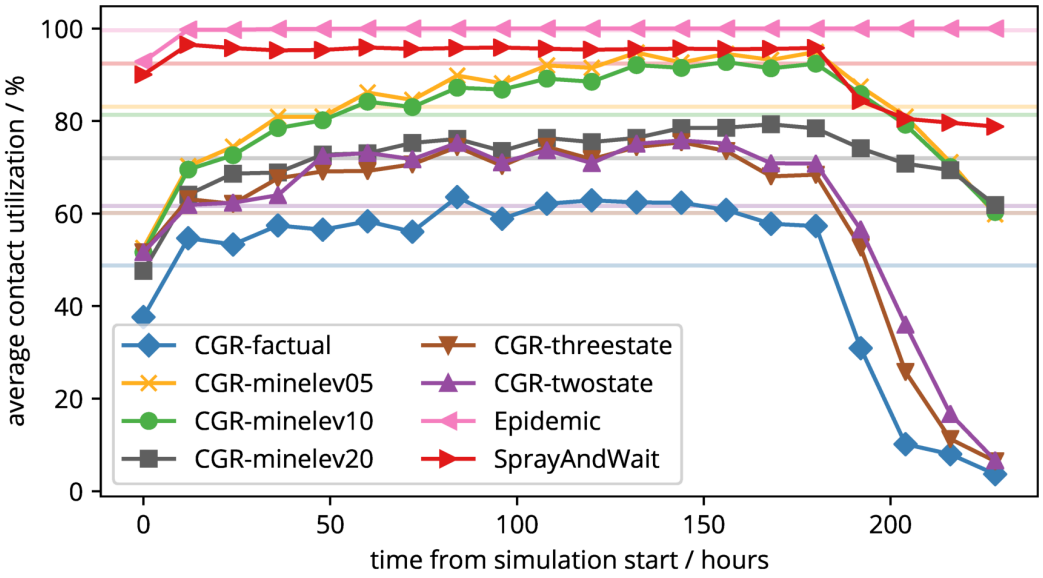

Figure C.11: /E6/: Average contact utilization over time with volume prediction, scenario S1, high traffic load

Research Context

[Fel+17] M. Feldmann, F. Walter, R. Böhm, J. Chorin, O. D. Jonckère, T. Hareau, M. Nitsch, and H. Ritter. “A Quality-Increasing Development Process for LEO Satellite Software”. In: *International Conference on Space Mission Challenges for Information Technology (SMC-IT)*. Alcalá de Henares, Spain, 2017. DOI: 10.1109/SMC-IT.2017.33.

[Fel+18] Marius Feldmann, Felix Walter, Tomaso de Cola, and Gianluigi Liva. “DTN Coding”. In: *Delay and Disruption Tolerant Networks: Interplanetary and Earth-Bound – Architecture, Protocols, and Applications*. Ed. by Aloizio Pereira da Silva, Scott C. Burleigh, and Katia Obraczka. CRC Press, 2018, pp. 139–203. DOI: 10.1201/9781315271156.

[FFW18] Marius Feldmann, Juan A. Fraire, and Felix Walter. “Tracking Lunar Ring Road Communication”. In: *IEEE International Conference on Communications (ICC)*. Kansas City, MO, USA, 2018. DOI: 10.1109/ICC.2018.8423031.

[Fra+19] Juan A. Fraire, Marius Feldmann, Felix Walter, Elena Fantino, and Scott C. Burleigh. “Networking in Interstellar Dimensions: Communicating with TRAPPIST-1”. In: *IEEE Transactions on Aerospace and Electronic Systems* 55.4 (2019). DOI: 10.1109/TAES.2018.2874149.

[FW15a] Marius Feldmann and Felix Walter. “Towards Ground Station Contact Discovery in Ring Road Networks”. In: *IEEE International Conference on Wireless for Space and Extreme Environments (WiSEE)*. Orlando, FL, USA, 2015. DOI: 10.1109/WiSEE.2015.7393096.

[FW15b] Marius Feldmann and Felix Walter. “µPCN - a Bundle Protocol Implementation for Microcontrollers”. In: *International Conference on Wireless Communications and Signal Processing (WCSP)*. Nanjing, China, 2015. DOI: 10.1109/WCSP.2015.7341252.

[FW17a] Marius Feldmann and Felix Walter. “Refining the Ring Road – Delays and Path Lengths in a LEO Satellite Message-Ferry Network”. In: *IEEE International Conference on Communications (ICC)*. Paris, France, 2017. DOI: 10.1109/ICC.2017.7996777.

[FW17b] Marius Feldmann and Felix Walter. “Routing in Ring Road Networks with Limited Topological Knowledge”. In: *Space Terrestrial Internetworking Publication Workshop (STINT-PUBS), IEEE International Conference on Wireless for Space and Extreme Environments (WiSEE)*. Montréal, Canada, 2017. DOI: 10.1109/WiSEE.2017.8124903.

[FWB17] Marius Feldmann, Felix Walter, and Ricardo Böhm. “Routing in Ring Road Networks: Leveraging a Spot of Maximum Knowledge”. In: *IEEE International Conference on Wireless for Space and Extreme Environments (WiSEE)*. Montréal, Canada, 2017. DOI: 10.1109/WiSEE.2017.8124894.

[Wal15] Felix Walter. “An Approach to a DTN Routing Component for Ring Road Networks powered by CubeSats”. Studienarbeit (Minor Thesis). TU Dresden, 2015.

[Wal16] Felix Walter. “A System for Inference of Contact Patterns in Delay-Tolerant Networks”. Diploma Thesis. TU Dresden, 2016.

[Wal17] Felix Walter. “Enhancing First-hop Probability Estimations in Ring Road Networks via Node Collaboration”. In: *Space Terrestrial Internetworking Publication Workshop (STINT-PUBS), IEEE International Conference on Wireless for Space and Extreme Environments (WiSEE)*. Montréal, Canada, 2017. DOI: 10.1109/WiSEE.2017.8124905.

[WF16] Felix Walter and Marius Feldmann. “Dynamic Discovery of Ground Stations in Ring Road Networks”. In: *IEEE International Conference on Wireless for Space and Extreme Environments (WiSEE)*. Aachen, Germany, 2016. DOI: 10.1109/WiSEE.2016.7877311.

[WF18] Felix Walter and Marius Feldmann. “Contact Capacity Prediction in Ring Road Networks”. In: *IEEE International Conference on Wireless for Space and Extreme Environments (WiSEE)*. Huntsville, AL, USA, 2018. DOI: 10.1109/WiSEE.2018.8637349.

[WF19] Felix Walter and Marius Feldmann. “Leveraging Probabilistic Contacts in Contact Graph Routing”. In: *IEEE Global Communications Conference (GLOBECOM)*. Waikoloa, HI, USA, 2019. DOI: 10.1109/GLOBECOM38437.2019.9014158.

Bibliography

[Ara+15] Giuseppe Araniti, Nikolaos Bezirgiannidis, Edward Birrane, Igor Bisio, Scott Burleigh, Carlo Caini, Marius Feldmann, Mario Marchese, John Segui, and Kiyohisa Suzuki. “Contact Graph Routing in DTN Space Networks: Overview, Enhancements and Performance”. In: *IEEE Communications Magazine* 53.3 (2015), pp. 38–46. DOI: 10.1109/MCOM.2015.7060480.

[Bas+18] M. Basharat, W. Ejaz, M. Naeem, A. M. Khattak, and A. Anpalagan. “A survey and taxonomy on nonorthogonal multiple-access schemes for 5G networks”. In: *Transactions on Emerging Telecommunications Technologies* 29.1 (2018). DOI: 10.1002/ett.3202.

[BB11] Scott C. Burleigh and Edward J. Birrane. “Toward a Communications Satellite Network for Humanitarian Relief”. In: *International Conference on Wireless Technologies for Humanitarian Relief (ACWR)*. Amritapuri, Kerala, India, 2011, pp. 219–224. DOI: 10.1145/2185216.2185280.

[BCT16] N. Bezirgiannidis, C. Caini, and V. Tsaoussidis. “Analysis of contact graph routing enhancements for DTN space communications”. In: *International Journal of Satellite Communications and Networking* 34.5 (2016), pp. 695–709. DOI: 10.1002/sat.1138.

[Ber+17] A. Berlati, S. Burleigh, C. Caini, F. Fiorini, J. J. Messina, S. Pozza, M. Rodolfi, and G. Tempesta. “Implementation of (O-)CGR in The ONE”. In: *International Conference on Space Mission Challenges for Information Technology (SMC-IT)*. Alcalá de Henares, Spain, 2017, pp. 132–135. DOI: 10.1109/SMC-IT.2017.30.

[BFB20] S. Burleigh, K. Fall, and E. Birrane. *Bundle Protocol Version 7*. Internet Draft draft-ietf-dtn-bpbis-24. Intended status: Standards Track. Internet Engineering Task Force (IETF), 2020. URL: https://tools.ietf.org/html/draft-ietf-dtn-bpbis-24.

[Bir12] Edward J. Birrane. “Building routing overlays in disrupted networks: inferring contacts in challenged sensor internetworks”. In: *International Journal of Ad Hoc and Ubiquitous Computing* 11.2 (2012), pp. 139–156. DOI: 10.1504/IJAHUC.2012.050271.

[BLV07] Aruna Balasubramanian, Brian Levine, and Arun Venkataramani. “DTN Routing as a Resource Allocation Problem”. In: *Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*. Kyoto, Japan: Association for Computing Machinery, 2007, pp. 373–384. DOI: 10.1145/1282380.1282422.

[Bur+06] John Burgess, Brian Gallagher, David Jensen, and Brian Neil Levine. “MaxProp: Routing for Vehicle-Based Disruption-Tolerant Networks”. In: *Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*. Barcelona, Spain, 2006. DOI: 10.1109/INFOCOM.2006.228.

[Bur10] Scott Burleigh. *Contact Graph Routing*. Internet Draft draft-burleigh-dtnrg-cgr-01. Experimental. Internet Research Task Force (IRTF), 2010. URL: https://tools.ietf.org/html/draft-burleigh-dtnrg-cgr-01.

[Bur+16] S. Burleigh, C. Caini, J. J. Messina, and M. Rodolfi. “Toward a Unified Routing Framework for Delay-Tolerant Networking”. In: *IEEE International Conference on Wireless for Space and Extreme Environments (WiSEE)*. Aachen, Germany, 2016, pp. 82–86. DOI: 10.1109/WiSEE.2016.7877309.

[Cai+11] Carlo Caini, Haitham Cruickshank, Stephen Farrell, and Mario Marchese. “Delay- and Disruption-Tolerant Networking (DTN): An Alternative Solution for Future Satellite Networking Applications”. In: *Proceedings of the IEEE* 99.11 (2011), pp. 1980–1997. DOI: 10.1109/JPROC.2011.2158378.

[CCSDS15] Consultative Committee for Space Data Systems (CCSDS). *CCSDS Bundle Protocol Specification*. CCSDS 734.2-B-1. Washington, DC, USA, 2015.

[CCSDS17] Consultative Committee for Space Data Systems (CCSDS). *TM Synchronization and Channel Coding*. CCSDS 131.0-B-3. Washington, DC, USA, 2017.

[CCSDS19] Consultative Committee for Space Data Systems (CCSDS). *Schedule-Aware Bundle Routing*. CCSDS 734.3-B-1. Washington, DC, USA, 2019.

[CL16] Honglong Chen and Wei Lou. “Contact Expectation Based Routing for Delay Tolerant Networks”. In: *Ad Hoc Networks* 36.P1 (2016), pp. 244–257. DOI: 10.1016/j.adhoc.2015.07.017.

[CMP15] Marco Cello, Mario Marchese, and Fabio Patrone. “HotSel: A Hot Spot Selection Algorithm for Internet Access in Rural Areas through Nanosatellite Networks”. In: *IEEE Global Communications Conference (GLOBECOM)*. San Diego, CA, USA, 2015. DOI: 10.1109/GLOCOM.2014.7417202.

[CMP16a] Marco Cello, Mario Marchese, and Fabio Patrone. “ColdSel: A Selection Algorithm to mitigate congestion situations over Nanosatellite Networks”. In: *IEEE Global Communications Conference (GLOBECOM)*. Washington, DC, USA, 2016. DOI: 10.1109/GLOCOM.2016.7841670.

[CMP16b] Marco Cello, Mario Marchese, and Fabio Patrone. “SatSel: A Satellite Selection Algorithm to reduce delivery time in DTN-Nanosatellite Networks for Internet Access in Rural Areas”. In: *8th Advanced Satellite Multimedia Systems Conference and the 14th Signal Processing for Space Communications Workshop, ASMS/SPSC 2016*. Palma de Mallorca, Spain, 2016.

[CS13] Yue Cao and Zhili Sun. “Routing in delay/disruption tolerant networks: A taxonomy, survey and challenges”. In: *IEEE Communications Surveys and Tutorials* 15.2 (2013), pp. 654–677. DOI: 10.1109/SURV.2012.042512.00053.

[CSH15] N.H. Crisp, K. Smith, and P. Hollingsworth. “Launch and deployment of distributed small satellite systems”. In: *Acta Astronautica* 114 (2015), pp. 65–78. DOI: 10.1016/j.actaastro.2015.04.015.

[DeJ19] O. De Jonckère. “Efficient Contact Graph Routing Algorithms for Unicast and Multicast Bundles”. In: *IEEE International Conference on Space Mission Challenges for Information Technology (SMC-IT)*. Pasadena, CA, USA, 2019, pp. 87–94. DOI: 10.1109/SMC-IT.2019.00016.

[DH07] Elizabeth M. Daly and Mads Haahr. “Social Network Analysis for Routing in Disconnected Delay-tolerant MANETs”. In: *ACM International Symposium on Mobile Ad Hoc Networking and Computing*. Montréal, Canada, 2007, pp. 32–40. DOI: 10.1145/1288107.1288113.

[Dhu+13] S. K. Dhurandher, D. K. Sharma, I. Woungang, and S. Bhati. “HBPR: History Based Prediction for Routing in Infrastructure-less Opportunistic Networks”. In: *IEEE International Conference on Advanced Information Networking and Applications (AINA)*. Barcelona, Spain, 2013, pp. 931–936. DOI: 10.1109/AINA.2013.105.

[Dia+17] R. Diana, E. Lochin, L. Franck, C. Baudoin, E. Dubois, and P. Gelard. “DTN routing for quasi-deterministic networks with application to LEO constellations”. In: *International Journal of Satellite Communications and Networking* 35.2 (2017), pp. 91–108. DOI: 10.1002/sat.1159.

[DP18] R. Dudukovich and C. Papachristou. “Delay Tolerant Network Routing as a Machine Learning Classification Problem”. In: *NASA/ESA Conference on Adaptive Hardware and Systems (AHS)*. Edinburgh, UK, 2018, pp. 96–103. DOI: 10.1109/AHS.2018.8541460.

[DV10] Amit Dvir and Athanasios V. Vasilakos. “Backpressure-based Routing Protocol for DTNs”. In: *ACM SIGCOMM Conference*. New Delhi, India, 2010, pp. 405–406. DOI: 10.1145/1851182.1851233.

[EM14] David Evans and Mario Merri. “OPS-SAT: A ESA nanosatellite for accelerating innovation in satellite control”. In: *International Conference on Space Operations (SpaceOps)*. Pasadena, CA, USA, 2014. DOI: 10.2514/6.2014-1702.

[Fra+17] J. A. Fraire, P. Madoery, S. Burleigh, M. Feldmann, J. Finochietto, A. Charif, N. Zergainoh, and R. Velazco. “Assessing Contact Graph Routing Performance and Reliability in Distributed Satellite Constellations”. In: *Journal of Computer Networks and Communications* 2017 (2017). DOI: 10.1155/2017/2830542.

[Fra+18] Juan A. Fraire, Pablo G. Madoery, Amir Charif, and Jorge M. Finochietto. “On route table computation strategies in Delay-Tolerant Satellite Networks”. In: *Ad Hoc Networks* 80 (2018), pp. 31–40. ISSN: 1570-8705. DOI: 10.1016/j.adhoc.2018.07.002.

[Gra+11] Samo Grasic, Elwyn Davies, Anders Lindgren, and Avri Doria. “The Evolution of a DTN Routing Protocol - PRoPHETv2”. In: *Proceedings of the 6th ACM Workshop on Challenged Networks*. Las Vegas, NV, USA, 2011, pp. 27–30. DOI: 10.1145/2030652.2030661.

[GT02] M. Grossglauser and D. N. C. Tse. “Mobility increases the capacity of ad hoc wireless networks”. In: *IEEE/ACM Transactions on Networking* 10.4 (2002), pp. 477–486. DOI: 10.1109/TNET.2002.801403.

[GW09] John S. Gilmore and Riaan Wolhuter. “Predicting Low Earth Orbit Satellite Communications Quality and Visibility over Time”. In: *Southern African Telecommunication Networks and Applications Conference (SATNAC)*. Swaziland, 2009.

[HCY11] P. Hui, J. Crowcroft, and E. Yoneki. “BUBBLE Rap: Social-Based Forwarding in Delay-Tolerant Networks”. In: *IEEE Transactions on Mobile Computing* 10.11 (2011), pp. 1576–1589. DOI: 10.1109/TMC.2010.246.

[HSL10] Theus Hossmann, Thrasyvoulos Spyropoulos, and Franck Legendre. “Know Thy Neighbor: Towards Optimal Mapping of Contacts to Social Graphs for DTN Routing”. In: *Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*. San Diego, CA, USA, 2010. DOI: 10.1109/INFCOM.2010.5462135.

[Ipp17] Louis J. Ippolito Jr. *Satellite Communications Systems Engineering: Atmospheric Effects, Satellite Link Design and System Performance*. Second Edition. John Wiley and Sons Ltd, 2017. DOI: 10.1002/9781119259411.

[Iss+14] J.-L. Issler, A. Gaboriaud, F. Apper, A. Ressouche, D. Evans, O. Koudelka, P. Romano, M. Unterberger, T. Dehaene, B. Lechevalier, G. Guillois, and M. Fernandez. “CCSDS Communication Products in S and X Band for CubeSats”. In: *AIAA/USU Conference on Small Satellites*. Logan, UT, USA, 2014.

[JFP04] Sushant Jain, Kevin Fall, and Rabin Patra. “Routing in a Delay Tolerant Network”. In: *ACM SIGCOMM Computer Communication Review* 34.4 (2004), p. 145. DOI: 10.1145/1030194.1015484.

[Kam+14] Mohamed Kameche, Haider Benzeniar, Ayhane Bey Benbouzid, Redha Amri, and Nadir Bouanani. “Disaster Monitoring Constellation Using Nanosatellites”. In: *Journal of Aerospace Technology and Management* 6.1 (2014), pp. 93–100. DOI: 10.5028/jatm.v6i1.281.

[KLV10] T. Karagiannis, J. Le Boudec, and M. Vojnovic. “Power Law and Exponential Decay of Intercontact Times between Mobile Devices”. In: *IEEE Transactions on Mobile Computing* 9.10 (2010), pp. 1377–1390. DOI: 10.1109/TMC.2010.99.

[KOK09] Ari Keränen, Jörg Ott, and Teemu Kärkkäinen. “The ONE Simulator for DTN Protocol Evaluation”. In: *SIMUTools ’09: Proceedings of the 2nd International Conference on Simulation Tools and Techniques*. Rome, Italy, 2009. DOI: 10.4108/ICST.SIMUTOOLS2009.5674.

[Kru+08] C. Krupiarz, C. Belleme, D. Gherardi, and E. Birrane. “Using SmallSats and DTN for Communication in Developing Countries”. In: *International Astronautical Congress*. IAC-08.B4.1.8. Glasgow, Scotland, 2008.

[Kup+19] Vishnupriya Kuppusamy, Udaya Miriya Thanthrige, Asanga Udugama, and Anna Förster. “Evaluating Forwarding Protocols in Opportunistic Networks: Trends, Advances, Challenges and Best Practices”. In: *Future Internet* 11.5 (2019). DOI: 10.3390/fi11050113.

[LD05] A. Lindgren and A. Doria. *Probabilistic Routing Protocol for Intermittently Connected Networks*. Internet Draft draft-lindgren-dtnrg-prophet-00. Internet Research Task Force (IRTF), 2005. URL: https://tools.ietf.org/html/draft-lindgren-dtnrg-prophet-00.

[LS+19] Julian Lopez-Salamanca, Laio O. Seman, Marcelo D. Berejuck, and Eduardo A. Bezerra. "Finite-State Markov Chains Channel Model for CubeSats Communication Uplink". In: *IEEE Transactions on Aerospace and Electronic Systems* (2019). Early Access. DOI: 10.1109/TAES.2019.2911769.

[Mad+18] Pablo G. Madoery, Juan A. Fraire, Fernando D. Raverta, Jorge M. Finochietto, and Scott C. Burleigh. "Managing Routing Scalability in Space DTNs". In: *IEEE International Conference on Wireless for Space and Extreme Environments (WiSEE)*. Huntsville, AL, USA, 2018, pp. 177–182. DOI: 10.1109/WiSEE.2018.8637324.

[Mat+19] Brian Mathason, Michael M. Albert, Doruk Engin, He Cao, Keith G. Petrillo, Jacob Hwang, Khoa Le, Kent Puffenberger, Slava Litvinovitch, Mark Storm, and Richard Utano. "CubeSat lasercom optical terminals for near-Earth to deep space communications". In: *Free-Space Laser Communications XXXI*. San Francisco, CA, USA, 2019, pp. 24–29. DOI: 10.1117/12.2508047.

[MF09] Alex McMahon and Stephen Farrell. "Delay- and Disruption-Tolerant Networking". In: *IEEE Internet Computing* 13.6 (2009), pp. 82–87. DOI: 10.1109/MIC.2009.127.

[MM09] Mirco Musolesi and Cecilia Mascolo. "CAR: Context-Aware Adaptive Routing for Delay-Tolerant Mobile Networks". In: *IEEE Transactions on Mobile Computing* 8.2 (2009), pp. 246–260. DOI: 10.1109/TMC.2008.107.

[NAS05] NASA. *Deep Impact Launch Press Kit*. 2005. URL: https://www.jpl.nasa.gov/news/press_kits/deep-impact-launch.pdf.

[NB18] Marc Sanchez Net and Scott Burleigh. "Evaluation of Opportunistic Contact Graph Routing in Random Mobility Environments". In: *IEEE International Conference on Wireless for Space and Extreme Environments (WiSEE)*. Huntsville, AL, USA, 2018, pp. 183–188. DOI: 10.1109/WiSEE.2018.8637316.

[PD12] Larry Peterson and Bruce Davie. *Computer Networks: A Systems Approach*. Version 6.1. License: CC BY 4.0. Elsevier, 2012. URL: https://github.com/SystemsApproach.

[PGR17] S. Pathak, N. Gondaliya, and N. Raja. "A survey on PROPHET based routing protocol in delay tolerant network". In: *International Conference on Emerging Trends Innovation in ICT (ICEI)*. Pune, India, 2017, pp. 110–115. DOI: 10.1109/ETIICT.2017.7977020.

[PP05] E. Papapetrou and F.-N. Pavlidou. "Analytic study of Doppler-based handover management in LEO satellite systems". In: *IEEE Transactions on Aerospace and Electronic Systems* 41.3 (2005), pp. 830–839. DOI: 10.1109/TAES.2005.1541433.

[RFC1122] R. Braden (Editor). *Requirements for Internet Hosts - Communication Layers*. Request for Comments. Internet Standard. Internet Engineering Task Force (IETF), 1989. URL: http://www.rfc-editor.org/rfc/rfc1122.txt.

[RFC1812] F. Baker (Editor). *Requirements for IP Version 4 Routers*. Request for Comments. Proposed Standard. Internet Engineering Task Force (IETF), 1995. URL: http://www.rfc-editor.org/rfc/rfc1812.txt.

[RFC3168] K. Ramakrishnan, S. Floyd, and D. Black. *The Addition of Explicit Congestion Notification (ECN) to IP*. Request for Comments. Proposed Standard. Internet Engineering Task Force (IETF), 2001. URL: http://www.rfc-editor.org/rfc/rfc3168.txt.

[RFC4271] Y. Rekhter (Editor), T. Li (Editor), and S. Hares (Editor). *A Border Gateway Protocol 4 (BGP-4)*. Request for Comments. Draft Standard. Internet Engineering Task Force (IETF), 2006. URL: http://www.rfc-editor.org/rfc/rfc4271.txt.

[RFC4838] Vint Cerf, Scott C. Burleigh, Adrian J. Hooke, Leigh Torgerson, Robert C. Durst, Keith L. Scott, Kevin Fall, and Howard S. Weiss. *Delay-Tolerant Networking Architecture*. Request for Comments. Informational. Internet Research Task Force (IRTF), 2007. URL: http://www.rfc-editor.org/rfc/rfc4838.txt.

[RFC5050] K. Scott and S. Burleigh. *Bundle Protocol Specification*. Request for Comments. Experimental. Internet Research Task Force (IRTF), 2007. URL: http://www.rfc-editor.org/rfc/rfc5050.txt.

[RFC6693] A. Lindgren, E. Davies, S. Grasic, and A. Doria. *Probabilistic Routing Protocol for Intermittently Connected Networks*. Request for Comments. Experimental. Internet Research Task Force (IRTF), 2012. URL: http://www.rfc-editor.org/rfc/rfc6693.txt.

[RFC7049] Carsten Bormann and Paul Hoffman. *Concise Binary Object Representation (CBOR)*. Request for Comments. Internet Standard. Internet Engineering Task Force (IETF), 2013. URL: http://www.rfc-editor.org/rfc/rfc7049.txt.

[SJB11] J. Segui, E. Jennings, and S. Burleigh. "Enhancing Contact Graph Routing for Delay Tolerant Space Networking". In: *IEEE Global Telecommunications Conference (GLOBECOM)*. Houston, TX, USA, 2011. DOI: 10.1109/GLOCOM.2011.6134460.

[Skl17] Bernard Sklar. *Digital Communications: Fundamentals and Applications*. 2nd Edition. Prentice Hall, 2017. ISBN: 0134724054.

[SMM04] A. Sekhar, B. S. Manoj, and C. S. R. Murthy. "MARVIN: Movement-Aware Routing oVer Interplanetary Networks". In: *IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks (SECON)*. 2004, pp. 245–254. DOI: 10.1109/SAHCN.2004.1381923.

[SPR05] T. Spyropoulos, K. Psounis, and C. S. Raghavendra. "Spray and Wait: An Efficient Routing Scheme for Intermittently Connected Mobile Networks". In: *ACM SIGCOMM workshop on Delay-tolerant networking (WDTN)*. Philadelphia, PA, USA, 2005, pp. 252–259. DOI: 10.1145/1080139.1080143.

[Sys02] BAE Systems. *RAD750 Microprocessor Technical Overview*. 2002. URL: https://web.archive.org/web/20090326020946/http://www.aero.org/conferences/mrqw/2002-papers/A_Burcin.pdf.

[TW11] Andrew S. Tanenbaum and David J. Wetherall. *Computer Networks*. Fifth Edition. Pearson, 2011. ISBN: 0132126958.

[Val+06] David A. Vallado, Paul Crawford, Richard Hujsak, and TS Kelso. “Revisiting Spacetrack Report #3”. In: *AIAA/AAS Astrodynamics Specialist Conference*. Vol. 6753. AIAA 2006-6753. Keystone, CO, USA, 2006. DOI: 10.2514/6.2006-6753.

[VB00] A. Vahdat and D. Becker. *Epidemic Routing for Partially-Connected Ad Hoc Networks*. Tech. rep. CS-200006. 2000. DOI: 10.1.1.34.6151.

[Wan+16] Guosheng Wang, Scott C. Burleigh, Ruhai Wang, Leilei Shi, and Yi Qian. “Scoping Contact Graph-Routing Scalability”. In: *IEEE Vehicular Technology Magazine* 11.4 (2016), pp. 46–52. DOI: 10.1109/MVT.2016.2594796.

[Wan+17] Peng Wan, Shi Chen, Tao Yu, and Zhongjie Hua. “A hybrid multiple copy routing algorithm in space delay-tolerant networks”. In: *Science China Information Sciences* 60.4 (2017). DOI: 10.1007/s11432-015-0954-6.

[WEH09] L. Wood, W. M. Eddy, and P. Holliday. “A Bundle of Problems”. In: *IEEE Aerospace conference*. Big Sky, MT, USA, 2009, pp. 1–17. DOI: 10.1109/AERO.2009.4839384.

[Whi+18] Dan White, Corey Shields, Pierros Papadeas, Agisilaos Zisimatos, Manolis Surligas, Matthaios Papamatthaiou, Dimitrios Papadeas, Eleytherios Kosmas, Vasileios Tsiligiannis, Alexandru Csete, Nikolaos Roussos, Alfredos-Panagiotis Damkalis, Konstantinos Triantafyllakis, and Georgios Vardakis. “Overview of the Satellite Networked Open Ground Stations (SatNOGS) Project”. In: *AIAA/USU Conference on Small Satellites*. Logan, UT, USA, 2018.

[Wya+09] Jay Wyatt, Scott Burleigh, Ross Jones, Leigh Torgerson, and Steve Wissler. “Disruption Tolerant Networking Flight Validation Experiment on NASA’s EPOXI Mission”. In: *International Conference on Advances in Satellite and Space Communications (SPACOMM)*. Colmar, France, 2009, pp. 187–196. DOI: 10.1109/SPACOMM.2009.39.

[Xue+09] J. Xue, X. Fan, Y. Cao, J. Fang, and J. Li. “Spray and Wait Routing Based on Average Delivery Probability in Delay Tolerant Network”. In: *International Conference on Networks Security, Wireless Communications and Trusted Computing*. Vol. 2. Wuhan, Hubei, China, 2009, pp. 500–502. DOI: 10.1109/NSWCTC.2009.284.

[Zha06] Zensheng Zhang. “Routing in Intermittently Connected Mobile Ad Hoc Networks and Delay Tolerant Networks: Overview and Challenges”. In: *IEEE Communications Surveys Tutorials* 8.1 (2006), pp. 24–37. DOI: 10.1109/COMST.2006.323440.

[Zha+17] Lichen Zhang, Xiaoming Wang, Junling Lu, Meirui Ren, Zhuojun Duan, and Zhipeng Cai. “A novel contact prediction-based routing scheme for DTNs”. In: *Transactions on Emerging Telecommunications Technologies* 28.1 (2017). DOI: 10.1002/ett.2889.

[Zho+17] Huan Zhou, Linping Tong, Tingyao Jiang, Shouzhi Xu, Jialu Fan, and Ke Lv. “Maximum data delivery probability-oriented routing protocol in opportunistic mobile networks”. In: *Peer-to-Peer Networking and Applications* 10.3 (2017), pp. 500–509. DOI: 10.1007/s12083-016-0512-x.