Utilizing Public Transport Networks for Bulk Data Transfer

Ayesha Ahmad

Helsinki May 14, 2019

UNIVERSITY OF HELSINKI
Department of Computer Science

HELSINGIN YLIOPISTO — HELSINGFORS UNIVERSITET — UNIVERSITY OF HELSINKI

Tiedekunta — Fakultet — Faculty	Laitos — Institution — Department
Faculty of Science	Department of Computer Science

Tekijä — Författare — Author
Ayesha Ahmad

Työn nimi — Arbetets titel — Title
Utilizing Public Transport Networks for Bulk Data Transfer

Oppiaine — Läroämne — Subject
Computer Science

Työn laji — Arbetets art — Level	Aika — Datum — Month and year	Sivumäärä — Sidoantal — Number of pages
	May 14, 2019	63 pages + 0 appendices

Tiivistelmä — Referat — Abstract

Public transport networks are a subset of vehicular networks with some important distinctions; the actors in the network include buses and bus-stops, they are predictable and they provide reliable physical coverage of an area. The Public Transport Network of a city can also be interpreted as an opportunistic network where nodes are bus-stops and communication between these nodes occurs when a bus travels between two bus-stops. How will a data communication network perform when built upon the opportunistic network formed by the public transport system of a city? In this thesis we explore this question basing our analysis on Helsinki Region's public bus transport system as a real example. We explore the performance of a public transport network when used for communication of data using both simulation of the network and graph analysis. The key performance factors studied are the data delivery ratio and data delivery time. Additional issues considered are the kind of applications such a system is suited for, the important characteristics governing the reliability and efficacy of such a data communications system, and the design guidelines for building such an application. The results demonstrate that data transfer applications can be built over a city's Public Transport Network.

ACM Computing Classification System (CCS):
C.2.1 [Computer-Communication Networks]: Network Architecture and Design - Store and forward networks, Wireless communication;
C.4 [Performance of Systems]: Modeling techniques;
D.2.8 [Software Engineering]: Metrics - performance measures;

Avainsanat — Nyckelord — Keywords
Delay Tolerant Networks, Public Transport Networks, Graph Analysis, Simulation

Säilytyspaikka — Förvaringsställe — Where deposited

Muita tietoja — övriga uppgifter — Additional information

Thesis for the Networking and Services subprogram

Contents

Abbreviations

BDT Bulk Data Transfer

DTN Delay Tolerant Networking

ISP Internet Service Provider

MANET Mobile Adhoc Network

MNO Mobile Network Operator

PTN Public Transport Network

PTDTN Public Transport Delay Tolerant Network

SAG Static Aggregated Graph

TVG Time Varying Graph

VANET Vehicular Adhoc Network

1 Introduction

The proliferation of mobile devices and wireless communication has created new types of networks that have different characteristics from the static, unchanging networks of the pre-wireless era. While networks used to be fairly static entities made up of physically tangible parts like phone lines, cables and wires connecting computers that did not move from their fixed locations - a 'hardcoded' network so to speak, present day networks are highly fluid, the physical, static part still exists but is now complemented by a potentially even larger network consisting of mobile devices which connect over wireless connections to the Internet. The reliable end-to-end paths of a stationary network are no longer a given, it is possible, and even highly likely that communication is occurring over both wired and wireless connections between mobile and stationary devices.

Communication can occur between mobile nodes even in the absence of cellular or wireless network; the devices only have to be within communicating distance in order to exchange data. When communicating nodes are mobile, and in the absence of a permanent physical infrastructure that connects them, there are going to be situations where nodes are unreachable for some time or cannot be found at all. Communication protocols developed for such an environment must take into account the challenges of temporal disconnect, and the possibility of data experiencing delays or even never reaching its destination. As an end-to-end connection is absent, communications will have a probabilistic nature rather than deterministic, and movement of data must be able to leverage the movement of nodes themselves as there is no guarantee of a cellular or wireless network being available. A class of networking protocols and applications has been developed based on the above-mentioned constraints and is described as Delay Tolerant Networking [Fal03].

Delay Tolerant Networking is a communications architecture for systems that exhibit intermittent connectivity and where an end-to-end path may not exist between the source and destination. This type of architecture is suitable for wireless communication systems like mobile and sensor networks, and systems involving communication over unreliable links with high-delay such as deep-space and ocean communication. DTNs are also often described as opportunistic networks because of the opportunistic way in which data is transferred when nodes come into contact. A special class of delay tolerant networks are Vehicular Adhoc Networks (VANETs). In such networks vehicles can exchange messages when they come physically close to one another, messages travel from node to node opportunistically until they reach their

destination.

This thesis investigates a specific delay tolerant networking application built over a Vehicular Adhoc Network environment, namely Bulk Data Transfer (BDT) over a Public Transport Network (PTN). It explores whether a data transfer network built over the framework of a Public Transport Network of an urban center can be used to provide reliable bulk data delivery throughout the network. The problem domain is that of Opportunistic Networks, Vehicular Networks, Delay Tolerant Networks and Static and Temporal Network Analysis. In addition to bulk data transfer over a public transport system, this topic has applications in the areas of cellular offloading and alternative or independent networks to the Internet.

Public transport networks are a subset of VANETs with some distinctive properties; the actors in the network include buses and bus-stops, the movement of nodes (buses) is regular and predictable, and the public transport network provides reliable physical coverage of an area. Public Transport Networks exist in almost all urban centers and provide a network of buses, and sometimes trains and subways, allowing people to move easily and efficiently between any two points in an urban area. These networks are designed to be reliable and periodic and they aim to provide full coverage of the urban area they are serving. If long-distance trains and bus services are included such a network can extend over an entire country and sometimes even across borders providing physical connectivity over potentially large areas.

Concentrating only on urban centers, like cities, it can be observed that dense bus networks make up a large part of the PTN. The PTN of a city can be interpreted as an opportunistic network where nodes are bus-stops and communication between these nodes occurs when a bus travels between two bus-stops. Bus-stops are disconnected from one another and are only connected when a bus travels between them. The connecting bus may transfer information from the source stop to the destination stop. How well does such a network perform, what are the parameters that define good performance, and what kind of applications is such a network best suited for? How will a data communication application perform when built upon the opportunistic network formed by the public transport system of a city? In this thesis we explore these questions basing our analysis on Helsinki Region's public bus transport system as a real example.

This research involves simulation and graph analysis of Helsinki region's public transport network. The primary method of evaluation is simulation of the network supported by graph analysis of the network using static and temporal network analysis.

1.1 Problem Statement

Before going on to discuss the research objectives, it is important to understand the problem that is being solved by such an application. Why would anyone consider transferring large amounts of data using any other method than a secure Internet connection from the comfort of their home? Certainly, the home user is able to produce and consume gigabytes of data, but the stumbling block becomes apparent when they want to move that data to the cloud.

Internet connections are asymmetric with high download speeds but low upload speeds. This is based on the assumption that users will download more than they will upload. But, with the explosion of cloud services and increase in image and video file sizes and traffic, it can be predicted that upload traffic will only increase. Uploading large files such as when taking a back-up of media files will take unacceptably long times on a home network. This is sometimes referred to as the 'last mile' problem where the bottleneck in the communication path from the user to the cloud is the user's connection to their Internet Service Provider.

Table 1: Internet Network Hierarchy Data Rates

Level	**Connects to**	**Data Rate**	**Connection**
Internet backbone	Backbones	Tbps	Fiber optic cables
ISP	Internet	Mbps - Gbps	Fiber optic cables
Home user	ISP	<= 100 Mbps	Broadband
Mobile user 3G	Internet	>= 0.2 Mbps[a]	3G wireless network
Mobile user 4G	Internet	100 Mbps - 1 Gbps[b]	4G wireless network

[a]https://en.wikipedia.org/wiki/3G
[b]https://en.wikipedia.org/wiki/4G

Table 1 highlights the differences in speeds available at different tiers of the Internet. Home and mobile users have the smallest data rates; in practice, the data rate for these users is usually lower than the limits described in the table. For example, Akamai's Connectivity Report [Aka17] describes the average download speeds in Finland as 20.5 Mbps (Megabits per second), well below the 100 Mbps provided by fixed broadband connections. To illustrate the difference in download and upload speeds; the speedtest[1] service recorded 49.35 Mbps and 21.36 Mbps download and

[1]http://www.speedtest.net/global-index/finland

upload speeds respectively for fixed broadband connections, and 37.06 Mbps and 13.07 Mbps for mobile users in Finland.

ISPs use rate limiting in order to share available bandwidth between all of their users. A user that is transferring large files will experience an initial spike in transfer rate followed by a drop as the ISP lowers bandwidth allocated to them. It is even more difficult to transfer large files from a mobile device to the cloud while on the go. A mobile user has two options to transfer information; public WiFi and mobile data. There are no security guarantees when using public WiFi and it has the same rate-limiting issues as a home connection. The alternative of using cellular data is not attractive either as it is expensive. The proposed **Bulk Data Transfer application over a Public Transport Network** (or **'BDT App'** for short) provides users with another alternative to move data to and from the cloud.

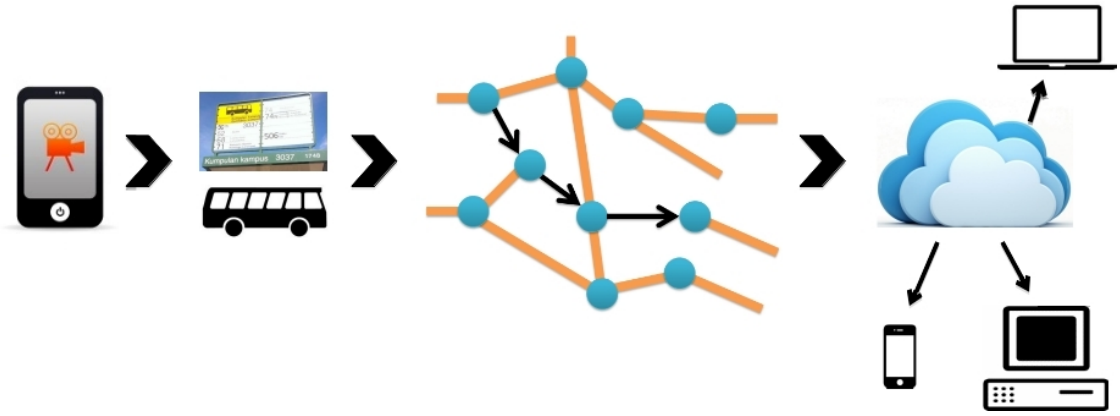
1.2 Bulk Data Transfer App

Use cases like backing up data and downloading data in bulk do not have real-time time requirements, it is enough that the data will be delivered within a reasonable amount of time. This lack of strict time constraints aligns BDT applications well with delay tolerant networking architectures. A user could offload their data back-up or large file transfer task to the public transport system when they encounter a bus or bus-stop and then be confident that the data will eventually reach its cloud destination. In practice this would mean that the buses in the transport system will physically move data from bus-stop to bus-stop, and data can make transfers from buses to stops and back to buses in order to reach its target. The buses and stops can be provisioned with storage so that data is not lost. They will also need to be fitted with hardware to allow short-range, high speed wireless transfers.

Figure 1 illustrates how a data transfer application built on public transport could work in practice. Suppose a person is walking around the city and records a video on their phone that they want to upload. If the video is large it is going to take some time to upload it to the cloud and the transfer will need either the user's mobile data connection or WiFi if accessible in the vicinity. In either case, bandwidth is limited and upload speeds are low causing long transfer times. A third option is for the person to walk to the nearby bus stop and use a high-speed WiFi connection to 'drop off' their video, then the bus network ensures that the data reaches the cloud.

BDT appears to be a promising application that can be deployed successfully on

Figure 1: User Offloading Video to Bus Network



a delay tolerant network built from the public transport system of a city. It complies with many of the properties for a 'killer app' for delay tolerant networking architectures as laid out in [LH09]. These properties are described below along with an explanation of how they are satisfied in the scenario under consideration in this thesis:

- providing the user with some benefit; the user can offload large transfers to the PTN and save time and mobile data costs
- favorable conditions for the app to be preferred; these conditions include low upload speeds, and paid mobile data plans which can both cause users to gravitate towards the BDT over PTN app
- and the existence of enabling technologies that make implementing the app feasible; these technologies are available such as high speed short-range WiFi connections, mobile nodes with unlimited battery (on buses and bus-stops) and are described in more detail in later sections

Other applications that can utilize a PTN as a data transmission network are *Cellular Offloading*, *Internet for Remote Areas*, and *Independent Communications Network*. *Cellular Offloading* refers to the diversion of data that would have normally traveled via a wireless 3G/4G network onto a wired network instead. In this way Mobile Network Operators (MNOs) can reduce the load on their cellular data networks. Several methods by which cellular offloading can be achieved are discussed in [WLW+17], and [DHHL11] demonstrate by simulation how much data can be offloaded from a 3G network. In the *Internet for Remote Areas* application, the PTN

connecting rural areas to urban areas can be used to provide access to the Internet. The buses act as data mules and carry requests from rural centers to the urban center that has Internet access, and carry back the requested data to the rural area. This topic has been studied in [PFH04] and [GVOM13]. An *Independent Communications Network* would allow residents linked by the PTN to exchange messages in a decentralized manner without the need for the Internet. This would include applications like a city-wide bulletin board, or targeted advertising. [HBDM14] presents a programming framework that can support the development of applications that run over the Public Transport System.

1.3 Research Methodology & Objectives

The research hypothesis for this thesis is that a data transfer network built over the framework of a Public Transport Network of an urban center can be used to provide reliable bulk data delivery throughout the network. The chosen method for testing this hypothesis is primarily network simulation supported by graph analysis of the network using static and temporal network analysis. The work includes designing and implementing a simulation of Helsinki Region's public transport system and modeling the transmission of data through this network. As explained in the previous section, the fact that PTNs already provide good coverage of an urban area and are reliable and predictable makes them an appropriate candidate for use in data transfer. Using a simulation provides the flexibility of modifying simulation parameters to test different routing algorithms and network conditions to examine how they affect the throughput of data in the system.

The significant questions when studying any network for transporting data are related to latency, throughput and data loss. This study aims to provide bounds for these metrics based on the various simulation parameters. It provides insight into the structure of the network and consequently give some rules that can be used to decide how the network should be constructed for optimum results. In short, the major research questions are:

- What is the maximum data capacity of this network?
- What is the data delivery ratio?
- What are the data delivery times?
- What are the performance bottlenecks in such a network?

- Is the network reliable?
- What kind of applications is this kind of network best suited for?

1.4 Thesis Outline

The introduction section has detailed the problem domain and the rationales for pursuing this research topic. The next two sections will introduce the background information regarding Delay Tolerant Networking, and Network Analysis for analyzing networks represented as graphs and relate it to the BDT over PTN application. Section 4 describes the related work done in this subject area, section 5 presents the results of the Static and Temporal Analysis of Helsinki's public transport network. Section 6 discusses the simulation and simulation results, and finally Section 7 presents the conclusions of this study.

2 Delay Tolerant Networking

In this section delay tolerant networks are described in more detail, their associations with other networking architectures with which they share overlapping features and applications is discussed, such as MANETs, Opportunistic Networks (ONs), and VANETs. The differences between these networking models are clarified and the relationship of the BDT App with respect to these architectures is made clear. This section will also give a summary of routing protocols used in DTNs and an overview of the types of applications that are suited to be deployed over Delay Tolerant Networks. Topics related to energy efficiency, security & quality of service in DTNs are not directly addressed in the BDT App and so will not be discussed in this thesis.

2.1 Terminology

Often the term DTN is mentioned in the same context as Mobile Ad-hoc Networks, Opportunistic Networks, and Vehicular Ad-hoc networks[2]. There are many com-

[2]There are many more acronyms for specifying the each type of network in this area, for example, Vehicular Delay Tolerant Network (VDTN) specifically defines a VANET which also experiences network partitioning, and Vehicle to Infrastructure (V2I) defines a network that includes roadside infrastructure. To simplify the terminology in this report, this study will define the meaning of

monalities between these networking paradigms but there are slight differences that should be highlighted. Table 2 attempts to summarize the different properties of these networks.

Table 2: Summary of Network Properties

Property	**MANET**	**DTN**	**VANET**
Node movement	low	low - high	high
Node distribution	dense	low-medium density	low density
Routing decisions	end-to-end route	per-hop	opportunistic
Connectivity	continuous	intermittent	intermittent

The concept of **Mobile Ad-hoc Network** (MANET) was formalized in RFC 2501 in 1999 [CM99]. A MANET is made up of mobile nodes that are connected via wireless links and each node has routing capabilities. It is possible to perform end-to-end routing in MANETs and several routing protocols exist, for example OLSR [CJ03] and AODV [PBRD03] among others. However these protocols do not handle network disconnection and will drop a packet if the next hop is unreachable [Zha06].

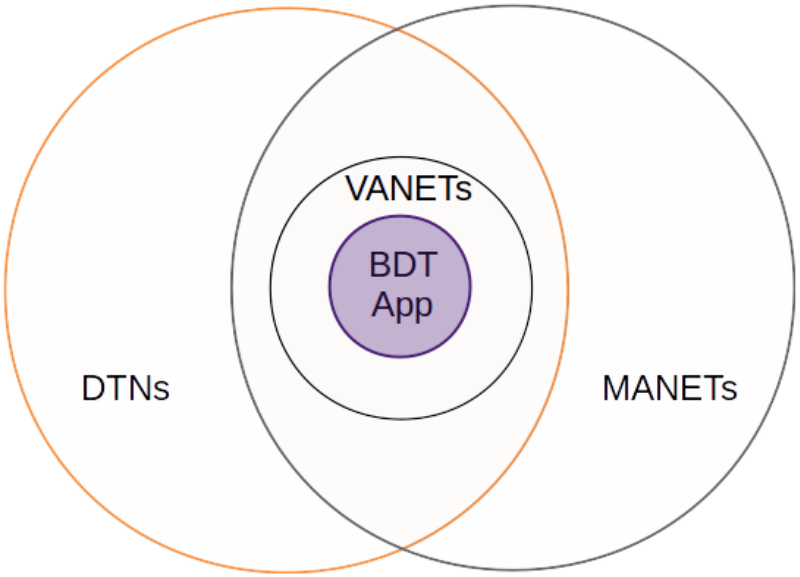
Opportunistic Networking is a more general concept than DTN, here there is no knowledge of the network topology and routing decisions are taken on a per-hop basis using local information and end-to-end paths between nodes may not always exist [PPC06]. ONs are also MANETs as they consist of mobile wireless nodes but differ from them in the lack of information about the network topology. Thus ONs have quite a broad definition and DTNs can be considered a subset of Opportunistic Networks. The terms Delay Tolerant Network and Opportunistic Network are often used interchangeably in the literature.

A **Vehicular Ad-hoc Network** is a MANET in the sense that it has mobile nodes (vehicles enabled with wireless communication) but is also a DTN as these nodes can communicate only when they are close enough to each other and are otherwise disconnected. Thus a VANET is a subset of a MANET and a DTN where the mobile nodes include vehicles.

Figure 2 shows where the BDT App lies with respect to the different network models. The BDT App has features from a DTN, a MANET, and a VANET. It is a VANET because the nodes include buses, it is a MANET because these nodes are mobile,

any acronym used. DTN related acronyms are specified in this section.

Figure 2: BDT App in Relation to Networking Architectures



and it is a DTN because the nodes in the network are not always reachable from one another at all times.

2.2 Definition

Delay Tolerant Networking was first introduced by Kevin Fall in 2003 [Fal03]. The paper introduced the idea that there are certain network environments or applications where traditional connection-oriented communication protocols would achieve very poor results. These 'challenged networks' suffer from network partitioning due to the movement or the short life-times of communicating nodes. In this network, an end-to-end path may not exist at any one instance in time between source and destination. This kind of network experiences disconnection due to missing nodes or missing linkages between nodes. It has high latency and low data rates due to missing links and low bandwidth communication mediums. And it has limited resources in terms of storage and energy [Fal03]. Such networks are often made up of heterogeneous nodes with varying storage, energy and transmission capabilities, for example a communication system including mobile phones, sensors and cars fitted with WiFi. Delay Tolerant Networking aims to enable delivery of messages in networks lacking continuous connectivity. Some examples of such environments/applications are

- space or deep-sea communication where distances are large, the risk of data

corruption is high and nodes may become unreachable because of periodic movements of planets or ships

- military applications where nodes might be equipment in the field that needs to communicate and those nodes may run out of power or be destroyed before a message reaches them
- mobile networks where nodes may be out of range of communication for some time, e.g. vehicular networks

DTNs overcome intermittent connectivity issues by being able to store data in transit in case the next hop is not available, once the link is available again, the node can forward the data. Therefore a *store-and-forward* mechanism must exist on the nodes in the network. In case the nodes are also mobile, then they may be able to carry the data to its next hop as well, this is called *store-carry-forward*. The storage requirement means that buffers must exist on DTN nodes.

In short, DTN networks are affected by two types of factors: The first factor includes constraints imposed by the resources available like **i.** buffer sizes on nodes, **ii.** energy usage by nodes and **iii.** the bandwidth available for communication between nodes. The second factor is related to problems caused by the mobility of the nodes themselves, such as breaks in connectivity between nodes and network partitioning, which bring routing challenges. These constraints are discussed in more detail in [ADC16, CS13].

DTN performance is most often evaluated using the metrics *delivery ratio* and *delivery delay*. The delivery ratio is the ratio of the amount of data that has reached its destination divided by the total amount of data that was injected into the network. The delivery delay is the time it has taken for the node to reach its destination. Other metrics have also been used in the literature such as buffer occupancy and energy consumption by nodes.

2.3 Routing Design

The most critical functionality for any communication network is the ability to route information through the network. As it is the foremost problem that needs to be solved for any network, there exists a plethora of research on routing in Delay Tolerant Networks. Recent survey papers discuss and compare different routing protocols in detail; [CS13] presents an overview of routing strategies for DTNs while

[TCCM15] specifically focuses on the protocols developed for VANETs. The fundamental ideas on routing in DTNs are presented in the seminal paper on routing in DTN [JFP04]. In this paper Jain et al lay out the decision points of a routing protocol for a DTN. These decisions include what metric(s) to optimize, how much information to use when computing routes, when to compute routes and whether to allow splitting of data. These ideas apply to all routing protocols for DTN as well as to the BDT App and are explained below:

Routing Objective: Routes are computed through the network based on minimizing or maximizing some metric. This metric can be delivery delay or delivery ratio or it could be some application specific cost like minimizing energy consumption. The BDT App computes routes with the objective of maximizing the delivery ratio and minimizing delivery delays.

Proactive vs Reactive Routing: In proactive routing, routes are computed once and remain static regardless of network changes occurring later. In reactive routing, routes are discovered by a node whenever traffic arrives that has a yet unseen destination. Reactive routing has the benefit of finding the optimum route at the time the data arrives but it takes longer to find the route and thus may increase the delay experienced by the data. The BDT App uses proactive routing because of its simplicity.

Source routing vs Per-hop routing: In source routing the route is computed once and then encoded into the data, the network then attempts to move the data according to this route. Even if the next hop in the route is unavailable and an alternate route exists or a better route appears, the source routing model cannot take advantage of it. In the per-hop routing case, a node can recompute the route and thus take advantage of latest changes in the network, however, this is computationally intensive in terms of space and time as more information must be computed and stored for each time interval. The BDT App uses source routing because of its simplicity.

Message Splitting: Any protocol must consider whether or not to allow message splitting. Message splitting can allow data to travel over multiple paths in case some paths are congested. But it requires the ability to compute multiple paths which is dependent on the path finding algorithm being used. Message splitting is not an option in the BDT App.

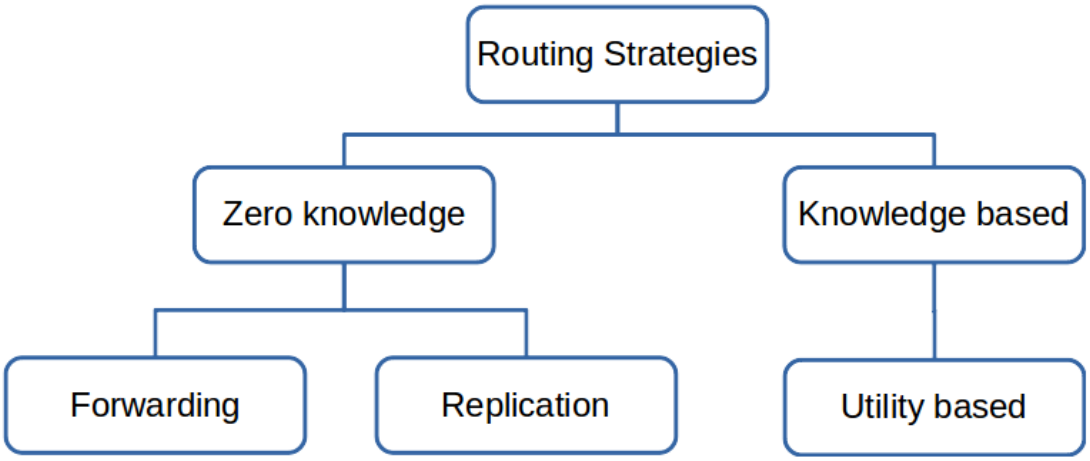
2.4 Routing Strategies

The routing strategy employed in a DTN depends a great deal on the type of application built on the system. Does the application require unicast messaging from one source to one destination? Does the application need to be able to send multicast messages? How much knowledge about the network is available? These considerations all affect the development of the routing algorithm that will give the best performance.

There are multiple ways to classify routing in DTNS. In [SRC11] the authors have classified routing protocols according to the nature of the network; whether it has a predictable topology (like a public transport network) or a dynamic topology that can be modeled stochastically (like people moving in the city center). In [dPSC16] the authors have classified routing based on how transfer of data between nodes occurs; using a deterministically determined route, or opportunistically transmitting to the first available encountered node and whether replication is used. And in [TCCM15] and [CS13] routing protocols have been categorized according to the mode of information dissemination whether unicast or multicast, and then at each further level based on the nature of the algorithm used.

Figure 3 broadly groups the routing strategies in terms of the amount of knowledge about the network and nodes. This classification provides enough granularity to discuss the general concepts in the different routing algorithms.

Figure 3: Routing in DTNs



- **Zero Knowledge:** No extra information is needed to select the next-hop node

  - *Forwarding:* These are suitable for unicast transmissions as there is no replication of data. The most basic implementation is that a node will transfer data only when it comes into contact with the destination node (the 'Direct Delivery' method [JFP04]). This is the simplest method but there are no guarantees that the data will ever reach its destination, i.e. it has unbounded delay. The only advantage is that there is only a single transmission of data thus conserving network resources. Direct Delivery relies on message forwarding, other forwarding based protocols include First Contact and Randomized Routing [SPR04]. In First Contact the node transfers data to the a randomly chosen node out of all of its immediate contacts, in Randomized Routing a probability is used to decide whether or not to forward the data. Both these algorithms perform better than Direct Delivery in terms of delay.
  - *Replication* Replication based protocols are suited to both unicast and multicast transmission of data. Examples of such protocols are Epidemic [VB+00] and Spray and Wait [SPR05]. Both employ flooding techniques in the hope that the packet will reach its target. These protocols spread messages quickly through a network but they may overload the network resources with replicated packets. There are many variations of these protocols that improve the network resource usage and are discussed in [CS13].

- **Knowledge Based:** These protocols use knowledge about other nodes and/or the network to select a node (or nodes) to which to forward or replicate data. So knowledge based protocols can also either forward or replicate data, the difference is that they select next-hop nodes on the basis of some 'utility' function that tries to compute the usefulness (or utility) of making the transfer. For example social network based and location based utility functions use the contact and location history of a node respectively for making routing decisions. Examples of such protocols are PRoPHET [LDS03] and Bubble Wrap [HCY11].

2.5 BDT App Design

The routing design and strategies for the BDT App depend on its particular requirements. The BDT App aims to route data to the cloud via a Public Transport

Network. The data travels from a single point to a known hub from where data can travel to the cloud. This means point-to-point data transfer from a specific bus-stop to a specific bus-stop must be supported. The network on which the BDT App is implemented is a PTN, therefore the node interactions are not random, rather they are predictable and periodic. Consequently there is enough information about the network to make routing decisions in advance rather than discover routes. Given these specifications, a routing design and strategy can be formulated for the BDT App.

Routing Design In a unicast application like bulk data transfer, the delivery delay is not as important as the fact that the data must be reach its destination. The goal of the BDT App is to route as much data as possible through the bus network to the cloud. This translates to a *routing objective* to maximize the delivery ratio. A secondary objective is to minimize the delay experienced by the data; more data traveling in the network means reduced resources available for additional data to enter the network. Thus the two objectives of maximizing delivery ratio and minimizing delivery delay are linked. The interesting factors here are the rate at which data can enter the network, that is, what is the theoretical maximum that the nodes in a PTN can inject data into the network, and then how efficiently can this data reach the hubs from where it can reach the cloud. *Proactive routing* and *source routing* will be the chosen routing mechanisms as there is enough information about the PTN available in the form of bus timetables that routes can be precomputed and data packets will have a route encoded in them. *Message splitting* will not be allowed in order to simplify the simulation.

Routing Strategy Nodes will forward data using a knowledge based strategy. The possible strategies that will be considered will involve minimizing transfers or minimizing hops experienced by the data.

The design decisions are summarized in Table 3.

Table 3: BDT App Design

Design Decision	**Description**
Routing objective	Maximize delivery ratio, minimize delivery delay
Route computation	Proactive Routing, Source Routing
Message splitting	No message splitting
Routing Strategy	Knowledge based

3 Network Analysis Using Graphs

The BDT App is studied using a simulation of Helsinki's Public Transport System. The simulation models the PTN as a graph with bus-stops represented as nodes and edges between nodes exist when a bus travels between two bus-stops. In addition, the nodes (bus-stops) between which data transfer is investigated have been selected with the help of graph metrics. Therefore it is useful to include some background on network analysis using graphs.

3.1 Networks as Graphs

Many real world systems spanning various disciplines can be represented by a graph. The entities in the system can be represented as vertices (nodes) of a graph and the interactions and/or relationships between these entities can be represented as edges of the graph. The graph (or network) formed by such systems can then be studied to extract measures and meanings that can be used to make inferences about the system. This kind of network analysis is used in many fields ranging from analysing the nervous system in biology, to exploring human social interactions in society, and studying technological systems like the World Wide Web or connectivity in a mobile network. It provides researchers a tool to understand these complex networks that are often too difficult to analyse without the simplifying construct of a graph [HS12].

For some systems, their graph representation remains constant over time with no changes in the nodes and edges. The connections between components do not change and the structure of such a system is independent of time. However, most complex systems do vary with time. Entities (nodes) may come and go and relationships (edges) may exist only at certain times instead of throughout the lifetime of the system. These systems or networks are *time-varying* and the graph used to model such a system needs to be able to capture the time dimension of the system. Such graphs are known by several names such as time-varying graphs, evolving graphs, dynamic graphs, dynamic networks, temporal graphs and temporal networks [HS12]. A public transport system can be modeled by a time-varying graph.

Traditionally, networks have been studied using models and metrics that do not take into account the dynamic nature of the network. These approaches focus on the statistical and structural properties of a network and do not illustrate the time-varying nature of the system [TMM+10]. Capturing the time dimension in the study of time-dependent systems is important because it allows extraction of more

accurate and meaningful information from the system. For example, if in a certain graph, edges only exist at specific times then it becomes important to know what those times are if we aim to find a path through the graph. Another reason is that detailed information, including time-related information, is now available from many systems like mobile networks and sensor networks. The real-network traces available from these systems show that connections vary with time, and researchers require methods for temporal network analysis to properly study these systems [TMML09]. In summary, temporal network analysis is important because:

- Systems whose topology changes over time can be better represented by time-varying networks
- There is now an abundance of fine-grained data available about dynamic systems that can be examined using temporal network analysis techniques
- Static representation of a time-varying system can result in incorrect inferences being drawn about the system

3.2 Static Networks and Temporal Networks

Static networks (or static graphs) are those networks that do not contain any time information, i.e. they remain constant in terms of nodes and edges possibly over a certain time interval. Such a graph, $\mathcal{G}$ is simply represented as a set of vertices, V and edges, E, without any additional time-related attributes. The edges can be assigned weights if necessary.

A static network can be used to analyse both time-varying and non-time-varying systems, with the drawback that in the case of time-varying systems the analysis may not be as correct as if a more representative model were used. An *aggregated graph* is built from a time-varying system by aggregating edges that appear during a selected window of time in the dynamic system - this graph is also a static graph over the given time interval. The aggregated graph is then analysed. Figure 4 shows an example of a static aggregated graph built from a time-varying system. The time-varying system has different edges present at times 1, 2 and 3, the final static graph (on the right side of the assignment) includes all these edges. It is clear that performing analysis on the static graph could give inaccurate results, an example is that the static graph shows a path connecting node A to F, whereas we can see

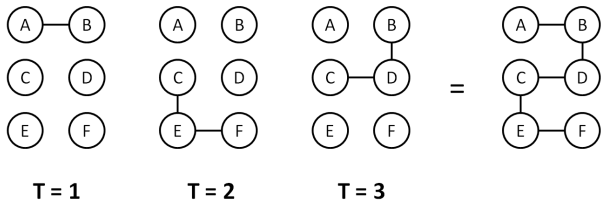


Figure 4: Time-varying network represented as a static graph

from the time-varying graph that no such path exists.

Temporal networks are a subset of complex networks. They can be used to represent systems that vary over time. The change in the system is captured by the appearance and disappearance of edges. Changes in the set of vertices (nodes) are not considered as this is equivalent to a node having no edges connecting it to the rest of the graph - this would be captured by the presence or absence of edges. Such a network takes into account when interactions between entities occur. The network can be defined as:

$$\mathcal{G}_t = (V, E_t)$$

where t is the time information.

In a temporal network, the edges exist only at certain times and for certain durations. The topology of the graph representing such a system depends on the time at which the graph is observed. The time-dependent interactions can be studied by applying various metrics. The meaning of the metrics will need to be interpreted with respect to the system under study. For example, betweenness centrality in a graph of email contacts identifies people who are important conduits of information, while the same metric in a graph for public transport networks identifies a node that lies on a busier section of road. In addition to time-dependent complex networks, temporal networks are also used in the analysis of *Delay-Tolerant Networks* and *Opportunistic Mobile Networks* [CFQS10].

3.3 Metrics

Graph metrics are used to describe a graph and its features. There are two types of metrics, local and global. Local metrics describe the vertex-level structure of the graph, they are computed on a single vertex at a time. Examples of such metrics are: degree, clustering coefficient, centrality metrics and similarity. Global metrics

describe features of the entire graph such as: radius, diameter, eccentricity, global clustering coefficient, reciprocity and similarity. A few of these metrics are briefly defined in Table 4. These metrics are also referred to as 'static metrics' as they have generally been applied on static graphs. Many of the static metrics have been adapted for studying temporal networks, and are called 'temporal metrics'. Some of these metrics have been used to analyse Helsinki Region's PTN and are described in more detail in a later section.

Table 4: Local and global static metrics

Metric	**Formula**	**Description**
Degree	$degree(i) = \Sigma_{j=1}^{N} A_{i,j}$	The number of edges incident on a vertex i, where $A_{i,j}$ is an entry in an adjacency matrix for the graph
Centrality	$degree\_centrality(i) = \frac{degree(i)}{N-1}$ $(N-1)$: max degree of a node	There are many types of centrality, the formula is for degree centrality. Centrality can measure importance of a node in terms of: position, connectedness, influence, prestige, and so on
Eccentricity	—	Maximum of the shortest paths to all other nodes
Diameter	—	Maximum distance between any pair of nodes in the graph

3.4 Representing Temporal Networks

In systems where the structural characteristics are more significant than the temporal variations, static graphs and metrics can provide insight into how the system

operates. However, as discussed earlier, static graphs are not powerful enough to represent time-varying systems. In general, for any time-varying system, we must consider the lifespan of the system, $[0, \mathcal{T}]$, a set of nodes (entities), $\mathcal{N} = [1, 2, ..., N]$, a set of relations between nodes (edges), $E = (i, j)$, then we can describe a time-varying graph by:

$$\mathcal{G} = (\mathcal{V}, E, \mathcal{T}, \rho, \zeta)$$

where $\rho : E \times \mathcal{T} \rightarrow \{0, 1\}$ is a *presence function* indicating whether or not an edge exists at a certain time, and ζ shows the duration that the edge exists for [SQF+11]. Such a formulation can represent graphs from various domains like [SQF+11]:

- Transportation networks, examples are public transport networks like buses, and air-flight networks
- Communication networks, examples are email or mobile communications between people
- Other Complex systems, an example could be relations between research papers and citations

Most research [TMM+10, CFQS10, GVOM13, NTM+13] has focused on two different methods of representing temporal networks, *Static Aggregated Graphs* and *Time-Varying Graphs*.

Static Aggregated Graphs. These graphs are constructed by aggregating information from the time-varying system and producing a **single** static aggregated graph. Information about contacts between pairs of vertices is stored into a single edge in the aggregated graph, consequently losing the information regarding the time at which the contact occurred, the contact duration and also the overall distribution of contact-occurrences in the dynamic system. The only information left is that a contact occurred between the pair of nodes during the lifespan of the time-varying system. As a result, the static aggregated graph representation of the temporal network is quite oversimplified.

One way to alleviate some of the information loss in an aggregated graph would be to store some additional information about the contacts, for example weights could be assigned to an edge in the aggregated graph corresponding to the number of times a contact occurred during the lifespan of the time-varying system. The edge attribute could take into account other features also, such as the duration or

frequency of contacts [NTM+13].

Time-Varying Graphs. Instead of a single static aggregated graph, consider if the time-varying system was broken down into multiple graphs each representing a time window from the original system. Each Time-Varying Graph (TVG), also called a *snapshot* or *fingerprint*, would contain aggregated information for only the time-window under consideration. The time-windows may be non-overlapping or a sliding window approach could be used. In either case, these snapshots would contain a greater amount of fine-grained information than the single static aggregated graph. Analysis would be performed on each of these TVGs and the evolution of these metrics over time can be studied.

This formalism produces more accurate results than the use of a single static graph, but its efficacy depends on the size of the time-window and the nature of contacts between nodes. If the window size is too big, information could be lost, on the other hand if the window size is too small, in a system with small contact durations most of the TVGs would contain very little information. Also, considering non-overlapping windows, how would contacts that are split across two windows be represented? These are questions for researchers to consider when analyzing a temporal system.

In the paper by Pan et al [PS11], analysis performed on both the single static aggregated graph and TVGs for the same real-world system showed that there is a correlation between distances between vertices in the static and temporal representations although there is a wide variation. The researchers found that despite two nodes being close together in the static graph, i.e. a path can be observed between the two nodes on the static graph, there may actually be no path between them, or it may take very long to travel between the two nodes. They also found that the pattern of contacts between nodes and the sequence of contacts affected the distance between nodes. In general it has been observed that single static aggregated graph representations overestimate the reachability of nodes and underestimate the time taken to travel between nodes in a time-varying system.

3.5 Analysing Temporal Networks

Many of the static metrics can be used to study temporal networks. A few of these are described below:

Static Metrics As we saw earlier a set of TVGs provides a better representation

of a dynamic system. Static metrics can be applied directly to each of the TVGs in turn, and then the evolution of these metrics over time can be studied. In addition to the static metrics introduced in Table 4, a few additional static metrics are density, clustering coefficient, modularity, and conductance [SQF+11]. All of these can be used to analyze a set of TVGs.

Temporal Metrics There are some metrics that can only be measured in a time-varying system. Practically such a metric is computed by iterating over each of the TVGs in the set representing the dynamic system. Most temporal metrics are based on a *time-respecting path* also known as *journey*. Many temporal metrics can be built upon the concept of a time-respecting path, the first is temporal distance. Temporal distance could in fact be interpreted in three ways: 1) Shortest distance: the distance in terms of hops between nodes, 2) Fastest distance: the distance in terms of latency (time taken to travel between two nodes) and 3) Foremost distance: the journey that arrives earliest at the destination node [SQF+11].

3.6 Graph Analyis and the BDT App

Helsinki Region's Public Transport System is most effectively captured by time-varying graphs as it is a time-varying system. Both static graph and temporal graph analysis have been used to study Helsinki's PTN and the various metrics have then been compared for the two graph representations. Additionally, the nodes selected to be a part of the simulation have been chosen on the basis of the graph metrics. Section 5 presents the results of the network analysis of Helsinki's PTN.

4 Related Work

This section provides an overview of research on using Public Transport Systems as a Delay Tolerant Network. Reviewing the related research in this area provides insight into the design decisions and performance metrics considered in the BDT-App developed in this thesis. Public Transport-based Delay Tolerant Networks (PTDTNs) have been studied from different perspectives; practical experiments, simulations, and network analysis. Often a mixture of these methods is used to study the performance of the PTDTN. Each of these research areas is expanded upon in the following sections.

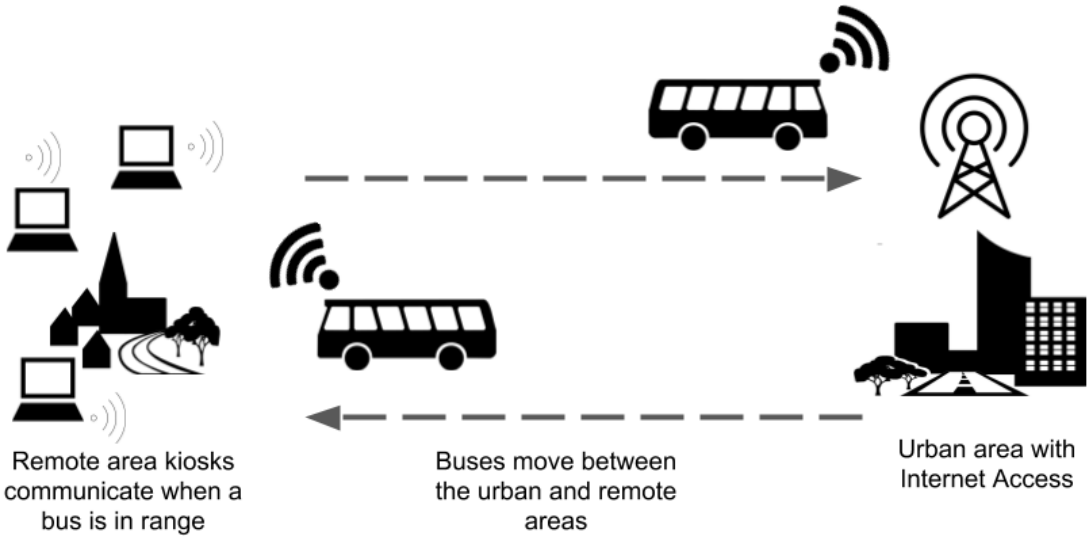
4.1 Real-World Experiments

Experimental implementation has a greater cost than running a pure simulation as hardware and software must be installed and maintained, and permission needs to be granted for installing the needed infrastructure on the Public Transport System. These constraints have meant that most PTDTN research has been conducted with the help of simulations. However, an experimental platform provides valuable information on how such a system will perform in real life.

The earliest experimental implementation of a PTDTN is DakNet [PFH04]. It was an attempt to provide Internet access to rural areas of India and Cambodia. In India, public buses were equipped with hardware to allow the wireless transmission and storage of data. Similarly, wifi-enabled kiosks were installed in the rural areas. Data transfers to and from buses could occur at these special kiosks, and when the bus reached the urban center, data could be exchanged over the Internet, Figure 5 demonstrates the setup. The focus was on the proof of concept with respect to provision of Internet services to remote areas. Although performance metrics, storage constraints and routing algorithms were not discussed, this was an important first step in demonstrating the workability of the PTDTN concept. The project was a success in terms of providing cheap connectivity to remote areas.

Another real-world implementation of a PTDTN was built by the developers of the MaxProp [BGJL06] routing algorithm. They created a testbed PTDTN called

Figure 5: Public Transport-based Delay Tolerant Network for Remote Areas



UMassDieselNet (DieselNet). DieselNet was deployed on 30 buses of a real public transport system in an area connecting several university campuses. Buses were equipped with hardware that would allow them to communicate with one another, the software running on these buses could make routing decisions for generated data. Buses could update their software whenever they had Internet access, allowing the researchers to test out different routing algorithms and packet sizes on a real PTN.

DieselNet was able to generate traces from the real-time data creation and transmission between buses. In DieselNet, packets are generated randomly and their destination is set to a bus currently running in the system. Stationary infrastructure such as bus stops, are not considered a part of the network. The aim is to route a packet from a source bus to a destination bus using bus-to-bus interactions alone.

The MaxProp routing algorithm was evaluated using the DieselNet testbed as well as by simulations using synthetic traces. The performance of the algorithm could then be compared on both real traces and the synthetic traces. The real-world experiment allowed the researchers to evaluate how their algorithm behaved in unpredictable situations such as a bus breaking down or the random assignment of buses to routes by the bus dispatcher service. The synthetic traces allowed the study of individual simulation parameters while holding other parameters constant. Thus, both real experiments and simulations provide useful information to the experimenter.

The MaxProp algorithm uses replication to deliver packets to its neighbors while assigning priorities to packets for both forwarding and dropping a packet. Packets are forwarded to buses that come into contact with the source bus on the basis of the likelihood of delivering the packet to its destination. Acknowledgements are also used so that buses can drop packets that have already reached their destination.

One more routing algorithm that was evaluated in a real-world scenario was RAPID [BLV07]. Balasubramanian et al used the DieselNet testbed to collect real traces for their RAPID routing algorithm, at this time DieselNet consisted of 40 buses. The RAPID algorithm aims to optimize a specific parameter of the PTDTN such as the delivery delay or the number of packets delivered within a certain time-frame. The decision of whether to forward a packet or not is dependent on the parameter being optimized. RAPID also gathers network state information like the location and number of replicas in the network to optimize its performance.

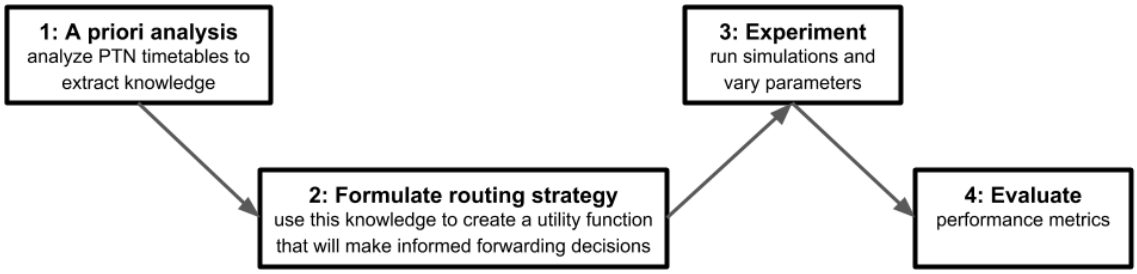
4.2 Simulations

Much of PTDTN research is based on simulations due to the difficulty of setting up the hardware mentioned earlier, and also the lack of flexibility in a real world experiment when compared to a simulation. It is easier to modify the parameters and try out different routing algorithms in the simulation environment. Consequently the majority of research uses simulations, based either on real traces of bus movements or on synthetic traces computed from the bus timetables. Simulations based on traces often make assumptions on transfer speeds and times for which two nodes in the PTDTN are in communication among others. A valid critique of simulation based approaches is the high number of simplifying assumptions that are made. Despite this, simulations can still provide useful information on the behavior patterns of the system.

Some routing protocols that have been tested using simulation on a PTDTN are BLER [SLL+08], CARPOOL [KT13], Op-HOP [GMRS12], HPDR [PK15], and CBF [AK07]. This is not an exhaustive list of studies on the topic of PTDTNs but the chosen studies broadly describe the type of research done in this area. All aim to optimize the routing objective based on a novel routing algorithm tuned to work on a PTN environment. Often a potential PTDTN application that is suited to the routing scheme is also suggested in the paper. The common steps performed in such studies are outlined in Figure 6, these steps also apply to the BDT App implementation. The parameters of the simulation in the various papers vary greatly with respect to number of buses, size of data packets, duration of the simulation, inclusion of buses and bus stops as DTN nodes, area covered by the PTN and so on. Each of these protocols is outlined next.

The BLER (Bus Line-based Effective Routing) routing protocol tries to route a

Figure 6: Steps in a PTDTN Simulation Study



packet from the source bus line to any bus belonging to a specific bus line. After that the data packet can jump within buses on the same bus line to reach a specific bus if that is needed. The routing is performed on the basis of shortest paths between bus lines. These paths are precomputed from the static graph of the PTN using Dijkstra's algorithm with the objective of maximizing contact times between bus lines. The paper demonstrates how deterministic routing using a contacts oracle is less effective than using a non-deterministic algorithm like BLER that is not dependent on meetings between specific buses. Simulations have been run on both real traces and artificial traces with 30 bus lines having 700 buses. One criticism of this approach is that shortest paths are computed over a static graph of the PTN network which will overestimate the contact opportunities between bus lines as buses are more and less frequent at different times of the day. The delivery delay could be improved by using time respecting shortest paths to find paths.

CARPOOL (Connectivity plAn Routing PrOtOcOL) is a routing protocol for a PTDTN that aims to supply free Internet access over an urban area. The idea here is that an urban area contains 'hotspots' where Internet access is free, the free Internet in these hotspots could be extended throughout the urban area using the public transport system. This would provide 'delay tolerant' Internet access. Buses and trams are envisioned as the data ferries and bus stops are 'offline gateways'. Ferries move data from an offline gateway to an online gateway. CARPOOL has global knowledge about the topology of network and which nodes are online gateways. CARPOOL uses this knowledge to perform routing. The researchers have tested their algorithm with 106 offline gateways (bus stops), 15 online gateways, and 60 buses employing the ONE simulator [KOK09]. An area where the authors could have expanded further is the type of content in the upload and download scenarios. It is likely that the size of upload and download requests would have an effect on the CARPOOL routing and overhead, and limit the number of users that could be supported by this network.

The objective of the paper introducing the Op-HOP (Opportunistic Hopping on Probabilities) protocol is to demonstrate how any urban public transport system can be used as an opportunistic network for data transfer. It introduces the probabilistic routing protocol Op-HOP and uses a custom simulator called URBeS to evaluate the protocol. The paper addresses unicast data delivery from one bus line to another bus line, thus handling applications involving content upload. Transfers occur when buses from different lines come into contact range of one another. The simulation traces are generated from the bus timetables from three different cities each with a

different transport system layout. Single copy forwarding is used to control traffic volume. Op-HOP calculates the probability of an encounter between buses of two bus lines by analyzing the graph constructed from the bus timetables and counting the number of times buses from two bus lines are in line-of-sight of one another. the algorithm is evaluated on all the cities and performs similarly on all. Delivery ratio, delivery delay and buffer usage are considered in the evaluation. One weakness of this study is that no real traces were used for evaluating the results. This might be an issue here as bus-to-bus contacts are more variable than for example, bus-to-bus stop contacts, and it could be argued that a real trace might produce fewer bus interactions than the traces generated from timetables.

The HPDR (Hybrid Position-based DTN Routing) scheme uses buses and bus stops to route data. Bus stops are assumed to have an access point (AP) installed that connects them to the Internet. The node (bus or bus stop) can query location information about destination buses before deciding where to forward data. An AP can simply send data via the Internet to another AP where the destination bus will be passing by. A criticism of this protocol is that it assumes that location data for all buses is available and can be accessed by nodes which may not be true, also it assumes that every bus stop has access to the Internet which will not be feasible in a real network. The simulation uses bus lines that connect the metropolitan area with smaller cities in the vicinity. These are long running bus routes along a multi-lane highway. The NS-2 simulator was used to run 123 buses from 8 bus lines.

The CBF (Cluster-based Forwarding) paper includes results from a simulation with 1163 buses running over 236 routes, simulated with NS-2 using real traces collected over a two week period. The main contribution of this paper is the dynamic programming based clustering algorithm which allows efficient computation of overlapping clusters for a PTDTN network. The clustering is performed on the basis of similarity in terms of encounters over bus lines. That is, if a bus line frequently encounters buses from another bus line then the two bus lines will belong in the same cluster. The CBF protocol will prefer to forward packets to buses that belong to the destination cluster using single copy forwarding. Once the packet is forwarded to a bus line in the same cluster, the forwarding switches to using Epidemic routing. The paper presents an interesting concept that deeper a-priori analysis of the PTN could provide insights that allow more intelligent forwarding decisions. A useful addition to the paper could be a comparison of how different topologies of PTN (due to geographic differences in cities) affect the clustering.

After surveying papers related to PTDTNs it is apparent that the routing decision is the most important decision in such studies. This decision is taken using both deterministic and probabilistic methods. The deterministic methods include computing routes based on timetables using variants of Dijkstra's shortest path algorithm (BLER, Op-HOP), and having extra knowledge about the topology, for example, the location of online gateways in CARPOOL. The forwarding decision is always knowledge-based, quite often using encounter probabilities between bus lines either directly or indirectly (BLER, Op-HOP, CBF), or on using metadata about the network (MAXPROP, RAPID), or on utilizing the timetable and topology information (CARPOOL, Op-HOP). All the papers rank delivery ratio as the most important metric for evaluating the routing protocol, closely followed by the delivery delay and buffer occupancy. One area lacking in these studies is an application-focused PTDTN; apart from CARPOOL, no study specifically discusses an application for the PTDTN, the application is usually a generic statement about delay tolerant spreading of content. It is probable that a particular application would require a particular routing strategy and design.

In this thesis the PTDTN is designed from an application perspective. The target Bulk Data Transfer application informs all design and routing decisions taken in the development of the BDT App. The BDT App development shares many features with the research discussed in this section; the bus timetables are analyzed a priori to compute the optimal paths for data, routing protocols are suggested for the BDT App, the routing protocols will be tested by simulation using synthetic traces generated from the bus timetables, the evaluation of the routing protocols is based on delivery ratio and delay. Table 5 summarizes the features of all the discussed protocols.

A key difference between the BDT App and the related studies is that in the BDT App data exchange will occur between buses and bus-stops. There will be no bus to bus transfers. The routing protocol is based on how commuters get on and off buses at stops in order to reach their destination. Another difference from previous studies is that the BDT App will include network analysis of the PTN to determine the most useful nodes for the given application. The studies discussed in the next section describe some of the work done specifically on network analysis for PTDTNs.

Table 5: Public Transport-based Delay Tolerant Networking Implementations

Algorithm	**Experiment**	**Routing Strategy**	**Nodes**	**Routing Objective**
DakNet	Real-world experiment	All data transferred at kiosk and at Internet hub	Buses and kiosks	-
MaxProp	Real-world experiment, simulation with real and synthetic traces	Bus to bus routing Multi copy forwarding with delivery acknowledgements to limit flooding Routing using packet priority and delivery likelihood	Buses	Maximize delivery ratio, minimize delivery latency
RAPID	Real-world experiment, simulation with real and synthetic traces	Bus to bus routing Multi copy forwarding with delivery acknowledgements to limit flooding Routing dependent on parameter to be optimized and metadata about network like the location and number of replicated packets	Buses	Minimizing average delay, minimizing worst-case delay, maximizing the number of packets delivered before a deadline
BLER	Real trace from bus GPS data, synthetic trace from bus timetables	Bus line to bus line routing Single copy forwarding Routing using precomputed path based on maximizing bus contact times	Buses	Maximize delivery ratio, minimize delivery delay
CARPOOL	Synthetic trace from bus timetables	Routing between bus, bus stop, and Internet gateway Target application: Provide Internet access to all via PTN Single copy forwarding Routing using simple shortest path computed from bus timetables	Buses, bus stops and gateway to Internet	Maximize delivery ratio, minimize delivery latency
Op-HOP	Synthetic traces from bus timetables	Bus line to bus line routing Single copy forwarding Routing using encounter probabilities between bus lines	Buses	Maximize delivery ratio, minimize delivery delay

Table 5: Public Transport-based Delay Tolerant Networking Implementations

Algorithm	**Experiment**	**Routing Strategy**	**Nodes**	**Routing Objective**
HPDR	Real traces from transport service	Bus to bus routing Single copy forwarding Routing based on GPS location of destination bus	Buses, bus stops with access points	Maximize delivery ratio, minimize delivery delay, storage usage
CBF	Real trace from bus GPS data	Bus line to bus line routing Single copy forwarding and Epidemic Routing using cluster membership	Buses	Maximize delivery ratio, minimize delivery delay
BDT App	Synthetic traces from bus timetables	Bus stop to bus stop routing Single copy forwarding Routing using shortest paths, described in detail in Section 6	Buses and bus stops	Maximize delivery ratio, minimize delivery delay

4.3 Network Analysis

The previous section focused on simulation as a way to evaluate a PTDTN, this section focuses on network analysis of a public transport network in order to create better forwarding strategies. This thesis performs both network analysis and simulation in the development of the BDT App. Most research has not combined both these methods into a single study. The simulation based papers do not perform much network analysis of the PTN, however, network analysis can identify bottlenecks and hubs in a PTN. The routing algorithm can then take into account these features of the network to build much more efficient PTDTNs.

In [GVOM13], Galati et al have performed a comprehensive temporal analysis of the public bus transport system of Zurich. The PTN contains 44 bus lines and 336 bus stations. They describe in detail how to model and analyze the transport system, and they employ several temporal metrics (temporal path length, global efficiency, betweenness centrality and closeness centrality) and describe how to apply them practically on the set of time varying graphs (TVGs) representing the system. Through their analysis they explore the connectedness of different parts of the transport system at different times of day such as peak and off-peak times. The study concludes that identification of 'hub' nodes, that is, nodes that are more connected with their neighbors and that will be more useful for forwarding data, will help to design better PTDTNs with more efficient forwarding algorithms. According to the authors, including hub nodes in routing paths will greatly increase the efficiency of the DTN, specifically, it will improve the delivery ratio and reduce network overhead. However, this hypothesis was not tested in the paper as there was no experimental evaluation of these claims. The BDT App will evaluate the effect of selection of hub nodes on the performance of routing algorithms.

Ahmed et al [AS10] have used a different set of metrics to identify hub nodes and to demonstrate various properties of nodes in a PTDTN. The previous research by Galati et al used bus stops as graph vertices and movement of buses between stops as an edge in the graph. This study considers buses as vertices and bus communications as edges in the graph. The static and temporal metrics are then computed for the nodes (buses). The selection of bus stop or bus as a vertex of the graph depends on the routing method in use; routing from bus to bus or from bus stop to bus stop. The BDT App will use bus stops as graph vertices as its routing is from bus stop to bus stop. It would be interesting to study cases where both bus to bus and bus stop to bus stop communication is allowed, of course, this is a more complex scenario.

[AS10] has analyzed real traces containing 1200 buses in a 24 hour period. The static metric used to identify hubs is node degree. It could be useful to compare if hubs found by static and temporal metrics are the same in a PTN. Static graph analysis is much simpler than temporal analysis and if the same hubs are identified by both metrics then it makes sense to use the less computation intensive metric. [AS10] also computes clustering coefficients using temporal analysis of the communications between buses in a 24 hour period with one hour time intervals. The clustering coefficient represents the connectedness of a node with other nodes in its vicinity

and is the metric that the authors suggest can be used to place extra infrastructure (like roadside gateways) where its value is low.

Table 6 summarizes the particulars of the studies discussed in this section and contrasts them with the BDT App.

Table 6: Public Transport-based Delay Tolerant Networking Network Analyses

Reference	**Experiment**	**Description**	**Results**
Galati et al [GVOM13]	Synthetic traces from bus timetables Analyzing temporal metrics	Temporal metrics: betweenness centrality, closeness centrality, path lengths, global efficiency Hubs found using temporal betweenness centrality Graph representation of PTN, bus stop = vertex, bus movement between stops = edge	Identification of hub nodes Interpretation of the temporal metrics for PTNs
Ahmed et al [AS10]	Real traces from bus GPS data Analyzing static and temporal metrics	Static metrics: node degree Temporal metric: clustering coefficient Graph representation of PTN: bus = vertex, buses can communicate = edge	Identification of hub nodes Placement of stationary gateways Suggestions on how to improve routing by taking metrics into account
BDT App	Synthetic traces from bus timetables Analyzing static and temporal metrics Simulation	Temporal and static metrics Graph representation of PTN, bus stop = vertex, bus movement between stops = edge Run simulation using hub nodes	Identification of hub nodes and other graph metrics Improve routing protocols by taking metrics into account Use hub nodes in simulation and evaluate performance

5 Network Analysis of Helsinki Region's Public Transport Network

As explored in the previous section, network analysis of a PTN enables researchers to make better design decisions when developing PTDTN applications. This section presents a network analysis of Helsinki region's public transport bus network, the results of which will be used to optimize the design for the BDT App. There are several significant questions to consider when planning a solution for a network application like the BDT App which aims to maximize delivery of large amounts of data to the Internet. Investigating these questions will aid in laying out the design and simulation settings for the App:

- Where in the PTN should hubs connected to the Internet be placed?
- How should the optimal path be computed from a bus stop to a hub?
- Which bus stops or links are more important for routing data?
- Given that PTNs are time-varying in nature, what is the state of the network at different times of the day? How could this affect routing?

The experimental setting for the BDT App is the area served by the the Helsinki Regional Transport Authority [HSL]. Figure 7 shows this area which comprises of the capital city of Helsinki and several of the surrounding municipalities including Espoo, Vantaa, Kauniainen, Kerava, Kirkkonummi, Sipoo, Siuntio and Tuusula. In the remainder of this work reference to the 'Helsinki region' implies the areas served by the Helsinki Regional Transport Authority (HSL).

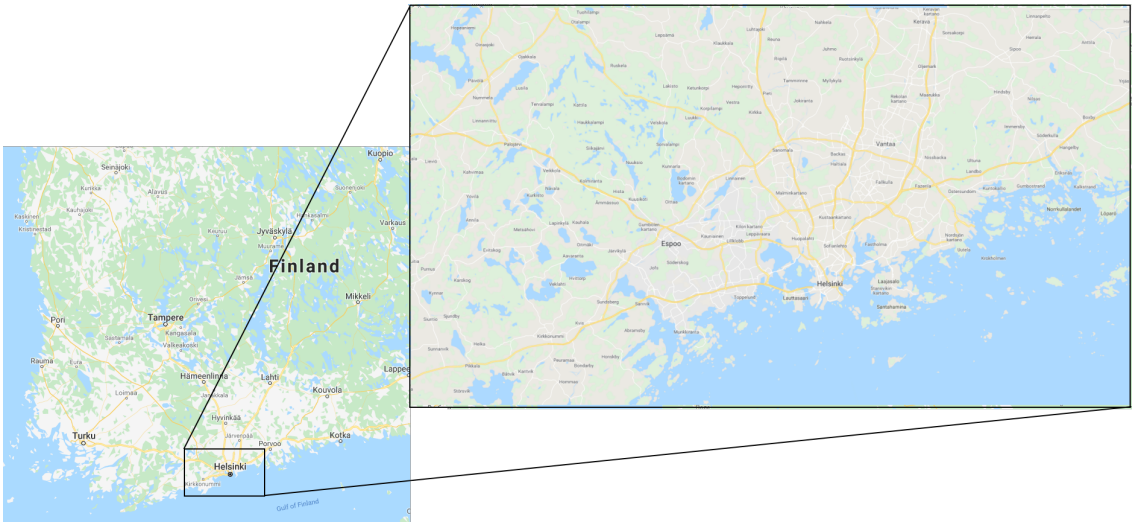


Figure 7: HSL Region

The HSL authority includes trams, metro, rail, ferry and bus services. Detailed maps for the various HSL services are available at `https://www.hsl.fi/en/timetables-`

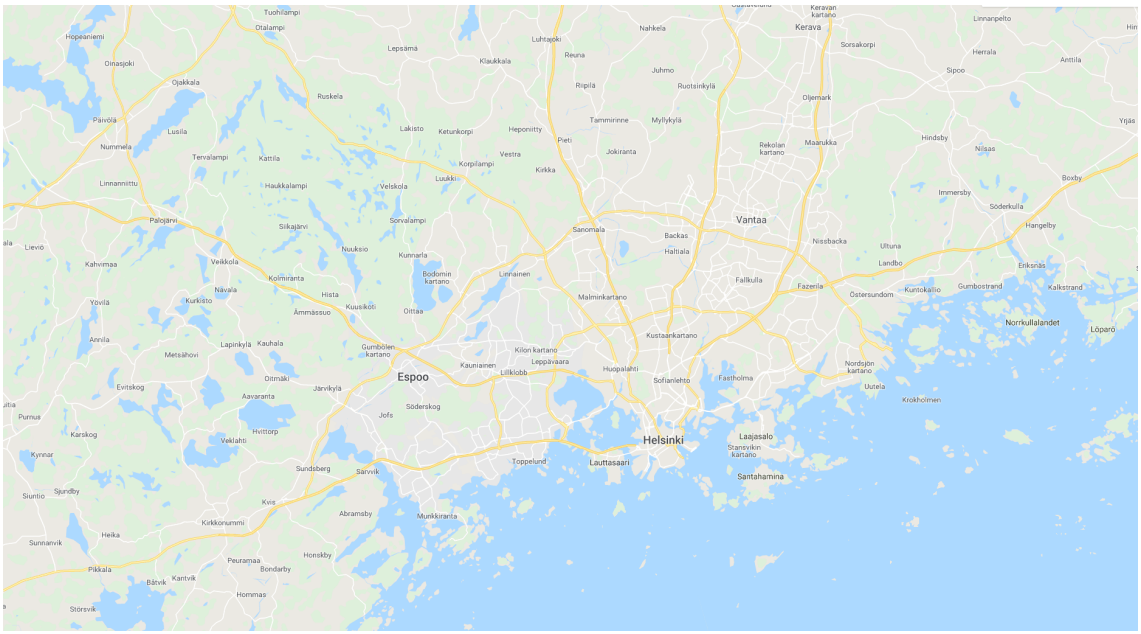


Figure 8: Road Map

`and-routes/routemaps`. A magnified map of the roads in this territory is shown in Figure 8[3] . The transport service routes are arranged in such a way as to facilitate efficient transport to and from the downtown area of Helsinki. Nearly all bus lines terminate in the downtown area, and a few bus lines link suburbs. The largest bus terminals are located around the Helsinki Central Railway station at Rautatientori, Elielinaukio and Kamppi.

The bus schedules from HSL provide an accurate representation of buses traveling through the Helsinki region. The network analysis is performed on a graph generated from these bus timetables. The BDT App design is based on the public bus system, this allows direct comparison with related research which often includes only bus services. Additionally, as elaborated in the thesis introduction, the bus service is the most available and frequent service in the Helsinki region, providing a suitable network for Bulk Data Transfer applications.

Both GTFS [gtf] format and Kalkati [kal] format bus schedules are available for HSL. A timetable dataset for January 2018 has been downloaded from the Kalkati service to provide data for generating graphs from the bus schedules. The Kalkati format specifies a single file containing all timetable and service data. The data has been parsed to create a graph with nodes representing bus stops and edges representing the movement of a bus between two stops[4].

[3]Google Maps Javascript API was used to generate all maps

[4]With thanks to Otto Waltari for providing the original Kalkati format parsing code

5.1 Static Aggregated Graph Analysis

The SAG analysis is performed on a graph of the network constructed by aggregating bus data over a 24 hour time window. Each link between two bus stops is weighted by the number of times a bus travels between the two stops. In this way the time varying information about buses traveling in the city is aggregated into bus capacities for each edge (bus link) in the graph. Exactly what time a bus traveled over a link is lost in the process, however, it is still possible to extract useful information from this simplified representation of a time-varying system. For example, constructing SAG graphs for different days such as weekdays and weekends demonstrates the variation in the bus service over a week. The SAG encoding of the system can also be used to analyze the distribution of bus stops and estimate the busiest bus links and bus stops in the 24 hour time window.

5.1.1 Bus Service Variation between Days

As is usual of most public transport systems, HSL's bus services are more frequent on weekdays and less frequent on the weekends. The number of services and buses in operation can be calculated from the HSL bus timetables. Figure 9 shows the differences in buses, bus routes and active bus stops between weekdays and weekends.

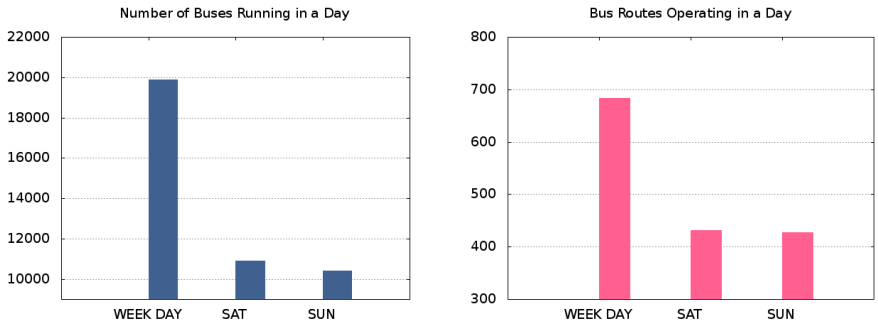
From the graphs it is apparent that the highest number of active stops, bus services, and bus routes occur on weekdays. Consequently the busiest day (any weekday) is selected for further analysis in order to leverage the most dense network. Any weekday can be selected as bus services are periodic and repeat at the weekday level.

5.1.2 Bus Stops, Bus Links and Bus Link Capacities

Considering a 24 hour SAG graph for a weekday, Figure 10 highlights the 6390 bus stops that exist in the area and Figure 11[5] shows the 7728 links between bus stops that represent that at least one bus travels between the two stops at some time in the 24 hour interval. The largest strongly connected component in the network encompasses 6042 out of the 6390 bus stops, showing that over a 24 hour interval, 95% of the bus stops are reachable from one another. This value increases to 99% if opposite stops are connected to each other. Figure 12 focuses on how busy different bus links are by visualizing the bus capacities as colors, thicker and darker edges on the graph indicate more buses traveling over that edge.

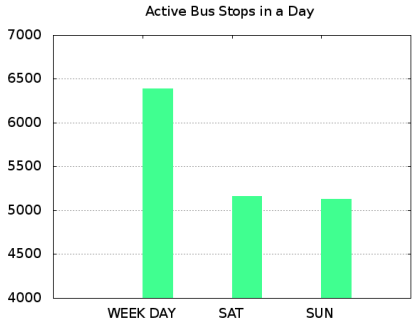
The bus link capacities are shown in Figure 13 and follow a Power Law distribution, demonstrating that a very small number of bus links have a large number of buses while the majority of bus links have a small number. The bus capacities can be used to favor certain routes over others as more buses correlates with more opportunities for data to travel in the network. A $\log_2$ plot of the capacities makes it easier to

[5]With thanks to Otto Waltari for providing the original link visualization code



(a) Buses

(b) Bus Routes



(c) Bus Stops

Figure 9: Buses on Weekdays and Weekends

compare the number of buses on bus links. The same Power Law distribution applies to most other PTN metrics such as node degree and node betweenness centrality; a few nodes (bus stops) have a large metric value while most have small values.

5.1.3 Bus Stop Distribution

Figure 14 is a heatmap based on the bus stop distribution in the region. Red corresponds to the most densely populated area, lighter colors represent less density. As can be seen there are a few locales where bus stops cluster together. Other clusters of bus stops occur at some railway stations, metro stations and large shopping centers. The suburbs have a fairly constant density of bus stops. It is possible to discover where it could be advantageous to place hubs by looking at the density information. Hubs should be placed where bus stops are most dense and these happen to be bus terminals.

5.1.4 Shortest Path Computation

Dijkstra's shortest path first algorithm can be used to discover routes between bus stops in the SAG graph. The computation is simple and there are many implemen-

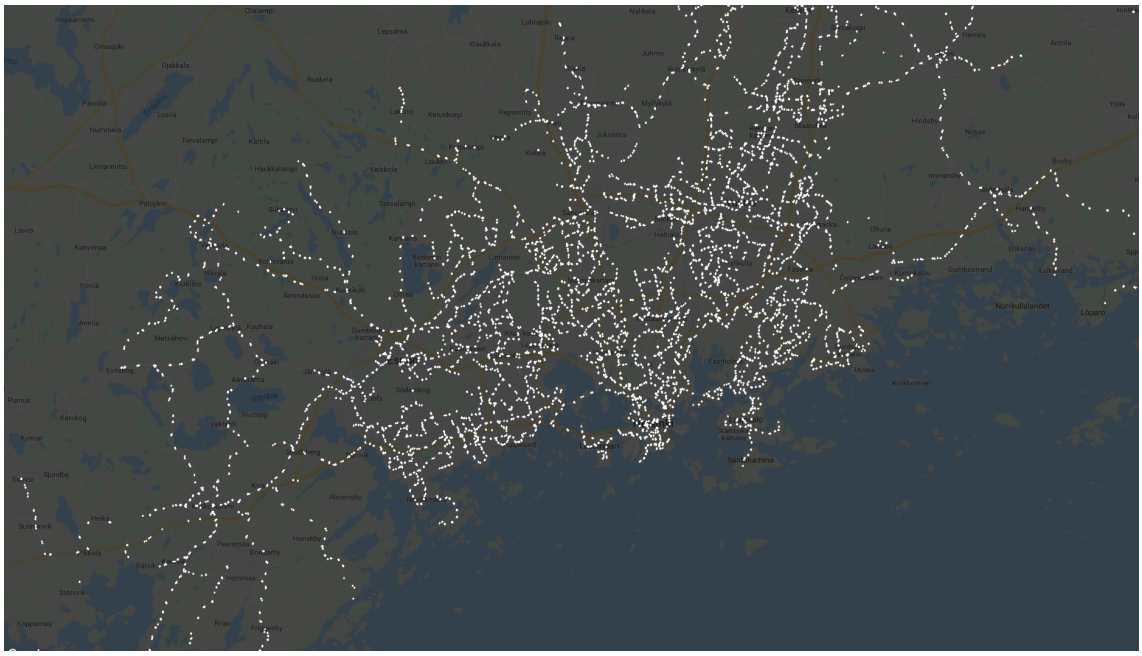


Figure 10: 6390 Bus Stops Highlighted

Figure 11: Bus Links Highlighted

tations of this algorithm for static graphs. One caveat of using a SAG graph to compute a path through a time varying system is that the end to end path may not exist in a real system with buses traveling at certain times. It is only known that a bus link on the path exists at some point in the 24 hour window. This may mean that data will reach its destination much more slowly than if it had followed

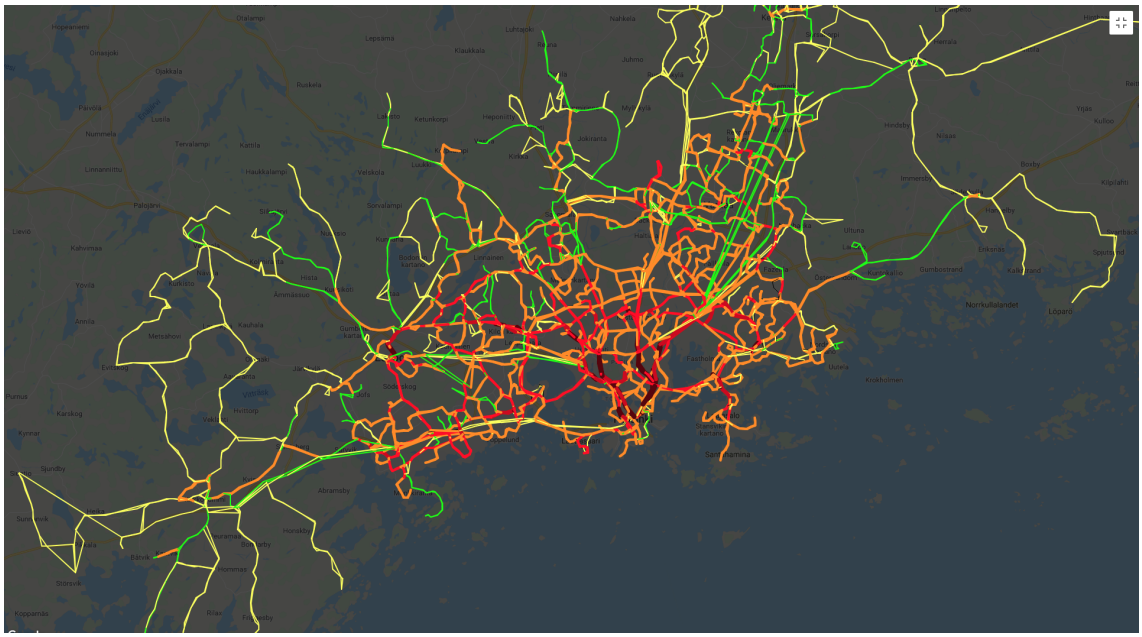


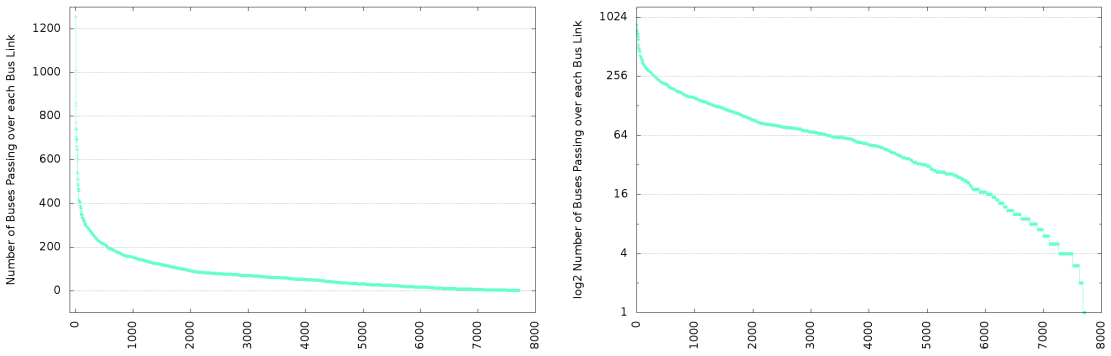
Figure 12: Bus Link Capacities Highlighted

a more intelligent path. As bus services are periodic at the daily level, this implies that if a piece of data was unable to traverse the computed path within the 24 hour window, in the next 24 hour window it could continue its travel. Unfortunately in the worst case data may take days to reach its destination, this is extremely poor performance in contrast to average bus commute times which rarely exceed two hours for anywhere in the HSL region.

5.2 Time Varying Graph Analysis

Aggregating bus schedule data over a 24 hour interval causes loss of fine-grained time information. Bus services in HSL operate between 10, 15, 20, 30 and 60 minute intervals and all bus services are not in operation over the entire day. This entails that the connectedness of the network varies with time, and a single SAG graph for a 24 hour period will overestimate the reachability of nodes and underestimate the time taken to travel between nodes. A path that exists in the single SAG graph may not actually exist at any one time in the network. Time varying graph analysis is more suited to a time varying system albeit this method is computationally more intensive and complex than working with a SAG graph. One way of performing time-varying graph analysis is to create multiple SAGs over smaller time intervals and then analyzing each of those.[6] In this section, HSL's PTN is analyzed over smaller time intervals to get a better understanding of the network's behavior.

[6]See Section 3 for a full discussion on static vs time varying network analysis



(a) Bus Frequencies (b) Log_2 Bus Frequencies

Figure 13: Distribution of Bus Frequencies over Bus Links

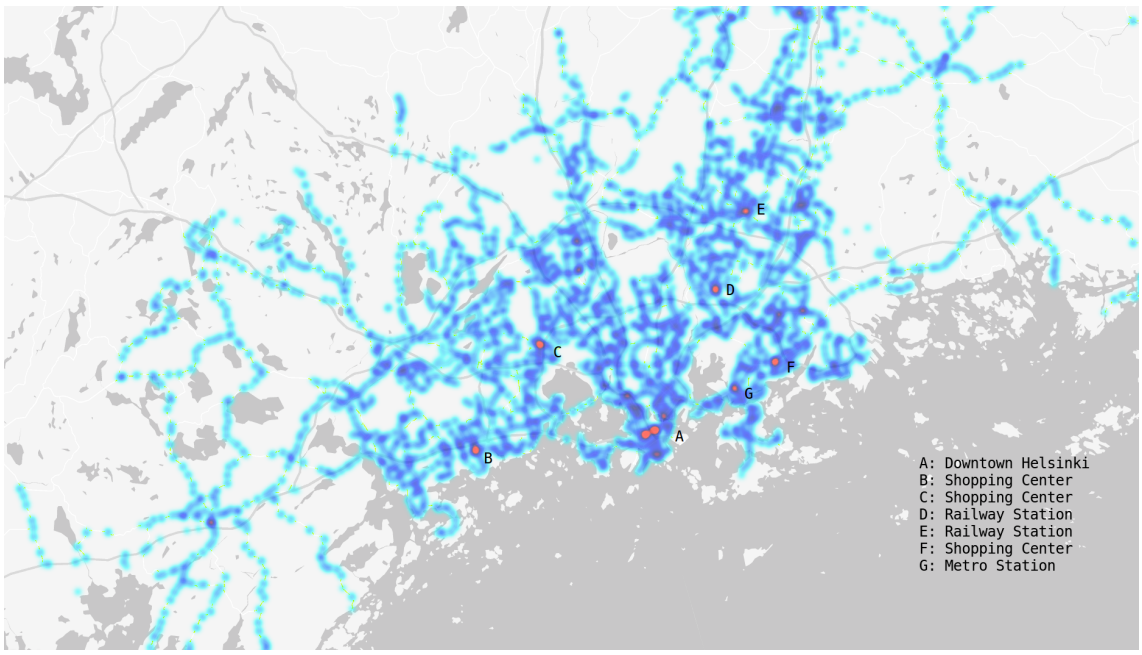


Figure 14: Bus Stop Heatmap

5.2.1 Bus Service Variation between Days using One Hour Intervals

The number of services in operation per hour is counted for weekdays and the weekend days. It is visible from Figure 15 that the bus frequencies vary between peak times and off times. On weekdays there are two busier periods of time when more buses are traveling in the region.

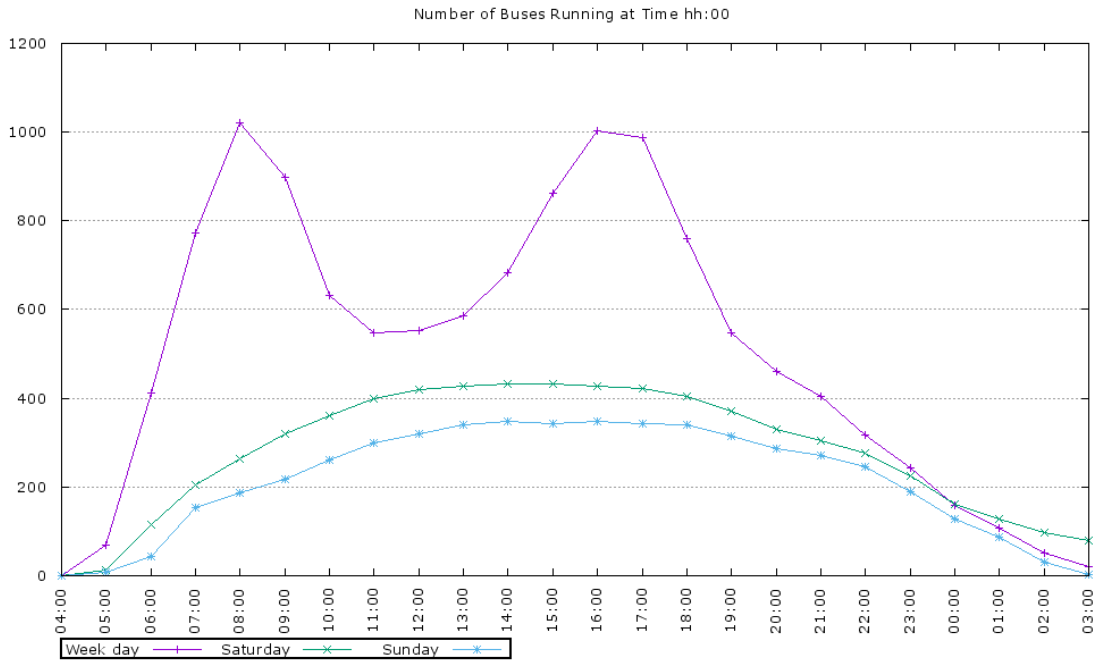


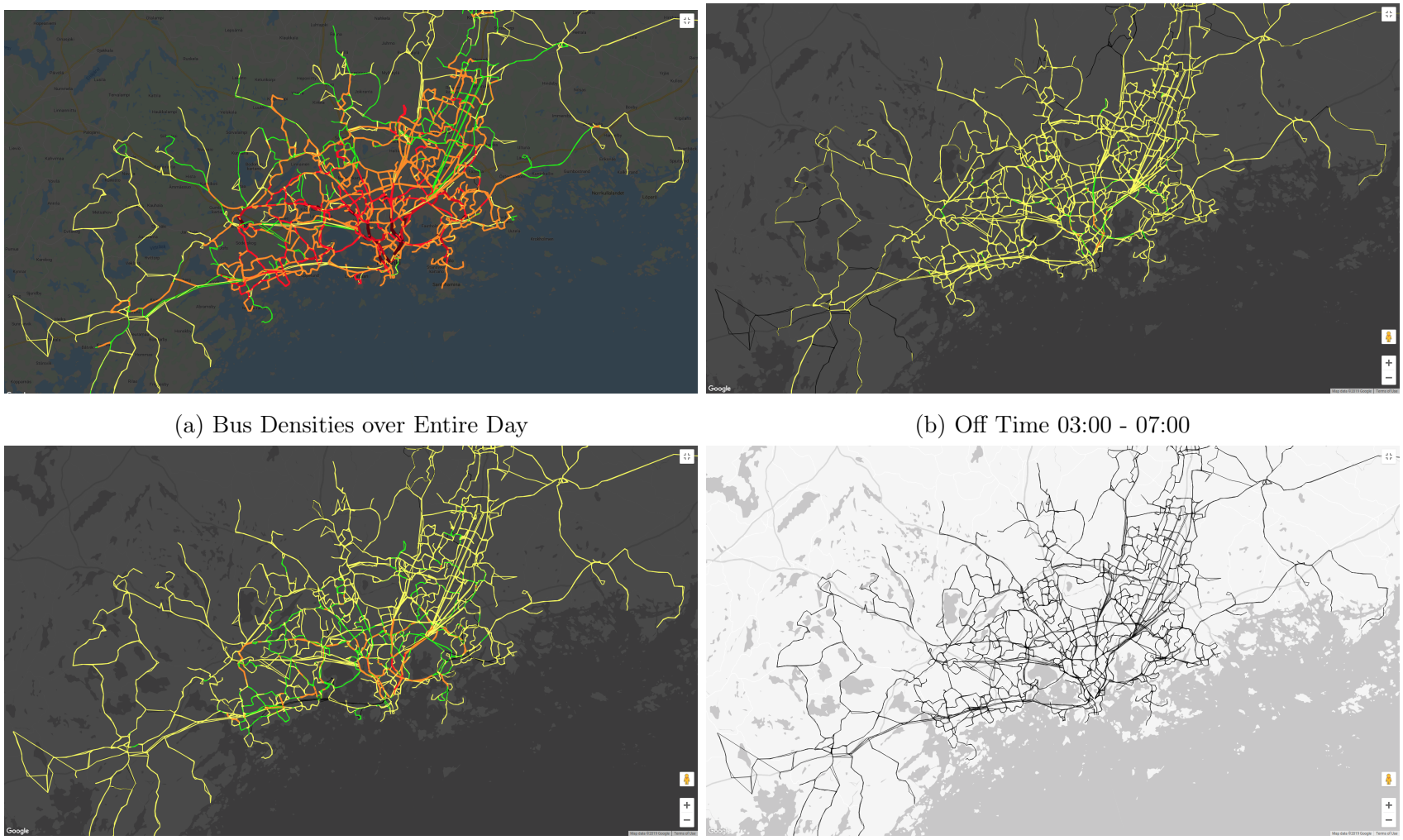
Figure 15: Buses over a Day

5.2.2 Bus Service Variation on a Weekday

Taking Thursday, the 4th of January as an example, there are 6390 bus stops running 690 bus routes, with a total of 19912 bus services active on this day. This information can be visualized as a graph; edges can be weighted by the number of buses traveling over the time period under study, then thicker and darker edges on the graph indicate more buses traveling over that edge. Figure 16 (a) is a static aggregated graph for the whole weekday. It is interesting to see how this graph varies over time over the day. Figure 16 shows the edge (bus) densities at different time periods on the same day.

5.2.3 Shortest Path Computation

The 'best' path through a time varying system would be a temporal shortest path. However there are hardly any publicly available, widely tested implementations on computing time-respecting shortest paths. Halfway between time-respecting and time agnostic shortest paths are those paths that have been computed on shorter time interval SAG graphs. These paths are more accurate than the paths computed using a 24 hour SAG graph, however incorporating multiple routes for different times of the day and using them in the simulation increases the complexity of the simulation. There is a trade-off between complexity and performance, and the next section examines these trade-offs and presents the design decisions for the simulation.



(a) Bus Densities over Entire Day

(b) Off Time 03:00 - 07:00

(c) Peak Time 07:00 - 10:00

(d) Off Time 24:00 - 03:00

Figure 16: Bus Densities at Different Times of the Day

5.3 Design Decisions based on Graph Analysis

The graph analysis has helped to answer the questions posed at the beginning of this section. These findings can be used to further define the simulation settings and design parameters on which the BDT App will be based. An initial set of design guidelines were developed in Chapter 2 which investigated the routing design and strategy for the BDT App with respect to DTN principles. The network analysis presented in the current chapter has helped to discover additional design rules which augment the previously stated guidelines. These design choices are summarized in Table 7 alongside the earlier design decisions (shown in italics).

Table 7: BDT App Design Guidelines

Design Decision	**Description**
Routing objective	*Maximize delivery ratio, minimize delivery delay*
Route computation	*Proactive Routing, Source Routing*
Message splitting	*No message splitting*
Routing Strategy	*Knowledge based*
Placement of hub nodes	At the highest density of bus stops
Optimal path computation	Dijkstra's algorithm running on a 24 hour SAG graph
Important stops for routing data	Bus link capacities correspond to importance of a link or stops
Time varying nature of the system	Bus movements are time respecting but computed paths are not

The routing objective is to maximize the delivered data in the system and to do this in the fastest possible time. All routes are precomputed and not re-computed during the simulation. Data is assigned a route and tries to follow it to its destination. A data packet travels as a single unit and is not split into smaller pieces. Routing decisions are taken based on the network analysis performed on the graph generated from the HSL timetables. This knowledge is used to decide locations of destination nodes, choose source nodes, decide cost functions for routes, and to compute shortest paths.

Hub nodes shall be placed in locations with the highest density of bus stops. High densities exist at bus terminals or other points where multiple bus services start or terminate. This is a good location for a hub as buses will always stop here and every bus stop located in low density bus stop areas has buses traveling through it whose destination is near a hub node. A bus stop will try to send data to its nearest hub node. The 'nearness' of the node being decided by finding an optimal path to the hub.

Optimal paths shall be computed on a 24 hour SAG graph of the network using Dijkstra's shortest path algorithm. The pros and cons of this approach have been

discussed earlier and in the end, the approach that simplifies the simulation has been selected.

The importance of stops or links with respect to routing corresponds to finding cost functions for selecting optimal paths through the graph. Cost functions will be based on the capacity of bus links, as more buses (more capacity) should be expected to increase the amount and throughput of data in the network. Another method to select optimal paths could be minimizing the number of traversed bus links without caring about bus link capacities. This may improve delivery times although throughput could decrease. A third option is to minimize bus transfers. Bus transfers often add minutes onto commuters' travel times and the same is expected for data that changes buses. Routes that minimize transfers between buses may provide better delivery times, this corresponds to following bus links of the same bus service as much as possible while traversing a route.

With respect to the time varying nature of the system the simulation will run based on bus schedules and so it will closely respect the time related features of the system. However path calculation will be performed on SAG graphs. Path accuracy could be improved by using paths calculated from a smaller SAG interval for a peak time of day. However, this is left as future work, as only paths generated from the 24 hour SAG graph have been tested in the simulation.

6 Simulation of Helsinki Region's Public Transport Network

The BDT App is a simulation of Helsinki region's Public Transport Network. This section describes how this simulation has been implemented based on the design guidelines developed in earlier sections. It also provides answers to the research questions posed in the introduction section of this thesis, such as what is the delivery ratio and delivery time for data traveling on a Public Transport Network, how much data can be throughput, how reliable is the network, and what are the performance bottlenecks? The section is divided into details on the simulation design, the various parameters that have been studied and the simulation results.

6.1 Design

There are two aspects to consider during the simulation; the first is the movement of buses, and the second is the movement of data. The movement of buses is easily discoverable from HSL region's bus timetables. The simulation uses the arrival times of buses at each stop to simulate buses traveling. The schedules themselves take into account rush hour and peak and off-peak times and therefore can be considered an accurate representation of bus movements over time. Buses must stop for some time at each bus stop in order to pick up and drop off passengers, this is referred to as

the ‘dwell time’ and is a function of the number of people getting on and off the bus plus some minimum wait time.

In the BDT App the dwell time is a random number between 5 and 18.75 seconds. This approximation of the dwell time is not representative of reality as it does not use any real information about the number of people that could be expected at a certain time at a certain bus stop. An improvement to the simulation could be adding realistic dwell times for all the bus stops and bus services. Even so, this approximation is a rather short time for a bus to wait at a bus stop, and it will provide a conservative estimate of the performance of the BDT App.

As to the movement of data, it is during the dwell time that data can transfer on and off the bus. As data only transfers to or from a bus when the bus is stopped at a bus stop, the simulation does not need to know exact locations of buses when they are between stops. Each bus stop knows a route to the nearest hub, this precomputed route is stored in the data packet when it is loaded at a bus stop. The ‘nearness’ of a hub depends on the cost function used to compute the shortest path to the hub. The specific cost functions used in the BDT App are discussed in the section on simulation parameters. Data enters and disembarks from buses according to the supplied route. As the dwell time is limited, it is possible that not all the data that wants to transfer to or from the bus can be transferred. Data waiting at a bus stop can wait until the next bus arrives, and data that has gone off its route because of a missed transfer needs to disembark at any next bus stop so that the stop can load a new route into the data packet.

When a data packet reaches a hub node, statistics about the data are stored. These include the route the data actually took, whether it had to be re-routed and how many times, and how long it took to reach a hub node. Similar statistics are also gathered about data that has not reached a hub node by the end of simulation.

The simulation was developed in Python using SimPy[7] which is a discrete event based simulator. The NetworkX[8] Python library was used to perform the network analysis. The steps taken to run the simulation are outlined in Algorithm 1.

Algorithm 1 Simulation Steps

1: Construct network graph from bus schedules
2: Improve graph connectivity by linking opposite stops
3: Mark hub nodes on graph
4: Assign costs to each edge
5: For every bus stop, compute route to nearest hub
6: Initialize simulation parameters
7: Run simulation
8: Collect statistics

[7]https://simpy.readthedocs.io/en/latest/contents.html
[8]https://networkx.github.io/

6390 bus stops running 690 Bus services are simulated on Thursday, the 4th of January 2018. Certain stops are selected as data producers; whenever a bus stops here, it can pick up data. The amount of data depends on how long the bus stays which is probabilistically decided. Important values for analyzing performance of the PTN were data throughput and average time to deliver data. Different routing algorithms produce different results for these performance metrics. One Gigabit per second transfer speeds were assumed between buses and stops. No limitations were placed on bus or bus stop buffer space, and there were no energy constraints.

6.2 Parameters

One of the advantages of testing with a simulation is that parameters can be easily modified to investigate different metrics like the data delivery ratio or the time taken for data delivery. The BDT App simulation parameters include cost functions to compute the routes between bus stops, the time-intervals during which data is produced and for which the simulation is run, and selection of data producing and hub bus stops. A brief description of each of the simulation parameters follows:

Simulation running time: The simulation is run for a predefined amount of time, for example one day, or two days and so on. The same bus schedule is repeated for each day mimicking the periodicity of the bus network. If all data does not reach its destination within the simulation set time, the running time can be increased, and it can be checked whether data can manage to reach its destination given more time.

Data production time: The time interval during which data enters the system is controlled, for example data is produced only during the peak hours from 7:00 to 10:00. It is then observed how much of this data reaches a hub, and how much remains in the network at the end of the simulation.

Hub node selection: Hub nodes have been selected from the bus stop density distribution map from bus terminals and other areas with many bus stops clustered together. By over-selecting hub nodes (selecting many more than are optimal) it is possible to observe which nodes are most used. The number of hub nodes can then be decreased to see how data delivery is affected. In this way the optimal location and number of hub nodes can be discovered.

Source node selection: Source bus stops are those from which data is being produced. These bus stops can be selected from suburbs (these have low betweenness centrality values), or from highly connected bus stops (these have high betweenness centrality values) or from any other interesting locations. Selection of different categories of source nodes allows comparison of BDT App's performance on differently connected bus stops.

Dwell time behavior: All data packets that want to disembark from a bus are able to do so regardless of dwell time. However, the number of packets boarding the bus is dependent on the time for which the bus is at the stop. This setting ensures

that no data packet can miss its stop. This setting can be changed so that the dwell time also limits disembarking data, in this case data can also miss getting off at the correct stop and then needs to be re-routed.

Data Transfer Speed: Data can be transferred between the bus and bus stop at the rate of 1Gbps. Data packets are 100Mb in size.

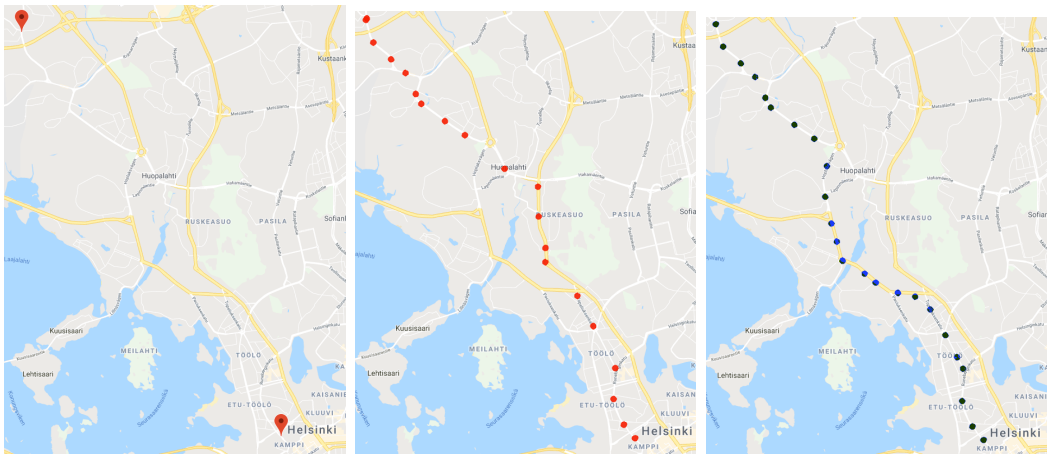
Routing algorithms for computing routes: Three routing algorithms (or cost functions) were used to discover routes between stops:

- **MINEDGES:** minimize edges traversed,
- **MINHOPS:** minimize edges traversed based on edge capacity, and
- **MINTRANSFERS:** minimize the number of transfers made by data

Each of these routing algorithms generates the shortest path to a hub using a different cost function. The MINEDGES algorithms assigns a cost of '1' to each edge and the shortest path tries to minimize the number of edges traversed. The MINHOPS algorithm uses a cost function that assigns lower costs to edges with more buses traveling over them, thus the route tries to follow edges with more buses on them. The final algorithm, MINTRANSFERS, provides a route and also strict transfer instructions to the data so that it only uses certain bus services in order to minimize transfers between buses. The data delivery ratio and data delivery times can be compared for the different routing suggestions.

6.3 Results

Each of the simulation settings described above can affect the performance and behavior of the BDT App, and there are numerous possible combinations of these settings. A specific set of parameters must be selected so that the important questions about latency, throughput and data loss in the BDT App can be investigated. The most important parameter is the routing algorithm, followed by network reliability. These relate directly to data latency, throughput and data loss in the simulation. The third parameter selected for study is the source node selection. The source nodes control how much data can be injected into the network. The interaction and effect of these three parameters is explored in the remainder of this section. The results provide enough information to illustrate the performance of the BDT App.



(a) Source and Destination (b) MINHOPS (c) MINTRANSFERS

Figure 17: Routes Generated per Algorithm

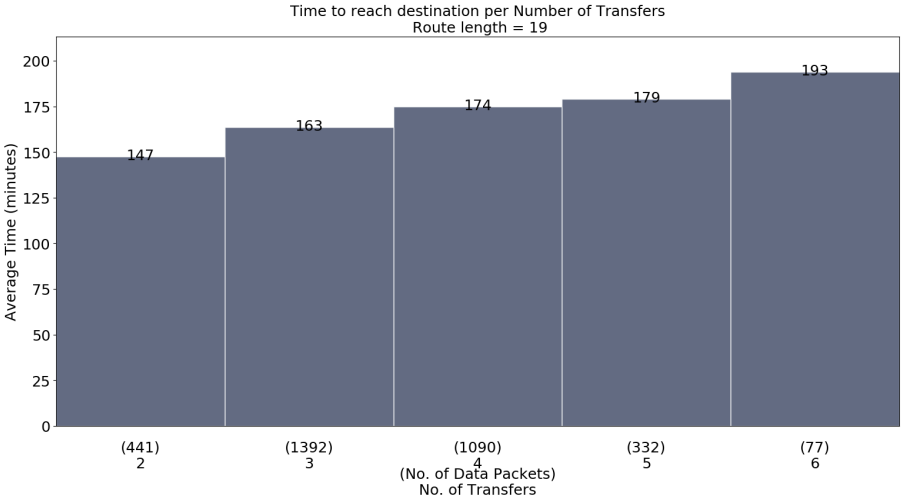
6.3.1 Routing Algorithm

The routing algorithms are first compared by generating routes between a single source-destination bus stop pair. This allows the examination of the characteristics of the two algorithms on a smaller example. Next, routes are computed between 870 randomly selected pairs of bus stops to demonstrate that the properties of the two algorithms are generalizable. Figure 17 shows the source and destination bus stops and the routes computed between these stops by the MINHOPS and MINTRANSFERS routing algorithms (MINEDGES is not shown as it produces routes very similar to routes generated by MINHOPS).

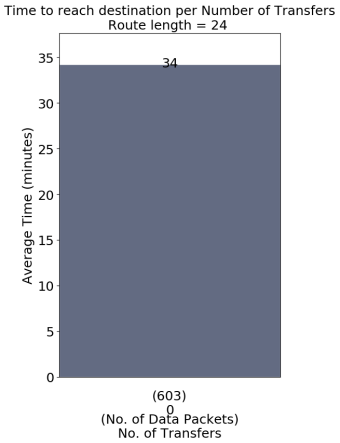
The MINHOPS algorithm allows data to jump on any bus that is travelling to the next stop on that data packet's pre-computed route, while trying to minimize the number of links traversed. MINHOPS puts no constraints on the number of times the data may have to transfer to another bus line in order to match the MINHOPS route's bus stop suggestions. Conversely, MINTRANSFERS focuses on minimizing the number of bus transfers and it provides strict instructions on which bus lines and bus stops the data must travel through.

Figure 18 demonstrates how data travels using the two algorithms. MINHOPS has allowed the injection of 3332 packets of data while MINTRANSFERS has allowed 603 packets. Data travelling using the MINHOPS algorithm has followed the same route but using different buses resulting in different delivery times. Data has made between 2 and 6 transfers using MINHOPS, and the number of transfers correlates with the time taken for delivery, with an average delivery time of 167 minutes. On the other hand, MINTRANSFERS has found a route to the destination that requires no transfers. The route is longer than MINHOPS (24 stops vs 19 stops) but the average time to reach the destination is 34 minutes.

In general, MINTRANSFERS generates longer routes but lower delivery times.



(a) MINHOPS Delivery Times



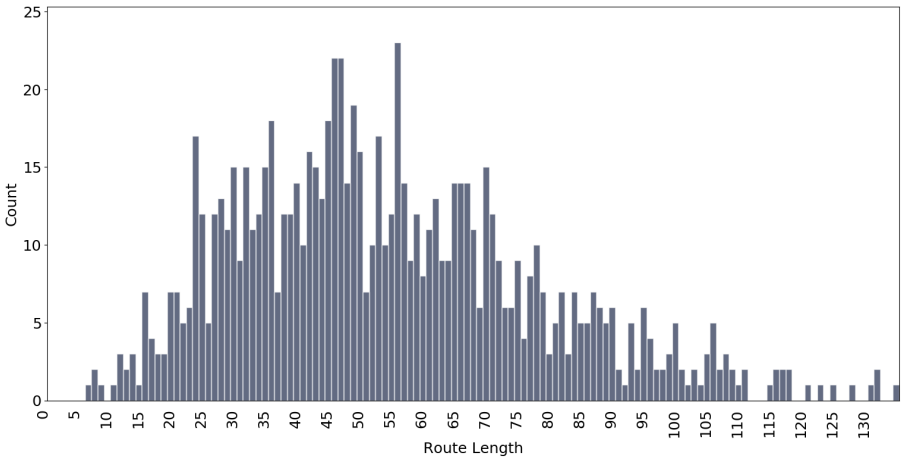
(b) MINTRANSFERS Delivery Time

Figure 18: Delivery Times per Algorithm

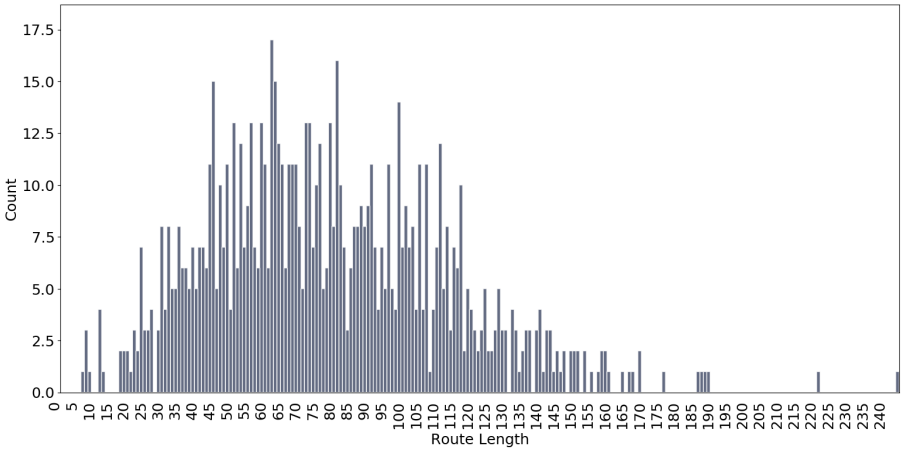
MINHOPS generates a shorter route length but data takes different buses and makes many transfers depending on which bus it comes across first while it is waiting at a stop. This results in longer delivery times. Each change of bus requires data to disembark, wait for a bus and then board. This can add minutes to the journey at each transfer. A greater amount of data can enter the bus network when using the MINHOPS algorithm as data can hop onto any bus going in the right direction (that is, the next stop on the data's route). In MINTRANSFERS data must wait for a bus from a specific bus line to arrive before it can start its journey, consequently, the data has fewer opportunities to enter the network. To summarize, the routing decisions affect the data production and delivery times in the BDT App simulation.

Figure 19 shows the distribution of route lengths for the two algorithms with a randomly selected set of 30 bus stops corresponding to around 870 routes. MINHOPS

average route length is 53 and MINTRANSFERS average route length is 77.



(a) MINHOPS Route Length Distribution



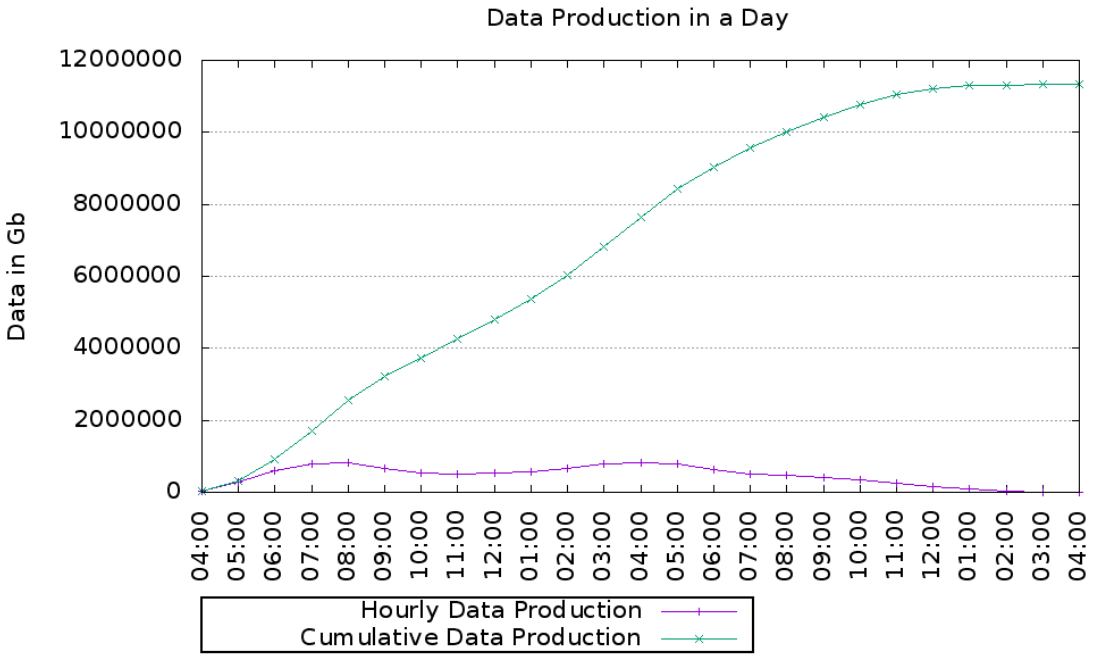
(b) MINTRANSFERS Route Length Distribution

Figure 19: Route Lengths per Algorithm

6.3.2 Theoretical Maximum Data Production Capacity

It is interesting to explore the maximum amount of data that could theoretically enter the BDT App simulation if all 6390 bus stops always have data to send and every bus stops at every bus stop for the maximum dwell time. The maximum dwell time is 18.75 seconds which results in approximately 11 million Gigabits (1400 TB) of data production in a day. Figure 20 shows how much data enters the simulation at every hour and the cumulative amount of data produced. This provides an upper limit to the amount of data that can enter the bus network. However, the BDT App is not able to support all 6390 bus stops as source nodes as the simulation runs out of memory before it can complete. Consequently, a smaller selection of source nodes is used to perform the experiments described next.

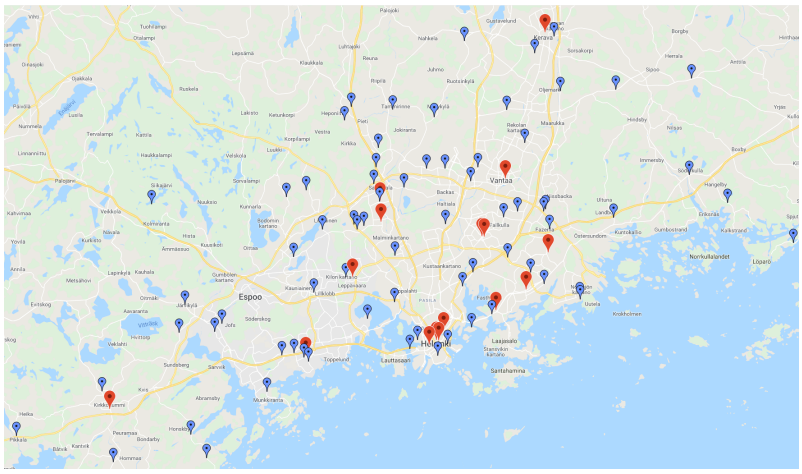
Figure 20: Theoretical Maximum Data Production



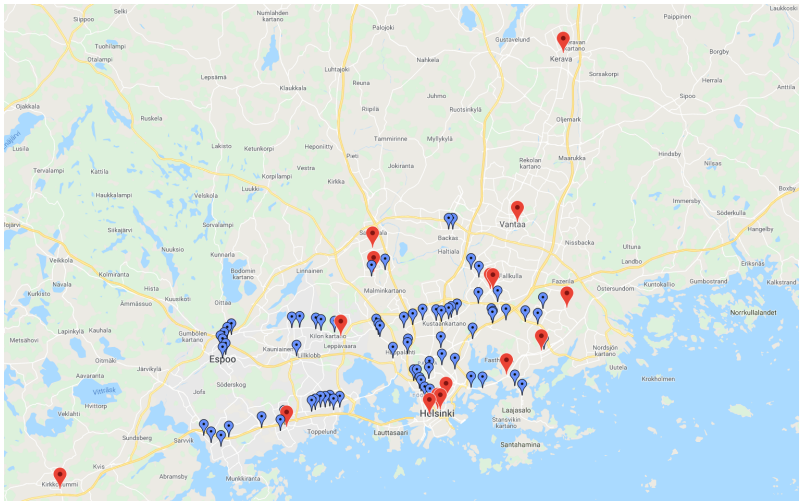
6.3.3 Selection of Source and Destination Nodes

A key parameter influencing the performance of the BDT App is the selection of source and destination nodes. As decided in the design for the BDT App, nodes located at hubs will serve as the destination nodes. The hubs have been selected using the bus stop heat map from Section 5. A source bus stop will try to send data to a bus stop in the physically nearest hub area.

The simulation experiments are conducted with 75 randomly selected stops and 75 stops selected on the basis of their high bus traffic. The high traffic nodes have been randomly selected out of those bus stops having between 300 and 500 buses visiting them during a 24 hour period. These bus stops are located along major roads in the urban areas of the HSL region, while the randomly selected nodes are spread evenly throughout the region. Figure 21 shows the selected source bus stops with blue markers and the hub locations with red markers.



(a) 75 Randomly Selected Bus Stops



(b) 75 High Frequency Traffic Bus Stops

Figure 21: Selection of Source Nodes

6.3.4 Reliable Network: Data Production, Delivery Ratio and Delivery Time

In this section the BDT App simulation is run on a set of source and destination bus stops and the data produced and delivered by each of the routing algorithms is compared. The simulation settings for experiments 1 and 2 simulate a reliable network, that is, a bus stops at every bus stop for a random dwell time and all data that wants to disembark a bus is able to do so at every stop. With these settings data cannot be delayed due to missed bus stops or remain stranded on buses because of missed transfers. The data produced and delivered by each of the routing algorithms can be studied without the effect of missed connections in the network.

Experiment 1: Reliable Network, 75 Randomly Selected Source Nodes The selected bus stops produce data between the times of 08:00 - 10:00. Figure 22 shows the amount of data produced and delivered using the MINHOPs and MINTRANSFERS simulations. MINHOPS routing allows the injection of almost double the data injected by MINTRANSFERS. MINHOPS also delivers 91% of the data within the same day while MINTRANSFERS is able to deliver 72%. No data needs to be re-routed as the simulation settings make the network reliable in terms of transfer opportunities as described earlier. Table 8 summarizes the behavior of the two routing algorithms.

Figure 22: Reliable Network, Random Source Nodes

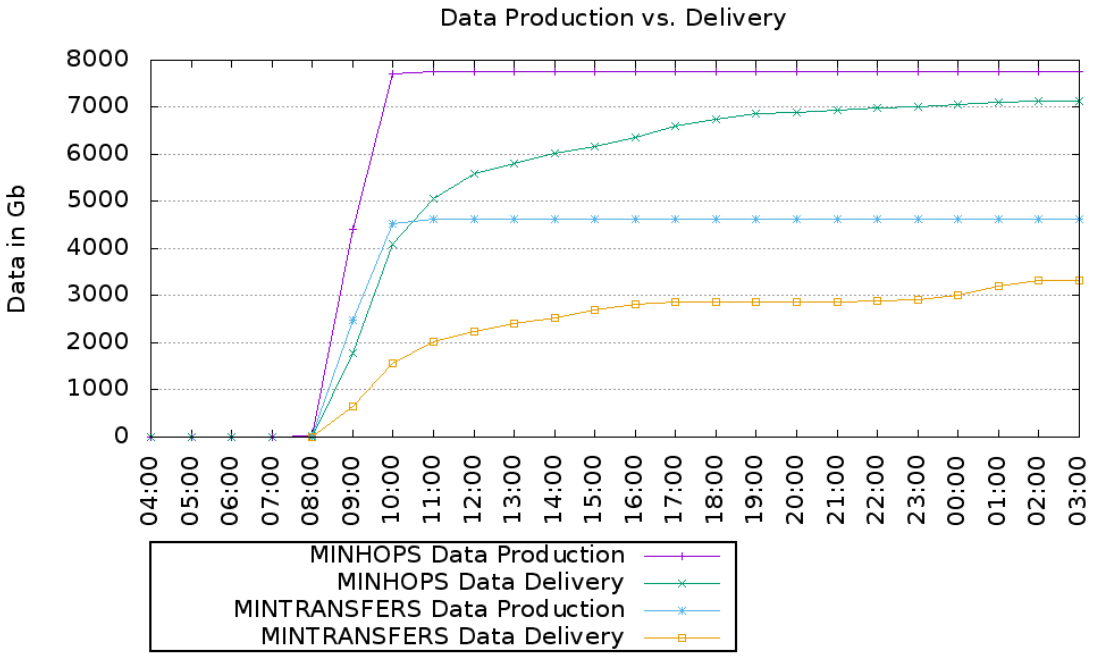


Table 8: Routing Algorithms: Reliable Network, Random Source Nodes

	MINHOPS	**MINTRANSFERS**
Average Route Length	18	25
Route Lengths vary between	1 - 58	2 - 82
Transfers vary between	0 - 7	0 - 3
Data Production	7754.4 Gb	4619.9 Gb
Data Delivered	91%	72%
Average Delivery Time (min)	110	120

Experiment 2: Reliable Network, 75 High Traffic Source Nodes Both routing algorithms allow the injection of more data into the network with MINHOPS allowing 4 times more data than MINTRANSFERS. This is because of the greater bus traffic at these nodes. The amount of data seems to have an effect on the delivery ratio with MINHOPS now delivering only 78% by day end and MINTRANSFERS delivering 94% of the data produced. The performance of MINTRANSFERS is greatly improved here. This is explained by the fact that high traffic nodes are often much closer to hub nodes than randomly selected source nodes. And related to this closer proximity is that MINTRANSFERS is able to find a low transfers route to a hub node. Figure 23 and Table 9 summarize the results.

Figure 23: Reliable Network, High Traffic Source Nodes

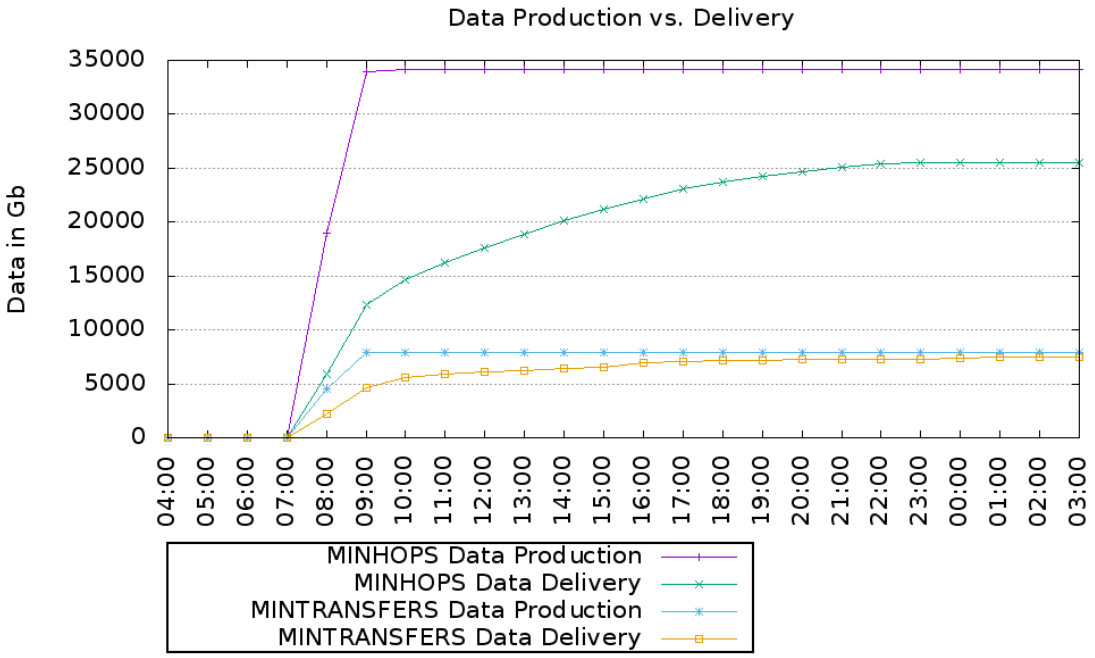


Table 9: Routing Algorithms: Reliable Network, High Traffic Source Nodes

	MINHOPS	**MINTRANSFERS**
Average Route Length	10	17
Route Lengths vary between	1 - 22	2 - 66
Transfers vary between	0 - 3	0 - 2
Data Production	26933.6 Gb	7944.1 Gb
Data Delivered	78%	94%
Average Delivery Time (min)	61	55

6.3.5 Unreliable Network: Data Production, Delivery Ratio and Delivery Time

The previous experiments simulated a reliable network, where all data was able to follow its assigned route as it always had the opportunity to board or disembark a bus at any bus stop. In the following experiments the simulation parameters are manipulated to make the network less reliable. Buses have a 10% chance of not stopping at a stop. This means that data will now be affected by missed buses and missed stops. The delivery ratio in the same time-frame can be expected to decrease while the delivery times can be expected to increase.

Experiment 3: Unreliable Network, 75 Randomly Selected Source Nodes
The same experimental setup as Experiment 1 is used. Figure 24 shows the data production and delivery patterns remain the same as in previous experiments. Table 10 summarizes some experiment metrics. As expected the delivery times have increased while delivery ratio has decreased for both routing algorithms. The percentage of data packets that had to be re-routed is more for MINHOPS. This is because data traveling using MINHOPS routes makes bus transfers more frequently than data traveling by MINTRANSFERS. When coupled with an unreliable network, MINHOPS data packets will have a higher chance of going off their route. Thus the higher number of re-routing events needed versus MINTRANSFERS data.

Figure 24: Unreliable Network, Random Source Nodes

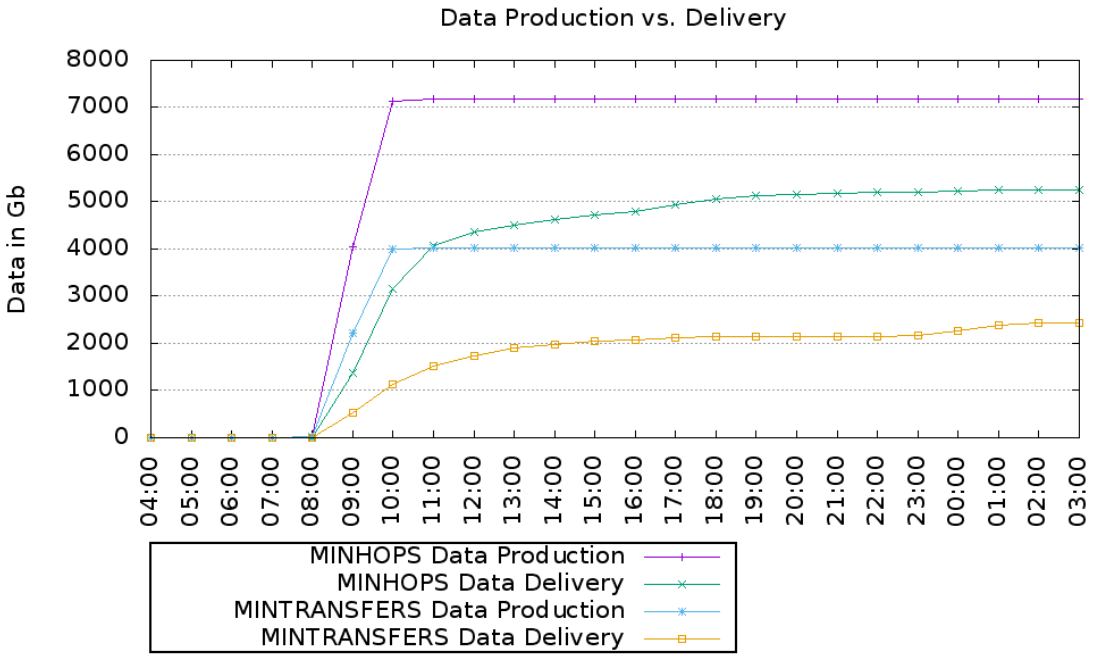


Table 10: Routing Algorithms: Unreliable Network, Random Source Nodes

	MINHOPS	**MINTRANSFERS**
Transfers vary between	0 - 9	0 - 3
Data Production	7171.2 Gb	5254.8 Gb
Data Delivered	73%	60%
Average Delivery Time (min)	125	120
% Data Re-routed	18%	9%

Experiment 4: Unreliable Network, 75 High Traffic Source Nodes The same experimental setup as Experiment 2 is used, and the same trends as seen in Experiment 3 are repeated. The performance degradation and re-routing events are similar in size and direction. Figure 25 and Table 11 summarize the results.

Figure 25: Unreliable Network, High Traffic Source Nodes

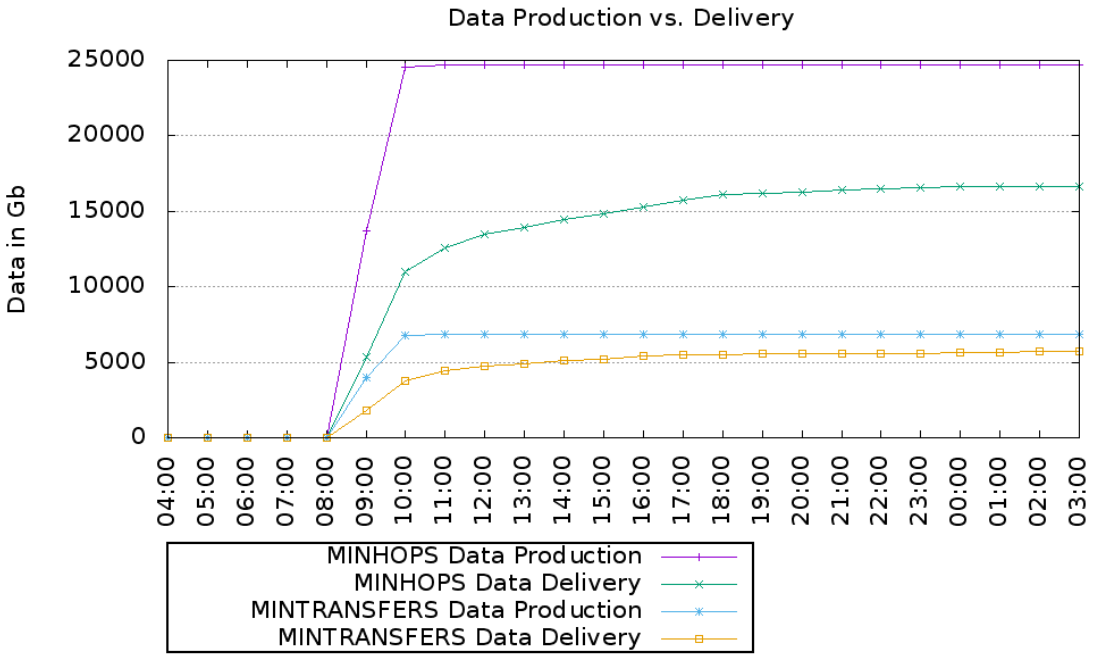


Table 11: Routing Algorithms: Unreliable Network, High Traffic Source Nodes

	MINHOPS	**MINTRANSFERS**
Transfers vary between	0 - 7	0 - 2
Data Production	24713.6 Gb	6856.9 Gb
Data Delivered	67%	84%
Average Delivery Time (min)	99	73
% Data Re-routed	14%	7%

6.4 Discussion

The routing algorithm, node selection and network reliability each have an effect on the delivery ratio, delivery time and injection of data in the network. The most important factor is the routing algorithm itself and the kind of routes it produces.

With respect to the routing instructions, the BDT App employs source routing which is very restrictive as the route is computed once and must be strictly followed. Even if the next hop in the route is unavailable and an alternate route exists or a better route appears, the source routing model cannot take advantage of it. A more flexible routing algorithm on a per-hop basis could increase the throughput of the BDT App.

Also related to the routing algorithm are the route length and number of transfers experienced by data traveling on the route. It is intuitive that larger route lengths and more transfers lead to longer delivery times. Transfers seem to have an even stronger negative effect on delivery ratio and time. In MINHOPS, for the same route, the average delivery time increases as the number of transfers experienced increases. In MINTRANSFERS, the number of transfers per route is pre-computed and fixed.

The properties of the bus stop also contribute to the performance of the system. Highly connected and frequented bus stops do better with MINTRANSFERS routing. While randomly selected nodes perform better using MINHOPS routing. This suggests that the routing algorithm may need to be modified depending on the bus stop's characteristics. Additionally, the average distance to a hub node is larger in the randomly selected nodes than the high traffic nodes. This also contributes to the poorer performance of MINTRANSFERS in all the random source node experiments, and superior performance in the high traffic experiments.

The BDT App performance begins to degrade when a certain amount of data enters the system. This is observed when comparing the results in experiments 1 and 2 for the MINHOPS algorithm. Much more data is injected into the network with MINHOPS in experiment 2, which results in a drop in the delivery ratio.

The performance also degrades when the network becomes less reliable. Some amount of data starts to go off its route and needs to be re-routed by assigning it a new path to a hub node. This means the same data is now in the network for a

longer time. A 10% drop in network reliability leads to double the re-routing events in the MINHOPS case than the MINTRANSFERS case in both experiments 3 and 4. MINTRANSFERS is less affected by the drop in network reliability because it produces routes optimized to reduce transfers. And transfer events are the ones that are affected most by missed bus stops.

The preceding experiments demonstrate that data production and delivery are possible using a Public Transport Delay Tolerant Network. The delivery times are in the order of hours, which means the BDT App can be employed for delivering data that does not have time critical constraints. Simulation experiments have demonstrated that the BDT App is able to deliver Gigabytes of data to hub destinations despite the constraints imposed by the restrictive MINHOPS and MINTRANSFERS routing algorithms and other simulation parameters. The simulation results provide a basis for further investigation employing improved routing algorithms and realistic dwell times. In addition, the BDT App is reliable as it does not lose any data. Data is always either stored on a bus or bus stop while it is traveling. From a bus stop, data can always continue its journey on the next day if it was not able to reach its destination on the first day due to the periodic nature of the bus network. Given enough time, data will manage to reach its destination.

7 Conclusions

The research hypothesis for this thesis is that a data transfer network built over the framework of a Public Transport Network of an urban center can be used to provide reliable bulk data delivery throughout the network. This application is referred to as 'Bulk Data Transfer over a Public Transport Network' or the BDT App. The contributions of this thesis are a set of design guidelines for developing BDT Apps, static and time-varying graph analysis of Helsinki's Public Transport Network, and a simulation of this network when used for data transfer.

PTNs already provide good coverage of an urban area and are reliable and predictable which makes them an appropriate candidate for use in data transfer. The PTN of a city can be interpreted as an opportunistic network where nodes are bus-stops and communication between these nodes occurs when a bus travels between two bus-stops. This thesis investigates how such a BDT over PTN application can be designed, and evaluates how well such an application performs by employing both graph analysis and simulation.

The design guidelines for building the BDT App are developed from studying the literature on Delay Tolerant Networks, and then further expanded by analyzing the features of the target transport network; namely, Helsinki's bus transport system. The design decisions are a rough outline of the major decisions that need to be taken to develop a BDT App. They are general enough to potentially be applied to any city setting in order to create a BDT App for it.

The chosen method for testing this hypothesis is primarily network simulation sup-

ported by graph analysis of the network using static and temporal network analysis. The network analysis has been used to set the parameters of the simulation like defining hub nodes, selecting source nodes and providing information to cost functions for computing routes. Using a simulation provides the flexibility of modifying the simulation parameters to test different routing algorithms and network conditions to examine how they affect the throughput of data in the system.

The simulation is used to model the transmission of data through Helsinki's bus network. The interesting parameters studied are the routing algorithm, the selection of data producing nodes and the effect of network reliability. The metrics used for performance comparison are delivery ratio, delivery time, data produced, and data re-routed.

The simulation experiments prove that even with restrictive simulation parameters and conservative estimates of dwell time, data is able to flow through the network. This provides a baseline against which improvements to the simulation can be benchmarked. The experiments also show how the different parameters interact with each other and affect the performance metrics. It is apparent that data can flow from source to hub nodes with relatively high delivery ratios and delivery time in the order of hours. However there is much room for improvement as ideally, delivery rates should be 100% on the first day. The experiments reveal multiple areas where the simulation can be improved and made more realistic.

There are several aspects of the design, simulation and graph analysis that can be developed or further tested. Some interesting future work could include, for example, designing a similar BDT App for a geographically different city. The same design guidelines would apply, the simulation parameters like source and hub nodes could be re-calculated via similar methods as in this thesis. Another line of study is making enhancements to the current simulation by improving the routing algorithm, and/or adding bus to bus interactions in addition to the bus-to-bus-stop interactions. Both these measures would improve the data throughput capability of the BDT App. Bringing the simulation closer to the real world is another aspect that can be developed. For example, using real GPS traces rather than synthetic traces, or at least using better dwell time approximations, either of these would make the simulation more realistic. One more area that has not been explored in this thesis is the actual hardware requirements a data transfer application operating over a PTN would need.

References

ADC16 Amirthavalli, R., Dhaya, R. and Chandrasoodan, M. S., A survey of routing algorithms in delay tolerant networks. *2016 2nd International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT)*, July 2016, pages 469–473.

AK07 Ahmed, S. and Kanhere, S. S., Cluster-based forwarding in delay tolerant public transport networks. *32nd IEEE Conference on Local Computer Networks (LCN 2007)*, Oct 2007, pages 625–634.

Aka17 Akamai, Akamai's state of the internet - connectivity report. 10,1(2017). URL `https://www.akamai.com/us/en/about/our-thinking/state-of-the-internet-report/global-state-of-the-internet-connectivity-reports.jsp`.

AS10 Ahmed, S. and Salil, S. K., Characterization of a large-scale delay tolerant network. *IEEE Local Computer Network Conference*, Oct 2010, pages 56–63.

BGJL06 Burgess, J., Gallagher, B., Jensen, D. and Levine, B. N., Maxprop: Routing for vehicle-based disruption-tolerant networks. *Proceedings IEEE INFOCOM 2006. 25TH IEEE International Conference on Computer Communications*, April 2006, pages 1–11.

BLV07 Balasubramanian, A., Levine, B. and Venkataramani, A., Dtn routing as a resource allocation problem. *SIGCOMM Comput. Commun. Rev.*, 37,4(2007), pages 373–384. URL `http://doi.acm.org/10.1145/1282427.1282422`.

CFQS10 Casteigts, A., Flocchini, P., Quattrociocchi, W. and Santoro, N., Time-varying graphs and dynamic networks. *CoRR*, abs/1012.0009. URL `http://arxiv.org/abs/1012.0009`.

CJ03 Clausen, T. and Jacquet, P., Optimized link state routing protocol (olsr). RFC 3626, RFC Editor, October 2003. URL `http://www.rfc-editor.org/rfc/rfc3626.txt`. `http://www.rfc-editor.org/rfc/rfc3626.txt`.

CM99 Corson, S. and Macker, J., Mobile ad hoc networking (manet): Routing protocol performance issues and evaluation considerations. RFC 2501, RFC Editor, January 1999.

CS13 Cao, Y. and Sun, Z., Routing in delay/disruption tolerant networks: A taxonomy, survey and challenges. *IEEE Communications Surveys Tutorials*, 15,2(2013), pages 654–677.

DHHL11 Dimatteo, S., Hui, P., Han, B. and Li, V., Cellular traffic offloading through wifi networks. *Mobile Adhoc and Sensor Systems (MASS), 2011 IEEE 8th International Conference on*, Oct 2011, pages 192–201.

dPSC16 del Pilar Salamanca, M. and Camargo, J., A survey on ieee 802.11-based manets and dtns for survivor communication in disaster scenarios. *2016 IEEE Global Humanitarian Technology Conference (GHTC)*, Oct 2016, pages 197–204.

Fal03 Fall, K., A delay-tolerant network architecture for challenged internets. *Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, SIGCOMM '03, New York, NY, USA, 2003, ACM, pages 27–34, URL http://doi.acm.org/10.1145/863955.863960.

GMRS12 Gaito, S., Maggiorini, D., Rossi, G. and Sala, A., Bus switched networks: An ad hoc mobile platform enabling urban-wide communications. *Ad Hoc Networks*, 10,6(2012), pages 931 – 945. URL http://www.sciencedirect.com/science/article/pii/S1570870511002204.

gtf Hsl gtfs, https://transitfeeds.com/p/helsinki-regional-transport/735. Accessed: 2019-03-18.

GVOM13 Galati, A., Vukadinovic, V., Olivares, M. and Mangold, S., Analyzing temporal metrics of public transportation for designing scalable delay-tolerant networks. *Proceedings of the 8th ACM Workshop on Performance Monitoring and Measurement of Heterogeneous Wireless and Wired Networks*, PM2HW2N '13, New York, NY, USA, 2013, ACM, pages 37–44, URL http://doi.acm.org/10.1145/2512840.2512846.

HBDM14 Harnie, D., Boix, E. G., D'hondt, T. and Meuter, W. D., Programming urban-area applications by exploiting public transportation. *ACM Trans. Auton. Adapt. Syst.*, 9,2(2014), pages 8:1–8:20. URL http://doi.acm.org/10.1145/2619999.

HCY11 Hui, P., Crowcroft, J. and Yoneki, E., Bubble rap: Social-based forwarding in delay-tolerant networks. *IEEE Transactions on Mobile Computing*, 10,11(2011), pages 1576–1589.

HS12 Holme, P. and Saramäki, J., Temporal networks. *Physics Reports*, 519,3(2012), pages 97 – 125. URL http://www.sciencedirect.com/science/article/pii/S0370157312000841. Temporal Networks.

HSL Helsinki regional transport authority, https://www.hsl.fi/en/helsinki-regional-transport-authority. Accessed: 2019-02-08.

JFP04 Jain, S., Fall, K. and Patra, R., Routing in a delay tolerant network. *SIGCOMM Comput. Commun. Rev.*, 34,4(2004), pages 145–158. URL http://doi.acm.org/10.1145/1030194.1015484.

kal Helsinki regional transport authority, http://developer.matka.fi/pages/en/kalkati.net-xml-database-dump.php. Accessed: 2019-02-08.

KOK09 Keränen, A., Ott, J. and Kärkkäinen, T., The ONE Simulator for DTN Protocol Evaluation. *SIMUTools '09: Proceedings of the 2nd International Conference on Simulation Tools and Techniques*, New York, NY, USA, 2009, ICST.

KT13 Komnios, I. and Tsaoussidis, V., Carpool: Extending free internet access over dtn in urban environment. *Proceedings of the 2013 ACM MobiCom Workshop on Lowest Cost Denominator Networking for Universal Access*, LCDNet '13, New York, NY, USA, 2013, ACM, pages 21–24, URL `http://doi.acm.org/10.1145/2502880.2502891`.

LDS03 Lindgren, A., Doria, A. and Schelén, O., Probabilistic routing in intermittently connected networks. *SIGMOBILE Mob. Comput. Commun. Rev.*, 7,3(2003), pages 19–20. URL `http://doi.acm.org/10.1145/961268.961272`.

LH09 Lindgren, A. and Hui, P., The quest for a killer app for opportunistic and delay tolerant networks: (invited paper). *Proceedings of the 4th ACM Workshop on Challenged Networks*, CHANTS '09, New York, NY, USA, 2009, ACM, pages 59–66, URL `http://doi.acm.org/10.1145/1614222.1614233`.

NTM+13 Nicosia, V., Tang, J., Mascolo, C., Musolesi, M., Russo, G. and Latora, V., Graph metrics for temporal networks. In *Temporal Networks*, Holme, P. and Saramäki, J., editors, Understanding Complex Systems, Springer Berlin Heidelberg, 2013, pages 15–40, URL `http://dx.doi.org/10.1007/978-3-642-36461-7_2`.

PBRD03 Perkins, C., Belding-Royer, E. and Das, S., Ad hoc on-demand distance vector (aodv) routing. RFC 3561, RFC Editor, July 2003. URL `http://www.rfc-editor.org/rfc/rfc3561.txt`. `http://www.rfc-editor.org/rfc/rfc3561.txt`.

PFH04 Pentland, A., Fletcher, R. and Hasson, A., Daknet: rethinking connectivity in developing nations. *Computer*, 37,1(2004), pages 78–83.

PK15 Park, H.-s. and Kim, J.-d., Wi-fi-based modeling and hybrid routing scheme for delay-tolerant public bus network. *Wirel. Commun. Mob. Comput.*, 15,7(2015), pages 1117–1130. URL `http://dx.doi.org/10.1002/wcm.2392`.

PPC06 Pelusi, L., Passarella, A. and Conti, M., Opportunistic networking: data forwarding in disconnected mobile ad hoc networks. *IEEE Communications Magazine*, 44,11(2006), pages 134–141.

PS11 Pan, R. K. and Saramäki, J., Path lengths, correlations, and centrality in temporal networks. *Phys. Rev. E*, 84, page 016105. URL `http://link.aps.org/doi/10.1103/PhysRevE.84.016105`.

SLL+08 Sede, M., Li, X., Li, D., Wu, M. Y., Li, M. and Shu, W., Routing in large-scale buses ad hoc networks. *2008 IEEE Wireless Communications and Networking Conference*, March 2008, pages 2711–2716.

SPR04 Spyropoulos, T., Psounis, K. and Raghavendra, C. S., Single-copy routing in intermittently connected mobile networks. *2004 First Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks, 2004. IEEE SECON 2004.*, Oct 2004, pages 235–244.

SPR05 Spyropoulos, T., Psounis, K. and Raghavendra, C. S., Spray and wait: An efficient routing scheme for intermittently connected mobile networks. *Proceedings of the 2005 ACM SIGCOMM Workshop on Delay-tolerant Networking*, WDTN '05, New York, NY, USA, 2005, ACM, pages 252–259, URL `http://doi.acm.org/10.1145/1080139.1080143`.

SQF+11 Santoro, N., Quattrociocchi, W., Flocchini, P., Casteigts, A. and Amblard, F., Time-varying graphs and social network analysis: Temporal indicators and metrics. *CoRR*, abs/1102.0629. URL `http://arxiv.org/abs/1102.0629`.

SRC11 Socievole, A., Rango, F. D. and Coscarella, C., Routing approaches and performance evaluation in delay tolerant networks. *2011 Wireless Telecommunications Symposium (WTS)*, April 2011, pages 1–6.

TCCM15 Tornell, S. M., Calafate, C. T., Cano, J. and Manzoni, P., Dtn protocols for vehicular networks: An application oriented overview. *IEEE Communications Surveys Tutorials*, 17,2(2015), pages 868–887.

TMM+10 Tang, J., Musolesi, M., Mascolo, C., Latora, V. and Nicosia, V., Analysing information flows and key mediators through temporal centrality metrics. *Proceedings of the 3rd Workshop on Social Network Systems*, SNS '10, New York, NY, USA, 2010, ACM, pages 3:1–3:6, URL `http://doi.acm.org/10.1145/1852658.1852661`.

TMML09 Tang, J., Musolesi, M., Mascolo, C. and Latora, V., Temporal distance metrics for social network analysis. *Proceedings of the 2Nd ACM Workshop on Online Social Networks*, WOSN '09, New York, NY, USA, 2009, ACM, pages 31–36, URL `http://doi.acm.org/10.1145/1592665.1592674`.

VB+00 Vahdat, A., Becker, D. et al., Epidemic routing for partially connected ad hoc networks.

WLW+17 Wang, T., Li, P., Wang, X., Wang, Y., Guo, T. and Cao, Y., A comprehensive survey on mobile data offloading in heterogeneous network. *Wireless Networks*. URL `https://doi.org/10.1007/s11276-017-1576-0`.

Zha06 Zhang, Z., Routing in intermittently connected mobile ad hoc networks and delay tolerant networks: overview and challenges. *IEEE Communications Surveys Tutorials*, 8,1(2006), pages 24–37.