Sticky Transfer Framework for Delay-and Disruption-Tolerant Networks

Project Report
CSE 4215: Mobile Communications
Winter 2014
York University

by: Farzana Yasmeen
Student # 211023199
Contents

Abstract 4

1. Introduction 5

2. Related Work 7

3. Sticky Transfer Framework
   3.1 Motivation 8
   3.2 Assumptions 8
   3.3 Concepts of Sticky Transfers 9
   3.4 Sticky Transfer Framework and Components 9
   3.5 Sticky Transfer Framework within the DTN Overlay 12
   3.6 Sticky Transfer Protocol 13
   3.7 Estimation of Expected Contact Duration 14
   3.8 Setting the Speed in Sticky Modes 15
   3.9 Performance Limitations of Sticky Transfers 15

4. Performance Evaluation
   4.1 Performance Metrics 16
   4.2 Simulation Settings 16
   4.3 Results and Analysis 17

5. Discussion and Conclusions 23

References 24
Abstract

In this report, we propose a sticky transfer framework to enhance successful message transfers in a Delay- and Disruption-Tolerant Network (DTN) where hosts transfer buffered messages opportunistically. In the proposed framework, ‘stick’ means two mobile devices remain within the communication range of each other beyond the contact duration that would naturally exist between them. In order to ‘stick’, nodes may modify movement behavior based on mutual agreements using movement status and preference information when the natural contact duration is not enough. Sticky modes define policies for changing nodes’ natural movement and speed, such that they may remain within the transmission range of another node for a pre-negotiated amount of time. We define and analyze several sticky modes, which involve nodes stopping, following each other, or slowing down. Sticky transfers improve the number of successful forwarded messages by allowing more messages to be transferred during the contact duration and by eliminating message transfer aborts. We evaluate the sticky transfer model through simulations via integrating it within three existing DTN routing protocols, Epidemic, PRoPHET and Spray-and-Wait, under a city-based network environment. Simulation results show that the network delivery ratio increased by as much as 38%, while the latency decreased by as much as 38% when using sticky transfers. However, using sticky transfer very frequently reduces the performance improvement for flooding based routing protocols due to higher overhead and decreased number of node encounters.
1. Introduction

Many networks need to be deployed quickly without infrastructure in ad hoc manner and need to be able to deliver messages even if no instantaneous end-to-end path can be found. Such networks include large-scale disaster recovery networks [1], mobile sensor networks for ecological monitoring [2], ocean sensor networks [3][4], people networks [5], vehicular networks [6][7] and projects such as TIER [8]. These types of networks can be realized with delay-and disruption-tolerant network (DTN) [9] technology. Generally, messages in DTNs are transferred hop-by-hop toward the destination in an overlay above the transport layer called the ‘bundle layer’. Unlike mobile ad hoc networks (MANETs) [10][11], 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. DTN routing protocols use different strategies in forwarding messages at each encounter. For example replication based routing protocols [12] - [14] create and forward multiple copies of a message when an encounter happens. Other protocols try to achieve good performance by using information (such as encounter history) to intelligently forward a limited number of copies [15] - [19].

The message delivery performance (such as delivery ratio and delay) in a DTN highly depends on time elapsed between encounters (inter-contact time) and the time two nodes remain in each other’s communication range once a contact is established (contact-duration). Limitations in the actual time to transfer data between nodes are observed due to the inherent mobility of hosts (Fig.1). When nodes move out of each others’ transmission range, data transfer is discontinued. Thus, many messages which are ready for forwarding cannot be forwarded. Additionally, if a message happens to be ‘on the fly’ during disconnection, the transmission is aborted. In a DTN where chances of encounters are crucial for network performance, this lack of time during node contacts results in an unutilized contact opportunity. Furthermore, messages will stay longer in the buffers, which may lead to message lifetime (TTL) expirations. Such problems are exacerbated in highly mobile DTNs that must handle large message sizes; e.g. vehicular networks [6][7].

![Fig. 1](image) Data transfer limitations due to natural movement
In this report we propose a novel and general framework for adequately utilizing node contacts for message transfers in DTNs called the sticky transfer framework. In this framework, once nodes have encountered each other, based on user preferences, they agree to stay within the transmission range of each other until message transfers can be completed. The amount of messages that are transferred depends on the underlying routing protocols message selection and forwarding strategy and on the agreed contact-time (i.e., ideally, all the messages that the routing protocol decides to transfer). It is likely that all mobile nodes in the network will not have the same capabilities and may necessitate different approaches for modifying their inherent mobility to adjust to the contact-time agreement. To accommodate user preferences in the sticky transfer framework, we propose and analyze five operational sticky modes that constitute actions such as stopping, slowing down or following other nodes and define changes to nodes’ inherent movement for sticky transfers. Once nodes agree to stick with each other, a sticky transfer begins.

Considering practical applications, 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 (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 delivered before this deadline (i.e. their TTL did not expired, but the use of the DNT is over). Therefore, it is crucial to deliver messages faster, and implicitly to avoid message aborts. The sticky transfer framework achieves 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 ensures that the K copies are forwarded faster; in fact, the copies will be forwarded during the first K contacts with different nodes. This leads to lower latencies and higher delivery ratios, which improves network performance.

Sticky transfer is an optional message transfer mechanism and any user may opt-in or opt-out from sticky transfers any time. Several applications exist where sticky transfers can be used to improve performance: (1) robots in a region survey application may be programmed to stick with each other for a while 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.

To evaluate the sticky transfer framework, we performed simulations in a city-based network environment with mobile nodes moving among certain points of interests (POIs). We analyzed the performance of sticky transfers by varying node density, network load, speed, and sticky behavior (i.e., the probability that nodes stick with each other) for Spray-and-Wait [12], PRoPHET [16] and Epidemic [13] routing protocols. Simulation results show that the delivery ratio of the network increases, and the message delivery latency decreases significantly with increased stick behavior of the nodes. The delivery ratio was increased by as much as 38%, while the latency was decreased by as much as 38%. We observed a decline in the effectiveness of sticky transfers for larger message sizes when nodes used the ‘stop’ mode with a high stick probability. Mobile nodes spent up to 25% of the simulation time not moving due to stopping for long periods to transfer, which affected the performance of the network due to decreased mobility [23]. Also, the overhead for transferring larger messages increased using flooding strategies with high stick probability, as more messages were dropped due to buffer limitations. These drawbacks can be easily mitigated by choosing a sticky behavior that is suitable for the application and tuning the stick time to motion time ratio.

The remainder of this report is structured as follows. We discuss related work in the following section. Section 3 describes the sticky framework in detail. Section 4 presents simulation results and their analysis. Section 5 summarizes the findings and concludes the report.
2. Related Work

The two main performance contributing factors in an opportunistic network are inter-contact time and contact-duration. The average inter-contact time measures how frequently nodes encounter other nodes 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 the mobility of the nodes and node density. Many nodes moving at faster speeds can reduce the average inter-contact time. Additionally, if the network is sparse, the inter-contact time can be reduced by introducing special components in the network, such as Ferries [20], or Data Mules [22] that move at relatively faster speeds on predefined routes and therefore increase contact opportunities. Inter-contact time directly affects the delay in the network.

The contact duration is the length of time for which two nodes remain within the transmission range of each other. The contact duration directly influences the capacity of opportunistic networks (e.g. DTNs) because it limits the amount of data that can be transferred between nodes. 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 movement patterns. Nodes adjust mobility at the instance of encounter, and return to normal movement behavior once the encounter is over. Thus, by encouraging cooperative and voluntary mobility changes the lightweight sticky transfer mechanism can substantially improve performance. To the best of our knowledge, there is no work closely related to the concept of sticky transfers.

The sticky transfer framework can be a feature of any mobile node and is independent of, but can function with, any DTN routing protocol. In our simulations which show the performance improvement with sticky transfers, we use three representative DTN routing protocols: Epidemic [14], PRoPHET [16], and Spray-and-Wait [12] which are briefly explained here.

In the Epidemic routing protocol [14], 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 [16] 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 Spray-and-Wait [12] 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 ⌊K/2⌋ copies to the other node and keeps ⌈K/2⌉ 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.
3. Sticky Transfer Framework

In this section we state the motivation for a sticky transfer framework along with assumptions. We also describe the concept and components of the framework, and the sticky message transfer protocol.

3.1 Motivation

In opportunistic networks such as DTNs, depending on the speed and mobile orientation of nodes, the contact duration can be very small; and depending on the size of messages and copy forwarding strategy, the time necessary for nodes to remain in contact for successful message transfers can be large. Insufficient contact duration during opportunistic communications results in underutilized contact opportunities, which limits the capacity of the network. When using sticky transfers, two mobile nodes voluntarily work together to extend the contact duration longer than otherwise possible and thus maximize utilization of contact opportunities.

3.2 Assumptions

The natural contact duration (or, contact time), $T_C$ is the length of time for which two nodes are expected to remain within the transmission range of each other. Here, we assume that a node is always in the transmission range of only one node (we discuss the solution for multiple nodes in the range of each other below). If at time $t_0$, $A$ comes into the transmission range of $B$ and moves away from $B$ at time $t_1$ then $T^c(A, B) = t_1 - t_0$. On the other hand, the time required for $A$ to complete transferring all messages (allocated by the routing protocol strategy) to $B$ is the required transfer duration, $T^r$. Let $R$ be the transmission speed of the nodes. If node $A$ has $p$ messages to be transferred to node $B$ where, $M_k$ is the size of the $k^{th}$ message for $1 \leq k \leq p$, and node $B$ has $q$ messages to be transferred to node $A$, where $M_l$ is the size of the $l^{th}$ message for $1 \leq l \leq q$, then the required transfer duration between $A$ and $B$ is:

$$T^r = \frac{\sum_{k=1}^{p} M_k + \sum_{l=1}^{q} M_l}{R}. \quad (1)$$

If multiple nodes are in the transmission range of each other, we assume the mutual encounter sequence comes naturally from the order in which nodes hear advertisement messages (i.e., "hello" beacons) from the other nodes. For example, if at time $t_0$, $A$ comes into the transmission range of $B$ and $C$, and hears $B$’s beacon first $A$ will finish sticky transfers with $B$ then will begin sticky transfers with $C$ (with 0 inter-contact time, given 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 $A$ was transferring messages to $B$ then $T^c(A, C) = 0$. Assuming that the message transfer starts immediately after nodes encounter, if $T^c(A, B) < T^r(A, B)$ then it is probable that aborts will happen and all of the messages which $A$ wants to forward to $B$ cannot be transferred within the expected contact time.
3.3 Concept of Sticky Transfers

To sufficiently utilize valuable contact opportunities, we propose the sticky transfer mechanism where encountered nodes will come to a mutual agreement on the time they will remain in each other’s communication range a-priori to message transfers. After sticky message transfers are complete, nodes un-stick and may end the encounter to proceed with their natural mobility behavior. Assuming that $T^c$ is the natural but insufficient contact duration to transfer messages, the additional time the nodes should remain in contact beyond the natural contact duration is $\delta = T^R - T^c$ where, $T^R = M/R$, (Eq. 1). Here, $\delta$ is the stick duration, which is calculated by nodes using the sticky transfer protocol, defined in section 3.4.

![Fig. 2](image.png)

Fig. 2 Transfer aborts can be avoided by extending contact duration

Note that if the natural contact duration is sufficient to complete message transfers, there is no need for the nodes to engage in sticky transfers. The pre-condition for sticky transfers is the natural contact duration between two nodes will be less than the required duration to transfer messages; i.e. $T^c \leq T^R$. 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. Second, sticky transfers will ideally eliminate message aborts. The next section describes the sticky transfer framework.

3.4 Sticky Transfer Framework and Components

A schematic of the sticky transfer framework is shown in Fig.3. There are three important entities: User Preference, Sticky Modes and Compatibility List, each of which is explained hereafter.

(1) User Preferences

Sticky transfers require changing the natural movement behavior of the nodes. As changing node movement depends upon user cooperation, to implement sticky transfers we realize users’ agreement of modified movement through user preferences, $P_i$, where '$i$' is any user in the network. 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.
(2) Sticky Modes

To enable user preference, we define several possible operational modes, called *sticky modes* for sticky transfers. The modes set in preferences represent how the users would respond to a sticky transfer request. Sticky transfers are realized by reducing the relative speed of the two encountered nodes such that they remain within the transmission range of each other for the required transfer duration (this may implicitly involve changes in the direction of motion as well). The relative speed of the two nodes can be reduced by changing the speed and/or movement direction of one or both nodes (section 3.8). We define five sticky modes: *Stop*, *Follow me*, *Follow you*, *Slow down* and *No stick*. Users can set one or more modes in their preference settings to be used for implementing sticky transfer decisions.

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 move one node with the same 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 node which adjusts its speed and direction to the other node’s speed and direction to follow the other node is called the *Follow you* (FLW2) mode. The relative speed of two nodes can be reduced by reducing one (or both) of the two node’s speed. It may be that one node is already moving slower than the other. So, only the node that is moving faster needs to adjust its speed by slowing down. When a node reduces its speed to increases the contact duration we call the mode *Slow down* (SLW) mode. Finally, a node may not agree to stick for message transfers or even may not respond to stick requests. This mode is *No Stick* (NO_STK) mode.

In the framework (Fig. 3), users A and B have input and stored their preferred sticky modes (i.e. SM$_1$, SM$_2$, etc.) in nodes A and B respectively under ‘preferences’. When setting preferences, some nodes may not be able to select some of the stick 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 prefer all modes and set a low priority for NO_STK, to ensure cooperative rescue operations. We assume that nodes will have sticky preferences set according to applications before engaging in any mission. When node A initiates a sticky transfer request to B, node B receives node A’s preference list and B can define the mode it will use to respond to A’s request from its own preference list. However, in order for sticky transfers to ‘work’, both of the nodes’ preferred modes should be compatible. We explain compatibility next.

---

Fig. 3 Sticky Transfer Framework
(3) Compatibility List

Sticky transfers are technically possible when preferences of two nodes have ‘compatible’ modes that allow for the mobility of each to mutually extend the contact. Based on the five specified sticky modes above, Table 1 shows the compatibility list between any two nodes. The table is constructed to determine sticky transfer compatibility based on user preference settings. For example, if both nodes preferences allow STP, then the modes are compatible. However, if one node allows only STP and the other node allows only FLW1, they are not compatible. Also, 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 and both are not moving in the same direction. Nodes have the information in Table 1 stored with them. Among compatible modes, the most preferred modes are used during the sticky transfer. In Fig. 3, if compatible modes are found between A and B, then sticky transfers are possible and B will send the resolved mode information as sticky mode $SM_A$ to A and will set its own sticky mode as the resolved mode $SM_B$. Then the message transfer will continue while both nodes are in these modes respectively. However, if compatible modes cannot be resolved, then sticky transfers are not possible and the nodes will exchange messages in a normal transfer mode, possibly with limited contact time.

<table>
<thead>
<tr>
<th>$SM_A$</th>
<th>$SM_B$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>STP</td>
</tr>
<tr>
<td>Stop (STP)</td>
<td>√</td>
</tr>
<tr>
<td>Slow down (SLW)</td>
<td>√</td>
</tr>
<tr>
<td>Follow me (FLW1)</td>
<td>×</td>
</tr>
<tr>
<td>Follow you (FLW2)</td>
<td>√</td>
</tr>
</tbody>
</table>

Table 1 Sticky Preference Compatibility (√ indicates compatibility, × indicates incompatibility and √/× indicates sticky modes may sometimes not be compatible due to user limitations)
3.5 Sticky Transfer Framework within the DTN Overlay

The DTN architecture was designed as an overlay to accommodate not only network connection disruption, but also to provide a framework for dealing with heterogeneity [30]. 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 (PDU) on each of the corresponding protocols at lower layers. The bundle protocol (BPL) is the most central component in the overlay and it requires detailed information of the state of the system upon which to base routing decisions [31].

Providing such information to the BPL is handled by the management plane. 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. Essentially, when sticky transfer is combined with a specific DTN routing protocol the overall network performance will depend upon 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.
3.6 Sticky Transfer Protocol

We assume that 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, \( M' \) (i.e. message size and id). Here, \( v' = \{v_p, \theta_p, (x_p, y_p)\} \) and \( M' = \{M_{j1d1}; M_{j2d2}; M_{j3d3}; ...M_{jd_k}\}; j=1...n \) and \( k=1...m \), where \( n \)=number of nodes in the network and \( m \)= number of messages to be transferred from node \( j \). The pool of messages in \( M' \) is decided for nodes by the routing protocol strategy. Suppose that nodes \( A \) and \( B \) have just come into each other’s transmission range which each has detected by beacons, and they have a number of messages to exchange. Fig. 5 shows the protocol sequence diagram for the sticky transfer of messages (where \( A \) sends the request first).

1. First, \( A \) sends a sticky transfer request to \( B \) along with its status information, \( I_A \) and sticky preferences, \( P_A^* \).

![Sticky transfer protocol sequence diagram](image)

2. After receiving the stick request from \( A \), \( B \) first calculates (Eq. 2, section 3.7) the natural expected contact duration, \( T^C \) between \( A \) and \( B \) using the status information in \( I_A \) and its own status information, \( I_B \). Then \( B \) finds all messages that \( A \) needs to send by removing from \( A \)’s message vector those message that \( B \) already has, thus making sure \( A \) does not send redundant messages. Depending on its own free buffer space \( B \) then decides the messages to receive from \( A \)’s message vector \( M'_A \) and specifies them in receive message vector \( M_r \) as the messages to receive from \( A \), thus making sure \( A \) does not send any messages \( B \) cannot accommodate in its buffer. Next, from its own message vector \( M'_B \) and from the free buffer space information \( BUF_A \) of \( A \), it will determine its own list of messages to send in the vector \( M_s \). \( B \) adds up the time needed

---

*Sticky requests can be included in the advertisement messages (i.e., “hello” beacons) to reduce the number of messages exchanged.
to send its own messages to A – this will be an upper bound because A may already have some of those messages. Both $M_r$ and $M_s$ are based on the empty buffer size of the intended receiver and expected transfer time. Next, $B$ can calculate an upper bound on the required transfer duration, $T^R$ based on message sizes of the send message vector, receive message vector, and the transmission speed (from Eq. 1). $B$ can compare $T^C$ and $T^R$ to determine if the natural contact duration is enough. $B$ also determines if a compatible stick mode exists between $A$ and $B$ from user preferences $P_A$ and $P_B$.

3. If $T^C$ is already enough for completing the message exchange, sticky transfers are not necessary and $B$ will notify $A$ through a reply with NO_STK as the stick mode, $SM$ and stick duration $\delta = 0$. The same response will result if $A$ and $B$’s sticky preferences are not compatible; i.e. mutually co-functioning sticky modes are not possible. However, if sticky transfers are necessary (i.e. $T^C > T^R$) and compatible sticky modes exist, then $B$ will set its own sticky mode and also set the mode of node $A$ in a variable $SM$. Finally, node $B$ will send an OK response (with send and receive message vector, stick duration, status information and stick mode) to $A$.

4. Node $A$ checks the OK response from $B$ and sets its stick mode as defined in $SM$. It then checks the messages $B$ wishes to send in $M_S$ and removes from $M_S$ the ones $A$ already has and redefines $M_S$. Node $A$ then sends messages indicated in the message vector $M_r$ to $B$, piggybacking the redefined $M_S$ vector on one of the data messages.

5. After receiving $M_r$, and the redefined vector $M_S$, $B$ will send messages indicated in $M_S$ to $A$ to complete the transfer. After completing sticky transfers, the nodes will resume their natural movement.

Since the computation 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 under assumptions (section 3.1), during sticky transfers if there are in fact multiple nodes in range, the requester (e.g. node $A$) picks one particular sequence of mutual encounters. This sequence may come naturally from the order in which $A$ hears the advertisement messages of other nodes.

### 3.7 Estimation of Expected Contact Duration

Consider two nodes $A$ and $B$ are in contact and moving on a plane in straight lines at angles of $0_A$ and $0_B$ ($0_A + 0_B < 180^\circ$), and at speeds of $v_A(>0)$ and $v_B(>0)$ respectively from time $t_0$. Suppose they move out of each other’s transmission range at time $t_1$. By projecting the speeds and direction of the two nodes along their connecting lines, the contact duration of the two nodes will be:

$$T^C = \begin{cases} 
\frac{W}{|v_A \pm v_B|} & \text{when,} \theta_A = \theta_B = 90^\circ \\
W \pm a & \text{otherwise} 
\end{cases}$$

(2)

where, $a$ is the distance between the two nodes at time $t_0$. When $\theta_A + \theta_B > 180^\circ$, numerator terms “$W-a$” and “$W+ a$” should be used for $\theta_A + \theta_B > 180^\circ$ and $\theta_A + \theta_B \leq 180^\circ$ conditions respectively.
3.8 Setting the Speed in Sticky Modes

To complete message transfers, 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 \( v_2 \). If the relative distance travelled by the two nodes before going out of contact is \( d \), then

\[
v_2 = \frac{d}{M} \quad \frac{1}{R}
\]  

(3)

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. In that case, the time \( T_2 \) for which they should stop or follow will be

\[
T_2 = \frac{M}{R} \cdot \frac{d}{v_1}
\]  

(4)

3.9 Performance Limitations of Sticky Transfers

Nodes using the sticky transfer protocol may experience transitory changes to their natural movement behavior to successfully transfer messages. An effect on the network performance, that is dependent on node mobility, is expected when sticky transfer is used. Generally in DTN, both the message delay and delivery ratio can be improved by reducing pair-wise inter-contact times. A smaller 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 critical from the point of view of the mission (e.g., it takes 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 make 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 which is a simulation environment capable of routing messages between nodes with various DTN routing algorithms and sender and receiver types [21]. We can find the benefit of using sticky transfers for message exchange by comparing the performances of routing protocols with and without sticky nodes. In our simulation we used a stick probability metric, SP, to represent the node willingness to sticky transfer requests. This SP is defined for each node in the network. A value of SP=0.0, 0.5 and 1.0 indicate that a node will not agree (the same as sticky transfer not being used), agree to 50% of the requests and always agree to a sticky transfer request respectively. We implement the STP mode as the mode for sticky agreements from the possible sticky transfer modes (section 3.3 – SLW, FLW1, FLW2) as it has the most critical effect on the natural mobility of nodes and thus is the most effective for knowing the lower-bound of sticky transfer performance. We ignore the time for calculating the stick mode at nodes as it is a machine process and this overhead is negligible compared to the message transfer time which depends on the wireless transmission capabilities of nodes.

4.1 Performance Metrics

We consider several performance metrics, to measure and compare the effectiveness of the sticky transfer protocol. The message delivery ratio is defined as the ratio of the number of successfully delivered messages to the total number of unique messages generated within the simulation time. The message delivery delay is the average end-to-end delay of the delivered messages only. We also consider the overhead ratio, which represents the quantity of excessive relays required to deliver the message to the destination and is representative of the bandwidth and energy consumption of the protocol. Specifically, we define the overhead ratio as: \( \frac{h_t - h_d}{h_d} \), where \( h_t \) is the number of hops a message (and all of its copies) traverses in the network, and \( h_d \) is the number of hops the delivered message had traversed before reaching the final destination.

4.2 Simulation Settings

In our simulations, we choose a map of the Helsinki downtown area of size 4500m x 3400m as our network area. Mobile nodes move according to the Shortest Path Map Based Movement model. Nodes randomly choose next locations from eleven disconnected Points of Interest (POIs). POIs are places on the map area used to model e.g. tourist attractions, shops, restaurants etc. Nodes move to selected POIs’ with random speeds and pause for a time randomly distributed between 1 second and 50 seconds before selecting the next POI. We used 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 were chosen as sources with random destinations. Each simulation ran for 30k seconds. Messages were generated every 5 seconds on average after a warm-up period of 1k seconds and generation stopped after 21k seconds. The average message generation rate was constant in the network; however loads in the network varied due to routing protocol copy forwarding strategies (i.e. flooding protocols incurred higher loads) and larger message sizes. Message sizes, were varied between 0.1Mbytes to 30 Mbytes with infinite lifetime (TTL) values. These sizes cover different types of messages such as text, photos, or short videos. Sticky transfer is expected to show its effectiveness especially for larger message sizes, which are desirable in DTNs considering the extra transmission effort [24]. The buffer size of each mobile node was 1GB. The transmission range of nodes were 100 meters with transmission speed of 11Mpbs (modeling IEEE 802.11b) and 54Mpbs (modeling IEEE 802.11g). In Table 2, we summarize important simulation settings and default values are used unless mentioned otherwise. Under these settings, we evaluate the benefits of the sticky transfer method in (i) Epidemic [14] (ii) Spray-and-Wait (SnW) [12] in binary mode with initial 4 copies of messages; and (iii) PRoPHET [16] routing protocols.
Table 2 Simulation Settings

<table>
<thead>
<tr>
<th>Mobile node type</th>
<th>Node speed (m/s)</th>
<th>No. of nodes</th>
<th>Buffer size (GB)</th>
<th>Msg. size (MB)</th>
<th>Transfer rate (Mbps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Range</td>
<td>-</td>
<td>1.25-25</td>
<td>10-30</td>
<td>1</td>
<td>0.1,30</td>
</tr>
<tr>
<td>Pedestrian</td>
<td>1.25-1.53</td>
<td>4</td>
<td></td>
<td>1</td>
<td>20</td>
</tr>
<tr>
<td>Cycler</td>
<td>2.5-8.33</td>
<td>6</td>
<td></td>
<td>1</td>
<td>11</td>
</tr>
<tr>
<td>Car</td>
<td>20-25</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

4.3 Results and Analysis

Simulation results presented in Fig. 6 to Fig. 10 considered scenarios with 20 nodes and 20 to 25 m/s moving speeds. We also present results for different node densities (Fig. 11), movement speeds (Fig. 12) and message sizes (Fig. 13). It should be noted that due to their design, Spray-and-Wait (SnW) leads to lower network load than the Epidemic and PRoPHET protocols.

Fig. 6 presents the delivery ratio as a function of stick probability (SP) with 20MB message size for both 11Mbps and 54Mbps transfer rates. At 54Mbps, a high delivery ratio was achieved even without sticky transfers due to 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 (e.g., Epidemic, PRoPHET) because of buffer overflows. For 11Mbps, the delivery ratio increased up to 38% with increasing SP values as more messages could be exchanged upon encounters due to nodes increasing willingness to stick for transfers.

![Fig. 6 Message delivery ratio as a function of stick probability](image-url)
At lower transfer rates (11 Mbps) when buffers were not a limitation, maximum delivery ratio was achieved at SP=1.0 in SnW. However, in Epidemic and PRoPHET, maximum delivery ratio was achieved for a SP value around 0.9. The delivery ratio decreased at SP=1.0 from SP=0.9 because nodes always agreed to stick (STP is the sticky mode) for message transfers, which reduced node movement. Nodes sticking longer cause the number of contacts per hour to decrease significantly, reducing contact opportunities. For small message sizes (e.g. 100kB), 100% delivery ratio was achieved in all three protocols even without sticky transfers and has been omitted for brevity. The benefit of sticky transfers is evident in scenarios with large message sizes and lower transfer rates.

Fig. 7 shows the message delivery delay for increased stick probability for 20MB message sizes. The reduction of delivery delay using sticky transfers was high especially at lower transfer rates. A maximum of 21% to 25% and 28% to 36% decrease in delivery delay were observed at 54Mbps and 11Mbps respectively. Obviously, the delays for 54Mbps were much smaller than those for 11Mbps due to higher transfer rates.

Fig. 8 shows the average buffer time as a function of stick probability for 20MB message sizes. The buffer time varied across protocols and stick probabilities. The buffer time increased as stick probability decreased due to reduced node movement and fewer contact opportunities.
With increasing SP values, messages spend less time in buffers on average (Fig. 8) and message copies were forwarded faster in the network, thus reducing delay significantly. For 100kB messages the improvement in delay was about 10% in general because in most cases stick was not required due to sufficient contact durations. We omit the graph here for brevity. Fig. 8 shows the average amount of time a message (20 MB size) spends in a node buffer. Buffer time for a message is the average time between when a message is received at a node to the time it is removed from that node, which also includes encounter delays. Message removals may happen either because the message was relayed to the next encounter or it was dropped from the node due to buffer overflow. As such, average buffer time can be higher than the average latency, because latency considers successfully delivered messages only. Average buffer time significantly impacts performance as it is an indicator at how fast nodes are forwarding and/or dropping messages. In Fig. 8, as SP gradually increased the average buffer time of messages decreased (up to 38%) compared to the no-sticky transfer case. The average time spent in buffers is lower for 54Mbps than 11Mbps due to faster transfer rates. In both cases at higher SP values (0.9 and 1.0), SnW had a higher average buffer time as it goes into the direct delivery (i.e., waiting) phase once all its copies are spread; thus messages remained longer in node buffers just prior to final delivery. Also, for Epidemic and ProPHET, more message drops occurred.

Fig. 8 shows the average amount of time a message (20 MB size) spends in a node buffer. Buffer time for a message is the average time between when a message is received at a node to the time it is removed from that node, which also includes encounter delays. Message removals may happen either because the message was relayed to the next encounter or it was dropped from the node due to buffer overflow. As such, average buffer time can be higher than the average latency, because latency considers successfully delivered messages only. Average buffer time significantly impacts performance as it is an indicator at how fast nodes are forwarding and/or dropping messages. In Fig. 8, as SP gradually increased the average buffer time of messages decreased (up to 38%) compared to the no-sticky transfer case. The average time spent in buffers is lower for 54Mbps than 11Mbps due to faster transfer rates. In both cases at higher SP values (0.9 and 1.0), SnW had a higher average buffer time as it goes into the direct delivery (i.e., waiting) phase once all its copies are spread; thus messages remained longer in node buffers just prior to final delivery. Also, for Epidemic and ProPHET, more message drops occurred.

Fig. 9 shows the average overhead ratio for 20 MB size messages. The overhead ratio is the number of extra forwards that messages (and all its copies) require for end-to-end delivery. For Epidemic and ProPHET, each time nodes encounter each other, copies of messages are spread 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 54 and 11Mbps). 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. 9 Average overhead ratio as function of stick probability](image-url)
Fig. 10 Stick time with varying stick probability

Fig. 10 shows a measurement of the total time nodes stick in the network as a percentage of total simulation time. Since we implement sticky transfers through STP mode, the stick time represents the total amount of time that a node stops (is stationary) and thus it is the cumulative delay on its journey to all of its destinations. For both 11Mbps and 54Mbps, SnW incurs less cumulative stick time in general as nodes have less loads to forward upon encounters. The stick time was higher for flooding protocols (up to 25%), particularly at SP=1, since nodes remained in contact with mutual encounters until all messages were forwarded. Too much stick time can reduce node mobility in the DTN, which can lead to fewer contact opportunities over time.

Fig. 11 shows the performance improvement achieved with sticky transfers at different node densities at 11Mbps transfer rates. The delivery ratio is shown in Fig 11(a). At low node density (5MH), the performance increase was up to 6%. At high densities (30MH), the performance increase was up to 43%. The performance improvement at higher densities compared to a less dense network came from the increase in encounters of MH’s due to more nodes in the same area. The performance improved significantly up to SP=0.9 and then fell slightly at SP=1.0 due to the same reasons as in Fig. 6. The delivery delay (Fig. 11(b)) decreased up to 22% at low densities and up to 43% at high densities. Sticky transfers increase benefits at higher node densities.

Fig. 11 Node densities with varying stick probability
In order to observe the performance improvement with sticky transfers for different node movement speeds in Fig. 12, we set the same speed for all nodes and varied the speeds at 11Mbps transfer rates. In the graphs we present results for 5m/s and 20m/s node speeds. Delivery ratios increased up to 12% (for 5m/s) and 49% (for 20m/s) in different protocols (Fig. 12(a)). Also delivery delay decreased to 15% and 45% respectively for 5m/s and 20m/s speeds (Fig. 12(b)). When sticky transfers were not used in the network, 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 was not enough to successfully complete the message transfers of large message sizes (20MB). However, as we increased the SP, the improvement in the delivery increased significantly for higher node speeds. Nodes were able to extend the contact duration to the required duration by sticking. Thus with the combination of meeting encounters faster and being able to transfer more messages, the delivery and latency performance of 20m/s speeds surpassed that of the lower speed scenario. Therefore, sticky transfers give higher benefit in higher mobility conditions where message transfer disruptions are more likely.

Fig. 12 Performance of sticky transfer for different movement speeds

Fig. 13 Performance of sticky transfer for different message sizes
Finally, in Fig. 13 we show the performance improvement using sticky transfers for 5MB and 30MB message sizes at 11Mbps transfer rates. Delivery ratio increased by 24% and 20% in Epidemic protocol while they increased by 43% and 37% in the other protocols for 5MB and 30MB messages respectively (Fig. 13(a)). Delivery delay decreased gradually from 30% to 39% in different protocols with increasing SP for 5MB messages. On the other hand for 30MB messages an increase in delay was observed around 0.3SP for all protocols from which the delay gradually dwindled and the increase became smaller towards SP=1.0 (Fig. 13(b)). This is due to the combined effects of restricting mobility (for larger messages) and reducing aborts. Therefore, sticky transfer improves delivery ratio for all message sizes, but may increase delivery delay for large message sizes depending on the routing protocols used.
5. Discussion and Conclusions

In DTNs, the performance of routing protocols depends largely on the mobility type and may not be the same if assumptions change suddenly (e.g., disaster situation). For example, if POIs are not used or the social behavior of the nodes change, PRoPHET will not perform well. However, the performance of any routing strategy is also bounded by the contact duration of the nodes, and this issue has not been addressed yet by the DTN research community. In this report, we proposed and analyzed a simple yet efficient method for increasing DTN performance by increasing the contact duration: sticky transfers - which can be coupled with any DTN routing protocol. 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. Simulation results show that for 20MB message sizes the delivery ratio was increased by up to 38% for 11Mbps transfer rate; also, the delay was decreased by up to 38%. For higher transmission rates (54Mbps), the data delivery ratio did not improve much, but the delay decreased by as much as 25%. In simulations, we used the sticky mode (e.g., STP) that gives the least performance improvement. It is expected that the performance will increase from the one observed when using “Slow down” and “Follow” modes as they have less effect on node movement.

The proposed sticky transfer scheme requires temporary change in movement of nodes upon sticky transfer agreements. Therefore, it is useful where the required change in node movement can be applied, such as human and robot assisted mobility scenarios. We believe that performance gains such as those achieved through our proposed method can open the path towards future widespread adoption of our proposed framework in DTNs.
References


