A Framework for Extending Contact Opportunities in Delay-and Disruption-Tolerant Networks

Farzana Yasmeen, Nurul Huda, Uyen Trang Nguyen
Department of Electrical Engineering and Computer Science
York University, Toronto, Canada
Email: {yasmeen,huda,utn}@cse.yorku.ca

Shigeki Yamada
Department of Informatics, Sokendai
National Institute of Informatics
Tokyo, Japan
Email: shigeki@nii.ac.jp

Cristian Borcea
Department of Computer Science
New Jersey Institute of Technology
Newark, USA
Email: borcea@njit.edu

***Abstract*—We propose a sticky transfer framework to enhance successful message transfers in a Delay- and Disruption-Tolerant Network (DTN). We propose a sticky transfer protocol within the framework that enables mobiles nodes to "stick" to each other for a longer period of time in order to complete the transmissions of the messages needed to be sent. A mobile node A makes the decision of whether to stick to another node B or not by exchanging information with its neighbors and negotiating an agreement to stick to B in order to prolong the contact duration for transferring messages. Sticky transfers improve the number of successfully forwarded messages by allowing more messages to be transferred during the contact duration and by minimizing the number of message transfer aborts. We evaluate the effectiveness of the proposed sticky transfer framework through simulations, which show that the average message delivery ratio increased by as much as 38%, while the average end-to-end delay decreased by as much as 36% when using sticky transfers.**

I. Introduction

Many infrastructure-less networks require quick, ad hoc deployment and the ability to deliver messages even if no instantaneous end-to-end path can be found [1]–[8]. Such networks can be implemented using delay-and disruption-tolerant network (DTN) technology [9]. The message delivery performance (such as delivery ratio and 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 (the *contact duration*) because 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 resources consumed by the failed transfer. In addition, many other messages which have been processed and were 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 such as vehicular networks [6], [7].

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 with which a node can perform sticky transfers is detected, 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. Based on the received information, a node A calculates the needed contact duration as a function of the amount of data to be exchanged with 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: (1) robots in a region-surveying 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.

To evaluate the effectiveness of the proposed sticky transfer framework, we performed simulations using a city-based network topology and the Spray-and-Wait [10], PRoPHET [13], and Epidemic [11] opportunistic routing protocols. We evaluated the performance of each routing protocol with and without sticky transfers. 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 message end-to-end delay decreased by as much as 36% with sticky transfers.

The remainder of this paper is structured as follows. We discuss related work in the Section II. Section III describes the sticky framework in detail. In Section IV, we present simulation results and our analysis. Section V summarizes the findings and outlines our future work.

978-1-4673-6305-1/15/$31.00 ©2015 IEEE

II. Background and Related Work

The two main contributing factors that determine the performance of an opportunistic network are *inter-contact time* and *contact duration* [21]. 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 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 [17] or data mules [19], that move at relatively faster speeds on predefined routes and therefore increase contact opportunities.

The contact duration is the length of time during which two nodes remain within the transmission range of each other. 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. By using the proposed sticky transfer framework, nodes can intelligently and cooperatively increase their contact durations to improve the capacity of the network by agreeing to brief, temporary modifications in their movement patterns and speeds.

Zhuo et al. [24] 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.

Our proposed sticky transfer framework is independent of, but can function with, any DTN routing protocol. DTN routing protocols use different strategies in forwarding messages at each encounter. For example, replication based routing protocols [10], [11], [22] create and forward multiple copies of a message when an encounter happens. In replication-based protocols, any node A can receive a copy of a message from any other node B when they come in contact. Other protocols try to improve the performance by using information (such as encounter history) to intelligently forward a limited number of copies [12]–[16].

III. The Proposed Sticky Transfer Framework

In this section, we first state the definitions and assumptions, and then describe the concept and components of the framework and the sticky message transfer protocol.

A. Definitions and Assumptions

The *natural contact duration* (or, *expected contact time*) 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 movement, the natural contact duration T^C of the two nodes can be predicted to be:

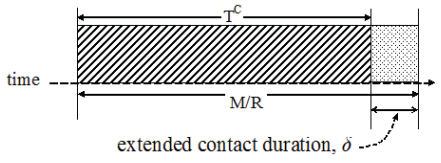

Fig. 1. Transfer aborts can be avoided by extending the contact duration.

$$T^C = \frac{-(ab+cd) + \sqrt{(a^2+c^2)W^2 - (ad-bc)^2}}{a^2+c^2} \quad (1)$$

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 $d = y_A - y_B$. Note that when $v_A = v_B$ and $\theta_A = \theta_B$, T^C approaches ∞. This predicted value is the expected contact duration between the two nodes. Here we assume that every node is equipped with a GPS, which helps to determine its speed and direction of movement.

Equation 1 also assumes that a node is 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 (required 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

$$T^R = \frac{\sum_{k=1}^{p} M_k + \sum_{l=1}^{q} M_l}{R} \quad (2)$$

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 from other nodes. For example, if at time t_0, A comes into the transmission range of B and C and receives B's message beacon first, A will finish sticky transfers with B first 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 was transferring messages to B then $T^C(A,C) = 0$.

Assuming that the message transfer starts immediately after nodes encounter each other, if $T^C(A,B) < T^R(A,B)$ then

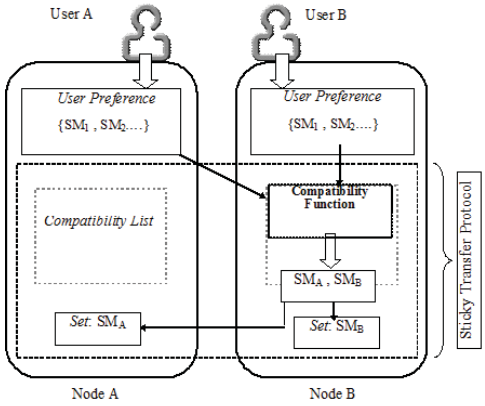

Fig. 2. Sticky Transfer Framework.

message aborts are probable and not all of the messages which A wants to forward to B can be transferred within the expected contact time.

B. Concept of Sticky Transfer

To maximize the use of valuable contact opportunities, the sticky transfer mechanism allows nodes to come to a mutual agreement, prior to the message transfers, 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 $T^R = M/R$, where M is the total size of all the messages to be transferred between the two nodes (Fig. 1). We call δ the *stick duration*, which is calculated by nodes using the sticky transfer protocol, described later in this section. 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. Second, sticky transfers will minimize the number of message aborts, improving message delivery ratio and network resource utilization.

C. Sticky Transfer Framework

The sticky transfer framework consists of three components: sticky modes, user preferences and compatibility lists. A schematic of the framework is shown in Fig 2.

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 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 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).

TABLE I
STICKY PREFERENCE COMPATIBILITY. ($\surd$ INDICATES COMPATIBILITY, $\times$ INDICATES INCOMPATIBILITY AND $\surd/\times$ INDICATES THE MODES MAY SOMETIMES NOT BE COMPATIBLE DUE TO USER LIMITATIONS)

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)	$\surd$	$\surd$	$\surd/\times$	$\times$

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).

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 accommodate all modes and set a low priority for NO_STK mode to ensure cooperative rescue operations. We assume that nodes will have sticky mode preferences set according to the application before engaging in any 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 shown in Fig 2, 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".

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. For example, when node A selects SLW mode and B selects FLW1 mode, they are compatible if B's speed is slower than A's speed. However, 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 I). 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

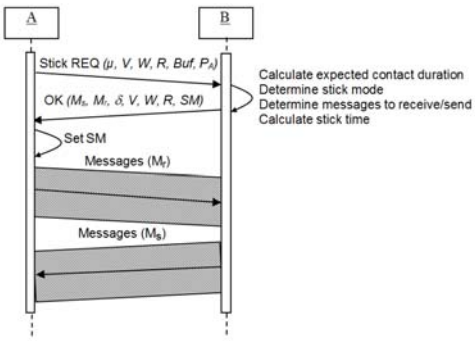

Fig. 3. Sticky transfer protocol sequence diagram

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.

D. 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, $V = \{v_j, \theta_j, (x_j, y_j)\}$ and $\mu = \{(\mu_1, id_1), (\mu_2, id_2), (\mu_3, id_3), \ldots, (\mu_k, id_k)\}$, $j = 1, \ldots, n$ and $k = 1, \ldots, m$, 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 [10], [11], [13]. Suppose that nodes A and B have just come into each other's transmission range and have a number of messages to exchange. Fig. 3 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, using Eq. (1), the expected contact duration, T^C between A and B using 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. (2). 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 from 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.
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.

IV. 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 [18]. We compare the performance of DTN routing protocols with and without sticky transfers. In particular, we evaluate the performance gain of the sticky transfer mechanism with the (i) Epidemic; (ii) Spray-and-Wait (SnW) in binary mode with initial four copies of messages; and (iii) PRoPHET routing protocols.

In our simulations, we assume a simple model in which a node agrees to a sticky transfer request with a probability value of $p(stick)$, where $0 \leq p(stick) \leq 1$. In our graphs we denote $p(stick)$ as stick probability SP. A value of SP = 0, 0.5, and 1 indicate that a node does not agree, agrees to 50% of the requests, or always agrees to sticky transfer requests, respectively. The SP of a node deos not change during the duration of a simulation run. In future work, we will develop algorithms to enable nodes to determine the optimal course of action 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).

TABLE II
SIMULATION SETTINGS

No. of nodes	Node speed (m/s)	Buffer size (GB)	Msg. Size (MB)	Transfer rate (Mbps)
20	20-25	1	0.1-30	11-54

We implement the 'Stop' mode as the mode for sticky agreements from the possible sticky transfer modes described in section III because it has 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 ignore the time for calculating the stick mode at nodes since this overhead is negligible compared to the message transfer time, which depends on the wireless transmission capabilities of nodes.

1) Performance Metrics: We use the following three performance metrics to evaluate the effectiveness of the proposed framework. 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 successfully delivered messages. The *average buffer time* for a message is the average time between the time a message is received at a node and the time it is removed from that node.

2) Simulation Settings: We choose a map of the Helsinki downtown area of size 4500 m x 3400 m. Mobile nodes move according to the Shortest Path Map Based Movement model [23]. Nodes randomly choose next locations from eleven disconnected points of interest (POIs). POIs are places on the map used to model tourist attractions, shops, restaurants, etc. Nodes move to selected POIs with random speeds and pause for a period with a value randomly distributed between 1 second and 50 seconds before selecting the next POI. We used 20 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 ran for 30,000 seconds. Messages were generated on average every 5 seconds after a warm-up period of 1,000 seconds. Nodes stopped sending messages after 21,000 seconds. Message sizes vary between 0.1 MBytes and 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 [20]. The buffer size of each mobile node was 1 GB. The transmission range of nodes were 100 meters with transmission rate of 11 Mbps (modeling IEEE 802.11b) and 54 Mbps (modeling IEEE 802.11g). Table II summarizes the simulation settings; the default values are used unless otherwise stated.

3) Results and Analysis: We present our simulation results in Figs. 4 to 7. Before delving into the results, let us note that the message dissemination strategy of Spray-and-Wait (SnW) leads to lower network load than the Epidemic and PRoPHET protocols.

Fig. 4 presents the delivery ratio as a function of stick probability (SP) with a message size of 20 MBytes and 11 Mbps and 54 Mbps transmission rates. 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 (e.g., 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 nodes 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 for ab 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 node movement. The above difference is due to the fact that SnW forwards fewer copies of a message than the Epidemic and PRoPHET protocols.

Fig. 5 shows the message delivery delay for increased stick probability for 20 MB message sizes. The reduction of delivery 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% decrease in the delivery 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 rates.

Fig. 6 shows the average amount of time a message (20 MB size) spends in a node buffer. In Fig. 6, 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 54 Mbps than 11 Mbps due to faster transfer rates.

Fig. 7 shows 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 STP 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. 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.

We also performed simulations for different node densities, movement speeds, message sizes, and other stick modes. The results from these experiments are consistent with those presented above. Specifically, sticky transfers improve the performance of the DTN, especially under high mobility speeds or high node densities. Sticky transfers improve the message delivery ratio for all message sizes, but may increase the delivery delay of large message sizes (30 MBytes or larger)

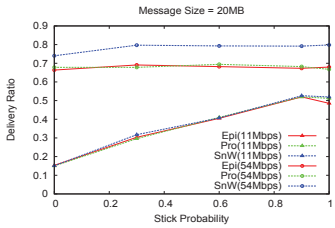

Fig. 4. Message delivery ratio as a function of stick probability.

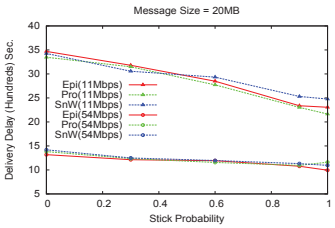

Fig. 5. Message delivery delay as a function of stick probability.

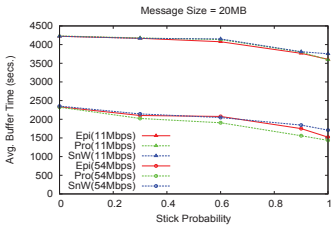

Fig. 6. Average buffer time as a function of stick probability.

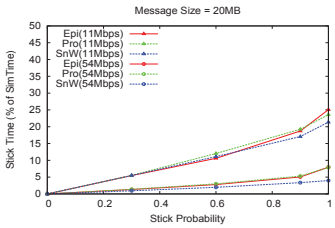

Fig. 7. Stick time with varying stick probability.

given certain stick probabilities and routing protocols.

V. Conclusion

The sticky transfer framework and protocol enable nodes in DTNs to collect neighbors' information and evaluate their movement patterns/amounts of data to transfer in order to make decisions of whether to " stick" with a neighbor to complete the necessary data transfers. Nodes intelligently negotiate sticky transfer parameters such as stick duration, mobility speed and movement directions based on user preferences and collected information. The sticky transfer framework can be combined with any DTN routing protocol to improves its performance. Our simulation results show that sticky transfers significantly improve the performance of the Spray-and-Wait, PRoPHET and Epidemic opportunistic routing protocols in terms of message delivery ratio and end-to-end delay.

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. We are developing an intelligent, adaptive algorithm 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 will also 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 with, 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.

VI. Acknowledgments

This research was supported in part by the NSF Grant No. CNS-1409523 and by a Discovery Grant from the Natural Sciences and Engineering Research Council of Canada.

References

[1] G. Zussman and A. Segall, *Energy Efficient Routing in Ad Hoc Disaster Recovery Networks*, IEEE INFOCOM, 2003.
[2] P. Zhang, C. M. Sadler, S. A. Lyon, and M. Martonosi, *Hardware design experiences in ZebraNet*, ACM SenSys, 2004.
[3] J. Partan, J. Kurose, and B. N. Levine, *A Survey of Practical Issues in Underwater Networks*, ACM WUWNet, 2006.
[4] A. Maei, K. Fall, and D. Chayes, *Ocean Instrument Internet*, AGU Ocean Sciences Conference, 2006.
[5] M. Motani, V. Srinivasan, and P. Nuggehalli, *PeopleNet: Engineering a Wireless Virtual Social Network*, ACM Mobicom, 2005.
[6] J. Ott and D. Kutscher, *A Disconnection-Tolerant Transport for Drive-thru Internet Environments*, IEEE INFOCOM, 2005.
[7] J. Burgess, B. Gallagher, D. Jensen, and B. N. Levine, *MaxProp: Routing for Vehicle-Based Disruption-Tolerant Networks*, IEEE INFOCOM, 2006.
[8] TIER Project, UC Berkeley. http://tier.cs.berkeley.edu/
[9] DTN Architecture RFC. http://tools.ietf.org/html/rfc4838
[10] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, *Spray and wait: An efcient routing scheme for intermittently connected mobile networks*, ACM SIGCOMM, 2005.
[11] A. Vahdat and D. Becker, *Epidemic routing for partially connected ad hoc networks*, Technical Report CS-2000-06, Department of Computer Science, Duke University, 2000.
[12] J. Burgess, B. Gallagher, D. Jensen, and B. N. Levine, *MaxProp: Routing for vehicle-based disruption-tolerant networks*, IEEE INFOCOM 2006.
[13] A. Lindgren, A. Doria, and O. Scheln, *Probabilistic routing in intermittently connected networks* ACM MobiHoc, 2003.
[14] B. Burns, O. Brock, and B. N. Levine, *MV Routing and Capacity Building in Disruption Tolerant Networks*, IEEE INFOCOM, 2005.
[15] T. Small and Z. Haas, *Resource and Performance Tradeos in Delay-Tolerant Wireless Networks*, ACM WDTN, 2005.
[16] A. Balasubramanian, B. N. Levine, and A. Venkataramani, *DTN Routing as a Resource Allocation Problem*, SIGCOMM, 2007.
[17] M.M.B. Tariq, M.H. Ammar, and E.W. Zegura, *Message ferry route design for sparse ad hoc networks with mobile nodes*, ACM MobiHoc, 2006.
[18] The ONE simulator. http://www.netlab.tkk.fi/tutkimus/dtn/theone
[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.
[20] C.V. Samaras, V. Tsaoussidis, *Adjusting Transport Segmentation Policy of DTN Bundle Protocol under Synergy with Lower Layers*, Elsevier Journal of Systems and Software, Vol. 84, Issue 2, 2011.
[21] T. Spyropoulos, A. Jindal, and K. Psounis, *An Analytical Study of Fundamental Mobility Properties for Encounter-Based Protocols*, Technical Report CENG-2007-8, University of Southern California, 2007.
[22] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, *Spray and focus: Efcient mobility-assisted routing for heterogeneous and correlated mobility*, IEEE PERCOM, 2007.
[23] A.Keranen, J.Ott, and T.Karkkainen, *The ONE Simulator for DTN Protocol Evaluation*, SIMUTools, 2009.
[24] X.Zhuo, Q.Li, W.Gao, G.Cao, and Y.Dai, *Contact Duration Aware Data Replication in Delay Tolerant Networks*, IEEE ICNP, 2011.