M.Sc. Engg. Thesis

A Novel Congestion Control Algorithm for Delay Tolerant Networks

by
Rajkumar Das

Submitted to

Department of Computer Science and Engineering
in partial fulfilment of the requirements for the degree of
Master of Science in Computer Science and Engineering

Department of Computer Science and Engineering
Bangladesh University of Engineering and Technology (BUET)
Dhaka 1000

May 2011

The thesis titled "**A Novel Congestion Control Algorithm for Delay Tolerant Networks**," submitted by Rajkumar Das, Roll No. 040805022P, Session April 2008, to the Department of Computer Science and Engineering, Bangladesh University of Engineering and Technology, has been accepted as satisfactory in partial fulfillment of the requirements for the degree of Master of Science in Computer Science and Engineering and approved as to its style and contents. Examination held on May 04, 2011.

Board of Examiners

1. _____

Dr. Md. Humayun Kabir — Chairman (Supervisor)
Associate Professor
Department of Computer Science and Engineering
BUET, Dhaka 1000

2. _____

Dr. Md. Monirul Islam — Member (Ex-officio)
Professor & Head
Department of Computer Science and Engineering
BUET, Dhaka 1000

3. _____

Dr. Md. Reaz Ahmed — Member
Associate Professor
Department of Computer Science and Engineering
BUET, Dhaka 1000

4. _____

Dr. Syed Faisal Hasan — Member (External)
Assistant Professor
Department of Computer Science and Engineering
Dhaka University, Dhaka 1000

Candidate's Declaration

It is hereby declared that this thesis or any part of it has not been submitted elsewhere for the award of any degree or diploma.

Rajkumar Das
Candidate

Contents

List of Figures

List of Algorithms

Acknowledgments

I express my heart-felt gratitude to my supervisor, Dr. Md. Humayun Kabir for his constant supervision of this work. He helped me a lot in every aspect of this work and guided me with proper directions whenever I sought one. His patient hearing of my ideas, critical analysis of my observations and detecting flaws (and amending thereby) in my thinking and writing have made this thesis a success.

I would also want to thank the members of my thesis committee for their valuable suggestions. I thank Professor Dr. Md. Monirul Islam, Dr. Md. Reaz Ahmed, and specially the external member Dr. Syed Faisal Hasan.

I also thank one of my friend Md. Anindya Tahsin Prodhan, who was my partner in the early works of my research. He was always there for me when I needed.

In this regard, I remain ever grateful to my beloved parents, who always exists as sources of inspiration behind every success of mine I have ever made.

Abstract

Due to intermittent connectivity and lack of persistent end-to-end path in Delay Tolerant Networks (DTNs), routing protocols adopt hop-by-hop store and forward mechanism to forward bundles towards the destination. The responsibility of reliable transmission is also delegated in a hop-by-hop fashion using custody transfer. A node accepting the custody of a bundle carries the liability of delivering or delegating it to the next hop. Therefore, the node is required to keep the bundles in a persistent storage. Due to limited storage capacity in the DTN nodes, storage congestion occurs when too many bundles are contending for this scarce resource. As a result, bundle loss increases and the delivery ratio of the network suffers.

As a reactive congestion mitigation scheme, a congested node can migrate some of its bundles to the uncongested neighbors in order to make room for the new bundles and pull back the bundles in its own storage when the congestion disappears. This scheme fails when the neighbors themselves are congested. As a proactive scheme, a node can decide whether to accept or reject the custody of a new bundle based on its current storage status.

There are many proactive schemes available in the literature. However, their performance can be improved further by introducing new features. In this thesis, we propose a novel proactive congestion control scheme for DTN. Our scheme ranks the incoming bundles using their priority and TTL. When congestion occurs, a node accepts or rejects the bundles based on their rank. We also introduce the concept of head of line blocking, confidence level, and dropping probability of the bundles to facilitate better decision making. Simulation results show that our scheme achieves 20% to 25% higher delivery ratio than that of the existing schemes.

Chapter 1

Introduction

Delay Tolerant Networks (DTNs) [2], [9] is a type of overlay network where the underlying networks suffer from intermittent connectivity (communication opportunities) due to scheduled or opportunistic contacts and host mobility. This leads to frequent network partitioning, lack of persistent end-to-end path between any two hosts, and/or highly variable link performance. Therefore, traditional Internet and Mobile Ad-hoc Network (MANET) protocols and algorithms cannot be directly applied on networks in this category. With the emergence of many real life DTNs, adaptation/modification of conventional network protocols to suit them in DTN and/or the development of the new protocols for DTN have drawn a lot of attention from the researchers of network community.

1.1 DTN Congestion

DTN nodes transmit data among themselves as the bundles rather than as the byte streams or the datagram. A bundle is composed of the whole application data. However, a bundle can be fragmented when the contact between two nodes cannot handle the original bundle as a whole. To route bundles from the source to the destination, DTN adopts a hop-by-hop store and forwarding mechanism which eliminates the requirement of persistent end-to-end path between them. Reliable delivery of bundles is also ensured by

a hop-by-hop technique called custody transfer. A DTN node not only accepts a bundle but also the responsibility of the reliable delivery of the bundle. This responsibility is called the custody of a bundle and the node holding this custody is known as the current custodian of the bundle. The node holds the bundle until it delegates the custody along with the bundle to the next custodian node. Therefore, a node is required to keep a bundle in a persistent storage. A new type of congestion arises if the amount of persistent storage in a node is limited and there are too many bundles to contend for this storage. This form of congestion is known as storage congestion and DTN suffers from very poor delivery ratio and delay, if it is not properly managed. Therefore, it is necessary to develop effective schemes to mitigate the problems. This is why congestion control in DTN stimulated keen interest in the researcher. Recent survey in [17] shows that congestion control is still an important and open issue that need to be further investigated.

1.2 Congestion Control in DTN

There are mainly two approaches to control congestion in DTN. One is called reactive; the other one is called proactive. In reactive approach, the current custodian node initiates the congestion recovery process. When congestion becomes evident, a congested node initiates a data migration process to get rid of it. The node selects a set of bundles from its persistent storage and a set of uncongested nodes from its neighbors, and forwards the selected bundles to those neighbors. Therefore, the node creates some rooms in its storage to accept and store the new bundles. Here, the node still holds the custody of the migrated bundles as if those bundles were kept in its own storage though they have been physically moved to the neighbor nodes. When the congestion disappears, the custodian node revokes all the migrated bundles. Seligman el. al. proposed a congestion control scheme [19] based on the reactive approach where they introduced the push-pull form of custody transfer. The push operation initiates the data migration process by selecting the bundles and the neighbor nodes, whereas the pull operation brings the bundles back. This approach ensures high delivery ratio but introduces much overheads and delays. It

also suffers from bundle loss when the neighbors themselves are congested.

The other approach takes proactive decisions to control congestion. The nodes manage a controlled allocation of its persistent storage to the incoming bundles. For each inbound bundle, a decision is made; the node either accepts or rejects the custody of a new bundle based on its properties and the current congestion level perceived by the node. When congestion in the network becomes evident, nodes rejects custody of the new bundles in order to restrain the downstream nodes from forwarding bundles upward. This process goes all the way back to the source nodes, thus, restraining them from injecting more bundles in the already congested network. When congestion fades away, the nodes start accepting bundles and it releases the whole path. Proactive approaches introduce less overhead and delay.

Congestion control schemes in [4] and [24] proactively decide whether to accept or reject a bundle. For each incoming bundle, these schemes take acception or rejection decisions by using some properties of the bundle (priority, TTL, size), properties of the node (Storage Space), and the congestion status of the network. These schemes have less overhead and delay compared to that of the scheme in [19]. However, they sometime take suboptimal decisions and reject bundles when it is unnecessary to do so.

1.3 Motivation

Although the proactive schemes do not suffer from the high overhead and delay issues like reactive schemes their performance can be further improved by introducing new and effective features in the decision making process. Head of Line Blocking (HLB) [11] is a regular phenomenon in DTN which affects the delivery ratio, delay, and fairness. HLB can be used with other parameters while acceptance or rejection decisions for the new bundles are taken in the existing proactive congestion control schemes. In this thesis, we propose a new proactive congestion control scheme using HLB in the above decision making process in order to achieve higher performance and to ensure fairness in DTN.

We introduce new metric cash-in-hand which calculates the current level of congestion in the network and defer the decision of rejecting a bundle until the cash-in-hand falls below a certain level. Queuing delay of a node is used to provide an immediate response to a bundle with smaller TTL. We also introduce the dropping probability of a bundle and the confidence level of a node to defer the rejection decision as later as possible. To evaluate the effectiveness of our scheme, we perform simulations using DTN SIM2 [1] and compare the delivery ratios and delay of our scheme with that of all the other existing protocols. Simulation results show that our scheme achieves 20% to 25% higher delivery ratios than that of the existing proactive approaches while keeping the delay minimum.

1.4 Deposition

The rest of the thesis is organized as follows. In Chapter 2, we introduce Delay Tolerant Network (DTN) briefly. In Chapter 3, we present a summary of congestion control schemes for DTN. We describe our congestion control scheme in Chapter 4. Chapter 5 presents experimental results and their analysis. Finally, Chapter 6 concludes this thesis with some future research directions.

Chapter 2

DTN Basics

DTN [2], [9] is an overlay network on top of some regional networks. The regional networks, which are subject to disruption and disconnection and high-delay, are connected by DTN. The networks with such characteristics are Interplanetary Internet or deep space networks, sensor-based networks using scheduled intermittent connectivity, terrestrial wireless networks that cannot ordinarily maintain end-to-end connectivity, satellite networks with moderate delays and periodic connectivity, and underwater acoustic networks with moderate delays and frequent interruptions due to environmental factors.

Though the Internet interconnects multiple networks with diverse communication characteristics, it has some basic assumptions that are not complied by the DTN regional networks. Those assumptions are as follows:

1. Continuous, bidirectional, end-to-end path are available between source and destination.

2. Relatively short and consistent round-trip delay between the source and destination.

3. Apparently symmetric data rate in both directions between the source and destination.

4. Comparatively low error-rate in data transmission links between the source and the destination.

DTN regional networks are rather characterized by [9],[7]:

1. Intermittent Connectivity.
2. Long or variable delays.
3. Asymmetric data rates.
4. High error rates.

DTN introduces a new protocol layer called "Bundle" to deal with the special characteristics of these challenged regional networks. The Bundle Layer [18] is designed to operate above the transport layer of OSI model to send the bundles from the source to the destination. Here, a bundle is a whole message from an application with the bundle protocol header. To overcome the issues associated with intermittent connectivity, long or variable delay, asymmetric data rates, and high error rates bundle layer uses store-and-forward message switching to transfer a bundle from one node to another node.

As opposed to establishing a TCP like connection, a Bundle sender sends application messages encapsulated into bundles in a similar fashion that a UDP sender encapsulates user data into UDP datagram. In order to provide communication reliability, bundle protocol uses a concept called custody transfer. Custody transfer implements node-to-node or hop-by-hop retransmission of lost or corrupt data.

2.1 DTN Characteristics

As mentioned above, DTNs are specially characterized by their intermittent connectivity, unpredictable and long delay, asymmetric data rate, and high error rate [2], [9], [7]. In this section, we discuss each of these characteristics briefly.

Intermittent Connectivity: Network partitioning due to, channel fading, line of sight loss, mobility, power conservation strategy, and security measures can cause intermittent connectivity between the source and the destination. This disruption in

network connectivity can happen at a set schedule. For example, everything is in motion in the deep space network. Communicating nodes move along predictable paths and predict or receive time schedule of their future positions and communications. This sort of contact among the nodes in DTNs is called scheduled contact. Network disruption may happen all of sudden or the nodes in the network may connect by chance and at an unscheduled time, this is called opportunistic contact in DTN. DTNRG has defined other types of contacts in [7], such as persistent contacts, on-demand contacts, and predicted contacts. We will discuss these contacts later in this paper.

In the traditional network intermittent connectivity is treated as network fault and the information is dropped when the connectivity is lost, which causes loss of data. DTN needs to support communication between intermittently connected nodes. For example, scheduled contacts in deep space network involve information exchange between nodes that are not in direct contact. PDA, vehicle, aircraft, or satellite may make contact and exchange information when they happen to be within the line-of-sight or close enough to communicate.

Long or Variable Delays: TCP assumes 500ms maximum round-trip delay between the source and the destination while round-trip time between Earth and Jupiter's moon Europa, for example, run between 66 and 100 minutes. This round-trip delay in Interplanetary Internet is excessively long to defeat TCP protocol. Intermittent connectivity and variable queuing delays in the nodes contribute to make the round-trip delay very much unpredictable. TCP's round-trip delay estimation method becomes ineffective in this highly unpredictable environment.

Asymmetric Data Rates: Concurrent communication in both directions with an approximately symmetric transmission rate between the source and the destination of the traditional networks does not happen in many challenged networks. For example, the communication in NASA's Deep Space Network (DSN) is only one direction. Even if bi-directional communication is possible the data rate in two directions may

vary. For example, uplink and downlink data rate of a satellite communication differs. Asymmetric communication makes the conversational protocols (TCP or SCTP), which are based end-to-end signaling, perform poor.

High Error Rates: Traditional networks are linked by copper wires or optical fiber cables, Bit error rates of copper (10^{-9}) and optical fiber (10^{-12}) are tolerable. On the other hand, most of the links in DTNs are wireless with a high bit error rate (10^{-2} to 10^{-4}). High bit error rate defeats the algorithms of traditional transport protocols, such as TCP or SCTP.

2.2 DTN Contacts

A DTN network can be considered as a multigraph, where vertices (nodes) are interconnected with more than one edge. Edges in this graph are, in general, time-varying with respect to their delay and capacity and directional because of the possibility of one-way connectivity. When an edge between two nodes has zero capacity, it is considered that they are not in contact. Edges may vary between positive and zero capacity. There might be a period of time interval during which the capacity is strictly positive and the delay and capacity are almost constant. This period of time is called a contact. The product of the capacity and the interval is known as a "contact's volume". A DTN node can send, receive, or forward a bundle when it is in contact. It stores the bundles in the persistent storage until it becomes in contact again. If contacts and their volumes are known ahead of time, intelligent routing and forwarding decisions can be made optimally. Several types of contacts are possible in DTNs, such as persistent, on-demand, intermittent-scheduled, intermittent-opportunistic, and intermittent-predicted [18].

Several types of contacts are possible in DTNs -

Persistent Contacts: No connection-initiation action is required to instantiate a persistent contact. Persistent contacts are always on. Internet connection through a DSL or Cable Modem gives a persistent contact.

On-Demand Contacts: Connection-initiation actions are required for on-demand contacts. It remains persistent until terminated. A dial-up connection is an example of an on-demand contact.

Intermittent - Scheduled Contacts: A scheduled contact is an agreement to establish a contact at a particular time for a particular duration. An example of a scheduled contact is a link with a low-earth orbiting satellite.

Intermittent - Opportunistic Contacts: Opportunistic contacts are unscheduled; they rather present themselves unexpectedly. For example, an unscheduled aircraft flying overhead and beckoning, advertising its availability for communication, would present an opportunistic contact.

Intermittent - Predicted Contacts: Predicted contacts are neither scheduled nor unexpected, rather predictable. The prediction of likely contact times and durations are based on a history of previously observed contacts or some other information.

2.3 Bundle Layer Protocol

Each node in a DTN runs a Bundle agent [18]. Bundle agents at DTN hosts accept application data unit and encapsulate them into bundles by adding bundle headers. Bundle routers and gateways forward the received bundles to the next-hop node until the destination DTN hosts are reached or the bundle life-time has expired. Bundle agents at the source and the destination nodes communicate with each other using simple sessions with no handshaking, synchronization, and connection setup. Acknowledgement from the receiving node is not mandatory though it is possible. The protocol just below the bundle layer may be conversational like TCP/SCTP or non-conversational like UDP or minimally-conversational like RTP/RTCP. If the connection between two bundle nodes is persistent, e.g., wired Internet, a conversational protocol, such as TCP, can be employed below the bundle layer. Non-conversational or minimally-conversational lower-layer protocols can be implemented otherwise. Delay Tolerant Network Research Group (DTNRG)

[2] has defined a transport protocol called Licklider Transmission Protocol (LTP) [8] to run below bundle layer in the Interplanetary Internet.

DTN Bundle layer depends on the lower layer protocols to insure the reliability of the communication. Bundle nodes while forwarding bundles terminate transport protocol connections at the bundle layer and the bundle layers are acting as the surrogates for end-to-end sources and destinations. As a result low-delay regions, where conversational lower-layer protocols are active, are isolated at the bundle layer from the long-delay or intermittently connected regions, where any conversational lower-layer protocol fails and non-conversational or minimally-conversational lower-layer protocols are implemented [7].

2.3.1 Custody Transfer in the Bundle Layer

Since no single transport-layer protocol operates end-to-end across a DTN, end-to-end reliability needs to be insured in the bundle layer by requesting custody transfer to the next-hop nodes. The nodes receiving these bundles along the way and agreeing to accept the reliable delivery responsibility are called "custodians". The movement of bundle and its delivery responsibility from one node to another is called a "custody transfer". A custody-transfer-request from a bundle node to its next-hop node asks for the delivery guarantee of the bundle. By accepting a custody-transfer-request a bundle node takes the responsibility of delivering the bundle reliably [18].

Custody transfer allows the source to delegate retransmission responsibility and recover its retransmission-related resources relatively soon after sending a bundle. Not all nodes in a DTN are required by the DTN architecture to accept custody transfers, so it is not a true 'hop-by-hop' mechanism. For example, some nodes may have sufficient storage resources to sometimes act as custodians, but may elect to not offer such services when congested or running low on power.

Custody acceptance by the next hop nodes also moves the points of retransmission progressively forward toward the destination. It minimizes the number of potential retransmission hops, i.e., the additional network load caused by retransmissions and the

total time to reliably deliver a bundle to the destination.

2.3.2 Store and Forward Message Switching

A custodian must store a whole bundle until either another node accepts the custody or the expiration of the bundle's time-to-live. Persistent storage, such as hard disk, is used to store the bundles. All types of DTN nodes need persistent storage to store the bundles until outbound links are available. Once an outbound link is available, a node moves the whole bundle in a single transfer though the fragmentation is possible. This is called store-and-forward bundle switching. Bundle layer usage of store-and-forward message switching to transfer a bundle from one node to another node helps it to overcome the issues associated with intermittent connectivity, long or variable delay, asymmetric data rates, and high error rates in DTN. Store-and-forward message switching together with custody transfer provides with the communication reliability [7].

2.3.3 Fragmentation and Reassembly

A bundle can be fragmented into smaller bundles when the contact volume between two nodes is not sufficient to transfer the large bundle atomically. Fragmented bundles are reassembled into the large bundle in the DTN. Two types of fragmentation/reassembly are possible in DTNs [18].

Proactive Fragmentation

A DTN node may divide a block of application data into multiple smaller blocks and transmit each such block as an independent bundle. In this case, the final destination is responsible for extracting the smaller blocks from incoming bundles and reassembling them into the original larger bundle and, ultimately, ADU. This approach is used primarily when contact volumes are known or predicted in advance.

Reactive Fragmentation

DTN nodes having an opportunistic contact in the DTN graph may fragment a bundle cooperatively when a bundle is only partially transferred. The receiving bundle layer modifies the incoming bundle to indicate it is a fragment, and forwards it normally. The sender may learn that only a portion of the bundle was delivered to the next hop, and send the remaining portion when subsequent contacts become available. Here, the fragmentation process occurs after an attempted transmission has taken place. For this reason, reactive fragmentation is more complicated than the proactive fragmentation, where fragmentation is done before the transmission starts. Reactive fragmentation requires a certain level of support from underlying protocols that may not be present, and presents significant challenges with respect to handling digital signatures and authentication codes on bundles. When a signed bundle is only partially received, most bundle authentication codes will fail. When DTN security is present and enabled, it may therefore be necessary to proactively fragment large bundles into smaller units that are more convenient for digital signatures.

2.3.4 Bundle Priority Classes

Traffic in DTN is generally not interactive and is often one-way. There are generally no strong guarantees of timely delivery, yet there are some forms of class of service, reliability, and security. The DTN architecture can offer different priority classes. These priority classes typically imply some relative scheduling prioritization among bundles in the queue at a sender, and based on these priorities the bundles may receive different types of services. The DTN architecture offers three (low, medium, and high) relative priority classes to date.

2.4 Congestion in DTN

When a node holds the custody of a bundle two events may happen. The node either forwards the bundle to the next hop during the next contact opportunity or the bundle's TTL expires before having the forwarding opportunity. Until any of the above events happens, a node has to retain all of its bundles in its persistent storage. Due to intermittent connectivity, and long and variable delay, a node might not have the contact opportunity for a long time. The bundle TTL is also reasonably large. Therefore, a node requires storing too many bundles for long time in its persistent storage. However, the nodes do not have unlimited persistent storage. If the storage capacity of a DTN node becomes filled up by the bundles, the node cannot accept the custody of the bundles anymore. This situation is known as congestion in DTN. This type of congestion is called storage congestion due to the fact that the storage capacity of a node is depleted. The network performance is affected severely due to the congestion. It increases both the bundle loss and the average bundle delivery time i.e., he network throughput. For this reason, congestion control mechanism is necessary in order to save the network from performance degradation.

2.5 Head of Line Blocking (HLB)

Head of line blocking [11] is a regular phenomenon in Delay Tolerant Networks. It occurs whenever traffic waiting to be transmitted prevents or blocks traffic destined elsewhere from being transmitted. HLB happens in DTN because of the mix up of intermittent contacts with persistent contacts. Figure 2.1 illustrates a simple DTN scenario with HLB. The network topology in Figure 2.1 has a sender node S and two destination nodes A and B. The path from the source to the destination nodes goes through an intermediate node R. The link from R to A is an always-on or persistent whereas the link from R to B is intermittent. R accepts bundle's custody destined to B and stores them in its persistent storage until it has the opportunity to forward the bundles to B. Consider a scenario

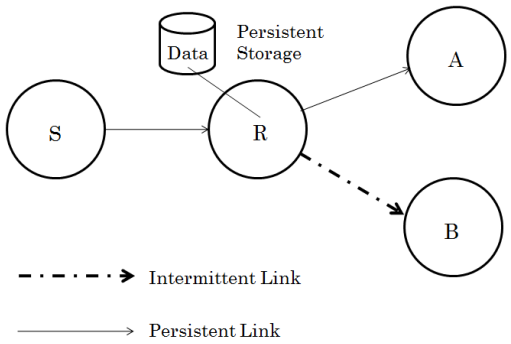

Figure 2.1: A simple DTN topology that could cause head-of-line blocking

when no R to B contact is currently available and node R's persistent storage is filled by bundles originating at S and destined for B. Now, the bundle traffic destined to A from S may be unable to transit R even if the R to A link is always available.

2.5.1 Congestion due to HLB

DTN nodes also suffer from congestion due to HLB [10].If the persistent contact to the next hop for a bundle is available, a node can immediately forward it. However, if only the intermittent contact to the next hop for a bundle is available it has to wait for the next contact in order to get forwarded. If the intermittent contact is infrequent, the bundle might have to wait in the persistent storage of the node for a long time. Now, if the custodies of a large number of bundles having intermittent contact to the next hop have already been accepted and the capacity of the persistent storage of the node has also been depleted the custody of the new bundles will be rejected even if their next hop nodes are in persistent contact with the node in concern. Therefore, the node needs to be more careful in order to avoid congestion due to HLB while accepting custody of the bundles.

2.6 Routing in DTN

DTN lacks persistent end-to-end path between any two hosts in the network which is a mandatory requirement for routing protocols in conventional networks. Therefore, conventional routing protocols based on TCP/IP network (such as RIP and OSPF) and ad-hoc routing protocols (DSDV, OLSR, AODV, DSR etc.) fail to perform in DTN. DTN routing adopts a hop-by-hop store and forward mechanism. Using different information available in a node, DTN routing protocols help it to choose the next hop for a bundle from the available contacts. Performance of the routing protocols depends mainly on the knowledge that they use in order to select the next hops. The knowledge includes contacts, contacts summary, queuing delays, and traffic demands. Routing algorithms can use zero knowledge, partial knowledge or complete knowledge in order to compute routes.

2.6.1 Routing protocols with zero knowledge

Routing protocols in this class do not use any knowledge. A typical example is First Contact (FC) [13] that forwards a bundle along a contact chosen randomly if multiple contacts are available. If all contacts are currently unavailable, the bundle waits for a contact to become available and is assigned to the first available contact. FC performs poorly in nontrivial topologies because the chosen next-hop is essentially random and forwarding along the selected contact may not make any progress toward the destination. A bundle may also oscillate forever among a set of nodes or be delivered to a dead end.

2.6.2 Routing protocols with Partial knowledge

The routing protocols in this category select paths using one or more of the available knowledge: contacts summary, contacts, and queuing delay. These protocols assign a cost to each edge in DTN and select a minimum cost path from the source to the destination based on the assigned costs. Costs are assigned to the edges to reflect the estimated delay

of the bundle through them.

Minimum Expected Delay (MED)

The MED [13] routing protocol uses the contact summary to compute the path costs. The cost of an edge along a path is assigned as the sum of the average waiting time of the bundles for the next contact to arrive, transmission delay, and propagation delay. This cost assignment is done statically, i.e., effective for scheduled contacts only and the path of a bundle for the same source-destination pair remains fixed. MED successfully minimizes the average waiting time along the path. However, no mechanism is employed here in order to get a way around a congested node along that fixed path. It fails to adapt the edge cost as well as the path from a source to a destination dynamically which is necessary for opportunistic or predicted contacts.

Minimum Estimated Expected Delay (MEED)

MEED [14] is a variant of the MED routing protocol. Instead of computing the expected waiting time using the future contact schedule, MEED uses the observed history of the past contacts. To compute the waiting time associated with a contact, a node records the connection and disconnection times of each contact over a sliding history window. When the local contact schedule of a node changes the node propagates the update to all the nodes in the network by using some flooding technique. For this reason, MEED routing protocol can perform routing in networks having both scheduled and predicted contacts.

2.6.3 Routing protocols with complete knowledge

Jain el.al. propose a routing protocol in [13] that uses the complete knowledge. They present a Linear Programming formulation that uses contacts, queuing delay and traffic load to determine the optimal routing for minimizing average delay in the network. However, routing protocols that require complete knowledge is far too difficult to implement

in DTN since acquiring all these as a priori information is not possible. Rather, routing protocols that use partial knowledge are practical to implement in DTN.

A comprehensive survey on different existing routing strategies is presented in [21]. Routing protocols in [16], [6] have addressed the issues regarding the construction of knowledge using available local information like contact schedule, past contact history. In [3], Bulut et. al. introduced a new metric called conditional intermeeting time using only the local knowledge of the past contacts. Incorporation of this metric improves the performance of the existing routing protocols like MED, MEED. However DTN routing is still an open and important issue that need to be further investigated [10], [17].

2.7 Summary

As a summary we can say that, DTN suffers from intermittent connectivity, long and variable delays, asymmetric data rates, and high error rates. Due to these special characteristics, conventional protocols like TCP/IP or ad-hoc wireless network protocols fail to perform in DTN. DTN introduced a new protocol layer called Bundle Layer Protocol which runs on top of the existing lower layer protocols like TCP, UDP, RTP/RTCP, LTP etc. Bundle protocol adopted a hop-by-hop store and forward mechanism to transmit the bundles from the source to the destination. The responsibility of reliable delivery of a bundle is considered as the custody and the custody of a bundle is also delegated to the next hop neighbor along with the bundle. A custodian node stores the bundle in its persistent storage until it transfers the custody to a next hop node. Storage congestion occurs if the storage capacity of a node depletes. DTN severely suffers from poor delivery ratio due to this storage congestion. HLB worsens the problem further. Effective congestion control schemes are essential to combat the poor delivery ratio problem. In the next chapter, we are describing some existing research works that proposed different congestion control schemes.

Chapter 3

State of the Art Congestion Control Schemes in DTNs

There are two ways to control storage congestion in DTNs. The first one is known as reactive approach. In reactive approach, a congested DTN node migrates its bundles to some of its uncongested neighbors and later retrieves those migrated bundles from the neighbors when congestion is over. The other approach is called proactive, which proactively decides whether it should accept or reject an incoming bundle based on its perceived level of congestion.

3.1 Reactive Congestion Control Scheme

Seligman et. al. proposed a reactive congestion control mechanism in [20] which is also known as Storage Routing (SR). They solved the problem of storage congestion by migrating stored data to the neighbors. They introduced the concept of push-pull custody transfer in their solution. In push operation, the current custodian node initiates the bundle transfer process to a suitable neighbor node. This is the traditional DTN custody transfer mechanism. In the pull operation, a node requests the custody of a bundle from its current custodian node.

When a bundle arrives at a congested node, SR dynamically selects a set of bundles to migrate them to a set of neighboring nodes using bundle selection and node selection algorithms respectively. Choosing the bundles to migrate by push custody transfer operation is analogous to choosing the packets to drop when an IP router exhausts its internal queue. Three factors are considered to select the bundles: arrival time, size, and priority. The bundle at the head or at the tail of the queue in a node can be selected based on the arrival time. The longest or the smallest bundle in the queue can also be selected if the size of a bundle is considered as the bundle selection factor.

The node selection algorithm searches for a set of nodes as the target of migration for the stored bundles and selects eligible storage neighbors from this set based on their storage availabilities and incident link characteristics. Here, incident link characteristics include the latency, bandwidth and up/down schedules of the links along the path from the current custodian node to the neighbors. Nodes those are selected as the migration targets are referred to as alternative custodians. SR restricts the set of nodes searched to become the alternative custodians to the congested nodes k-neighborhood. It employs a form of expanded ring search in which all the nodes m hop away from the congested node are considered for migration prior to any node m+1 hop away, up to a maximum of k hops from the congested node. If no alternative custodian is available, the node drops the bundle.

When congestion occurs, SR employs push custody transfer operation to transfer the selected bundles to the alternative custodians. When congestion subsides, the bundles stored in the alternative custodians need to be retrieved. SR invokes a retrieval selection algorithm to determine which nodes it will contact and then employs the pull form of custody transfer with a custody request operation to get the bundles back. Same factors are considered to select the alternative custodians and the bundles during the pull operation.

However, this reactive approach suffers bundle loss and performs poor when all the neighbor nodes are also congested and there is no way to migrate the stored bundles. Moreover, the cost of implementing the "push-pull operation" is very high and it intro-

duces too much overhead to the networks.

3.2 Proactive Congestion Control Scheme

In [4], Burleigh et. al. proposed a proactive congestion control mechanism based on a financial model. The proposed solution is rule based which has a set of predefined rules for accepting or rejecting the incoming bundles. Accept or reject decision is taken based on the local information, such as remaining storage capacity, bundle's priority, and bundle's TTL. In the worst case, an incoming bundle might reside in the persistent storage of a node until the bundle's TTL expires. Therefore, they compute the projected net growth in persistent storage occupancy over the interval of the incoming bundle's residual TTL. Here, net growth in the persistent storage occupancy over a given interval is the sum of the sizes of all bundles accepted over that interval minus the sum of the sizes of all bundles forwarded over that interval. Equation 3.1 shows the computation of projected net growth in persistent storage occupancy.

$$D = G_T + OC + b_l \tag{3.1}$$

where,

G_T = Net growth in persistent storage occupancy over interval T;

T = TTL of the incoming bundle;

OC = Storage capacity of the node currently occupied by the bundles and;

b_l = Length of the incoming bundle itself.

The projected net growth might overrun the remaining storage capacity of the node during the bundle's stay time at that node and the storage space(S) might completely filled up before the bundle is forwarded to the next hop. They accept the incoming bundle if the storage overrun does not occur, that is, the projected growth of storage space occupancy over the bundle's residual TTL period does not exceed the total storage space capacity of the node. Therefore, the bundle is accepted only if $D \leq S$. This is the

first condition to accept a bundle. If the first condition fails, they compute the bundle's acceptance risk (R)to give the bundle a second chance. The risk of accepting a bundle is the worst-case number of byte-seconds of storage space that the bundle will consume. It is simply the product of the bundles size and its residual TTL ($b_l \times T$).They assign a value (V) to each of the bundles which are computed as a function of the priority (Pr) of the bundles as shown in Figure 3.2.

$$V = b_l \times Pr \tag{3.2}$$

Then, they compute the risk rate of the bundle as the risk divided by the value. They also keep records regarding the aggregate risk and the aggregate value of the bundles accepted so far to compute the mean risk and the mean value. Mean risk rate over an interval is then computed as the ratio of the mean risk for that interval to the mean value for the same interval. If the risk rate of an incoming bundle exceeds the mean risk rate over the bundle's residual TTL, then the bundle is rated as the above-average risk bundle; they refuse it. Otherwise, they accept the bundle. This is the final rule.

They carried their simulation in a network scenario with only five nodes connected in a straight line. Therefore, the scheme is susceptible to perform well in network scenarios with moderate or large number of nodes. Simulation result proves that their scheme keeps the storage capacity underutilized and the obtained delivery ratio in networks with larger number of nodes is small.

In [24], Zhang et. al. proposed another proactive congestion control scheme based on revenue management. They considered two conflicting forces that are governing the receiving node's actions. On one hand, it is beneficial to accept a large number of bundles as it can potentially advance the bundles toward their ultimate destinations and that can maximize the network utilization. On the other hand, if the receiving node over-commits itself by accepting too many bundles, it may find itself setting aside an excessive amount of storage and thereby preventing itself from receiving further potentially important, high yield (in terms of network utilization) bundles. They used a benefit function and

an opportunity cost to balance the above two conflicting forces. The benefit function denotes the gain of moving a bundle to the next hop. By accepting a bundle, a node accumulates a certain amount of benefit. The benefit function is composed of the bundle size with different weights based on their corresponding traffic priorities or traffic types. The opportunity cost measures the value of the storage capacity that will be occupied if the node accepts the bundle. The opportunity cost measures the loss of benefits by accepting a lower benefit bundle which prevents a potentially higher benefit bundle to be accepted in future. The opportunity cost increases as the remaining capacity of a node decreases. At any moment, the node takes decision based on the benefit function and the opportunity cost. If the benefit of accepting a bundle is greater than the current opportunity cost, the bundle is accepted. Otherwise, the bundle is rejected.

Though this approach takes proactive decision, its decision making process is relatively slow and generally fails to react quickly. It also did not consider HLB, which is a regular phenomenon in DTN, in decision making. As a result the delivery ratio falls significantly.

3.3 Summary

In this chapter, we have studied both types of congestion control schemes that are currently existing: reactive and proactive. We have seen that the reactive schemes suffer from bundle loss and impose too much overhead on the network and proactive schemes take suboptimal decision by rejecting bundles too early although it is not necessary to do so. They also perform poor in networks with head of line blocking. DTN demands a new generation of congestion control schemes to mitigate all the shortcomings of the existing schemes. In the next chapter, we propose a new congestion control scheme in this regard.

Chapter 4

Proposed Congestion Control Scheme

In this Chapter, we present our congestion control scheme for DTN. The main goal of a congestion control algorithm is to increase the delivery ratio of the network. Although the delay is tolerated in DTN but it should not go up beyond a reasonable limit. Overhead is an issue that should get paramount importance in DTN. Any congestion control algorithm should not overwhelm the network by creating too much overhead to the already congested one. Since the propagation of information to distribute the global knowledge is difficult in DTN, any algorithm that expects using some sort of global information is rather impractical to implement in real network scenario. To devise a new congestion control algorithm we have taken all these factors into our consideration.

4.1 Proposed Scheme

We propose a congestion control mechanism that is using the local information only. In our congestion control algorithm, we adopt the proactive approach of decision making since it performs better in highly and regularly congested networks. Proactive approach initiates a sort of flow control in the network by rejecting the incoming bundles. The flow

control is originated from the node at the point of congestion and propagated back to the source node. Proactive approach also introduces less overhead to the network comparing to its counterpart, reactive approach.

Our algorithm is based on an economic concept. Though other research works [4] [24] used economic concepts to mitigate congestion in DTN. Our concept is different from them. The concept introduced in [4] such as conveyance fee paid by the sender and receiver of a bundle is different from the price of a bundle and cash-in-hand of a node proposed in our work. The concepts used in [24] such as benefit function, and opportunity cost are also different from the concepts introduced in our scheme.

The decision of accepting or rejecting a bundle in our algorithm mimics the decision of purchasing a perishable commodity. Let us consider a scenario where the producers produce perishable commodities and the consumers consume the commodities. There is a market for transporting these products from the producers to the consumers and a chain of store keepers constitutes the market. The goal of a store keeper is to purchase the perishable commodities and store them until it has been purchased by a consumer or another store keeper. The key constraint is the capacity of the store. Each store has a limited capacity. Each of the commodities has a price, and the store keeper has some cash to buy them. The cash-in-hand of a storekeeper is directly related to the capacity of the storage. The commodities can be perished during its storage time. It is the duty of the store keeper to sell them before their life time comes to an end. But marketing is another hard job to accomplish. The store keeper might need to wait for a long time before getting the opportunity to meet another keeper to whom he can sell his stored commodities. So the trading is fully dependent on the market condition. The current market condition can be determined by its liquidity. If the market for a certain commodity is liquid enough, it will be sold within a short time. Otherwise, it will remain in the storage and block the capacity for a long period. Therefore, purchase decision by a store keeper depends on the price of the commodity itself, its lifecycle, the storage capacity of the store and the current market condition of that commodity. The decision will go in favor of those commodities whose prices are reasonable, lifecycles are long enough to sustain the market uncertainty,

and the market condition is liquid enough. The ultimate goal is to take optimal decision and forward as much products as possible to the consumers.

Routing the bundles in DTN by store and forward resembles the above market scenario. DTN nodes are responsible for forwarding the bundles from the source host to the destination host. A bundle has to wait in a node's persistent storage until the node has the opportunity to forward the bundle to the next hop neighbor during the next contact period. Like perishable commodities, each bundle has a limited life time which is represented by the time-to-live (TTL) of the bundle. If the TTL of a bundle expires before it reaches the final destination, the bundle will be dropped immediately. A bundle has to be stored in a node's persistent storage because of its custody and this storage period may be long due to frequent network partition. It is the nodes' responsibility to forward as many bundles as possible before they are getting dropped.

Like the above market scenario, each bundle has a price that is its priority and TTL. Each node can rank the incoming bundles according to the prices of the bundles and regulate the purchasing decision based on this ranking. To purchase, that is, to accept a bundle a node requires cash which can be represented as a function of the free storage space of the node. Therefore, each node has an amount of cash-in-hand for purchasing the incoming bundles and if the node's cash-in-hand is abundant, it can accept all the bundles. The price of the bundle is indifferent then. But, as we know the persistent storage capacity of a node is limited in DTN, it will be depleted during the congestion period and that certainly lowers the cash-in-hand of that node. Therefore, the node should go for the low or medium price bundles during congestion period.

Market liquidity can also play a significant role in decision making. Some bundles may have to wait for a long time if the contacts with their destination come less frequently. The TTL of some bundles might expire before they got forwarded. However, there might be some bundles that got forwarded quickly because of the frequent contact of their destination. A node can easily determine this liquidity factor by keeping information regarding the previous contact schedules and previous waiting time for the bundles with

respect to a particular contact. By using this information, a node can compute a dropping probability for each incoming bundle. The dropping probability of a bundle determines the probability of the bundle to be dropped from the node.

In DTN, a node has different set of neighbors at different time because of the varying contact schedules. Sometimes, a node may have neighbors with a large storage capacity free to hold the incoming bundles. A node might have the chance to forward its stored bundles to these neighbors during their contact periods. On the other hand a node sometimes may have neighbors with small amount of free storage to store the future bundles. This may happen especially when the neighbors are in congestion. This reduces the possibility of the node to forward its bundle to those neighbors. By using this information, a node can compute a confidence level. The confidence level determines how confident a node is to forward the bundle to its next hop in time. Finally, a node can make decision to accept or reject a bundle based on the dropping probability of that bundle and the confidence level of that node.

In our congestion control scheme, when a custody request of a bundle arrives in a DTN node we propose the node to compute the price and the dropping probability of the bundle. We also propose the node to compute its cash-in-hand and confidence level at the same time. Bundle priority, TTL, and head-of-line blocking property of the node is used to compute the price. Priority of a bundle determines its intended class of service from the network. A high priority bundle should get the highest possible service from the network. The node includes priority in the price calculation to ensure this. We also propose the node to incorporate the TTL of a bundle into its price calculation since the bundles with different TTL values require different responses. Bundles with low TTL value need immediate response whereas the bundles with relatively high TTL value can tolerate deferred response. The head-of-line blocking property of the node also contributes to the price of a bundle to ensure fairness in DTN [11]. It also ensures a high delivery ratio. Detail description of price calculation of a bundle is presented in Section 4.2.

The dropping probability of a bundle is computed using the remaining TTL of the

bundle and the average queuing delay of the node. The detail calculation of the dropping probability is presented in Section 4.2.

The remaining persistent storage capacity of a node is used to compute the cash-in-hand of that node since the amount of remaining persistent storage capacity of a node determines the store ability, i. e., the purchase ability, of that node. Detail description of cash-in-hand computation is presented in Section 4.2.

Finally, the current state of the remaining storage capacity of the neighbors is used to compute the confidence level of the node. A node is confident to forward a bundle when the remaining capacities of its neighbors are high. Detail description of confidence level calculation is presented in Section 4.2.

In order to accept or reject the custody of the bundle, we propose the node first examines the size of the bundle. If the size is larger than the free storage space of the node, the bundle is rejected. Otherwise we propose the node to examine the amount of cash-in-hand. If the cash in hand of the node is greater than a predefined cash-in-hand threshold (C_{th}), i.e., the node has adequate space to store bundles, it will accept the bundle. But if the cash-in-hand falls below that threshold, the node assumes it as congestion and the bundle acception and rejection decision becomes complicated.In this case, the price of the bundles is examined. If the price of a bundle is high, it assures that the three contributing factors are higher than their normal values. There are some negative aspects with the bundle and the bundle is a good candidate for rejection at the time of congestion. Moderate values of prices ensure positive characteristics of the contributing parts and decision goes in favor of accepting bundles with moderate price. Therefore, the price is compared with a predefined price threshold (P_{th}). If the price is smaller than the threshold, the node is prepared to accept the bundle. However, we propose the node to check the current confidence level of the node before accepting the bundle. Confidence Level determines how much a node is confident to forward its stored bundles to the next hop. If the node's current confident level is larger than a predefined confidence level threshold (CL_{th}), the node will accept the bundle. Otherwise, the bundle

will be rejected.

If the price of the bundle is larger than the price threshold the dropping probability of the bundle is examined. If the dropping probability is smaller than half, the bundle has a good chance to get forwarded in time without being dropped. However, we propose the node to accept the bundle after examining the current confidence level of the node. If the node's current confident level is larger than the confidence level threshold (CL_{th}), the node will accept the bundle. Otherwise, the bundle will be rejected.

If the dropping probability of the bundle is greater than the dropping probability threshold DP_{th}, the node will reject the bundle since the chance of the bundle to be forwarded in time without being expired at that node is small.

Algorithm 1 depicts the steps of our congestion control scheme.

4.2 Detailed Calculation

Our congestion control scheme uses different parameters for taking its decisions. A node calculates its cash-in-hand to determine the level of congestion. When the custody of a bundle arrives at a node, it computes the price as the rank of the bundle. The node also computes the dropping probability and confidence level to facilitate better decision making. In this section, we present a detailed calculation of these parameters.

4.2.1 Price (P)

In our congestion control scheme, we propose a DTN node to assign a price to each incoming bundle whenever it receives the custody request. It employs the price of bundles to rank them and based on this ranking a node either accepts or rejects a bundle. Price is calculated in such a way that a low priced bundle is ranked to a higher level and acception decision goes in favor of a higher level ranked bundle. High priced bundles are potential candidate for rejection at the time of congestion. The node computes bundle's price (P)

Algorithm 1 *procedure* CongestionControl

When a node i receives custody request of a bundle from another node j

Node i computes the price (P) and dropping probability (D_p) of the bundle
Node i computes its cash-in-hand (C), and confidence level(CL)
if $bundle\_length < FreeStorageSpace$ **then**
    Reject the bundle
else
    if $C < C_{th}$ **then**
        Accept the bundle
    else
        if $P < P_{th}$ **then**
            Accept the bundle
        else
            if $D_p < 0.5$ && $CL > CL_{th}$ **then**
                Accept the bundle
            else
                Reject the bundle
            end if
        end if
    end if
end if

using some properties of the bundle, such as priority, TTL, and the HLB, using Equation 4.1

$$P = \frac{1}{Pr_d} + \frac{T_d}{L_d} + \frac{\sum S_d}{S} \tag{4.1}$$

where,

Pr_d - *Priority* of the incoming bundle having destination d;

T_d - Time-To-Live of a bundle having destination d;

L_d - Life Time of a bundle having destination d;

S_d - Storage space occupied by each Bundle at the Node having the destination d and;

S - Storage Capacity of the Node.

Bundle price calculated in Equation 4.1 has three additive components to emphasize each of the three contributing factors. This price rewards bundles with high priority as priority reflects the intended class of service for a bundle. A high priority bundle should receive the highest possible service from the network whereas a bundle with low priority might be served with the best effort way. The first component of the Equation 4.1 ensures this by taking the inverse of the priority. The bundles with lower TTL are preferred over other bundles as they have earliest deadline and require immediate response from the network. Therefore, they are assigned lower price. The second term of Equation 4.1 ensures this by normalizing the TTL with the life time of a bundle. The third contributing part is for HLB property of a bundle with respect to a node. A node finds the number of bundles stored in its persistent storage destined towards d, where d is the destination of the incoming bundle whose acception or rejection decision is under consideration. A smaller count of such bundles represents less traffic towards this destination from this node, i.e., less HLB for this bundle. A bundle with less HLB is worth to accept, i.e., should be assigned to a low price. In order to normalize the HLB with the storage capacity of the node, we find the sum of the storage occupancies of these bundles and add the ratio of the sum and the storage capacity in Equation 4.1.

4.2.2 Cash-in-Hand(C)

Cash-in-hand of a node represents the current capacity of the node to store further bundles and calculated as a function of the remaining storage capacity of the node. A large free storage allows a node to store more incoming bundles. i.e., more cash-in-hand. Equation 4.2 computes cash-in-hand of a node as the proportion to the size of its current free storage by normalizing it to its total storage capacity.

$$C = \frac{F}{S} \tag{4.2}$$

where,

F = Storage Capacity that is currently free at the node and;

S = Storage Capacity of the node

Therefore, the value of the attribute cash-in-hand of a node indicates the level of congestion of the network perceived by the node. If the cash-in-hand decreases below a certain level and remains lower for a certain time period, it indicates congestion. Whenever a node perceives such an indication, it takes appropriate actions to control the congestion.

4.2.3 Confidence Level (CL)

Confidence Level determines the level of confidence of a node to forward its stored bundles to their respective next hops in time. At a particular time, forwarding capability of a node depends on the current number of neighbors that the node has and the total amount of free storage available in those the neighbors. A DTN node can easily receive the information about the free storage available in the neighbors and sum them up in order to determine its confidence level. Equation 4.3 shows the computation of the confidence level of a node.

$$CL = \sum_{i \in N} (\frac{F}{S})_i \tag{4.3}$$

where

N = List of current neighbors

4.2.4 Dropping Probability (D_p)

Every bundle in a DTN is dropped from the network when the bundle's TTL expires. TTL expiration of a bundle may happen at a node if its TTL is smaller than the queuing delay of the node. Here, queuing delay of a node is the average waiting time of the bundles in its persistent storage. Therefore, the bundle will be dropped at the node with a high probability if the TTL of the bundle is less than the queuing delay of the node. In other words, if the queuing delay increases while TTL remains the same, dropping probability increases. On the other hand, if the TTL increases but the queuing delay does not increase, drooping probability decreases. Therefore, the dropping probability is directly proportional to queuing delay but indirectly proportional to TTL. For this reason, we compute the dropping probability as follows in Equation 4.4.

$$D_p = \frac{Q_d}{T + Q_d} \tag{4.4}$$

where,

D_p = Dropping Probability of an incoming bundle;

Q_d = Queuing delay of a node;

T = Time to Live of the incoming bundle.

In Equation 4.4, we divide Q_d by $(T + Q_d)$ so that the values of D_p always remain less than one. According to [18] the queuing delay in a node can be computed by Equation 4.5 as follows.

$$Q_d = \frac{1}{\lambda - \mu} \tag{4.5}$$

where,

Q_d = Queuing delay of a node;

λ = Incoming rate of bundles in a node and;

μ = Outgoing rate of bundles in a node.

4.2.5 Thresholds

Our congestion control scheme uses four threshold values to decide whether to accept or reject an incoming bundle. They are i) Cash-in-hand threshold ii) Price threshold iii) Confidence level threshold iv) Dropping probability threshold.

Cash-in-hand Threshold C_{th}

In Section 4.2.2, we have defined cash-in-hand which denotes the free storage capacity of a node to hold further bundles. Low cash-in-hand indicates congestion in the node. Whenever the cash-in-hand falls below a limit, the node must be conservative to accept new bundle. Cash-in-hand threshold defines this limit. We use cash-in-hand threshold as one of our algorithm parameter.

Price Threshold P_{th}

In Section 4.2.1, we have defined the price of a bundle as its rank and based on this rank a bundle is either accepted or rejected at the time of congestion. We use the maximum price of the bundles that are currently stored in the persistent storage of a node as its current price threshold. The maximum price changes when the set of bundles in the persistent storage changes and it corresponds to a bundle that is the worst among the bundles currently stored in the persistent storage with respect to price. We should not accept any bundle that has price higher than this maximum price. With this rationale, we use this maximum price as the price threshold and compare the price of a bundle with the price threshold to accept or reject the bundle during congestion.

Confidence Level Threshold CL_{th}

In Section 4.2.3, we have defined the confidence level of a node as the level of its confidence to forward stored bundles to their next hops and computed the confidence level as the average free storage capacity at its neighbors. If the average free storage capacity at the neighbors is larger than at least α fraction of the storage capacity that is currently occupied by the bundles at the nodes it gives an indication that the node has the higher probability that it will forward the currently stored bundles to the neighbors. For this reason, we compute the confidence level threshold as follows in Equation 4.6.

$$CL_{th} = \alpha \times \frac{O}{S} \tag{4.6}$$

where,

O = Storage Capacity Occupied by bundles;

S = Storage capacity of the node and;

α is an algorithm parameter.

We compare the confidence level of a node with the confidence level threshold to accept or reject a bundle.

Dropping Probability Threshold DP_{th}

In Section 4.2.4, we have defined the dropping probability of a bundle. If the dropping probability of a bundle is high, it is likely to be rejected. Otherwise, the bundle is accepted. Therefore, we compare the dropping probability of the bundle against a threshold value, DP_{th}. We use DP_{th} as one of our algorithm parameter.

4.3 Summary

In this chapter, we described a novel congestion control scheme based on proactive approach. Our novel congestion control scheme accepts or rejects an incoming bundle by

using the following four concepts - i) Price of a bundle ii) Cash-in-hand of a node iii) Confidence Level of node and iv) Dropping Probability of a bundle. Price of a bundle represents its rank and has been computed using bundle's priority, TTL and the HLB. Cash-in-hand denotes the current capacity of a node and has been computed as a function of the free storage capacity in that node. Confidence level of a node represents how much a node is confident to forward its stored bundles to the next hops and has been calculated as the function of the storage capacity free in the neighbor nodes. Each bundle also has a dropping probability which is related with the queuing delay of the node and the TTL of the bundle. We also used three threshold values in order to facilitate the decision making. A bundle is rejected when the node is congested, the bundle has a high price (low rank), the confidence level of the node is low and the bundle has a high dropping probability. By deferring the rejection decision, a node keeps as many bundles as possible in its persistent storage. In the next chapter, we evaluate our proposed scheme.

Chapter 5

Performance Evaluation

In this chapter, we present the evaluation of our congestion control scheme. We first present the metrics that we have used to evaluate our scheme, followed by a brief description of our simulation setup. Finally, we present a comprehensive performance comparison of our congestion control scheme with that of some existing congestion control schemes.

5.1 Performance Metrics

Usually DTN congestion control schemes are evaluated using two metrics- Delivery Ratio (DR), and Median Delay (MD).

- **Delivery ratio(DR)** is defined by the ratio of the total number of bundles delivered (b_{del}) to the total number of bundles created (b_{cre}). The computation of Delivery Ratio has been shown in Equation 5.1.

$$DR = \frac{b_{del}}{b_{cre}} \tag{5.1}$$

- **Median Delay (MD)** is defined as the median of the time required for a bundle to reach its destination. Equation 5.2 shows the expression for Median Delay.

$$Lat = Median(\forall_{delivered}(t_{del} - t_{cre})) \tag{5.2}$$

where,

t_{cre} = bundle creation time;

t_{del} = bundle delivery time;

$delievered$ = list of delivered bundles.

- **Effective Storage Utilization (ESU)**): In order to compare the performance of our proposed congestion control scheme with that of existing congestion control schemes in terms of buffer utilization we propose a new performance metric called Effective Storage Utilization, which will represent the ratio of the persistent storage that has been effectively used by the successfully delivered bundles to the total capacity of the persistence storage of the network.The number of successfully delivered bundles is the delivery ratio multiplied by the total number of bundles created, i.e., $DR \times b_{cre}$. Median Delay or MD represents the approximate time a bundle stays in the network. If it is multiplied by the average bundle length, b_l, the result represents the approximate storage usage by a successfully delivered bundle. Therefore, the approximate storage usage by all the successfully delivered bundles can be computed as $DR \times b_{cre} \times MD \times b_l$. Again, the total persistence storage capacity of the network can be computed as $SC \times N \times T$. Here, SC is the storage capacity of a node, N is the total number of nodes in the network, and T is the total simulation time. Therefore, ESU can be computed by Equation 5.3 as follows:

$$ESU = \frac{DR \times b_{cre} \times MD \times b_l}{SC \times N \times T} \tag{5.3}$$

5.2 Simulation Setup

To evaluate the effectiveness of our congestion control scheme, we compare the performance of our scheme with that of the two existing proactive congestion control schemes using DTNSim2 (Delay Tolerant Network Simulator 2) [1]. DTNSim2 is a publicly-available, widely accepted and used discrete-event driven simulator for evaluating DTN protocols

though the actual implementation is missing custody transfer of bundles. The default implementation of DTNsim2 has been customized first to incorporate the custody transfer.

DTNSim2 comes with the implementation of some practical single-copy routing protocols of DTN - MED (Minimum Expected Delay) [13], MEED (Minimum Estimated Expected Delay) [14]. In our simulation, we used these routing protocols for forwarding bundles to the next hops.

Two types of nodes - i) leaf nodes with low degree, responsible for generating network traffics at regular interval, and for absorbing them and ii) core nodes with high degree, responsible for constituting the core of the network, routing network traffics from source to destination, absorbing traffic destined to it were considered in our simulation. Different network topologies were simulated by varying the node count from 10 to 50. The ratio between the number of core nodes and the number of leaf nodes is 50%. The network grew from a simple to a complex structure.

Two types of links or contacts were considered: i) Persistent and ii) Intermittent. Persistent contacts between two nodes are always on. One node can forward bundles to the other node whenever there are such bundles ready to send. Intermittent contacts have a fixed schedule. It maintains a duty cycle for each intermittent contact. Duty cycle represents the proportion of time period when an intermittent link remains up. Duty cycles vary from 0.30 to 0.80 with an average of 0.50 (50% uptime and 50% down time). Each link of the networks has a bandwidth of 50kbps and latency of 20ms. The link intermittency schedule was arranged in such a way that the networks became partitioned at some point in time. At the start of simulation (at simulation time t = 0), all links are assumed up.

Leaf nodes can generate bundles of five priority classes, with a rate of 1 bundle per second. The bundle length is normally distributed with an average length of 100 bytes. Bundles are generated in burst with around 15 bundles per fire and kept in persistent storage before being injected in the network. The storage capacity to keep the bundles is same, which is typically 100M, for all the nodes of the network. Here, M is the average

bundle length. Both newly generated bundles and incoming bundles from other nodes share the same storage space. Therefore, contention might arise in a node between newly generated bundles and the bundles received from the other nodes. To keep things simple, core nodes are assigned with the only responsibility of routing bundles. Therefore, a core node never generates bundles, only accepts them from the other nodes. A bundle is immediately forwarded to the upper layer when it reaches the final destination. No contention for the storage is assumed at the destinations. Therefore a bundle can be destined to a core node.

Our algorithm can address any type of nodes mobility as long as the contacts between nodes are not opportunistic. In case of opportunistic networks (OpNets)[12] [5], different routing strategies [23], [22], [15] based on flooding of multiple copies of a bundle are required. Congestion control in OpNets works on limiting the multiplication factor. Since the concept of custody transfer is not adopted in OpNets and multiple copies of a bundle exist in the network, a node can safely drop some of its stored bundles when congestion occurs. Therefore, congestion control algorithm in OpNets decides which bundle should be dropped. Our congestion control scheme can cope with semi-opportunistic contacts where the contact schedule can be predicted.

The amount of storage at each node in a congested DTN has a strong effect on the delivery ratio of the network. Congestion occurs in DTN only due to storage scarcity and if storage capacity of a node can be made infinite, there will be no congestion. In our simulation, we also evaluated the impact of the storage capacity on the different metrics in order to compare the performance of different schemes. To evaluate the impact of storage, we gradually increase the storage capacity of a node from 20M to 300M where M is the average bundle length.

Cash-in-hand represents the free storage space in a DTN node to hold future bundles. This is being compared with a cash-in-hand threshold to accept or reject a new bundle. We empirically computed the optimum value for the cash-in-hand threshold. The following two figures show the experimental results. Figure 5.1 plots the delivery ratio of a typical

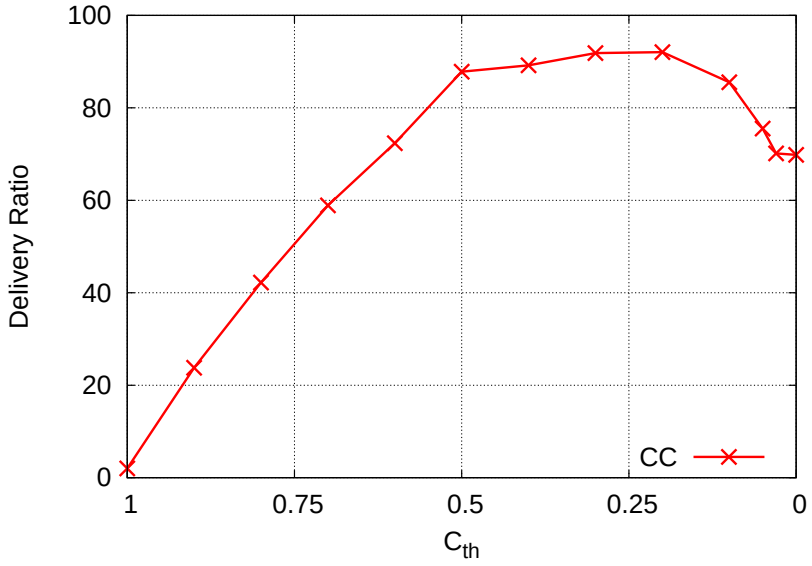

Figure 5.1: Delivery Ratio vs. Cash-in-hand Threshold. Here, CC: Our Congestion Control Scheme.

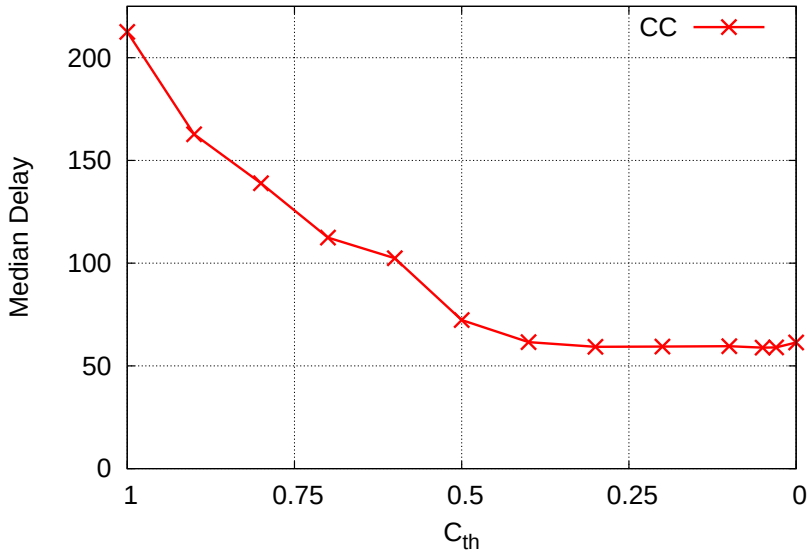

Figure 5.2: Median Delay vs. Cash-in-hand Threshold. Here, CC: Our Congestion Control Scheme.

DTN scenario against the whole range of cash-in-hand threshold values, [1, 0] and Figure 5.2 plots corresponding delay. From the above two figures it is evident that any value between the range [0.1 - 0.4] is a good choice for the cash-in-hand threshold as it ensures

a high delivery ratio along with a low delay. We used 0.3 as the cash-in-hand threshold in all our experiments.

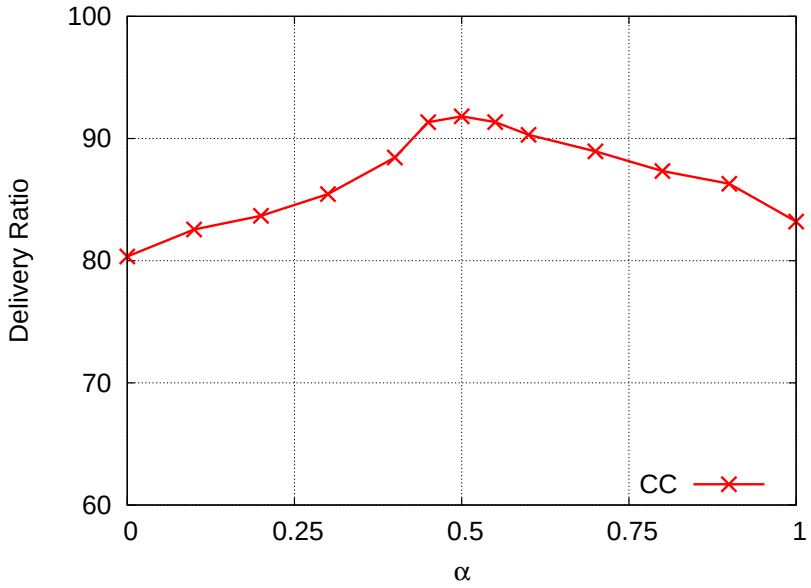

Figure 5.3: Delivery Ratio vs. α. Here, CC: Our Congestion Control Scheme.

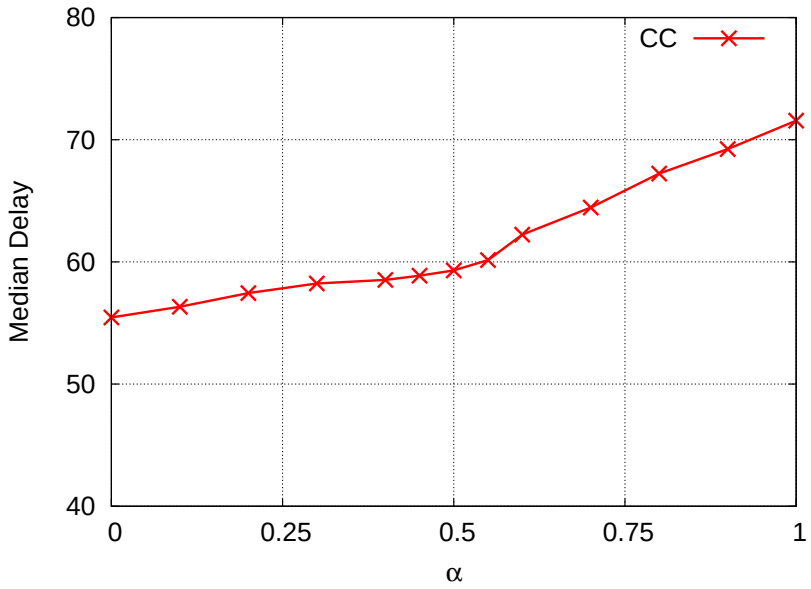

Figure 5.4: Median Delay vs. α. Here, CC: Our Congestion Control Scheme.

Our scheme needs to compute the confidence level of a node and compare it with the confidence level threshold. Equation 4.6 in Chapter 4 computes the confidence level

threshold. In Equation 4.6, we use an algorithm parameter, α. In order to find a suitable value for α, we studied the effect of choosing different values of α on the delivery ratio and the median delay. The Figure 5.3 and 5.4 show the experimental results.

From Figure 5.3, we see that the delivery ratio is maximum at $\alpha = 0.5$. From Figure 5.4, we see that the median delay at $\alpha = 0.5$ is little more than that at $\alpha = 0.0$. The median delay abruptly increases with the increase in α after 0.5. Therefore, $\alpha = 0.5$ is the best choice for our simulation scenario.

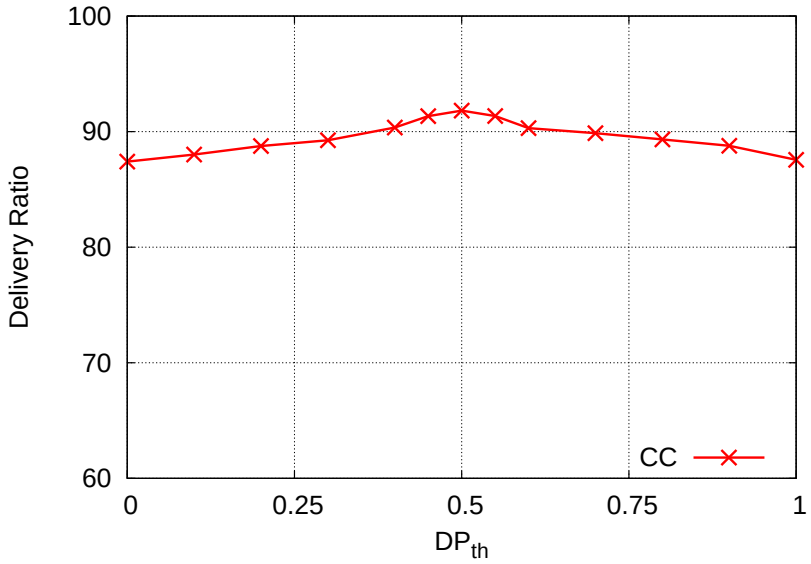

Figure 5.5: Delivery Ratio vs. DP_{th}. Here, CC: Our Congestion Control Scheme.

Our scheme needs to compute the dropping probability of a bundle and if the dropping probability of the bundle is greater than a predefined threshold, DP_{th}, it is rejected. In order to find a suitable value for DP_{th}, we find the delivery ratios and the median delays varying the DP_{th} from 0.0 to 1.0. The experimental results are shown in Figure 5.5 and 5.6 respectively. It is evident from both figures that 0.5 is the best choice for DP_{th} for our simulation scenario since the delivery ratio is maximum and the median delay is also moderate at this value.

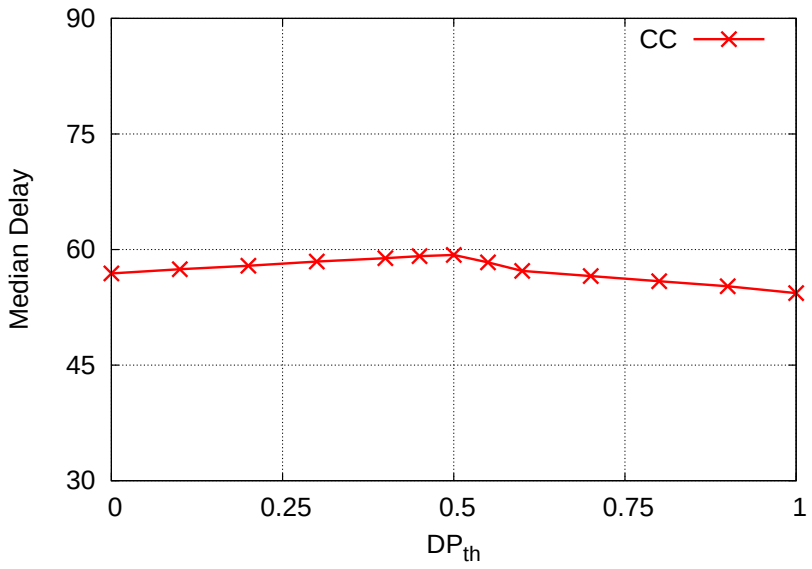

Figure 5.6: Median Delay vs. DP_{th}. Here, CC: Our Congestion Control Scheme.

5.3 Simulation Results

At first, we present the impacts of our congestion control scheme on the two famous routing protocols (MED, MEED) of DTN. Figure 5.7 shows the delivery ratios of MED routing protocol with and without our congestion control scheme varying the persistent storage capacity. Figure 5.8 shows those of MEED routing protocol. It is obvious that the delivery ratio increases with the increase in the storage capacity in the DTN nodes. More storage capacity eases the congestion in the network, which enables it to deliver more bundles and suffer less bundle loss, i.e., to achieves higher delivery ratio. For this reason, we see the delivery ratio in the above figures is increasing along with the increase in storage capacity irrespective of the use or not use of congestion control scheme with the routing protocols. With the very low storage capacity nodes are facing severe congestion which cannot be recovered by the use of congestion control scheme either. For this reason, the delivery ratio remains the same and low even after using the congestion control scheme with the routing protocols. With the very high storage capacity nodes are not facing the congestion at all. For this reason, the delivery ratio becomes high even if the congestion control scheme is not used with the routing protocols. However, the delivery ratio of the

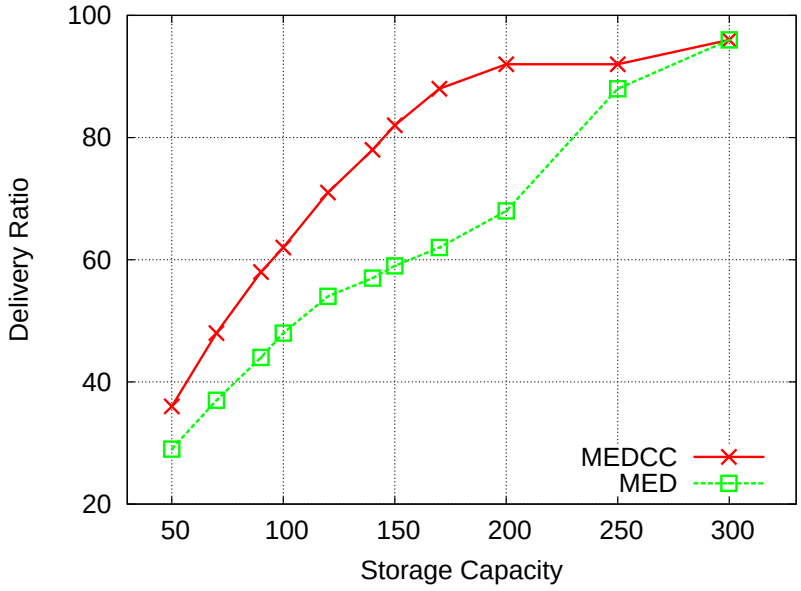

Figure 5.7: Delivery Ratio vs. Storage Capacity for MED Routing with (MEDCC) and without our scheme.

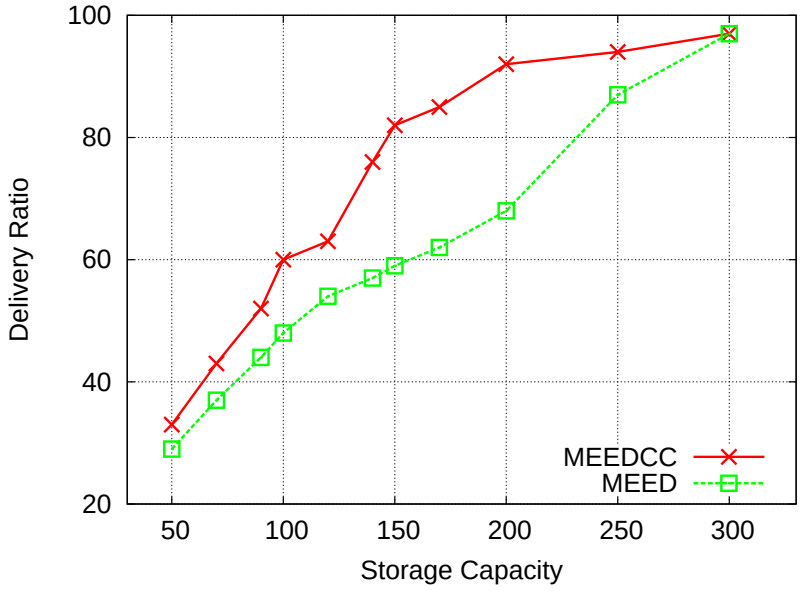

Figure 5.8: Delivery Ratio vs. Storage Capacity for MEED Routing with (MEEDCC) and without our scheme.

routing protocols improves by 25% to 30% for medium range storage capacity (100M to 250M) by using our congestion control scheme with the routing protocols. In this range, delivery ratio depends heavily on the effective use of the storage and our congestion control

scheme ensures the effective use of the storage by selectively accepting the bundle custody.

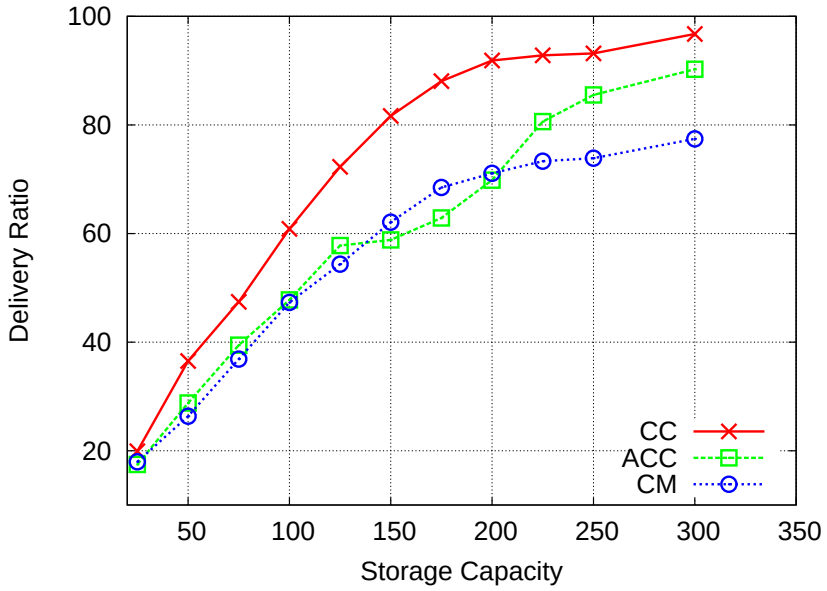

Figure 5.9: Delivery Ratio vs. Storage Capacity for CC, ACC [4], CM [24] : MED Routing.

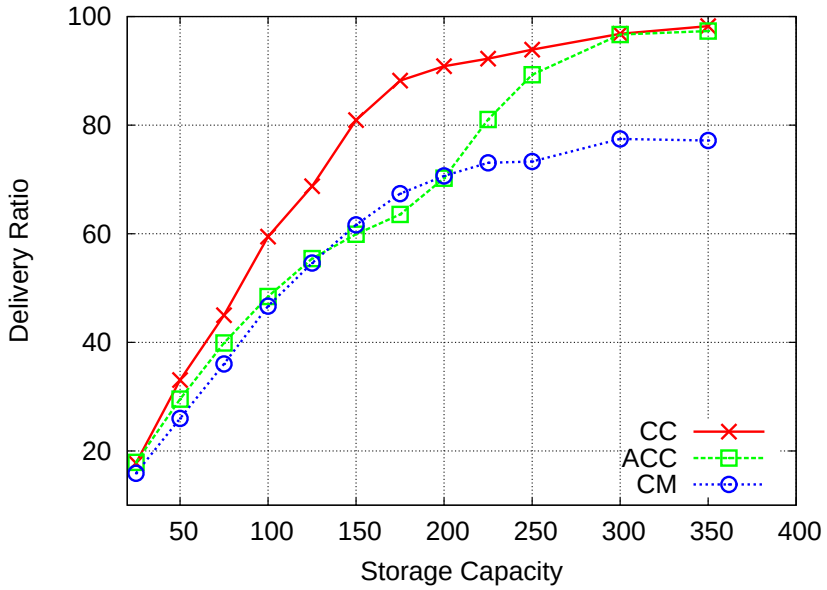

Figure 5.10: Delivery Ratio vs. Storage Capacity for CC, ACC [4], CM [24] : MEED Routing.

Figure 5.9 and 5.10 compares the delivery ratios of different congestion control schemes. Figure 5.9 shows the delivery ratios of different congestion control schemes with MED rout-

ing protocol while Figure 5.10 depicts the same with MEED routing protocol.According to these figures, the delivery ratio of the congestion control schemes increases with the increase in the storage capacity in the node for both routing protocols. As the storage capacity of a node grows, it can accommodate more bundles that ease the congestion in the network. Therefore, the delivery ratio is increasing along with the increase in storage capacity. With very low storage capacity nodes are facing congestion severely and frequently. No congestion control scheme can handle this recurrent congestion. Therefore, the congestion control schemes perform very poor and suffer from low delivery ratio. However, our congestion control scheme (CC) outperforms the existing schemes. It achieves 25% to 35% higher delivery ratio than ACC and CM. Existing schemes take suboptimal decisions by rejecting bundles when it is not required. They do not consider the head of line blocking which affects the delivery ratio of DTNs with both persistent and intermittent contacts. By deferring the rejection decision as far as possible, and considering the head of line blocking, our scheme achieves higher delivery ratios.

In Figure 5.11 and 5.12, the performance of the congestion control schemes in terms of Median Delay is presented. Figure 5.11 and 5.12 shows the impact of varying storage capacity on the Median Delay for MED and MEED routing protocols respectively. These figures illustrate that our congestion control scheme maintains reasonable delay in comparison with that of other schemes. The delays show a downward trend as the storage capacity in a node increases. As the storage capacity is increased in the nodes, the average number of custody transfers is increased due to the higher availability of the storage at the next hop nodes. This is reducing the average custody transfer delay at the nodes, i.e., the median-delay to deliver the bundle.

Figure 5.13 and 5.14 shows the effect of changing the number of created bundles on the delivery ratio of the network while keeping the network capacity constant. Here, we consider a DTN network scenario consisting of 30 nodes with core to leaf node ratio 40%. Network capacity per node is 100M, where M is the average size of bundles. The Figures suggest that Delivery ratio decreases with the increase in the number of bundles. As the number of created bundles increases, the network load grows. Since the total capacity is

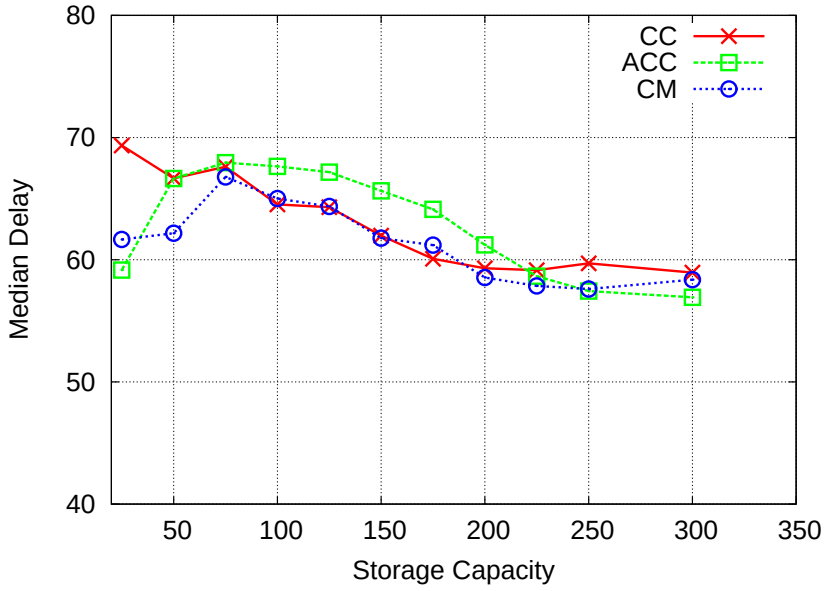

Figure 5.11: Median Delay vs. Storage Capacity for CC, ACC [4], CM [24] : MED Routing.

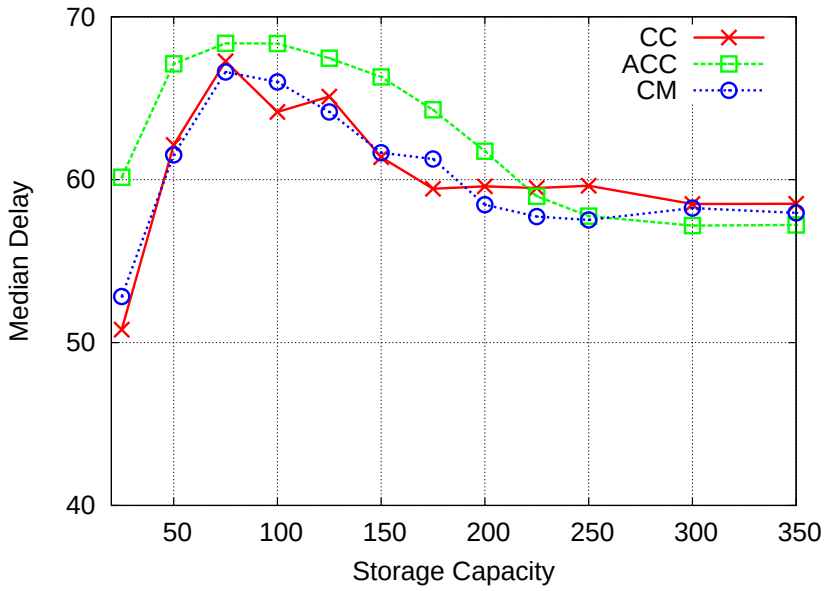

Figure 5.12: Median Delay vs. Storage Capacity for CC, ACC [4], CM [24] : MEED Routing.

constant, the network becomes more congested with the increasing load. Therefore, DTN nodes face congestion more frequently, and frequent bundle loss occurs. Consequently, the overall delivery ratio of the network falls with the increase in the number of bundles. However, our congestion control scheme achieves higher delivery ratio than that of the

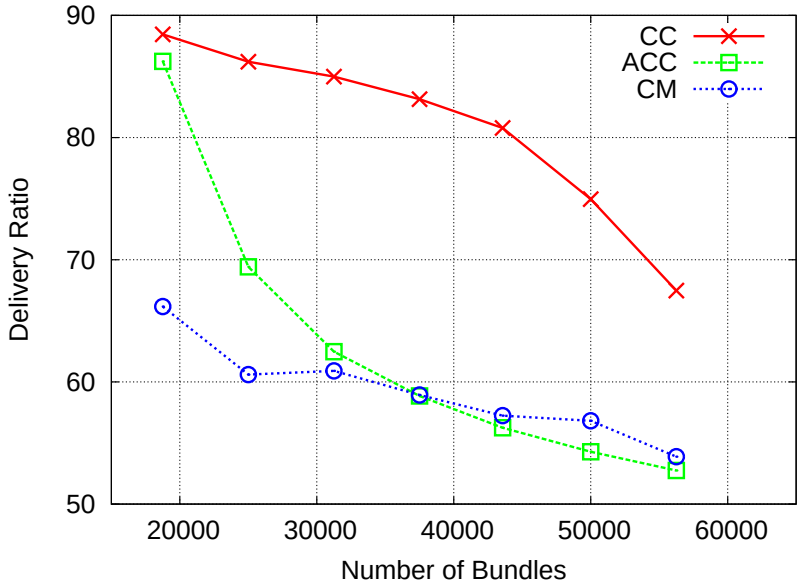

Figure 5.13: Delivery Ratio vs. Number of Bundles for CC, ACC [4], CM [24] : MED Routing.

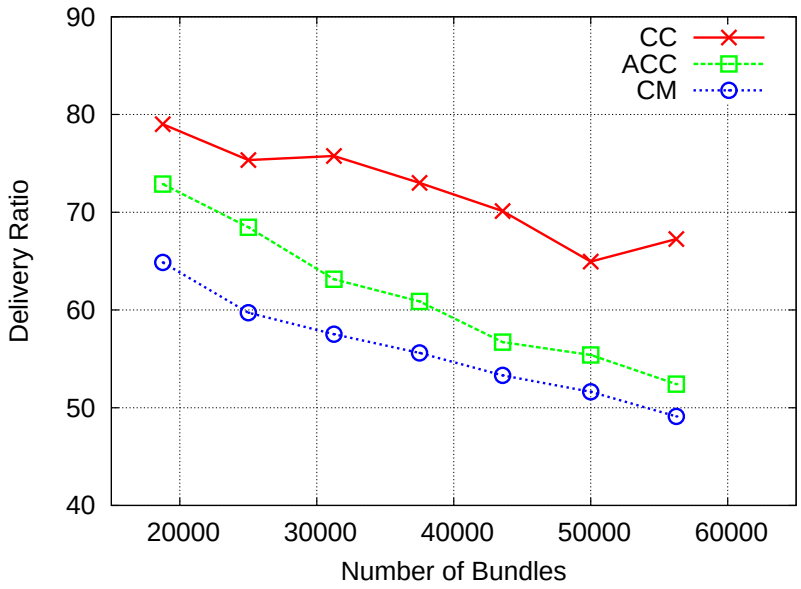

Figure 5.14: Delivery Ratio vs. Number of Bundles for CC, ACC [4], CM [24] : MEED Routing.

existing schemes by more than 22% on the average. Our scheme attains this higher delivery ratio by allocating the persistent storage carefully and selectively to the high ranked bundles. It emphasizes on accepting bundles with smaller TTL values, therefore, more

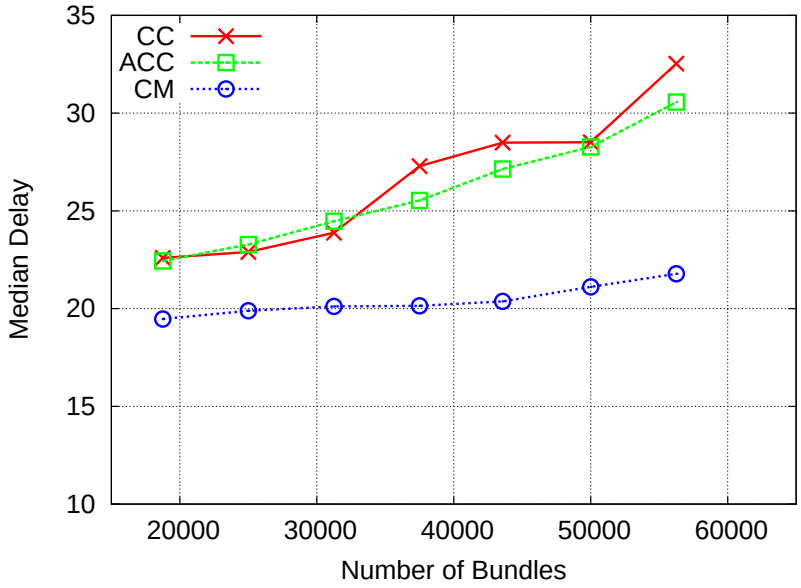

Figure 5.15: Median Delay vs. Number of Bundles for CC, ACC [4], CM [24] : MED Routing.

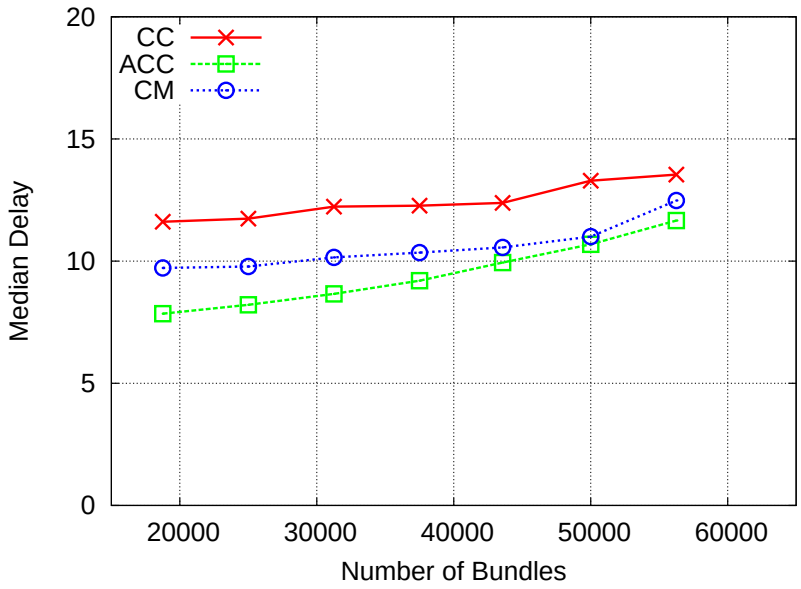

Figure 5.16: Median Delay vs. Number of Bundles for CC, ACC [4], CM [24] : MEED Routing.

bundles can reach their destination before getting expired. Our scheme also reserves a portion of the storage capacity for bundles that require persistent contact to get forwarded to the respective next hop. This also ensures more bundles to reach their next hops.

Figure 5.15 and 5.16 illustrates the effect of changing the number of created bundles on

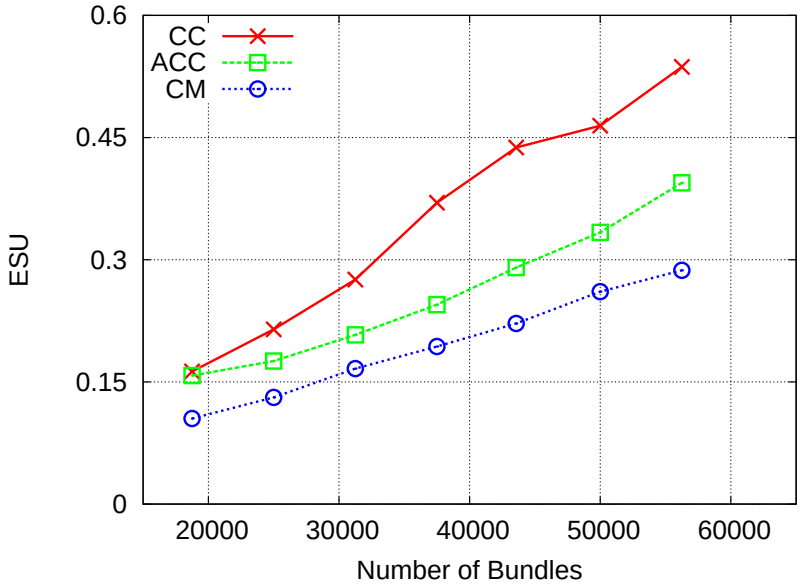

Figure 5.17: Effective Storage Utilization vs. Number of Bundles for CC, ACC [4], CM [24] : MED Routing.

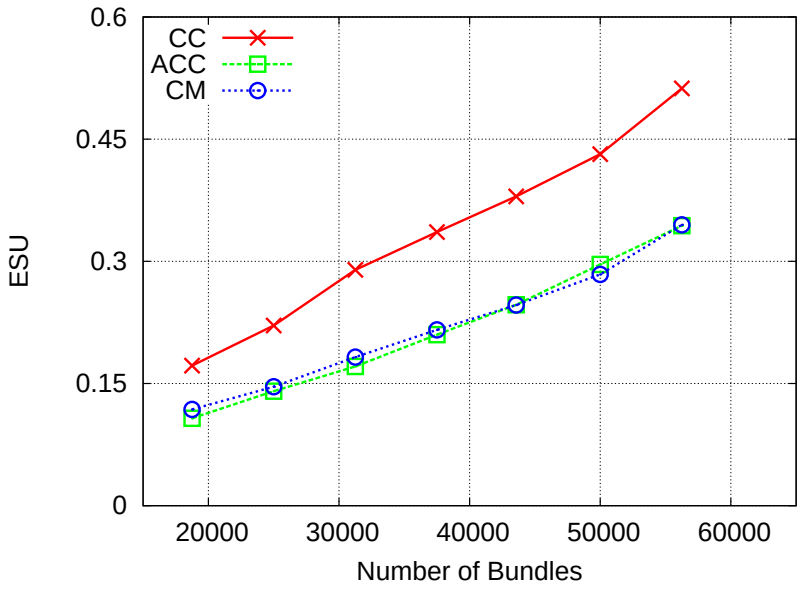

Figure 5.18: Effective Storage Utilization vs. Number of Bundles for CC, ACC [4], CM [24] : MEED Routing.

the median delay of the successfully delivered bundles. From the figure it is obvious that, the median delay increases with the increase in the number of generated bundles. Since

the storage capacity of the network is kept constant, the network faces more congestion as the number of generated bundles grows from low to high. Therefore, more contention for the storage occurs among the incoming bundles in a node and a bundle faces congestion in most of the nodes along the source to the destination. Consequently, the delay for the successfully delivered bundles increases. The figures also show that our congestion control scheme suffers from higher delay than that of the existing schemes, but is able to keep the delay within some reasonable limit. In DTN, a certain delay is tolerable.

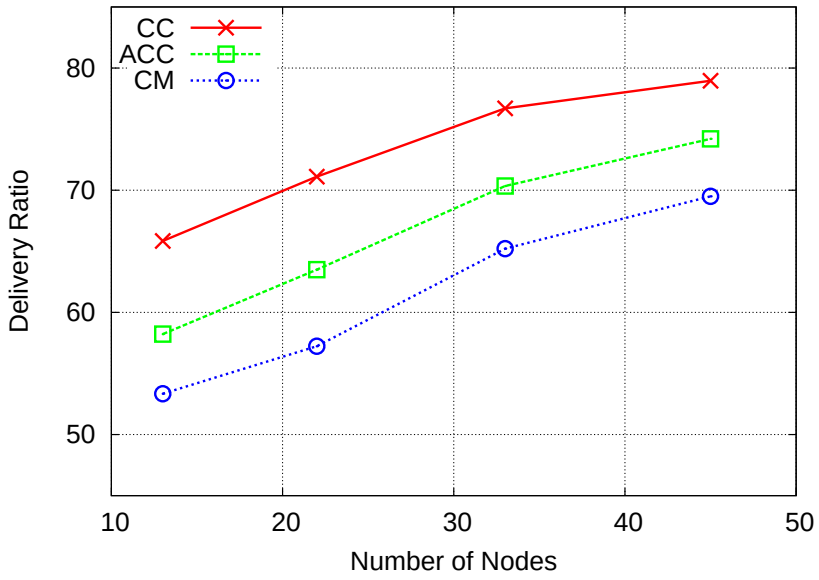

Figure 5.19: Delivery Ratio vs. Number of Nodes for CC, ACC [4], CM [24] : MED Routing.

If we consider both the delivery ratio and the delay in one metric, we can compare the overall performance of the schemes. Figure 5.17 and 5.18 depicts such performance in terms of Effective Storage Utilization (ESU) of the network which considers both the delivery ratio and the delay. The figures demonstrate that, our congestion control scheme utilizes the persistent storage more effectively than the existing schemes do, thus improves the overall performance of the network. It is obvious that our congestion control scheme achieves the highest delivery ratio while keeping the delay reasonable. The storage in a node is mostly occupied by the bundles that are ultimately delivered to its final destination. Therefore, the storage is effectively used by the delivered bundles.

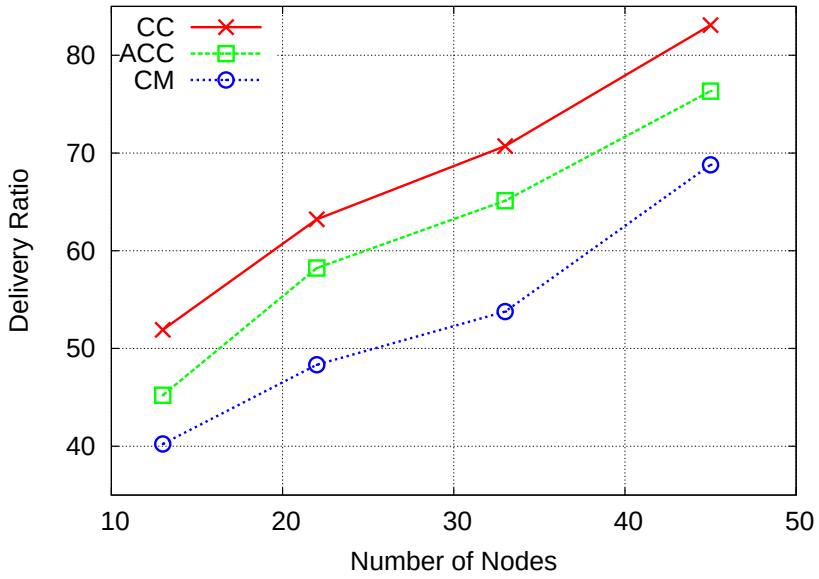

Figure 5.20: Delivery Ratio vs. Number of Nodes for CC, ACC [4], CM [24] : MEED Routing.

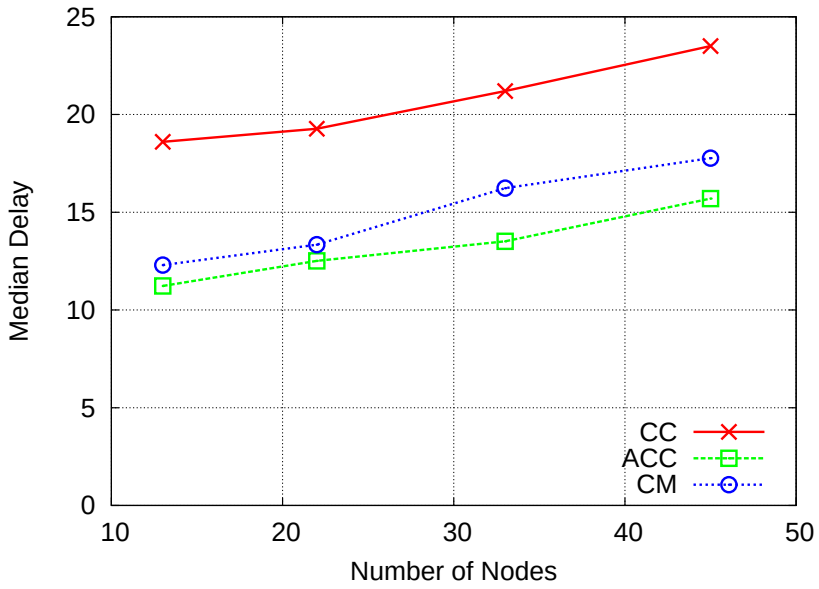

Figure 5.21: Median Delay vs. Number of Nodes for CC, ACC [4], CM [24] : MED Routing.

Figure 5.19 and 5.20 demonstrates the effect of changing the number of nodes on the delivery ratio of the network. As the number of nodes grow, the capacity of the network increases. The nodes can keep more bundles in their persistent storage as the total storage capacity of the network increases. The average contact opportunity of the nodes

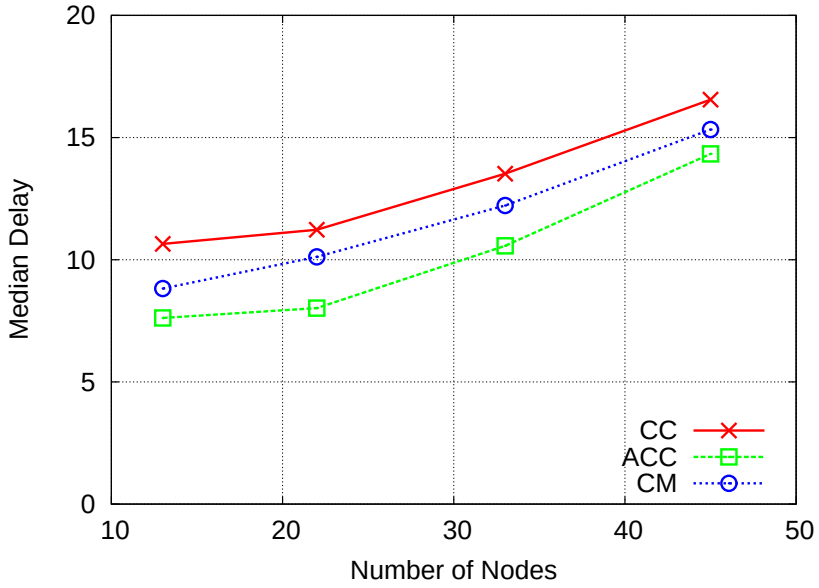

Figure 5.22: Median Delay vs. Number of Nodes for CC, ACC [4], CM [24] : MEED Routing.

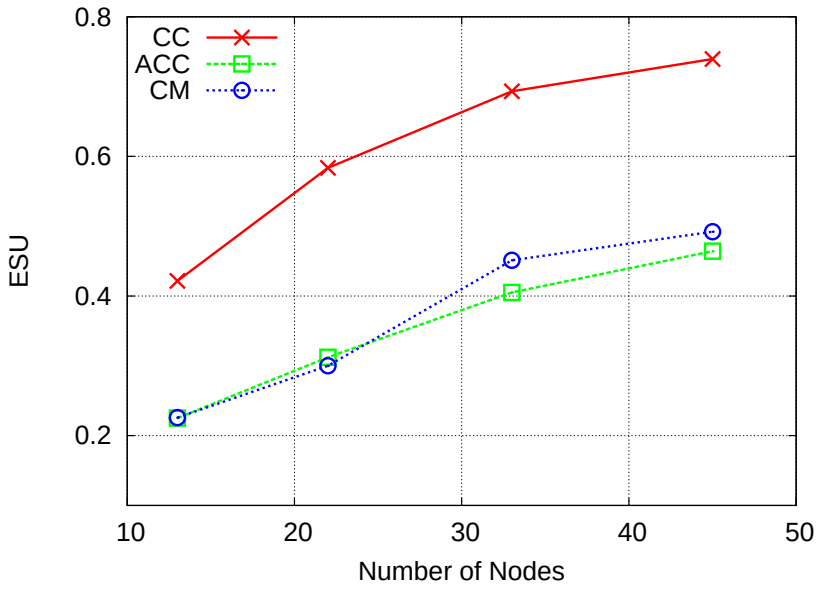

Figure 5.23: Effective Storage Utilization vs. Number of Nodes for CC, ACC [4], CM [24] : MED Routing.

also increases due to the increment in their number. Growths in the network capacity and the contact opportunity together ease the congestion from the network. Therefore, the delivery ratio increases with the increase in the total number of nodes in the network.

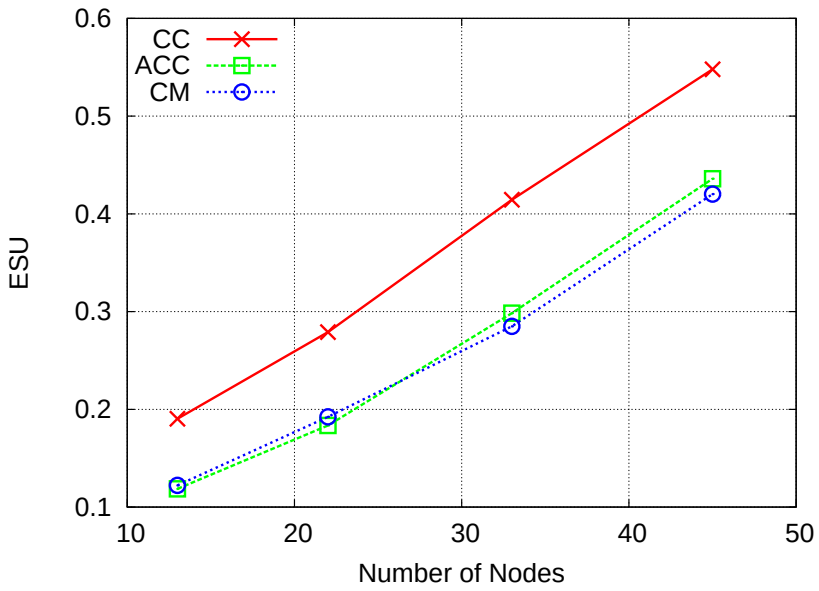

Figure 5.24: Effective Storage Utilization vs. Number of Nodes for CC, ACC [4], CM [24] : MEED Routing.

Figure 5.19 and 5.20 also shows that our congestion control scheme outperforms the existing schemes in terms of delivery ratio for both small and big sized networks. Our scheme attains this higher delivery ratio by allocating the persistent storage carefully and selectively to the high ranked bundles. It emphasizes on accepting bundles with smaller TTL values, therefore, more bundles can reach their destination before getting expired. Our scheme also reserves a portion of the storage capacity for the bundles that require persistent contact to get forwarded to the respective next hop. This also ensures more bundles to reach their next hops.

Figure 5.21, 5.22 and Figure 5.23, 5.24 illustrate the performance of our scheme in terms of Median Delay and Effective Storage Utilization respectively for varying network size respectively. Median Delay is increasing with the increase in the number of nodes. As the number of nodes increases, the network itself becomes larger which makes the average source to destination path length larger. As a result, bundle delivery time increases. Both the median delay and the delivery ratio of the network increase with the number of nodes in the network. As a combined effect the effective storage utilization is increased, which

is demonstrated in Figure 5.23 and 5.24.

5.4 Summary

In this chapter, we have presented a comprehensive performance comparison of our congestion control scheme with that of some existing schemes. For comparison, three matrices are evaluated - i) Delivery Ratio, ii) Median Delay, and iii) Effective Storage Utilization. DTNSim2 is used as the simulator. We have varied - i) Persistent Storage Capacity of a node ii) Number of generated bundles, and iii) Number of nodes in the network in order to study the performance. We used i) MED and ii) MEED routing protocols. Simulation result confirms that our congestion control scheme attains higher delivery ratio than that of the existing schemes for both routing protocols. Our scheme ensures this higher delivery ratio by taking optimal decision based on congestion level, by deferring the bundle rejection decision as far as possible and by introducing the HLB in decision making. Median delay is increased slightly since our scheme keeps as many bundles as possible in the persistent storage of a node. Our scheme, however, increases effective storage utilization significantly.

Chapter 6

Conclusion and Future Direction

DTN has the lack of persistent connectivity among the nodes. Traditional network protocols fail to cope with these challenged networks since the protocols have assumed a persistent end-to-end connectivity between the source and the destination nodes. To deal with this inconsistent connectivity, DTN adopts a hop-by-hop store and forward routing protocols along with the custody transfer of the bundles to ensure the reliable delivery. Due to the limited storage capacity in the nodes congestion occurs when too many bundles contend for this limited space. To control congestion and to make effective use of limited storage space, DTN comes up with both reactive and proactive congestion control strategies. In reactive approach, a congested node pushes its bundles to uncongested neighbor nodes. This approach suffers from bundle loss in highly congested networks and introduces too much overhead. Proactive approach adopts a form of flow control where a congested node rejects some of the incoming custody request to enforce the previous nodes to regulate their flow. This prevents bundle loss and lessens the overhead. But existing schemes based on this approach suffers from poor delivery ratio. In this thesis, we proposed a novel congestion control algorithm based on proactive approach that outperforms the existing schemes.

Our congestion control scheme introduces the monitoring of the network and measures the present level of congestion. When a node detects congestion, it starts controlling the

flow by rejecting some of the incoming custody requests. It introduces the concept of head of line blocking in order to take better decisions. Confidence level of a node facilitates the decision making further.

In our simulation, we demonstrate the superiority of our congestion control scheme by comparing its performance with that of other existing schemes. Our scheme achieves more than 20%-25% higher delivery ratio than that of existing protocols while maintaining a reasonable delay in comparison with them. It also ensures higher buffer utilization.

6.1 Future Direction

Our work can be extended in various directions. First of all, we can combine our congestion control scheme with some reactive approach. In this new scheme, DTN nodes can regulate the flow of bundles as well as they can migrate the bundles to some of their uncongested neighbors at the time of congestion. This will ensure a high delivery ratio along with the maximal use of the persistent storage capacity of the nodes. Extending our idea in order to incorporate it with the routing protocols in opportunistic networks would be very interesting and also very challenging. Opportunistic routing protocols adopt flooding based bundle forwarding mechanism. In flooding, a node replicates its stored bundles in the nodes with which the concerned node has the opportunity to contact. Controlling this replication factor using the concepts of our scheme might be a direction of this integration. Dropping of bundles in opportunistic networks is a safe operation since multiple copies of a bundle are present in the network at a time. Selecting the candidates for dropping at the time of congestion by using our ideas will also be an extension of our work. Introduction of social awareness in decision making will be another direction of extension.

Bibliography

[1] *Delay Tolerant Network Simulator*. http://watwire.uwaterloo.ca/DTN/sim/.

[2] *Delay Tolerant Networking Research Group*. http://www.dtnrg.org.

[3] E. Bulut, S. C. Geyik, and B. K. Szymanski. Efficient routing in delay tolerant networks with correlated node mobility. In *MASS*, pages 79–88, 2010.

[4] S. Burleigh, E. Jennings, and J. Schoolcraft. Autonomous congestion control in delay-tolerant networks. In *SpaceOps*, pages 70–79, 2006.

[5] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, and J. Scott. Pocket switched networks: Real-world mobility and its consequences for opportunistic forwarding. Technical Report UCAM-CL-TR-617, Computer Laboratory, University of Cambridge, February 2005.

[6] X. Chen, J. Shen, and J. Wu. A novel information model for efficient routing protocols in delay tolerant networks. In *IPDPS*, pages 1–8, 2009.

[7] Cerf et. al. *Delay-Tolerant Network Architecture*. RFC 4838, April 2007.

[8] S. Burleigh et al. *Licklider Transmission Protocol - Motivation*. RFC 5325, September 2008.

[9] K. R. Fall. A delay-tolerant network architecture for challenged internets. In *SIGCOMM*, pages 27–34, 2003.

[10] K. R. Fall and S. Farrell. Dtn: An architectural retrospective. *IEEE Journal on Selected Areas in Communications*, 26(5):828–836, 2008.

[11] K. R. Fall and W. Hong. Custody transfer for reliable delivery in delay tolerant networks. Technical Report IRB-TR-03-030, Intel Research Berkeley, 2003.

[12] C. N. Chuah J. LeBrun and D. Ghosal. Knowledge based opportunistic forwarding in vehicular wireless ad hoc networks. In *IEEE VTC Spring*, pages 2289–2293, 2005.

[13] S. Jain, K. R. Fall, and R. Patra. Routing in a delay tolerant network. In *SIGCOMM*, pages 145–158, 2004.

[14] E. P. C. Jones, L. Li, J. K. Schmidtke, and P. A. S. Ward. Practical routing in delay-tolerant networks. *IEEE Trans. Mob. Comput.*, 6(8):943–959, 2007.

[15] A. Lindgren, A. Doria, and O. Schelén. Probabilistic routing in intermittently connected networks. *SIGMOBILE Mob. Comput. Commun. Rev.*, 7(3):19–20, 2003.

[16] C. Liu and J. Wu. Scalable routing in delay tolerant networks. In *MobiHoc*, pages 51–60, 2007.

[17] I. Psaras, L. Wood, and R. Tafazolli. Delay-/disruption-tolerant networking: State of the art and future challenges. Technical report, University of Surrey, UK, 2010.

[18] K. Scott and S. Burleigh. *Bundle Protocol Specification.* RFC 5050, November 2007.

[19] M. Seligman, K. R. Fall, and P. Mundur. Alternative custodians for congestion control in delay tolerant networks. In *SIGCOMM*, pages 229–236, 2006.

[20] M. Seligman, Kevin R. Fall, and P. Mundur. Storage routing for dtn congestion control. *Wireless Communications and Mobile Computing*, 7(10):1183–1196, 2007.

[21] J. Shen, S. Moh, and I. Chung. Routing protocols in delay tolerant netowrks: A comparative survey. Technical report, The 23rd International Technical Conference on Circuits/Systems, Computers and Communications, 2008.

[22] T. Spyropoulos, K. Psounis, and C. S. Raghavendra. Spray and wait: An efficient routing scheme for intermittently connected mobile networks. In *SIGCOMM*, pages 252–259, New York, NY, USA, 2005.

[23] A. Vahdat and D. Becker. Epidemic routing for partially-connected ad hoc networks. Technical Report CS-200006, Duke University, 2000.

[24] G. Zhang, J. Wang, and Y. Liu. Congestion management in delay tolerant networks. In *WICON*, page 65, 2008.