A Message Transfer Framework for Enhanced Reliability in Delay-Tolerant Networks

Farzana Yasmeen, Uyen Trang Nguyen

Department of Electrical Engineering and Computer Science,

York University, Toronto, Canada

E-mail: {yasmeen, utn}@cse.yorku.ca

Nurul Huda

Ted Rogers School of Information Technology Management,

Ryerson University, Toronto, Canada

E-mail: nurul.huda@ryerson.ca

Cristian Borcea

Department of Computer Science, New Jersey Institute of Technology, USA

E-mail: borcea@njit.edu

Shigeki Yamada

National Institue of Informatics, Tokyo, Japan

E-mail: shigeki@nii.ac.jp

Received: August 30, 2015 Accepted: Nov. 1, 2015 Published: Nov. 28, 2015

DOI:10.5296/npa.v7i3.8201 URL: http://dx.doi.org/10.5296/npa.v7i3.8201

Abstract

Delay-tolerant networks (DTNs) can tolerate disruption on end-to-end paths by taking advantage of temporal links emerging between nodes as nodes move in the network. Intermediate nodes store messages before forwarding opportunities become available. A series of encounters (i.e., coming within mutual transmission range) among different nodes will eventually deliver the message to the desired destination. The message delivery performance in a DTN (such as delivery ratio and end-to-end delay) highly depends on the time elapsed between encounters and the time two nodes remain in each others communication range once a contact is established. As messages are forwarded opportunistically among nodes, it is important to have sufficient contact opportunities in the network for faster, more reliable delivery of messages. We propose a simple yet efficient method for improving the performance of a DTN by increasing the contact duration of encountered nodes (i.e., mobile devices). Our proposed sticky transfer framework and protocol enable nodes in DTNs to collect neighbors' information, evaluate their movement patterns and amounts of data to transfer in order to make decisions of whether to "stick" with a neighbor to complete the necessary data transfers. The sticky transfer framework can be combined with any DTN routing protocol to improve its performance. We evaluate our framework through simulations and measure several network performance metrics. Simulation results show that the proposed framework can improve the message delivery ratio, end-to-end delay, overhead ratio, buffer occupancy, number of disrupted message transmissions and so on. It can be well adopted for challenged scenarios where larger messages sizes need to be delivered with application deadline constraints. Furthermore, performance of the DTN improved (upto 43%) at higher node densities and (up to 49%) under increased mobility conditions.

Keywords: Delay-tolerant network, enhanced reliability, message transfer protocol, mobility management framework, opportunistic routing.

1 Introduction

Opportunistic networks [1] can be realized and implemented by both Delay-Tolerant Networks (DTNs) [2] and Mobile Ad-hoc NETworks (MANETs) [3]. However, in sparse, mobile, infrastuctureless environments, frequent network partitions and per-link instability render conventional MANET routing protocols [4–6] ineffective, due to a high number of timeouts and retransmission requests in the underlying transmission protocol and required pre-established source-to-destination connections before data transfers. DTN protocols [7–15], on the other hand, can handle long, frequent, and intermittent link conditions in the network by using the store-and-forward mechanism [37] and opportunistic contacts for routing, given that the sequence of connectivity graphs over a time-interval forms a virtual end-to-end path. The trade-off is generally high delivery delays. Suitable applications for DTNs are non-real time, such as file transfer, sensor data collection, and messaging. Examples of DTNs in practice include (but are not limited to) 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, underwater acoustic networks with moderate delays and frequent interruptions due to environmental factors, vehicular networks with cyclic but non-deterministic connectivity, disaster recovery situations, rural area communication scenarios, wildlife monitoring systems, and battlefield operations.

The message delivery performance (such as message delivery ratio and end-to-end delay) in a mobile DTN highly depends on the time elapsed between encounters (the inter-contact time) and the time two nodes remain in each other's communication range once a contact is established (contact duration) as node contacts are opportunistic and limited in such mobile networks. Node mobility may cause nodes to move out of each other's transmission range in the middle of a transmission, interrupting the transmission and wasting the resource consumed by the failed transfer. In addition, many other messages which have been processed and ready for transmission cannot be forwarded. These messages will stay longer in buffers of limited sizes, which may eventually be discarded due to buffer overflow, wasting node resources. The end result is low message delivery ratio and long end-to-end delay. The above problems are exacerbated in DTNs with highly mobile nodes that must handle large messages, such as vehicular networks [16, 17].

The objective of our research is to maximize node contact durations for message transfers in delay-tolerant networks. To solve the above problems, we propose a novel framework called the sticky transfer framework that enables nodes to prolong their contact durations for message transfers in DTNs. In this framework, nodes send out periodic beacons for neighbor discovery. Once a neighbor is detected with which a node can perform sticky transfers, the nodes exchange information such as their mobility speed and direction, current location, transmission range, available buffer size, the amount of data to be sent and the corresponding destination, using our proposed sticky transfer protocol within the framework. Based on the received information, a node A calculates the needed contact duration based on the amount of data to be exchanged/transferred with a neighbor B, determines the required mode of movement (e.g., slowing down or stopping in order to stay in contact), and negotiates an agreement to stick with B (e.g., negotiating the mode of movement and contact duration). After the sticky transfer is over, nodes A and B resume their original movement behavior.

Sticky transfers can be used to improve the network performance of many applications, such as (1) robots in a region survey application may be programmed to stick with each other longer when needed to improve message delivery ratio and delay; (2) emergency response team members could be asked to stop or follow each other when necessary to improve the network performance; (3) a network of mobile sensors engaged in ecological monitoring could use sticky transfers to enable faster message delivery to the sink. Note that the sticky transfer mechanism is optional; nodes may choose not to run the protocol or ignore sticky transfer requests from other nodes.

In the remainder of this paper, *'messages'* denotes any type of payload content, including but not limited to text, audio, video, and image. We also use the terms 'messages', 'packets' and 'bundles' interchangeably. To evaluate the effectiveness of the proposed sticky transfer framework, we performed simulations using a city-based network topology and the Spray-and-Wait [9], PRoPHET [8], and Epidemic [7] routing protocols. We evaluated the performance of each routing protocol with and without sticky transfers. We performed simulations on the Opportunistic Network (ONE) simulator, a simulation environment capable of routing messages between nodes using various DTN routing algorithms [27]. Our simulation results show a significant improvement in the performance of the routing protocols in the presence of sticky transfers. The message delivery ratio increased by as much as 38%, while the end-to-end delay decreased by as much as 36% with sticky transfers.

In summary, our contributions include:

- a novel framework called the *sticky transfer framework* that enables mobile nodes in a DTN to modify their mobility and prolong contact durations to enhance successful message transfers.
- a *sticky transfer protocol* within the framework that governs how pair-wise mobiles nodes will "stick" to each for a negotiated period of time in order to complete the transmissions of messages.

Sticky transfers can increase delivery ratio and decrease delivery delay by ensuring faster forwarding and reducing the number of message transfer aborts. Our framework is especially beneficial for large message sizes and/or high mobility situations, where contact times allow for few successful transfers.

The remainder of this paper is organized as follows. We discuss background information and related work in section 2. In section 3, we describe our proposed sticky transfer framework and sticky transfer protocol. In section 4, we present simulation results and analysis of our proposed framework and protocol. We conclude the paper in section 5 with a of our findings and an outline of future work. This article is an extended version of [43].

2 Related Work

In DTNs, the encounter delay can be quite large, especially if the network is sparse. The success of message delivery highly depends on the likeliness of nodes encounters and the time elapsed between encounters greatly influences other performance metrics of the network, such as delay, storage occupancy, and so on. DTNs utilize node mobility to achieve message delivery. Nodes forward messages only when they encounter the appropriate relay or the destination node. Due to this dependence on the mobility of nodes participating in the network, understanding mobility characteristics of mobile nodes plays an important role in the analysis of routing algorithms and the overall performance of the network.

Two main contributing factors that determine the performance of a DTN and are directly related to the mobility characteristic of the network are *inter-contact time* and *contact duration* [20]. We define them as:

- *Average Inter-contact Time, I_C*: The *average inter-contact* time measures how frequently nodes encounter other nodes on average in the network. Specifically, it is the duration from the moment a node A moves out of the transmission range of node B until node A encounters another node (which could be B again). Inter-contact time depends primarily on node mobility and node density in the network. In sparse networks, the inter-contact time can be reduced by introducing special components, such as ferries [18] or data mules [19], that move at relatively faster speeds on predefined routes and therefore increase contact opportunities. For example, in disaster recovery operations, such as those performed after earthquakes or tsunami's, where communication infrastructure has been damaged, mobile devices in that area may use a temporarily deployed DataMule [19] to carry data to and from a sink remotely connected to the Internet.

- *Average Contact Duration, T_C*: The *average contact duration* is the length of time during which pair-wise nodes remain within the transmission range of each other on average in the network. The contact duration directly influences the capacity of opportunistic networks (e.g., DTNs) as it limits the amount of data that can be transferred successfully between nodes.

The inter-contact time and contact duration directly impact the throughput of the network. We can characterize the average throughput (THP), which is the maximum data rate that can be sent between two nodes as:

$$THP = \frac{T_C \cdot R}{(T_C + I_C)} \tag{1}$$

where, R is the transmission rate in bits per time unit.

Thus, the performance of DTNs can be improved by reducing the inter-contact time and increasing the contact duration, which is observed from 1.

Numerous routing protocols [7–24] have been proposed for DTNs. Routing algorithms can exploit mobility, but they can not essentially change the mobility characteristics inherent of a networks participating nodes. For example, replication based routing protocols [7–9] create and forward (i.e., *flood*) multiple copies of a message when an encounter happens. The heuristic behind this policy is that, since there is no knowledge of a possible path towards the destination nor of an appropriate next-hop node, a message should be sent everywhere. It will eventually reach the destination by passing node by node. These protocols, standalone however, do not reduce the inter-contact time nor increase the contact duration of pair-wise nodes in the network. By using the proposed sticky transfer framework, nodes can intelligently and co-operatively increase their contact durations to improve the capacity of the network by agreeing to brief, temporary modifications in their movement patterns and speeds.

In our proposed framework, we evaluate the following three replication-based routing protocols: Epidemic [7], PRoPHET [8], and Spray-and-Wait (SnW) [9]. In the Epidemic routing protocol [7], whenever two nodes encounter each other, a node forwards all the messages the other node has not received previously. Epidemic routing tends to find optimal paths and returns lower average end-to-end delays at a cost of high resource consumption. PRoPHET [8] reduces the overhead of flooding by forwarding copies probabilistically using a delivery predictability table to each known node in the network. The delivery probability increases upon encounters, and decreases over time in case no meeting occurs. PRoPHET does not perform well under suddenly changed social communication patterns (such as in a disaster-stricken areas) because the probability of meeting two specific nodes (calculated based on their history) no longer applies. In the Spray-and-Wait [9] protocol, the source of a message initially starts with *K* copies. When any node with *K*>1 message copies (source or relay) encounters another node with no copies of the message, it transfers $\lfloor K/2 \rfloor$ copies to the other node and keeps $\lceil K/2 \rceil$ copies for itself. When a node is left with only one copy, it waits to meet the destination and directly delivers the message upon encounter. The performance of Spray-and-Wait depends largely on the number of initial copies, *K* at the source.

Considering practical applications and assuming that mobility and traffic load are given, the delivery ratio of the DTN is affected by two factors. Firstly, messages may have an expiration timeout or time-to-live (TTL) set by applications. The messages may be useless after this TTL expires and therefore are dropped. Secondly, generally there is a “mission deadline”, and some messages may not be delivered before this deadline (i.e. their TTL did not expire, but the use of the DTN is over). Therefore, it is crucial to deliver messages faster, and implicitly to avoid message aborts. Our proposed sticky transfer framework would be able to achieve just that. For example, if a routing protocol wants to forward K copies of a message to different nodes in order to improve delivery, sticky transfer will ensure that the K copies are forwarded faster; in fact, the copies would be forwarded during the first K contacts with different nodes. Consequently, this would improve overall performance of the network with observably lower

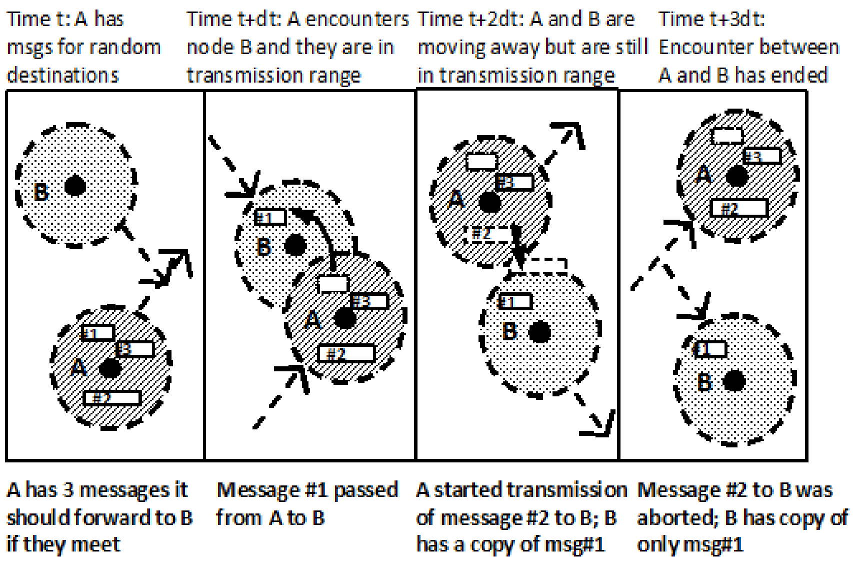

Time t: A has msgs for random destinations
Time t+dt: A encounters node B and they are in transmission range
Time t+2dt: A and B are moving away but are still in transmission range
Time t+3dt: Encounter between A and B has ended
B
A
#1
#2
#3
A has 3 messages it should forward to B if they meet
Message #1 passed from A to B
A started transmission of message #2 to B; B has a copy of msg#1
Message #2 to B was aborted; B has copy of only msg#1

Figure 1: Data transfer limitations due to natural movement

latencies and higher delivery ratios. Our proposed sticky transfer framework is independent of, but can function with, any DTN routing protocol.

Zhuo et al. [21] propose a packet-level replication protocol, which uses erasure coding to encode large messages into smaller packets, to address the problem of limited contact duration. However, this technique requires replication of packets at each node, which is expensive in a DTN. Our sticky transfer protocol requires no additional mechanism nor infrastructure other than the simple beaconing mechanism which has been used for many other purposes in wireless ad hoc networks. To the best of our knowledge, our work is the first to propose the concept of 'sticky' message transfers to extend contact durations.

3 The Proposed Sticky Transfer Framework

In opportunistic networks such as DTNs, limitations in the actual time to transfer data between nodes result from the inherent mobility of hosts, as illustrated by the example shown in Figure 1.

In this example, at time t node A has three variable-length messages {#1, #2, #3} queued in its buffer to send to the next encountered node using a flooding-based routing strategy for message forwarding. At time t+δt, A is presented with contact opportunity B as node B comes within the wireless communication range of A. A successfully transmits message $\#1$ to B, and retains metadata of the copy of message $\#1$ sent to B. At time t+2δt, A proceeds to transfer message $\#2$ to B, while A and B are gradually moving out of each other's communication range. Due to their velocity, the contact duration to transmit message $\#2$ is insufficient. Hence at time t+3δt, node A was unsuccessful to transmit message $\#2$ to B, which may have been a better candidate (next hop node) for delivery on the time-varying path to the message's eventual destination. Also, node A now requires to re-transmit the message to the next viable encounter.

Moreover, the DTN Bundle Protocol specification [29] does not limit bundle size or specify content of bundles. Indeed, a bundle may: (i) contain a single file (e.g., a photograph), multiple files (such as small-sized engineering telemetry files), or a file segment (possibly part of highquality video); (ii) be of fixed size determined by application type or network management procedures; (iii) be of variable size set by application, and containing a coherent bundling of application data. Key DTN functionality is that each bundle is kept in memory in its entirety, and is deleted upon receipt of acknowledgment for its successful delivery to the next node on the path to the destination.

Fragmentation allows smaller sized messages to be forwarded within smaller contact duration's. However, splitting complicates routing because, in addition to determining the sizes of the fragments, corresponding paths for the fragments also need to be determined. Moreover, if the message fragmentation size is fixed then the actual size of the splits may not be optimal to 'fit' within the contact duration.

When nodes move out of each other's transmission range, data transfer is discontinued. Thus, many messages which are ready for forwarding cannot be forwarded. Additionally, if a message happens to be in transit during the disconnection, the transmission is disrupted, which can result in an unsuccessful transmission. Therefore, insufficient contact duration during opportunistic communications results in underutilized contact opportunities (i.e. wasted contact time and bandwidth), which limits the capacity of the network. Furthermore, messages will stay longer in limited buffers, which may lead to message lifetime (TTL) expiration. Such problems are exacerbated in highly mobile DTNs (e.g. vehicular networks) that must handle large message sizes [16], [17].

To solve the above problem, we propose a novel framework called the *sticky transfer framework* for maximizing node contacts for message transfers in DTNs . In the proposed framework, to *'stick'* means mobile devices remain within the communication range of each other longer than the contact duration that would be expected between them without using the framework. By using the proposed sticky transfer scheme, nodes can cooperatively increase the contact duration to improve the capacity of the network by agreeing to brief, temporary modifications in their movement patterns. Nodes adjust mobility at the instance of an encounter, and return to normal movement behaviors after the encounter is over. Thus, by encouraging cooperative and voluntary mobility changes, the sticky transfer mechanism can substantially improve network performance.

As part of the framework, we also propose a *sticky transfer protocol* which governs how nodes engage and participate in sticky message transfers. The protocol allows nodes to exchange information, upon encounter, and calculate a compatible mode and duration for message exchanges. The agreement is made based on information gathered about current network parameters and *user preferences* (discussed in section 3.3.2). Based on this mutual decision, nodes may remain within the transmission range of each other until message transfers are completed. The amount of messages that are transferred depends on the underlying routing protocols message selection and forwarding strategy and the agreed stick-time. Ideally, all the messages that the routing protocol decides to transfer.

In this section, we first state the definitions and assumptions, and then describe the concept and components of the sticky message transfer framework and protocol. We also explain how the proposed framework can be seamlessly integrated into the existing DTN architecture.

3.1 Definitions and Assumptions

The *natural contact duration* T_C is the length of time during which two nodes are expected to remain within the transmission range of each other, and can be estimated as follows. Consider two nodes A and B that are in contact (i.e., within the transmission range W of each other) and moving on a plane at angles of θ_A and θ_B $(0 \leq \theta_A, \theta_B \leq 2\pi)$, and at speeds of v_A and v_B $(v_A, v_B > 0)$, respectively. Let (x_A, y_A) and (x_B, y_B) be the coordinates of A and B, respectively. By projecting the speeds and directions of the two nodes along their movements, the natural contact duration T_C of the two nodes can be estimated to be:

$$T_C = \frac{-(ab+cd) + \sqrt{(a^2+c^2)W^2 - (ad-bc)^2}}{a^2+c^2} \tag{2}$$

where $a = v_A \cos\theta_A - v_B \cos\theta_B$, $b = x_A - x_B$, $c = v_A sin\theta - v_B sin\theta_B$, and $d = y_A - y_B$. This estimated value is the expected contact duration between the two nodes. Note that when $v_A = v_B$ and $\theta_A = \theta_B$, T_C approaches ∞. That is, A and B will always stick to each other.

We make the following assumptions regarding equation 2:

1. – We assume that every node is equipped with a GPS, which helps to determine its speed and direction of movement.
2. – All nodes have the same transmission range, W. This can easily be modified to include nodes with heterogeneous transmission ranges [35].
3. – A node is in the transmission range of only one node. If at time t_0, A comes into the transmission range of B and moves away from B at time t_1, then $T_C(A, B) = t_1 - t_0$.

   If multiple nodes are in the transmission range of each other, we assume the mutual encounter sequence comes naturally from the order in which a nodes receives beacon messages [40] from other nodes. For example, if at time t_0, A comes into the transmission range of B and C and receives B's beacon message first, A will finish sticky transfers with B first and then begin sticky transfers with C (assuming that C has not already moved away). Assuming that at time t_1, A moves away from the transmission range of B and C, then $T_C(A, B) + T_C(A, C) = t_1 - t_0$. However, if C has already moved out of range while A is transferring messages to B then $T_C(A, C) = 0$.

On the other hand, the time required for A to complete transferring all messages scheduled by the routing protocol to B is the *required transfer duration* T_R. Let R be the transmission rate of the nodes. (The calculation can easily be extended to nodes transmitting at different rates). If node A has p messages to send to node B, B has q messages to send to A, and M_i denotes the size of message i, then the required transfer duration between A and B is

$$T_R = \frac{\sum_{k=1}^{p} M_k + \sum_{l=1}^{q} M_l}{R} \tag{3}$$

Assuming that the message transfer starts immediately after nodes encounter each other, if $T_C(A, B) < T_R(A, B)$ then message aborts are probable and not all of the messages that A wants to forward to B can be transferred within the estimated contact time.

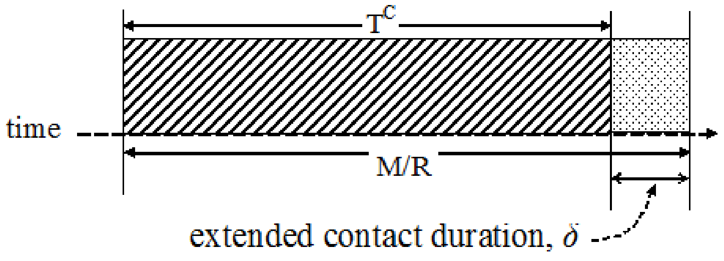


Figure 2: Extending the contact duration

We envision the sticky transfer protocol to be a part of the bundle layer, and hence any lower layer technology for each layer in the protocol stack can be implemented. For example,

– At the MAC layer, any MACAW (Multiple Access with Collision Avoidance for Wireless) [38] protocol, such as IEEE 802.11 RTS/CTS, could be used to cope with collisions and hidden/exposed terminal problems.

– At the transport layer, we can assume a point-to-point reliability protocol such as *Saratoga* [41], [42]. Saratoga is a UDP-based protocol, supporting a Selective Negative Acknowledgment (SNACK) with holes-to-fill strategy [41] for reliable retransmissions. Additionally, for reliable transfer, data packets are sent using UDP with checksums turned on.

– 802.11b, or 802.11g can be used at the physical layer.

3.2 The Concept of Sticky Message Transfers

To maximize the use of valuable contact opportunities, the sticky transfer mechanism allows nodes to come to a mutual agreement on the time during which they will remain in each other's communication range. Once all message transfers are complete, nodes may "unstick" and resume their natural mobility patterns. Assuming that T_C is the natural but insufficient contact duration for the message transfer, the additional time the nodes should remain in contact beyond the natural contact duration is $\delta = T_R - T_C$, where $T_R = M/R$ and M is the total size of all the messages to be transferred between the two nodes (Fig. 2). We call δ the *stick duration*, which is calculated by nodes using the sticky transfer protocol described later on in section 3.5.

We expect this mechanism to improve the performance of the network in two ways.

First, sticky transfers will be able to deliver messages faster in the network, hence minimize the end-to-end delay. For example, if a routing protocol wants to forward K copies of a message to different nodes in order to improve delivery, the sticky transfer mechanism ensures that the K copies are forwarded faster; in fact, they will be forwarded during the first K contacts with encountered nodes. This leads to lower latencies and higher delivery ratios, which improves network performance.

Second, sticky transfers will minimize the number of message aborts, improving message delivery ratio and network resource utilization, as it allows the atomic transfer of messages.

The sticky transfer scheme is optional. Any user may opt-in or opt-out from exchanges using the sticky transfer protocol at any time. Several applications exist where sticky transfers can be used to improve performance: (1) robots in a region survey application may be pro-

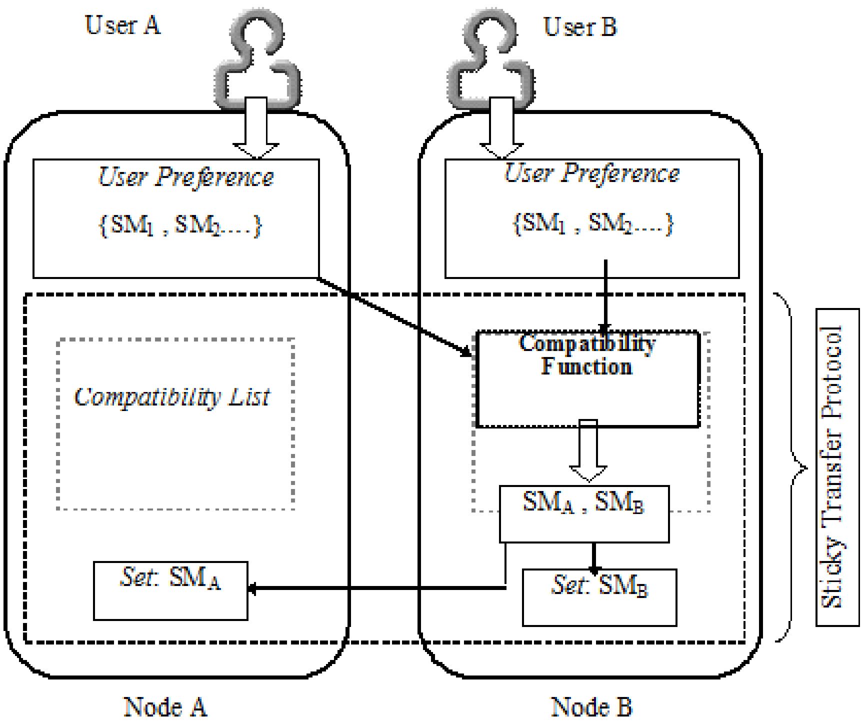


Figure 3: The sticky transfer framework

grammed to stick with each other for a certain stick time to improve message transfers, (2) emergency response team members could be asked to stop or follow each other for a while to improve the network performance, or (3) people may be asked to stick with each other to help in a socially-aware forwarding protocol, thus helping with data delivery in their community.

3.3 The Proposed Sticky Transfer Framework

The sticky transfer framework consists of three components: *sticky modes*, *user preferences*, and *compatibility lists*. Sticky modes are modes of operation defining users agreement to modify their natural movement. A *user preference* consists of an ordered list of acceptable sticky modes. Modes set as preferences represent how the users would respond to a sticky transfer request. A compatibility list, stored on each device/node/user, defines if two modes chosen by two encountered nodes allow for the engagement in a sticky transfer. A schematic of the framework is shown in Fig 3. We discuss each of these components as follows.

3.3.1 Sticky Modes

Two neighbor nodes can “stick” to each other by reducing their relative speed so that they remain within the transmission range of each other for the required transfer duration. The relative speed of the two nodes can be reduced by changing the speed and/or movement direction of one or both nodes. We define five sticky modes: *Stop, Follow me, Follow you, Slow down* and *No stick*.

The *Stop* (STP) mode is implemented by changing the relative speed of two nodes to zero. One way to achieve this is to change both of the nodes velocity to zero, i.e., stopping the nodes. Another way of achieving zero relative speed is to allow the two nodes to move at the same

Table 1: Sticky mode compatibility

SM_A	SM_B			
	STP	SLW	FLW1	FLW2
Stop (STP)	$\surd$	$\surd$	$\times$	$\surd$
Slow down (SLW)	$\surd$	$\surd$	$\surd/\times$	$\surd$
Follow me (FLW1)	$\times$	$\surd/\times$	$\times$	$\surd/\times$
Follow you (FLW2)	$\surd$	$\surd$	$\surd/\times$	$\times$

speed as the other in the same direction, i.e., one node follows the other node. The mode of the node which is followed by the other node is called *Follow me* (FLW1) mode. The mode of a node that adjusts its speed and direction to the other node's speed and direction in order to follow it is called *Follow you* (FLW2). When a node reduces its speed to match that of a slower moving node, its mode is called *Slow down* (SLW). Finally, a node may not agree to stick for message transfers. This mode is called *No Stick* (NO_STK).

Users (e.g., network administrators) can set one or more sticky modes in mobile nodes (e.g., sensors or robots), which make decisions based on the pre-defined modes and collected information (e.g., mobility speeds, movement directions, buffer sizes, and message sizes).

3.3.2 User Preferences

When setting sticky modes in mobile nodes, a user (e.g., network administrator) may not be able to select some of the modes at all. For example, a robot performing region surveys may not be able to use FLW2 mode due to its fixed route and schedule, but it may be able to use SLW mode for a very short duration. On the other hand, emergency response team members in a disaster stricken area may choose to accommodate all modes and set a low priority for NO_STK mode to ensure the highest level of cooperation among team members. We assume that nodes will have sticky mode preferences set according to the application before engaging in a mission.

A user preference consists of an ordered list of acceptable sticky modes. The order defines the priority of user preferences, with higher priority modes coming first in the list. In the framework (Fig. 3), users A and B have input and stored their preferred sticky modes (i.e. SM_1, SM_2 , and so on).

3.3.3 Compatibility List

Among the five sticky modes defined above, modes may or may not be compatible depending on the speeds and movement directions of the nodes involved in the negotiation. Sticky transfers are technically possible when nodes have *compatible* modes that allow for the mobility of each to mutually extend the contact. Two modes may or may not be compatible based on their speed limitations and movement direction. For example, when A's mode allows SLW and B's mode allows FLW1 then they are compatible if B's speed is slower than A's speed. They are not compatible if B's speed is faster than A's speed or both are not moving in the same direction.

For this reason, we construct a table that determines the compatibility between any two sticky modes (Table 1). In the table, $\surd$ indicates mode compatibility, $\times$ indicates incompatibility and $\surd/\times$ indicates the modes may sometimes not be compatible due to user limitations.

When implementing the framework, each node has a copy of the compatibility table. Among compatible modes, the most preferred modes are used during the sticky transfer. If compatible modes cannot be found, then sticky transfers are not possible and the nodes will exchange messages in the normal transfer mode, possibly with limited contact time.

3.4 The Sticky Transfer Framework within the DTN Overlay Architecture

The DTN architecture [36] was designed as an overlay to accommodate not only network connection disruption, but also to provide a framework for dealing with heterogeneity [25]. In Fig. 4 we show a conceptual overview of the sticky transfer protocol and components integrated in a layered DTN architecture. The overlay spans several planes where the sticky transfer module resides in the management plane along with the DTN routing module. As DTN can use a multitude of different delivery protocols including TCP/IP, raw Ethernet, serial lines, or hand-carried storage drives for delivery, the convergence layer provides the functions necessary to carry DTN protocol data units (PDUs) on each of the corresponding protocols at lower layers. The bundle protocol (BPL) is the central component in the overlay; it requires detailed information of the state of the system upon which to base routing decisions [26]. The management plane handles providing such information to the BPL. The sticky transfer module interacts with the routing module and adds components to existing routing management features to implement sticky forwarding decisions. The sticky control database contains compatibility information which is used by the control management system to calculate sticky modes. The sticky control management system communicates sticky transfer decisions to physical movement systems; such as - to the motion sensors on an environmental monitoring robot/vehicle. Essentially, when sticky transfer is combined with a specific DTN routing protocol, the overall network performance will depend on the performance of the routing protocol and the benefit of sticky transfers.

In the following section we describe the sticky message transfer protocol in detail.

3.5 The Proposed Sticky Transfer Protocol

We assume nodes have user preferences P and status information I consisting of movement vectors V (i.e., speed, direction, and current location), transmission range W, transmission rate R, free buffer size Buf, and message vectors μ (i.e., containing the message size and ID). Here,

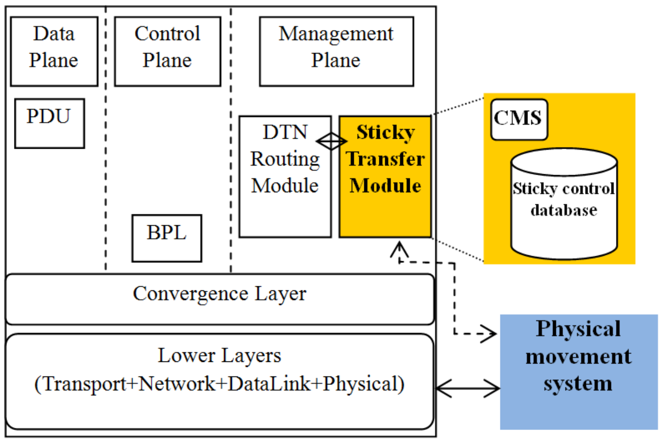


Figure 4: Sticky transfer framework integration in the DTN architecture

$V = \{v_j, \theta_j, (x_j, y_j)\}$ and $\mu = \{(\mu_1, id_1), (\mu_2, id_2), (\mu_3, id_3), \dots, (\mu_k, id_k)\}$, $j = 1, \dots, n$ and $k = 1, \dots, m$, where n is the number of nodes in the network and m is the number of messages to be transferred from node j. The set of messages in μ is decided for node j by the routing protocol strategy [7–9]. Suppose that nodes A and B have just come into each other's transmission range and have a number of messages to exchange. Fig. 5 shows the protocol sequence diagram for the sticky transfer of messages, assuming A sends the request first.

1. First, A sends a sticky transfer request to B along with its status information I_A and user preferences P_A. Sticky requests can be included in the beacon messages to reduce the number of messages exchanged.

2. After receiving the stick request from A, B first calculates the expected contact duration T_C between A and B using Eq. 2, the status information in I_A and its own status information I_B. B then determines the messages it needs from A by removing from μ_A the messages B already received. B then records the IDs of the messages it needs from A in a 'receive' vector M_r. B could remove some messages from M_r if it does not have enough buffer for all messages in M_r. Next, if B has messages to send to A from its own message vector μ_B, it will use A's free buffer space information Buf_A to determine the messages it wants to send to A without overflowing A's buffer and records their IDs in a 'send' vector M_s.

   B then calculates an upper bound on the required transfer duration T_R using the total size of the messages recorded in M_r and M_s, the transmission rate and Eq. 3. B can compare T_C and T_R to determine if the natural contact duration is sufficient. B also determines if a compatible stick mode exists between A and B using user preferences P_A and P_B.

3. If T_C is sufficient for completing the message exchange, sticky transfers are not necessary. B will notify A through a reply with the stick mode SM set to NO_STK and stick duration $\delta = 0$. B would also send a NO_STK message to A if A's and B's sticky preferences are not compatible. On the other hand, if compatible sticky modes exist between them and sticky transfers are necessary (i.e., $T_C < T_R$), B will update its own sticky mode (e.g., *Follow you*) and record the mode it expects A to use in a variable SM_A. B then sends an OK message to A, which contains the following information: message vectors μ_B, M_r and M_s, stick duration δ, status information I_B, and stick mode SM_A.

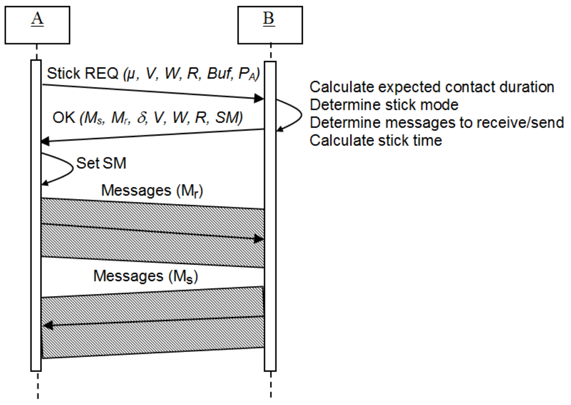

A
B
Stick REQ (μ, V, W, R, Buf, P_A)
Calculate expected contact duration
Determine stick mode
Determine messages to receive/send
Calculate stick time
OK (M_s, M_r, δ, V, W, R, SM)
Set SM
Messages (M_r)
Messages (M_s)

Figure 5: The sticky transfer protocol sequence diagram

4. A receives the OK message from B and sets its stick mode as defined in SM_A. Next it updates vector M_s by removing from M_s the messages it had received. A then sends to B the data messages B has indicated in vector M_r. A also sends the updated vector M_s to B by piggybacking M_s onto some of the data messages.

5. After receiving M_r and the updated vector M_s, B will send messages indicated in M_s to A to complete the transfer. After completing the message transfers, the nodes will resume their natural movements.

Since the estimation of the required transfer time may be slightly lower than necessary or nodes may opt-out from the stick transfer agreement, a limited number of aborts are still possible. As previously mentioned in section 3.1, during sticky transfers if there are in fact multiple nodes in range, the requester (e.g., A) selects one particular sequence of mutual encounters. This sequence may come naturally from the order in which A hears the advertisement messages of other nodes.

3.6 Setting the Speed in Sticky Modes

To complete message transfers, as described in section 3.1, it is necessary to extend the natural contact duration, T_C when it is smaller than the required transfer duration, T_R. This extension can be achieved by reducing the relative speed of the two encountered nodes. Let the initial relative speed of the two nodes be v_1 and the required relative speed of the two nodes for completing the transfer be v_2 (Figure 6). If the relative distance traveled by the two nodes before going out of contact is d, then

$$v_2 = \frac{d}{M/R} \tag{4}$$

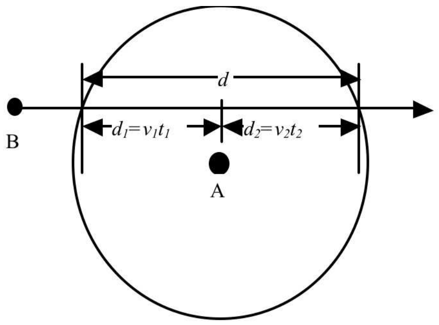


Figure 6: Relative speed in the sticky transfer framework

Here, M is the message size and R is the transmission rate. If they are going in the same direction the required relative speed can be achieved by either increasing the speed of the slower node or by reducing the speed of the faster moving node. But if they are going in opposite directions, this can be achieved by slowing down one or both nodes. Also, extension of the contact duration can be achieved by reducing the relative speed of the two nodes to zero, i.e., stopping them or by one node following the other at matching velocities. In that case, the time t_2 for which they should stop or follow will be

]

$$t_2 = \frac{M}{R} - \frac{d}{v_1} \tag{5}$$

3.7 Performance Limitations of Sticky Transfers

Nodes using the sticky transfer protocol may experience transitory changes to their natural movement in order to successfully engage in the sticky transfer of messages. An effect on the network performance, which depends on node mobility, is expected when sticky transfer is used, as mobility itself is known to increase the capacity of wireless ad hoc networks [39]. Generally in DTN, both the end-to-end delay and delivery ratio can be improved by reducing pair-wise inter-contact times. A shorter inter-contact time means getting a forwarding opportunity faster, which results in less delivery delay. Sticky transfer is achieved by reducing nodes movement speed which may increase the inter-contact time in the network. Thus, if sticky transfers are used persistently (e.g., all nodes engage in sticky transfers with a high probability) in highly loaded network conditions, the effectiveness of sticky transfers may be reduced due to lower mobility (i.e., nodes stopping for long periods to finish their exchanges). Lower mobility can also be detrimental from the point of view of the mission. For example, it may take longer for the robots to map an area, or it takes longer for the first responders to do their job). Finally, sticky transfers may lead to incessant buffer overflows when the network load is very high and the routing protocols disseminate many copies of each message.

4 Performance Evaluation

In this section, we provide performance evaluations of the sticky transfer framework using the Opportunistic Network (ONE) simulator, a simulation environment capable of routing messages between nodes using various DTN routing algorithms and sender/receiver types [27]. We compare the performance of DTN routing protocols with and without sticky transfers. In particular, we evaluate the performance gain resulting from the sticky transfer mechanism with the following DTN routing protocols: Epidemic [7]; Spray-and-Wait (SnW) [9]; and PRoPHET [8].

In our simulations, we assume that a node agrees to a sticky transfer request with a probability value of SP (stick probability), where $0 \leq SP \leq 1$. A value of SP = 0, 0.5, and, 1 indicates that a node does not agree, agrees to 50% of the requests, or always agrees to sticky transfer requests, respectively. Our study mainly observes the trade-offs of various SP values on parameters such as delivery reliability, traffic overhead, and memory utilization. The SP of a node does not change during the duration of a simulation run. Also, we assume that all nodes in the network will have the same stick probability. In future work, we will develop algorithms to enable nodes to determine the optimal stick probability when receiving a stick request based on the collected information (e.g., mobility speed, direction, and message sizes) and network conditions (e.g., network density).

We implement the "*Stop*" (STP) mode as the mode for sticky agreements from the possible sticky transfer modes described in section 3.3.1. The implementation choice is based on the observation that the "*Stop*" mode has the most inhibiting effect on the natural mobility of nodes, as not only the relative but also the actual speed of the two nodes agreeing to participate in the sticky transfer becomes 0 m/s. They must momentarily *stop* on their natural movement path(s) until sticky transfers are completed. Implementation of the "*Stop*" mode will provide a worst-case scenario analysis of the effect of our proposed sticky transfer framework. When using alternative modes such as "*Follow*" and "*Slow down*" (SLW) only the relative speeds of both nodes become 0 m/s. Nodes participating in the sticky transfer do not have to stop moving. An alteration (i.e. change in direction) on one or both of the nodes natural movement paths might be required. For example when participating in the sticky transfer with the mode set to "*Follow*", the node using the "*Follow me*" (FLW1) mode may not have to change the direction of its natural movement path, but only adjust its speed to accommodate and enable the other node to follow it. The node using the "*Follow you*" (FLW2) mode might have to change both direction and speed to enable the sticky transfer to be successful. In a realistic case, this may happen when one node's (node in "*Follow me*" mode) message has a higher delivery priority than that of other nodes in the network and delivering that package may be critical for the mission. Hence, other nodes will have incentive to engage in the "*Follow you*" mode. Nodes engaging in sticky transfers in the "*Slow down*" (SLW) mode will also have a greater advantage than nodes engaging in sticky transfers in the "*Stop*" mode, as they only need to move at a relative speed of 0 m/s and do not require a change in direction of their natural movement path. These observations leads us to believe that implementation of the "*Stop*" (STP) mode will be the most effective for observing the lower-bound of sticky transfer performance. The implement and comparison of other sticky modes are part of our future work.

In our implementation of the *sticky transfer protocol* (see section 3.5), we ignore the time for calculating the stick mode agreements at nodes during the initiation of the protocol as this overhead is negligible compared to the overall message transfer time, which depends on the wireless transmission capabilities of nodes.

4.1 Performance Metrics

As discussed in section 2, replication-based DTN routing protocols such as Epidemic [7] and PRoPHET [8] forward multiple copies of a message to increases the probability of message delivery to the destination. Replication-based routing is also know as multi-copy forwarding, or flooding. In this process, a source node generates multiple copies of a message m in its buffers and forwards them to nodes/contacts that have not yet 'seen' (i.e. received) m. The specific number of copies depend on each individual routing protocol strategy. Flooding-based routing guarantees a higher delivery probability compared to other DTN routing strategies at the expense of higher energy consumption and lower scalability. We design/select performance metrics with multi-copy message forwarding in mind and take into account the multiple copies of messages in our analysis.

We use the following performance metrics to evaluate the effectiveness of the proposed framework.

1. *Average message delivery ratio*. The message delivery ratio of a destination d is the ratio of the number of unique messages for d generated by its source(s) and successfully received by d to the total number of unique messages created. The *average* message delivery ratio is the average of the delivery ratios of all the destinations in the network.

2. *Average end-to-end delay*. The end-to-end delay of every message successfully received at every destination is recorded. Once the copy of a particular message is received at the final destination, subsequent copies are disregarded. The average over all the messages received is then computed.

3. *Average buffer time*. The buffer time for a message *m* is the interval between the time the message is saved in the buffer at a node to the time it is removed from the buffer of that node. Essentially, it is the time the message was queued in the node's buffer. The *average* buffer time is the average taken over all messages stored at every node in the network. [1] Average buffer time significantly impacts performance as: (a) it is an indicator of how fast messages are being forwarded at nodes, and thus are propagating through the network, as longer buffer time leads to longer end-to-end delays; and (b) the longer messages are stored in buffers the more buffer overflows will occur, leading to higher loss rates.

4. *Average overhead ratio*. We consider direct delivery as a 0-hop delivery. Direct delivery is when the source itself meets the destination and forwarding to an intermediate node is not necessary. When there is one intermediate node (relay) involved in the end-to-end delivery, we consider it a 1-hop delivery. Direct or 0-hop delivery is considered the ideal case. Any additional hop after a 0-hop delivery is considered as overhead. Hence, if the message required 2 relays to be delivered, the overhead is 2. The overhead ratio is defined as follows:

$$\frac{H - h_d}{h_d} \tag{6}$$

   where H is the number of hops a message *m* and all copies of *m* traverse in the network before a copy is successfully delivered to the intended recipient, and h_d is the number of hops the delivered message (or its copy) had traversed before reaching the final destination. It represents the quantity of excessive relays required to deliver the message and quantifies the bandwidth and energy consumption of the routing protocol. The *average* overhead ratio is the total number of transmissions required over all messages successfully delivered to their destinations.

5. *Average number of disrupted transmissions*. A successful message transfer from one hop (sender) to another (receiver) entails that a message is transmitted in its entirety (all bits) from the senders queue and all bits of the message are received at the receivers queue. Some messages may be received at the destination with bit errors, which are not counted as disrupted messages. Unsuccessful/disrupted transmissions in our DTN network environment can be caused by nodes moving out of the transmission range of each other in the middle of message transfers, or by packet collisions. Obviously, these cases can not be avoided. The number of disrupted transmissions is an implicit measure of the wasted bandwidth in the network. The *average* number of disrupted message transmissions is the sum of the number disrupted transmissions observed by each sender over the total number of nodes in the network.

6. *Average number of contact opportunities per hour*. We define a 'contact opportunity' as two nodes coming within the wireless transmission range of each other, presenting an opportunity to transfer messages. The *average* number of contact opportunities per hour is the total number of contact opportunities for the duration of a simulation run divided

[1]Note that messages may be removed from nodes either because the message was relayed to the next encountered node or it was dropped from the current node due to buffer overflow. Hence, the average buffer time can be higher than the average end-to-end delay, as end-to-end delay considers successfully delivered messages only.

by the total simulated time (in hours). To be counted as a contact opportunity, nodes do not necessarily have to use that contact opportunity to transfer messages. The number of contact opportunities in the network is important, particularity for DTN's where the routing strategy is multi-copy forwarding because more contact opportunities increase the delivery probability by providing the opportunity to spread more message copies in the network.

7. *Average stick time*. We define 'stick time' as the time after two nodes come within the wireless communication range of each other, and use the contact opportunity to engage in message transfers using one of the sticky transfer modes. This is measured at each node by the δ-value introduced in section 3.2. The *average* stick time is the sum of the stick time of pair-wise contacts during each simulation run over the total number of pair-wise contact opportunities used for sticky transfers in that simulation run. In our results, we represent the stick time as a percentage of the total simulation time for one run. For example, lets assume the stick time was measured as 3,000 simulated seconds. The stick time in this case would be represented as 10% given the total time for each experiment run was 30,000 simulated seconds.

Table 2: Default simulation parameters

Parameter	**Value**
Network size (map area)	4500m x 3400m
Number of nodes	5 to 30
Points of interest (POI's)	11
Movement model	Shortest Path Map Based Movement [28]
Pause time at POI's	random between 1 to 50 seconds
Movement speed	Total 20 nodes 4 nodes (pedestrians): 1.25 - 1.53 m/s 6 nodes (cyclist): 2.5 - 8.33 m/s 10 nodes (cars): 20 - 25 m/s
Traffic model at sources	0.2 msgs/sec
Message generation rate	5 seconds on average
Simulation time	30,000 seconds of simulated time
Message generation start time	at 1,000 simulated seconds
Message generation stop time	at 21,000 simulated seconds
Message size	100 Kbytes ; 20 Mbytes
Time-to-live (TTL) of messages	∞
Buffer size of nodes	1GB
Transmission range	100 meters
Transmission rate	11 Mbps (modeling IEEE 802.11b) 54 Mbps (modeling IEEE 802.11g)
DTN routing protocols	Epidemic [7] PRoPHET [8] Spray-and-Wait [9]
Number of runs per data point	10

4.2 Simulation Parameters

We choose a map of the Helsinki downtown area of size 4500 m x 3400 m as our network area. Mobile nodes move according to the Shortest Path Map Based Movement model [33] [34]. Nodes randomly choose next locations from eleven disconnected points of interest (POIs). POIs are places on the map used to model office buildings, tourist attractions, shops, restaurants and parks. Nodes move to selected POIs with random speeds between 1 m/s to 25 m/s and pause for a period with a value randomly distributed between 1 second and 50 seconds before selecting the next POI. We use 5 to 30 mobile nodes in the network. In the graphs, we refer to nodes as hosts or mobile hosts (MH).

We consider a uniform traffic model where all mobile nodes have data to send to destinations selected randomly. Each simulation runs for 30,000 seconds in simulated time. Messages are generated on average every 5 seconds after a warm-up period of 1,000 seconds. Sources stop generating messages after 21,000 seconds to allow for all message copies already generated in the network a chance to propagate and be distributed fairly compared to messages previously generated. The average message generation rate in the network is 0.2 messages/second, as messages are generated randomly at the i^{th} second of every 5 second generation interval.

Message sizes vary between 0.1 Mbytes to 30 Mbytes with infinite lifetime (TTL) values. These sizes cover different types of messages such as text, photos, or short videos. The sticky transfer mechanism is expected to show its effectiveness especially for larger message sizes, which are desirable in DTNs. Given the same amount of data to be sent, larger messages mean less data units to be transmitted, and thus lower overhead incurred by data unit headers at different layers of the TCP/IP protocol stack. The buffer size of each mobile node is 1 GB. The transmission range of nodes are 100 meters with transmission rates of 11 Mbps (modeling IEEE 802.11b) and 54 Mbps (modeling IEEE 802.11g).

Table 2 summarizes the simulation settings; the default values are used unless otherwise stated. Under these settings, we will evaluate the performance of the sticky transfer mechanism with the following routing protocols: Epidemic [7]; Spray-and-Wait (SnW) [9]; and PRoPHET [8]. In our experiments we implement SnW in binary mode with K=4. The K value for SnW was chosen after reviewing existing literature on the protocol and observing standard values from simulations performed in the DTN research community.

Each simulation was run 10 times. Each data point plotted on graphs is an average value over 10 runs.

We conducted a total of five sets of experiments to provide in-depth insight into the performance of the sticky transfer mechanism. In each experiment we varied the stick probability (SP) and one of the following parameters:

- *Transmission rate at the physical layer* [section 4.3.1]: The first set of experiments was performed by varying the transmission rates of nodes on both small and large message sizes. We measured several performance metrics for this experiment set mentioned in Section 4.1. The results depict the performance of the sticky transfer mechanism under networks that support different transmission rates at the physical layer.
- *Node density* [section 4.3.2]: The second set of experiments was performed by varying the number of nodes in the network area. We measured two performance metrics for this

Table 3: Parameter settings for experiments

Function of	Parameter	Value
Different Transmission Rates	Transmission rate	11 Mbps ; 54 Mbps
	Message size	100 KB ; 20 MB
	No. of nodes	20
	Movement speed	default (Table 2)
	TTL	∞
Different Node Densities	Transmission rate	11 Mbps
	Message size	20 MB
	No. of nodes	5 ; 30
	Movement speed	5 nodes: 1 pedestrian, 1 cyclist, 3 cars (Table 2) 30 nodes: 4 pedestrians, 6 cyclist, 20 cars (Table 2)
	TTL	∞
Different Node Mobility Speeds	Transmission rate	11 Mbps
	Message size	20 MB
	No. of nodes	20
	Movement speed	5 m/s ; 20 m/s
	TTL	∞
Different TTL Values	Transmission rate	11 Mbps
	Message size	20 MB
	No. of nodes	20
	Movement speed	default (Table 2)
	TTL	1 simulated hour 5 simulated hours
Different Message Sizes	Transmission rate	54 Mbps
	Message size	5 MB ; 30 MB
	No. of nodes	20
	Movement speed	default (Table 2)
	TTL	∞
Supporting Message Sizes	Transmission rate	54 Mbps
	Message size	100 KB, 5 MB, 20 MB, 30 MB
	No. of nodes	20
	Movement speed	default (Table 2)
	TTL	∞
	Routing protocol	Spray-and-Wait

experiment set: (i) the average message delivery ratio, and (ii) the average end-to-end delay. In this experiment set, we show the performance of the sticky transfer mechanism with varying node densities at 5 mobile hosts(MH) to represent a low density network and 30 mobile hosts(MH) to represent a higher density. The transmission rate of nodes is 11Mbps and the message size is 20MB.

- *Node speed* [section 4.3.3]: The third set of experiments was performed by varying the speed of nodes in the network. We measured two performance metrics for this experiment set: (i) the average message delivery ratio, and (ii) the average end-to-end delay. In this experiment set, we present results for 5m/s and 20m/s node speeds with varying stick probabilities for 20MB message sizes with nodes transmitting messages at 11Mbps. In order to observe the performance improvement with sticky transfers for different node movement speeds, we set the same speed for all 20 nodes per experiment, as opposed to the default simulation setting in table 2.

- *Time-to-live (TTL) value* [section 4.3.4]: The fourth set of experiments was performed by varying the TTL value of messages in the network. We measured two performance metrics for this experiment set: (i) the average message delivery ratio, and (ii) the average end-to-end delay. We run experiments for 20MB message sizes at 11Mbps transmission rates on two time-to-live (TTL) values for messages: 1 hour and 5 hours, respectively.

- *Message size* [section 4.3.5]: The fifth set of experiments was performed by varying the size of the messages generated in the network. We measured two performance metrics for this experiment set: (i) the average message delivery ratio, and (ii) the average end-to-end delay. We show the performance improvement using sticky transfers varying the message size at 5MB and 30MB message sizes. In each experiment 20 nodes are in the network transmitting messages at 54Mbps. We choose a higher transmission rate for this set of experiments as it involves larger message sizes. By having a faster transmission rate we could reduce the simulated time required to run the experiments, yet still observe the effect of the sticky transfer protocol.

We summarize the set of parameters used in the five experiments in Table 3. Note that the sixth column in the table, 'supporting message sizes', is an accumulated analysis from previous experiments and did not require new simulation runs.

4.3 Simulation Results and Analysis

We present simulation results in Fig 7 to Fig 18.

When analyzing simulation results we note the following:

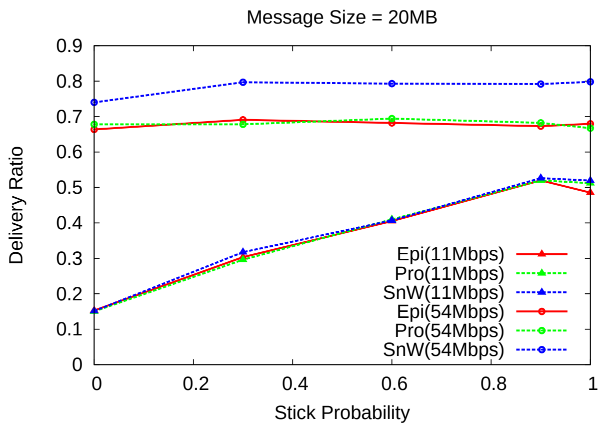
- due to the nature of the protocol design (i.e., message dissemination strategy), the Spray-and-Wait (SnW) protocol incurs lower network traffic than the Epidemic and PRoPHET protocols, as the number of copies of a message that can exist in the network is limited by the K parameter in SnW. In PRoPHET and Epidemic, in worst-case scenarios, it is theoretically possible for each node to be carrying a copy of every message generated in the network, leading the number of copies of a message m to be equal to the number of nodes N in the network.

- traffic in the network impacted by the message forwarding strategy of the routing protocol. For example, the Epidemic protocol incurs higher loads than the PRoPHET or Spray-and-Wait protocol as it is a flooding protocol and copies of messages are forwarded to every encountered node.

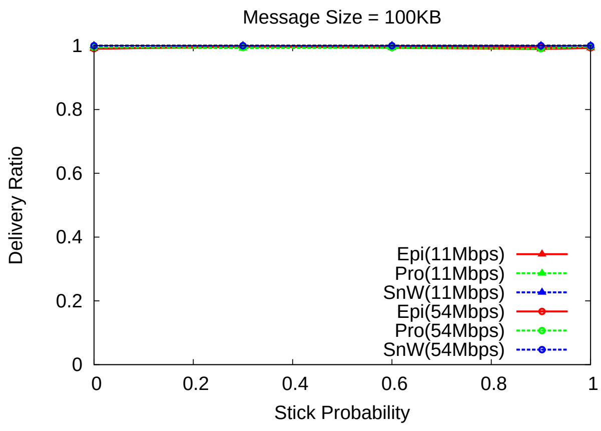
4.3.1 Different Transmission Rates

In the first set of results we measure the performance of the sticky transfer protocol using two different transmission rates at the physical layer: 11 Mbps and 54 Mbps. The results presented in this section show the effect of using the sticky transfer protocol for delivering messages in the network considering several performance metrics as defined in section 4.1 and presented as follows:

4.3.1.1 Average Message Delivery Ratio



(a) Message size = 20MB.



(b) Message size = 100KB.

Figure 7: Average Delivery Ratio as a function of Stick Probability.

Fig. 7 presents the delivery ratio as a function of the stick probability (SP). The graph in Fig. 7(a) depicts the performance of the average message delivery ratio with a message size of 20 MBytes and messages being transmitted at 11 Mbps and 54 Mbps transmission rates, respectively.

At 54 Mbps, a high delivery ratio was achieved even without sticky transfers due to the shorter transfer time of messages. Gradually increasing SP values increased the delivery ratio of SnW up to 6% but did not significantly increase the delivery ratio of flooding based protocols (Epidemic, PRoPHET) because of buffer overflows.

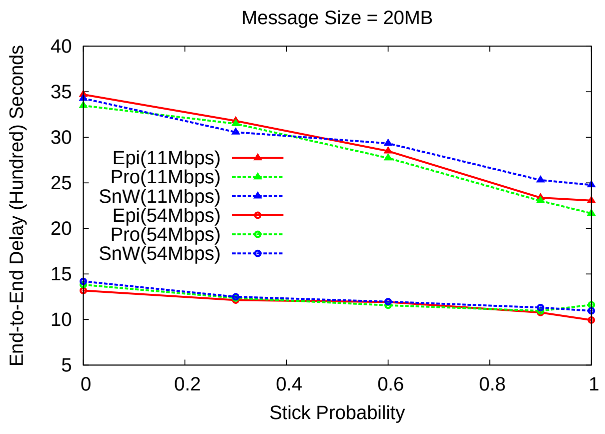
At 11 Mbps, the delivery ratio increased up to 38% with increasing SP values as more messages could be exchanged upon encounters due to mobile hosts' increasing willingness to stick for transfers.

At the lower transmission rate (11 Mbps) when buffers were not a limitation, the maximum delivery ratio was achieved at SP=1.0 in SnW. However, in Epidemic and PRoPHET, the maximum delivery ratio was achieved at a SP value around 0.9. In these two algorithms, the delivery ratio decreased as SP increased from 0.9 to 1 because nodes always agreed to stick for message transfers (i.e., the stop mode), which reduced the node movement. The above difference is due to the fact that SnW forwards fewer copies of a message than the Epidemic and PRoPHET protocols.

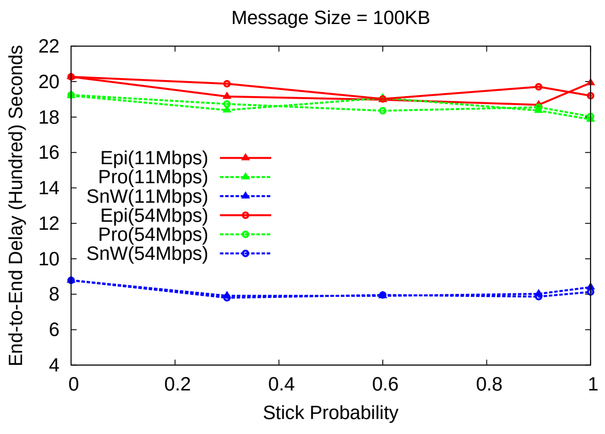
The graph in Fig. 7(b) depicts the performance of the average message delivery ratio for smaller message sizes (e.g., 100KB). On average, a 100% delivery ratio was achieved in all three protocols even without the sticky transfer mechanism (i.e. at SP=0) at both transmission rates due to the smaller size of messages and sufficient contact opportunities (see Fig. 12(b)).

In summary, the benefit of sticky transfers is clearly evident in scenarios with larger message sizes and lower transmission rates, which indicates that DTN networks with these characteristics can benefit from sticky transfers.

4.3.1.2 Average End-to-End Delay



(a) Message size = 20MB.

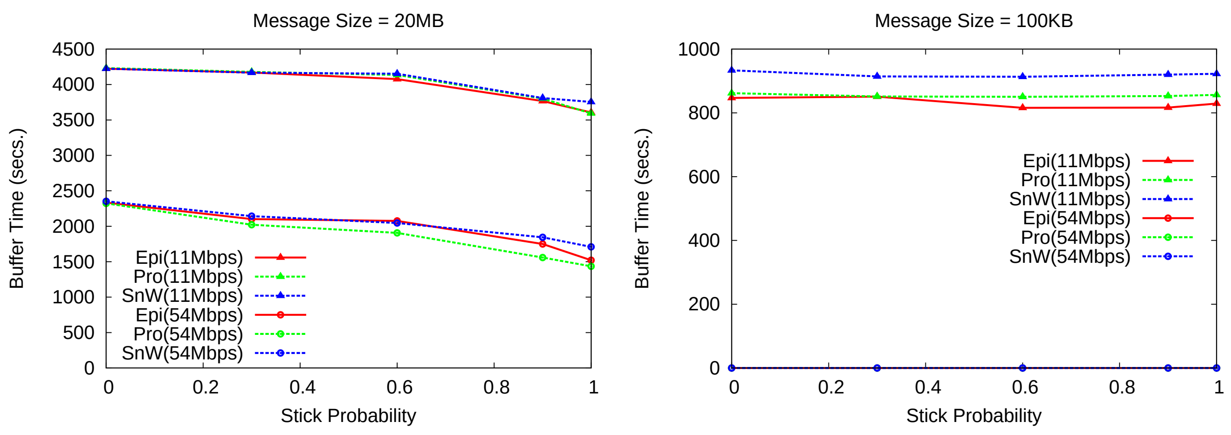


(b) Message size = 100KB.

Figure 8: Average End-to-End Delay as a function of Stick Probability.

Fig. 8 presents the average end-to-end delay of messages as a function of the stick probability (SP). In general, the end-to-end delay decreased gradually with increasing SP. Fig. 8(a) shows the end-to-end delay with increasing stick probabilities for 20 MB message sizes. The reduction of the delay using sticky transfers was significant especially at the lower transfer rate of 11 Mbps. A maximum of up to 25% and up to 36% reduction in the end-to-end delay was observed at 54 Mbps and 11 Mbps, respectively. As expected, the delays for 54 Mbps were much lower than those for 11Mbps due to the higher transmission rate.

Fig. 8(b) shows the end-to-end delay with increasing stick probabilities for 100 KB message sizes. The improvement on the delay performance was about 10% overall because in most cases sticky transfers was not used due to the nature of contact duration's available. However, this does show that increasing the stick probability despite the delivery ratio being almost 100% (see Fig. 7(b)) for all stick probabilities did improve the end-to-end delay of those delivered messages. Note that the delay is minimum around SP=0.9, instead of at SP=1.0 as would be expected, for flooding based protocols (e.g., Epidemic) because persistent stick restricts movement in high loads.



(a) Message size = 20MB.
(b) Message size = 100KB.

Figure 9: Average Buffer Time as a function of Stick Probability.

4.3.1.3 Average Buffer Time

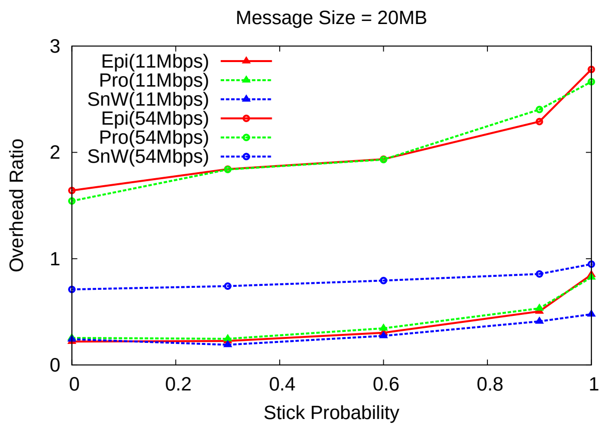
Fig. 9 shows the average amount of time messages spent in node buffers before being forwarded to the next node. In Fig. 9(a), as the SP was gradually increased, the average buffer time of messages decreased (by 38%) compared to the no-stick transfer case (SP=0). With increasing SP values, messages spent less time in buffers and message copies were forwarded faster in the network, thus reducing delay significantly. The average time spent in buffers is lower for 54 Mbps than 11 Mbps due to faster transmission rates. In both cases at higher SP values (0.9 and 1.0), SnW has a higher average buffer time as it goes into the direct delivery (i.e., waiting) mode once all its copies are forwarded; thus messages remained longer in node buffers just prior to the final delivery. Also, for Epidemic and PRoPHET, we can expect that more message drops occurred.

A similar trend was observed for the three routing protocols at a transmission rate of 11 Mbps for smaller messages sizes of 100KB as shown in Fig. 9(b). However, we observe in this graph that at the faster transmission rate of 54 Mbps, the buffer occupancy time is negligible across all SP values. This is of course due to the smaller messages along with faster transmission speeds and nodes having sufficient contact durations upon encounters. Also comparing Fig. 9(a) and Fig. 9(b), we can see that the average time messages spent in buffers was much less for messages sizes of 100KB compared to the larger messages of 20MB overall.

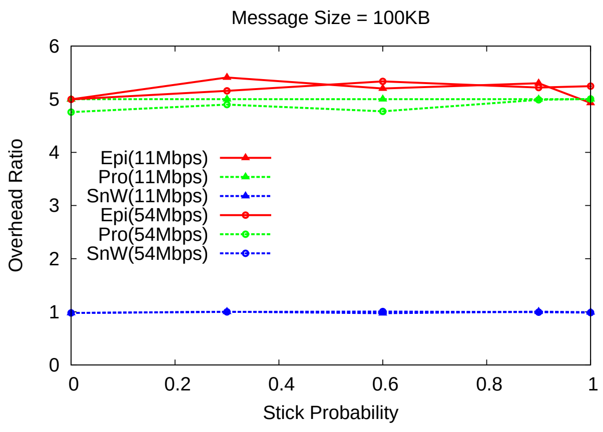
4.3.1.4 Average Overhead Ratio

Fig. 10 shows the average overhead ratio which signifies the number of ‘extra’ (additional) hops a messages requires for end-to-end delivery. As mentioned in section 4.1, the overhead ratio quantifies the bandwidth and energy consumption of the routing protocol.

In Fig. 10(a), for Epidemic and PRoPHET, each time nodes encounter each other, copies of messages are forwarded in the network until any one of the copies reaches the final destination. For SnW once all 4 copies are spread, nodes go into direct delivery mode. So, there is naturally less overhead for SnW and the number of extra hops a copy is spread before being finally delivered is less than 1 (for both 54Mbps and 11Mbps).



(a) Message size = 20MB.



(b) Message size = 100KB.

Figure 10: Average Overhead Ratio as a function of Stick Probability.

At gradually higher SP values and 54Mbps transfer rates, more copies are spread faster due to nodes always successfully forwarding copies to other encounters because of sticky agreements. This fulfills the strategy of flooding protocols at a cost of overhead. At SP=1.0, this overhead can cause ‘thrashing’. Thrashing is a condition where due to buffer limitations messages are dropped incessantly to make room for new ones. This can lead to reduced network performance.

Fig. 10(b) shows the average overhead ratio for 100KB sized messages. Thrashing was not noticed in this scenario as buffers were sufficient to accommodate messages. Hence, the overhead ratio was higher as when messages were forwarded hop-by-hop, they could be accommodated and not dropped from the buffers. SnW incurred the least overhead due to the design of the protocol as there is the initial ‘spray’ phase which accounts for 1 hop, then the ‘wait’ phase commences and messages wait in buffers until encountering the final destination (i.e. intended recipient), aligning with the results of buffer occupancy being higher for SnW (Fig. 9). Epidemic and PRoPHET’s overhead ratio was much higher on average, as all copies forwarded are accounted for in the overhead ratio.

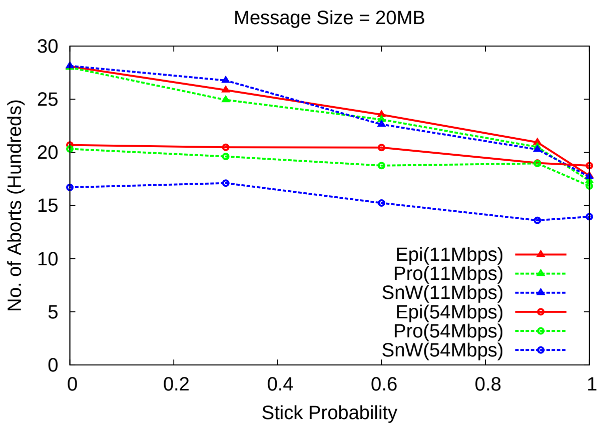
4.3.1.5 Average Number of Disrupted Message Transmissions

Fig. 11 shows the average number of message aborts with varying stick probabilities. We first analyze Fig. 11(a) with 20MB message sizes. In the SnW protocol, a limited number of copies of the message are created and hence the number of transfer requests (and aborts) is smaller than those from the other two protocols.

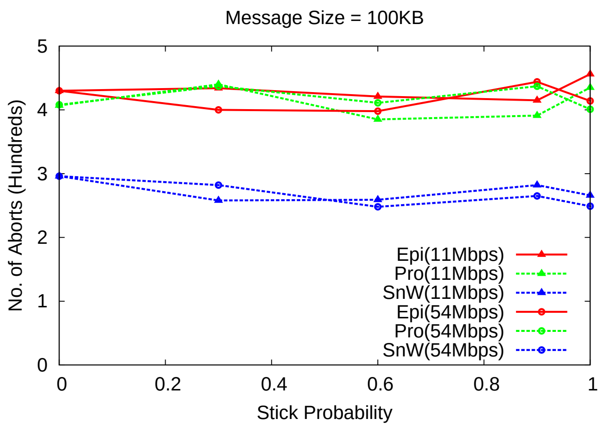
For smaller message sizes of 100KB sticky transfer was rarely used and thus the reductions in the number of aborts were small (Fig. 11(b)). However, for 20MB message sizes the number of aborts were large without sticky transfers. An improvement of (decrease in the number of aborts) 9% to 19% and 36% to 38% were observed at 54Mbps and 11Mbps transmission rates respectively in different protocols when the $SP>0$.

4.3.1.6 Average Number of Contact Opportunities per Hour

Fig. 12 shows the number of contacts per hour as a function of stick probability.

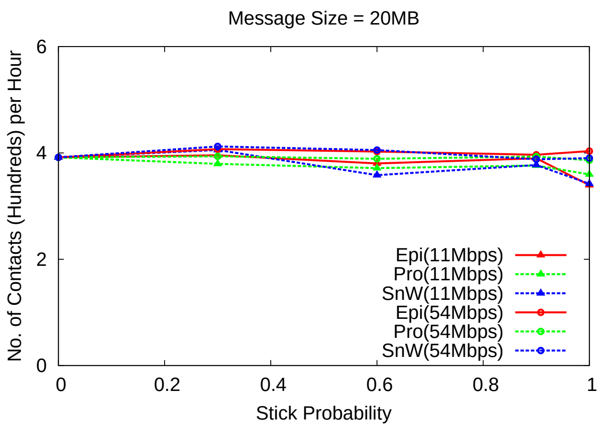


(a) Message size = 20MB.

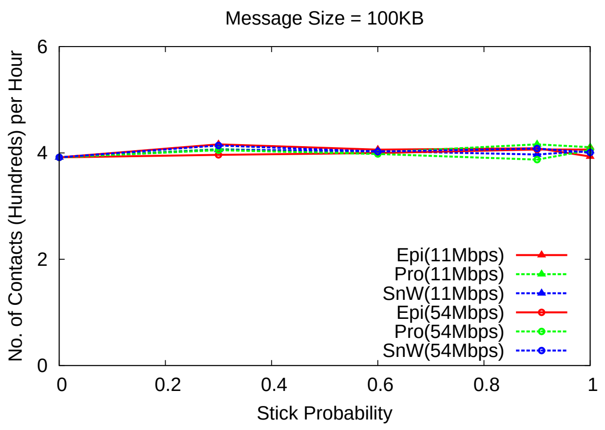


(b) Message size = 100KB.

Figure 11: Average Number of Disrupted Message Transmissions as a function of Stick Probability.



(a) Message size = 20MB.



(b) Message size = 100KB.

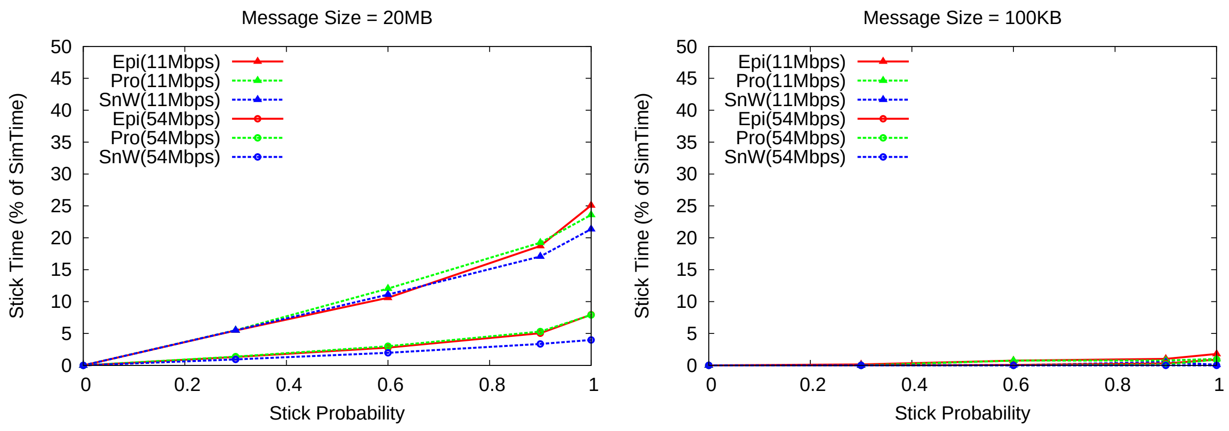
Figure 12: Average Number of Contact Opportunities per Hour as a function of Stick Probability.

We observe for 100KB-sized message (Fig. 12(b)), the number of contacts remained almost linear across all stick probabilities due to the fact that as the message sizes were small, sticky transfers were not required for long durations. We expect that the average stick time in these cases was negligible, as messages were being transmitted almost instantly upon mutual encounters.

For 20MB-sized messages (Fig. 12(a)), the stick time is expected to be longer. Hence, as the SP is increased, the number of contacts per hour decrease, as pair-wise contacts stick longer to successfully transfer message and thus meet fewer new contact within the same amount of time (i.e., in comparison to smaller message sizes). This is even more observable in the case of the flooding-based Epidemic routing protocol, as gradually every pair-wise encounter will lead to a sticky transfer.

In Fig. 12(a), in the case of flooding based protocols the number of contacts decreased by 13% when SP=1.0. This is because the movement of the nodes were restricted during sticky transfers and thus the number of contacts decreased. However, compared to the benefit of sticky transfers (via improving delivery performance, reducing message transfer aborts etc.), this negative impact on node mobility is insignificant. Also, instead of the "Stop" mode if other modes, like "Follow" or "Slow down" are used, then the above discussed impeding effect of sticky transfers on the network performance will be very small.

4.3.1.7 Average Stick Time



(a) Message size = 20MB.   (b) Message size = 100KB.

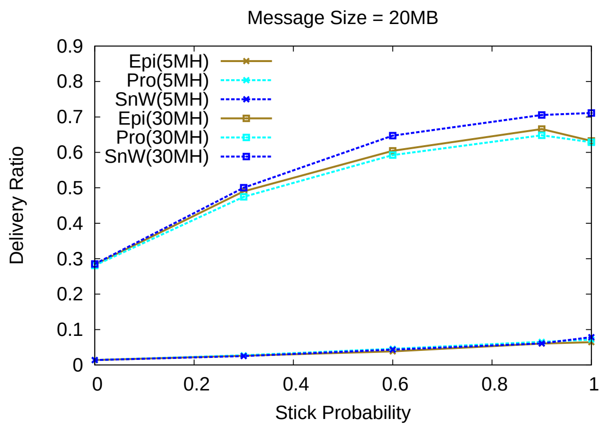
Figure 13: Average Stick Time as a function of Stick Probability.

Fig. 13 illustrates a measurement of the total time nodes stick in the network as a percentage of the total simulation time. Since we implement sticky transfers through the "STOP" mode, the stick time represents the total amount of time that a node stops (is stationary) and thus is the cumulative delay on its journey to all of its destinations. We notice the same trend for this parameter for both larger and smaller sized messages, shown in Fig. 13(a) and Fig. 13(b), respectively. The stick time was non-existent (zero value) for all routing protocols at SP=0, since the probability of sticky transfers is zero.

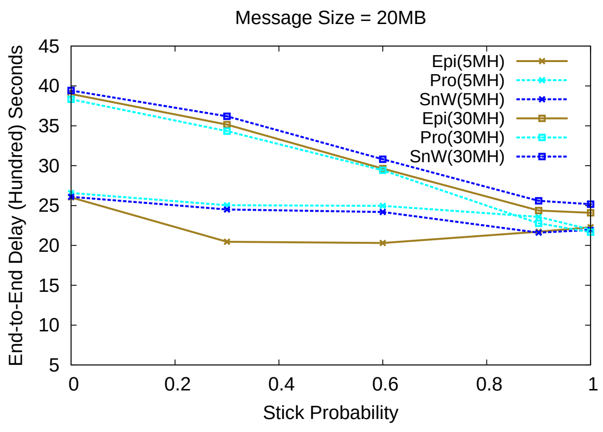
In Fig. 13(b), since the message size was smaller, nodes needed not engage much in sticky transfers and the stick time was less than 2% across all protocols. In general, for both graphs, the stick time was higher for flooding protocols (up to 25%), particularly at SP=1, since nodes remained in contact upon mutual encounters until all messages were forwarded. For both 11Mbps and 54Mbps, SnW incurs less cumulative stick time in general as nodes have message copies to forward upon encounters. Too much stick time can reduce node mobility in the DTN, which can lead to fewer contact opportunities over time.

4.3.2 Different Node Densities

Fig. 14 shows the performance improvement achieved with sticky transfers at different node densities for large message sizes (MS=20MB). The delivery ratio is shown in Fig. 14(a). At the low node density (5MH), the performance increased by 6%. At the higher node density (30MH), the performance increased by 43%. The performance improvement at higher densities compared to lower node densities comes resulted from more contact opportunities due to more nodes in the same area. The performance at higher node densities improved significantly from SP=0 until up to SP=0.9, then fell slightly at SP=1.0 due to the same reasons as for Fig. 7(a). The end-to-end delay (Fig. 14(b)) was reduced by as much as 22% at low node densities and



(a) Average Delivery Ratio as a function of SP.



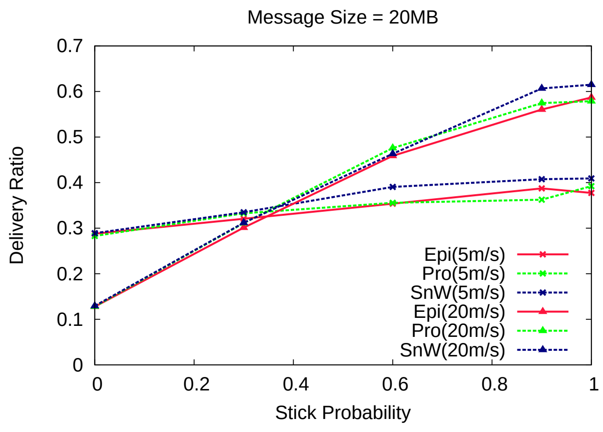
(b) Average End-to-End Delay as a function of SP.

Figure 14: Performance of sticky transfers at different node densities.

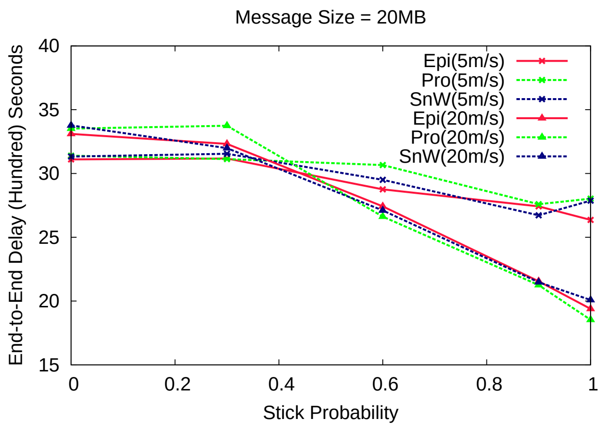
by as much as 43% at higher node densities as the SP increased.

In summary, using sticky transfers provides higher delivery ratio and lower end-to-end delay, especially in dense networks.

4.3.3 Different Node Mobility Speeds



(a) Average Delivery Ratio as a function of SP.



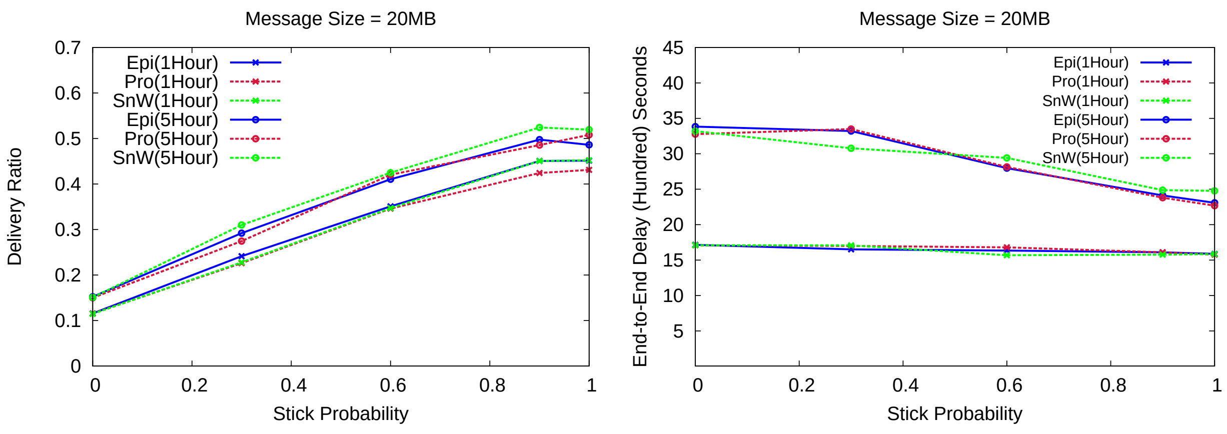
(b) Average End-to-End Delay as a function of SP.

Figure 15: Performance of sticky transfers at different node speeds.

Delivery ratio improved by 12% (for node speeds of 5m/s) and by 49% (for node speeds of 20m/s) across different DTN protocols (Fig. 15(a)). Also end-to-end delay decreased by 15% and 45% respectively for 5m/s and 20m/s speeds (Fig. 15(b)). When sticky transfers were not used in the network (SP=0), delivery performance at lower speeds (5m/s) were much better than at higher speeds (20m/s), because at faster node speeds the natural contact duration of the nodes were not enough to successfully complete the message transfers of larger message sizes (20MB). However, as we increased the SP, the improvement in the delivery performance increased significantly at higher node speeds. Nodes were able to extend the contact duration to the required duration by using the sticky transfer protocol. Thus with the combination of meeting encounters faster (faster movement speed) and being able to transfer more messages (longer contact durations), the delivery and latency performance at 20m/s speeds surpassed that of the lower speed scenario. Therefore, sticky transfers give higher benefits in higher mobility conditions where message transfer disruptions are more likely.

4.3.4 Different Time-to-Live(TTL) Values

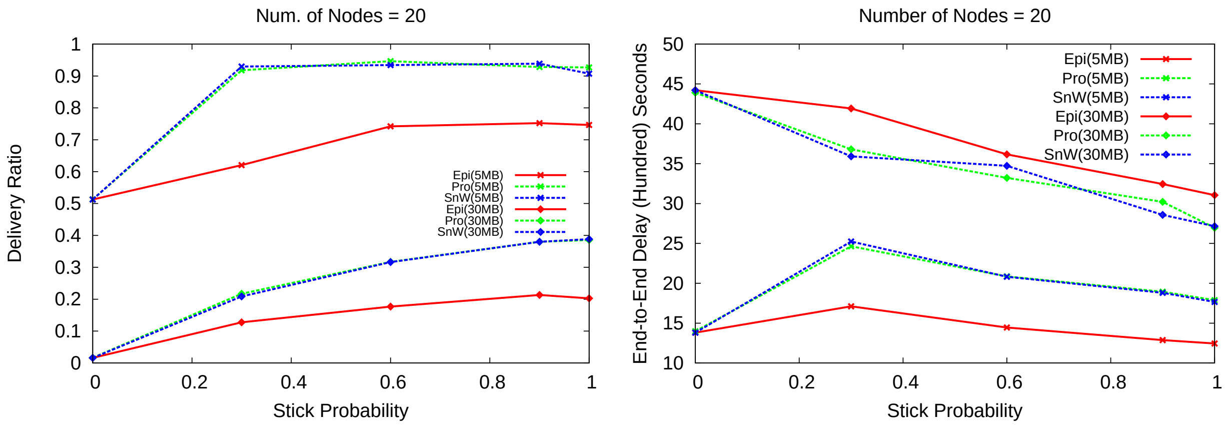
As we know from our literature review, DTN technology can be applied in the case of a wide range of applications, starting from underwater to outer space. Messages may have an expiration timeout (TTL) set by applications. Generally there is a mission deadline, to be met by nodes/participants in the network. Messages may have TTL values set to meet these deadline requirement. The messages may be useless for the mission after this and therefore, once the TTL expires, are dropped. For space applications the TTL of messages may be very large, maybe upto several months. For military applications the TTL could be in the range of days or hours. For emergency situations the TTL could be in the range of minutes. To keep the simulation time feasible and to shed insight into how the sticky transfer mechanism can assist applications with different TTL values, the results in In Fig. 16 show the average delivery ratio and average end-to-end delay with TTL values for 1 and 5 simulation hours, respectively.



(a) Average Delivery Ratio as a function of SP. (b) Average End-to-End Delay as a function of SP.

Figure 16: Performance of sticky transfers at different message time-to-live (TTL) values.

Delivery ratio improved by 34% (for TTL of 1 hour) and by 37% (for TTL of 5 hours) across different DTN protocols (Fig. 16(a)). Also end-to-end delay decreased by 8% and 32% respectively for 1 hour and 5 hour TTL values (Fig. 16(b)). We notice in Fig. 16(b), the end-to-end delay was almost constant across all SP values for messages having TTL value of 1 hour, meaning using sticky transfers do not have much impact on the delay performance (i.e. were not able to reduce the end-to-end delay) due to the already shortened lifetime of the messages. However, the delay was significantly reduced at larger TTL values, meaning when applications have a large message (20MB), and TTL values ranging in the hours - our sticky transfer mechanism can help to meet application critical deadlines by delivering the messages faster in the network. Furthermore, the delivery ratio for 20MB messages was greatly improved (by up to 37%) by using sticky transfers. Therefore, the sticky transfer mechanism can help with guaranteed delivery for time critical applications.



(a) Average Delivery Ratio as a function of SP.

(b) Average End-to-End Delay as a function of SP.

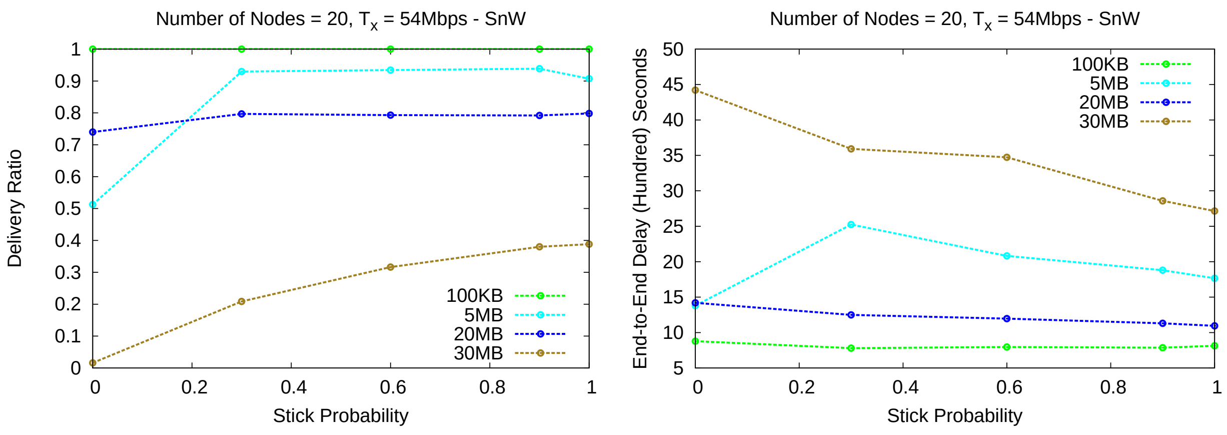
Figure 17: Performance of sticky transfers at different message sizes.

4.3.5 Different Message Sizes

In (Fig. 17), the delivery ratio increased by 24% and 20% in Epidemic protocol while it increased by 43% and 37% in the other protocols for 5MB and 30MB messages respectively (Fig. 17(a)). End-to-end delay (Fig. 17(b)) decreased gradually from 30% to 39% with increasing SP for 30MB-sized messages. This shows the benefit of using higher SP values when message sizes are large. The overall end-to-end delay for 5MB message sizes is much lower compared to 30MB sizes in general. However, an increase in delay was observed around SP=0.3 for all protocols when transmitting 5MB-sized messages (see Fig. 17(b)). We observe in the corresponding delivery ratio graph (Fig. 17(a)) that at this point the delivery ratio increased as well from a ratio of 0.5 to 0.9. At SP=0 sticky transfers were not being used and we anticipate only a few packets were being delivered, and the delivery was mostly within a single hop. However, at SP=0.3 as more packets were delivered and since $SP \neq 0$, multiple hops were required contributing to the higher end-to-end delay. The delay from this point (SP=0.3) gradually dwindled and the increase became smaller towards SP=1.0, as expected with increasing stick probability. Thus, sticky transfer improves delivery ratio for all message sizes, but may increase end-to-end delay in some cases if more packets have to travel longer distances.

4.3.6 Supporting Message Sizes: 100kB, 5MB, 20MB, 30MB

DTN Protocol Data Units (PDU) are called bundles (a.k.a 'messages'). A key DTN functionality is that each bundle is kept in memory in its entirety, and is deleted upon receipt of an acknowledgment for its successful delivery to the next node on the path to the destination. However, the DTN Bundle Protocol specification [29] does not limit bundle size or specify content of bundles. A bundle may be of any size depending on the application. As discussed in [30], the bundle protocol uses bundle as a protocol data unit which may be of various lengths. Ivancic et al. have used bundle sizes of 160 MB to transfer images from orbit to a ground station [31]. On the other hand in [32], Scott Burleigh suggests use of small bundles, less than 64KB long, to enable partial data delivery at application-appropriate granularity. It is obvious that the scientific community has not yet defined a commonly accepted formula of encapsulating application data into specifically sized bundles.



(a) Average Delivery Ratio as a function of SP. (b) Average End-to-End Delay as a function of SP.

Figure 18: Performance of sticky transfers at different message sizes for Spray-and-Wait

This lack of a fixed bundle size for DTNs suggests that a mechanism to improve the performance of the DTN across various message sizes would be beneficial and would address an area which requires attention. We believe that the sticky transfer protocol is able to achieve this goal, which we show by Fig. 18. These graphs are composed by integrating results from 4.3.5, 4.3.1.1 and 4.3.1.2. It shows the performance improvement achieved by the sticky transfer protocol across a range of message sizes: 100kB, 5MB, 20MB, and 30MB. The number of nodes is 20 and the transmission rate (T_x) is 54 Mbps. We perform a brief analysis of the graphs as the results have already been covered in previous sections. The graph shows results for the SnW routing protocol. The trend is similar for Epidemic and PRoPHET. Fig. 18(a) shows the average delivery ratio across message sizes. The delivery ratio was improved in each case by using sticky transfers. Fig. 18(b) shows the average end-to-end delay across the different message sizes. Using the sticky transfer protocol, end-to-end delay was reduced across all message sizes (with the exception of 5MB, which has been explained in section 4.3.5).

4.4 Summary of Simulation Results

Based on our results and analysis of the sticky transfer mechanism, we can conclude that using the sticky transfer protocol can improve several performance parameters such as the delivery ratio, end-to-end delay, overhead ratio, and so on. It can be well adopted for challenged scenarios where larger messages sizes need to be delivered with application deadline constraints. Performance of the DTN improved (upto 43%) at higher node densities (30 nodes) with sticky transfers. Message transfer disruptions are more likely in increased mobility conditions, and by using sticky transfers the performance of the DTN improved (up to 49%) at higher movement speeds (20 m/s). Furthermore, sticky transfers improve the delivery ratio at all message sizes (5MB-30MB), but may increase end-to-end delay for larger message sizes (30MB) at certain stick probabilities depending on the routing protocol. This is due to the combined effects of restricting mobility and reducing message aborts simultaneously, while sticking to transfer larger messages. Specifically, sticky transfers improve the performance of the DTN, specially under high mobility speeds or high node densities.

We implemented the "*Stop*" mode as the mode for sticky agreements from the possible sticky transfer modes described in section 3.3.1. As we mentioned at the beginning of this section, we chose this mode as it will have the most inhibiting effect on the natural mobility of nodes and thus will be the most effective for observing the lower-bound of sticky transfer performance. We expect the other sticky transfer modes, namely "*Follow*" and "*Slow down*", to perform at least as well as the "*Stop*" mode or better.

Our current assumption is that nodes have preset SP values; e.g. set by a network administrator. Algorithms to predict optimal SP values for nodes on-the-fly, which will be calculated based on the network environment and relevant node parameters, is part of our future work.

5 Conclusion

Delay-tolerant networks provide routing solutions in scenarios where the network comprises mostly of wireless mobile nodes/devices with network constraints, such as intermittent connectivity, which persist due to the mobility of nodes, power cycle of nodes, or geographical sparsity. No assumption is made with regard to the existence of a complete path between two nodes wishing to communicate. Source and destination nodes might never be connected to the same network, at the same time. Furthermore, nodes may not possess or acquire any knowledge about the network topology. Routes are built dynamically, while messages are in route between the sender and the destination(s), and any possible node can opportunistically (i.e. when the chance arises) be used as a next hop, provided it brings the message closer to the destination than the current node.

An *encounter* refers to two pair-wise nodes coming within the communication range of each other, presenting a possibility to exchange data. When a encounter occurs in an opportunist network, it is usually of limited duration. To be able to analyze situations that include limited bandwidth, we discussed the properties of *inter-contact time* (i.e. how frequently nodes encounter other nodes in the network on average) and *contact duration* (i.e. the average time two nodes have to exchange data during an encounter) and established that they directly impact the throughput (and, performance) of the network.

Several routing protocols have been proposed and implemented for DTNs which use strategies such as flooding messages, or using historical encounters to predict future encounter probabilities, and so on. Irrespective of the forwarding technique used by routing algorithms, the actual time to transfer messages between two nodes is limited by the *contact duration* of the nodes, which depends inherently on node mobility. If the expected contact duration is not sufficient for the entire message to be transmitted, which can change at any moment due to node mobility or other factors, messages tend to fail to be forwarded to the next hop. This can cause the routing protocol to retransmit the message, thus wasting valuable bandwidth, and wasting the resource consumed by the failed transfer. In addition, many other messages which have been processed and ready for transmission cannot be forwarded. These messages will stay longer in buffers of limited sizes, which may eventually be discarded due to buffer overflow, wasting node resources. The end result is low message delivery ratio and long end-to-end delay. The above problems are exacerbated in highly mobile DTNs that must handle large messages.

Our objective was to:

- Prolong the natural contact duration of mobile nodes in a DTN to improve the network performance, measured by the delivery ratio, end-to-end delay, and average buffer time.

- Propose a solution that can be integrated into the existing DTN architecture without requiring additional infrastructure or storage.

We addressed the above issues by proposing a novel framework called the *sticky transfer framework* that enables nodes to prolong their contact durations for message transfers in DTNs. In this framework, nodes send out periodic beacons for neighbor discovery. Once a neighbor is detected with which a node can perform sticky transfers, the nodes exchange information such as mobility speed and direction, current location, transmission range, available buffer size, the amount of data to be sent and the corresponding destination, using our proposed *sticky transfer protocol* within the framework.

5.1 Summary of Contributions

Main contributions of this paper include:

1. A novel framework called the sticky transfer framework that enables mobile nodes in a DTN to modify their mobility and prolong contact durations to enhance successful message transfers. The framework allows users of mobile devices to cooperatively adjust their natural movement behaviors and ensures users flexibility in the agreement to 'stick' using the following three components: *user preference*, *sticky modes*, and *compatibility list*.

2. We demonstrated the seamless integration of the sticky transfer framework with the existing DTN network management modules.

3. We evaluated our framework with extensive simulations showing the performance of the sticky transfer protocol under various network setting and measuring several perforamce parameters. To evaluate the effectiveness of our framework, we ran tests on three DTN routing protocols: Epidemic, PRoPHET, and Spray-and-Wait; with and without sticky transfers in a realistic mobile environment. Our analysis showed that the sticky transfer protocol improved the performance parameters we considered, and that our framework is especially beneficial for large message sizes and/or high mobility situations, where contact times allow for few successful transfers.

5.2 Future Work

The sticky transfer framework and protocol can be enhanced and extended in many ways. Currently, we use a constant stick probability in our simulations for all nodes in the network. A more intelligent, adaptive algorithm can be developed which allows nodes to dynamically decide whether to stick or not using available information such as its mobility speed relative to its neighbors' speeds and local node density. We also plan to consider the overhead of sticky negotiations and time required for resolving medium contention prior to sticky data transfers. Other future research issues include:

- Developing new algorithms using Bayesian networks and Markov models that consider past network performance to predict future optimal stick decisions without requiring intervention from the network administrator.

- Developing algorithms to select the best neighbors to perform sticky transfers, which will take into account several factors such as neighbors' residual energy, transmission rates, and amounts of data to be exchanged.

- Evaluating the effectiveness of sticky transfers in multi-rate networks.
- Implementing sticky message transfers in a realistic network using Mindstorm® NXT robots.

References

[1] L. Pelusi, A. Passarella, M. Conti, *Opportunistic Networking: data forwarding in disconnected mobile ad hoc networks*, IEEE Communications Magazine, Vol. 44, No. 11, Nov. 2006. `http://dx.doi.org/10.1109/MCOM.2006.248176`

[2] S. Jain, K. Fall, R. Patra, *Routing in a Delay Tolerant Network*, ACM SIGCOMM, 2004. `http://dx.doi.org/10.1145/1030194.1015484`

[3] E. Royer, C-K Toh, *A Review of Current Routing Protocols for Ad-hoc Mobile Wireless Networks*, IEEE Personal Communications Magazine, 1999. `http://dx.doi.org/10.1109/98.760423`

[4] Charles E. Perkins , Elizabeth M. Royer, *Ad-hoc On-Demand Distance Vector Routing*, IEEE Workshop on Mobile Computer Systems and Applications, Feb. 1999.

[5] Charles E. Perkins , Pravin Bhagwat, *Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers*, ACM SIGCOMM, Aug. 1994. `http://dx.doi.org/10.1145/190314.190336`

[6] J. Broch, D.B. Johnson, D.A. Maltz; *The Dynamic Source Routing Protocol for Mobile Ad hoc Networks*, IETF MANET Working Group, Internet-Draft, Oct. 1999.

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

[8] A. Lindgren, A. Doria, and O. Scheln, *Probabilistic Routing in Intermittently Connected Networks*, ACM MobiHoc, 2003. `http://dx.doi.org/10.1145/961268.961272`

[9] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, *Spray and wait: An Efficient Routing Scheme for Intermittently Connected Mobile Networks*, ACM SIGCOMM, 2005. `http://dx.doi.org/10.1145/1080139.1080143`

[10] B. Burns, O. Brock, B. N. Levine, *MV routing and capacity building in disruption tolerant networks*, IEEE INFOCOM, 2005. `http://dx.doi.org/10.1109/INFCOM.2005.1497909`

[11] D. Nain et al., *Integrated Routing and Storage for Messaging Applications in Mobile Ad Hoc Networks*, WiOpt: Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks, 2003. `http://dx.doi.org/10.1023/B:MONE.0000042498.60917.e8`

[12] F. Tchakountio, R. Ramanathan, *Tracking Highly Mobile Endpoints*, IEEE WoWMoM, 2001. http://dx.doi.org/10.1145/605991.606003

[13] M. Musolesi, S. Hailes, C. Mascolo, *Adaptive Routing for Intermittently Connected Mobile Ad Hoc Networks*, IEEE WoWMoM, 2005. http://dx.doi.org/10.1109/WOWMOM.2005.17

[14] J. Leguay, T. Friedman, V. Conan, *Evaluating Mobility Pattern Space Routing for DTNs*, IEEE INFOCOM, 2006. http://dx.doi.org/10.1109/INFOCOM.2006.299

[15] T. Spyropoulos, K. Psounis, C. S. Raghavendra, *Single-copy routing in intermittently connected mobile networks*; IEEE SECON (Sensor and Ad Hoc Communications and Networks), 2004. http://dx.doi.org/10.1109/SAHCN.2004.1381922

[16] J. Ott, D. Kutscher, *A Disconnection-Tolerant Transport for Drive-thru Internet Environments*, IEEE INFOCOM, 2005. http://dx.doi.org/10.1109/INFCOM.2005.1498464

[17] J. Burgess, B. Gallagher, D. Jensen, B. N. Levine, *MaxProp: Routing for Vehicle-Based Disruption- Tolerant Networks*, IEEE INFOCOM, 2006. http://dx.doi.org/10.1109/INFOCOM.2006.228

[18] M.M.B. Tariq, M.H. Ammar, E.W. Zegura, *Message ferry route design for sparse ad hoc networks with mobile nodes*, ACM MobiHoc, 2006. http://dx.doi.org/10.1145/1132905.1132910

[19] R. C. Shah, W. Brunette, *Data MULEs: Modeling a Three-tier Architecture for Sparse Sensor Networks*, Intel Research Tech Report IRS-TR-03-001, 2003. http://dx.doi.org/10.1109/SNPA.2003.1203354

[20] T. Spyropoulos, A. Jindal, K. Psounis, *An Analytical Study of Fundamental Mobility Properties for Encounter-Based Protocols*, Technical Report CENG-2007-8, University of Southern California, 2007. http://dx.doi.org/10.1504/IJAACS.2008.019198

[21] X.Zhuo, Q.Li, W.Gao, G.Cao, Y.Dai, *Contact Duration Aware Data Replication in Delay Tolerant Networks*, IEEE ICNP, 2011. http://dx.doi.org/10.1109/ICNP.2011.6089057

[22] T. Spyropoulos, K. Psounis, C. S. Raghavendra, *Spray and focus: Efcient mobility-assisted routing for heterogeneous and correlated mobility*, IEEE PERCOM, 2007. http://dx.doi.org/10.1109/PERCOMW.2007.108

[23] P.Hui, J. Crowcroft, E. Yoneki, *Bubble rap: Social-based forwarding in delay-tolerant networks*, IEEE Transactions on Mobile Computing, Vol. 10, No. 11, 2011. http://dx.doi.org/10.1109/TMC.2010.246

[24] A. Balasubramanian, B. N. Levine, A. Venkataramani, *DTN Routing as a Resource Allocation Problem*, SIGCOMM, 2007. http://dx.doi.org/10.1145/1282380.1282422

[25] K. Fall, S. Farrell, *DTN: An Architectural Retrospective*, IEEE Journal on Selected Areas in Communications, Vol. 26, Issue. 5, Jun. 2008. http://dx.doi.org/10.1109/JSAC.2008.080609

[26] M. Demmer, E. Brewer, K. Fall, S. Jain, M. Ho, R. Patra, *Implementing Delay Tolerant Networking*, IRB-TR-04-020, Dec. 2004.

[27] *The ONE simulator*, http://www.netlab.tkk.fi/tutkimus/dtn/theone [retrieved: August, 2015]

[28] A.Keranen, J.Ott, T.Karkkainen, *The ONE Simulator for DTN Protocol Evaluation*, SIMUTools, 2009. http://dx.doi.org/10.4108/ICST.SIMUTOOLS2009.5674

[29] K. Scott, S. Burleigh, *Bundle Protocol Specification*, RFC 5050, The MITRE Corporation, NASA Jet Propulsion Laboratory, Nov. 2007.

[30] N. Bezirgiannidis, V. Tsaoussidis, *Packet size and DTN transport service: Evaluation on a DTN Testbed**, ICUMT, Oct. 2010. http://dx.doi.org/10.1109/ICUMT.2010.5676669

[31] W. Ivancic, W. M. Eddy, D. Stewart, L. Wood, J. Northam, C. Jackson, *Experience with delay-tolerant networking from orbit*, 4th Advanced Satellite Mobile Systems, Aug. 2008. http://dx.doi.org/10.1109/ASMS.2008.37

[32] *Interplanetary Overlay Network (ION) Design and Operation*, V1.11, Jet Propulsion Laboratory, California Institute of Technology, Dec. 2009. https://ion.ocp.ohiou.edu [retrieved: August, 2015]

[33] A.Keranen, *Opportunistic Network Environment Simulator*, Special Assignment Report, 2008.

[34] http://tier.cs.berkeley.edu/drupal/ [retrieved: August, 2015]

[35] Y. Cao, H. Cruickshank, Z. Sun, *A Routing Framework for Delay Tolerant Networks Based on Encounter Angle*, IWCMC, 2011. http://dx.doi.org/10.1109/IWCMC.2011.5982776

[36] V. Cerf, S. Burleigh, A. Hooke, L. Torgerson, R. Durst, K. Scott, K. Fall, H. Weiss, *Delay-Tolerant Networking Architecture*, RFC 4838, Google, NASA/Jet Propulsion Laboratory, The MITRE Corporation, Intel Corporation, SPARTA, Inc., Apr. 2007.

[37] *Message Switching*, Wikipedia, http://en.wikipedia.org/wiki/Message_switching [retrieved: August, 2015]

[38] V. Bharghavan et. al., *MACAW: A Medium Access Protocol for Wireless LAN's*, ACM SIGCOMM, Aug. 1994. http://dx.doi.org/10.1145/190314.190334

[39] M. Grossglauser, D. N. C. Tse, *Mobility Increases the Capacity of Ad Hoc Wireless Networks*, IEEE/ACM Transactions on Networking (TON), Vol. 10 Issue.4, Aug. 2002. http://dx.doi.org/10.1109/TNET.2002.801403

[40] https://en.wikipedia.org/wiki/Beacon_frame [retrieved: August, 2015]

[41] L. Wood, W. Eddy, W. Ivancic, C. Jackson, *Saratoga: A scalable Data Transfer Protocol*, Internet Draft, Sept. 2011. URL: https://tools.ietf.org/html/draft-wood-tsvwg-saratoga-10

[42] P. Matzakos, C. Bonnet, *Transport Layer Protocols and Strategies for Delay and Disruption Tolerant Networks*, EUROCOM Research Report RR-13-285, Jul. 2013.

[43] F. Yasmeen, U.T. Nguyen, N. Huda, S. Yamada, C. Borcea, *A Framework for Extending Contact Opportunities in Delay-and Disruption-Tolerant Networks*, IEEE SCPA Workshop, Jul. 2015.