Effective Implementation of Conditional Shortest Path Routing in Delay Tolerant Networks

Dr.G. Rajitha Devi.
Asst.Prof. in Computer Science
Email:rajithareddygurram@gmail.com

Abstract *- In this article Delay tolerant networks are characterized by the sporadic connectivity between their nodes and therefore the lack of stable end-to-end paths from source to destination. Delay tolerant networks (DTNs) where each node knows the probabilistic distribution of contacts with other nodes. Since the future node connections are mostly unknown in these networks, opportunistic forwarding is used to deliver messages. However, making effective forwarding decisions using only the network characteristics (i.e. average intermeeting time between nodes) extracted from contact history is a challenging problem. Based on the observations about human mobility traces and the findings of previous work, we introduce a new metric called conditional intermeeting time, which computes the average intermeeting time between two nodes relative to a meeting with a third node using only the local knowledge of the past contacts. We then look at the effects of the proposed metric on the shortest path based routing designed for delay tolerant networks. We propose Conditional Shortest Path Routing (CSPR) protocol that routes the messages over conditional shortest paths in which the cost of links between nodes is defined by conditional intermeeting times rather than the conventional intermeeting times. Through trace-driven simulations, we demonstrate that CSPR achieves higher delivery rate and lower end-to-end delay compared to the shortest path based routing protocols that use the conventional intermeeting time as the link metric.*

Keywords *- Delay Tolerant Networks, CSPR, ROUTER*

I. INTRODUCTION

Routing in delay tolerant networks (DTN) is a challenging problem because at any given time instance, the probability that there is an end-to-end path from a source to a destination is low. Since the routing algorithms for conventional networks assume that the links between nodes are stable most of the time and do not fail frequently, they do not generally work in DTN's. Therefore, the routing problem is still an active research area in DTN's [1].Routing algorithms in DTN's utilize a paradigm called *store-carry-and-forward*. When a node receives a message from one of its contacts, it stores the message in its buffer and carries the message until it encounters another node which is at least as useful (in terms of the delivery) as itself. Then the message is forwarded to it. Based on this paradigm, several Research was sponsored by US Army Research Laboratory and the UK Ministry of Defence and was accomplished under Agreement Number W911NF-06-3-0001. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the US Arm y Research Laboratory, the U.S. Government, the UK Ministry of Defence, or the UK Government. The US and UK Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

Routing algorithms with different objectives (high delivery rate etc.) and different routing techniques (single-copy [2] [3], multi-copy [4] [5], erasure coding based [6] etc.) have been proposed recently. However, some of these algorithms [7] used unrealistic assumptions, such as the existence of oracles which provide future contact times of nodes. Yet, there are also many algorithms (such as [8]-[10]) based on realistic assumption of using only the contact history of nodes to route messages opportunistically.

Recent studies on routing problem in DTN's have focused on the analysis of real mobility traces (human [11], vehicular [12] etc.). Different traces from various DTN environments are analyzed and the extracted characteristics of the mobile objects are utilized on the design of routing algorithms for DTN's. From the analysis of these traces performed in previous work, we have made two key observations. First, rather than being memory less, the pair wise intermeeting times between the nodes usually follow a log-normal distribution [13] [14]. Therefore, future contacts of nodes become dependent on the previous

contacts. Second, the mobility of many real objects are non-deterministic but cyclic [15]. Hence, in a cyclic Mobi Space [15], if two nodes were often in contact at a particular time in previous cycles, then they will most likely be in contact at around the same time in the next cycle.

Additionally, previous studies ignored some information readily available at transfer decisions. When two nodes (e.g., A and B) meet, the message forwarding decision is made according to a delivery metric (encounter frequency [9], time elapsed since last encounter [16] [17], social similarity [18] [19] etc.) of these two nodes with the destination node (D) of the message. However, all these metrics depend on the separate meeting histories of nodes A and B with destination node D[1]. Nodes A and B do not consider their meetings with each other while computing their delivery metrics with D.To address these issues, we redefine the intermeeting time concept between nodes and introduce a new link metric called *conditional intermeeting time*. It is the intermeeting time between two nodes given that one of the nodes has previously met a certain other node. For example, when A and B meet, A (B) defines its conditional intermeeting time with destination D as the time it takes to meet with D right after meeting[1] Some algorithms ([9], [17]) use transitivity to reflect the effect of other nodes on the delivery capability of a node but this update can be applied for all delivery metrics and it does not reflect the metric's own feature.

B (A). This updated definition of intermeeting time is also more convenient for the context of message routing because the messages are received from a node and given to another node on the way towards the destination. Here, conditional intermeeting time represent the period over which the node holds the message.

To show the benefits of the proposed metric, we adopted it for the shortest path based routing algorithms [7] [10] designed for DTN's. We propose *conditional shortest path routing* (CSPR) protocol in which average conditional intermeeting times are used as link costs rather than standard[2] intermeeting times and the messages are routed over conditional shortest paths (CSP). We compare CSPR protocol with the existing shortest path (SP) based routing protocol through real-trace-driven simulations. The results demonstrate that CSPR achieves higher delivery rate and lower end-to-end delay com-pared to the shortest path based routing protocols. This shows how well the conditional intermeeting time represents inter-node link costs (in the context of routing) and helps making effective forwarding decisions while routing a message.

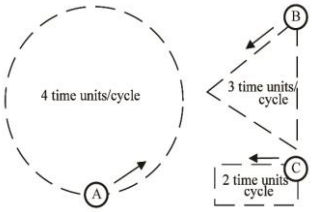
The rest of the paper is organized as follows. In Section II, the proposed metric is described in detail. In Section III, we present CSPR protocol and in Section IV, we give the results of real-trace-driven simulations. Finally, we provide conclusions and outline the future work in Section V.

II. CONDITIONAL INTERMEETING TIME

An analysis of real mobility traces has been done in different environments (office [13], conference [20], city [23], skating tour [14]) with different objects (human [11], bus [12], zebra [24]) and with variable number of attendants and led to significant results about the aggregate and pair wise mobility characteristics of real objects. Recent analysis [13] [14] [20] on real mobility traces have demonstrated that models assuming the exponential distribution of intermeeting times between pairs of nodes do not match real data well. Instead up to 99% of intermeeting times in many datasets is log-normal distribution. This makes the pair wise contacts between nodes depend on their pasts. Such a finding invalidates a common assumption [8] [17] [25] that the pair wise intermeeting times are exponentially distributed and memory less. More formally, if X is the random variable representing the intermeeting time between two nodes, P (X > s + t | X > t) 6= P (X > s) for s, t > 0. Hence, the residual time until the next meeting of two nodes can be predicted better if the node knows that it has not met the other node for t time units [13].

To take advantage of such knowledge, we propose a new metric called *conditional intermeeting time* that measures the intermeeting time between two nodes relative to a meeting with a third node using only the local knowledge of the past contacts. Such measure is particularly beneficial if the nodes move in a cyclic so-called MobiSpace [15] in which if two nodes contact frequently at particular time in previous cycles,[2] We use the terms *conventional* and *standard* interchangeably while refer-ring to the commonly used intermeeting time metric.

Fig. 1. A physical cyclic MobiSpace with a common motion cycle of 12 time units.



they will probably be in contact around the same time in the next cycle. Consider the sample cyclic MobiSpace with three objects illustrated in Figure 1. The common motion cycle is 12 time units, so the discrete probabilistic contacts between A and B happen in every 12 time units (1, 13, 25, ...) and between B and C in every 6 time units (2, 8, 14, ...). The average intermeeting time between nodes B and C indicates that node B can forward its message to node C in 6 time units. However, the conditional intermeeting time of B with C relative to prior meeting of node A indicates that the message can be forwarded to node C within one time unit.

- In a DTN, each node can compute the average of its standard and conditional intermeeting times with other nodes using its contact history. In Algorithm 1, we show how a node, s, can compute these metrics from its previous meetings. The notations we use in this algorithm (and also throughout the paper) are listed below with their meanings:

- $\tau_A(B)$: Average time that elapses between two consecutive meetings of nodes A and B. Obviously when the node connections are bidirectional, $\tau_A(B) = \tau_B(A)$.
- $\tau_A(B|C)$: Average time it takes for node A to meet node B after it meets node C. Note that, $\tau_A(B|C)$ and $\tau_B(A|C)$ are not necessarily equal.
- S: N × N matrix where S(i, j) shows the sum of all samples of conditional intermeeting times with node j relative to the meeting with node i. Here, N is the neighbor count of current node (i.e. N (s) for node s).
- C: N × N matrix where C(i, j) shows the total number of conditional intermeeting time samples with node j relative to its meeting with node i.
- β_i: Total meeting count with node i.

In Algorithm 1, each node first adds up times expired between repeating meetings of one neighbor and the meeting of another neighbor. Then it divides this total by the number of times it has met the first neighbor prior to meeting the second one. For example, if node A has two neighbors (B and C), to find the conditional intermeeting time of $\tau_A(B|C)$, each time node A meets node C, it starts a different timer. When it meets node B, it sums up the values of these timers and divides the results by the number of active timers before deleting them. This computation is repeated again each time node B is encountered. Then, the total of times collected from each timer is divided by the total count of timers used, to find

Algorithm 1 update (node m, time t)

```
if m is seen first time then
firstTimeAt[m] ← t
else
increment β_m by 1
lastTimeAt[m] ← t
end if
for each neighbor j ∈ N and j ≠ m do
start a timer t_mj
end for
for each neighbor j ∈ N and j ≠ m do
for each timer t_jm running do
S[j][m] += time on t_jm
increment C[j][m] by 1
end for
delete all timers t_jm
end for
for each neighbor i ∈ N do
for each neighbor j ∈ N and j ≠ i do
if S[j][i] ≠ 0 then
τ_s(i|j) ← S[j][i] / C[j][i]
end if
end for
τ_s(i) ← (lastTimeAt[i] − firstTimeAt[i] ) / β_i
end for
```

the value of $\tau_A(B|C)$.

While computing standard and conditional intermeeting times, we ignore the edge effects [12] by including intermeeting times of atypical meetings. That means that we include the values of $\tau_A(B)$ for the first and last meetings of node B with node A. Likewise, we include the values of $\tau_A(B|C)$ for the first meeting of node A with node C and the last meeting of that node with node B. Although this may change the results, this change will be negligible if long enough time passed to collect many other meeting data. The drawback of this computation can also be minimized either by updating the computation

by including the edge effects as in [12] or by keeping an appropriate size of window of the past contacts [10].

Consider the sample meeting times of a node A with its neighbors B and C in Figure 2. Node A meets with node B at times {5, 16, 25, 30} and with node C at times {11, 14, 23, 34}. Following the procedure described above, we find that $\tau_A(B|C) = (5 + 2 + 2)/3 = 3$ time units and $\tau_A(C|B) = (6 + 7 + 4 + 9)/4 = 6.5$ time units.

III. Conditional Shortest Path Routing

A. Overview

Shortest path routing protocols for DTN's are based on the designs of routing protocols for traditional networks. Messages are forwarded through the shortest paths between source and destination pairs according to the costs assigned to links between nodes. Furthermore, the dynamic nature of DTN's is also considered in these designs. Two common metrics used to

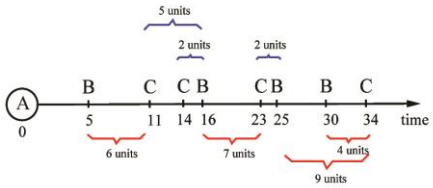


Fig. 2. Sample meeting times of node A with nodes B and C . While the values in the upper part are used in the computation of τ_A (B|C), the values in the lower part are used in the computation of τ_A (C |B).

define the link costs are minimum expected delay (MED [7]) and minimum estimated expected delay (MEED [10]). They compute the expected waiting time plus the transmission delay between each pair of nodes. However, while the former uses the future contact schedule, the latter uses only observed contact history.

Routing decisions can be made at three different points in an SP based routing: i) at source, ii) at each hop, and iii) at each contact. In the first one (source routing), SP of the message i s decided at the source node and the message follows that path. In the second one (per-hop routing), when a message arrives at an intermediate node, the node determines the next hop for the message towards the destination and the message waits for that node. Finally, in the third one (per-contact routing), the routing table is recomputed at each contact with other nodes and the forwarding decision is made accordingly. In these algorithms, utilization of recent information increases from the first t o the last one so that better forwarding decisions are made; however, more processing resources are used as the routing decision is computed more frequently.

B. Network Model

We model a DTN as a graph G = (V , E) where the vertices (V) are mobile nodes and the edges (E) represent the connections between these nodes. However, different from previous DTN network models [7] [10], we assume that there may be multiple unidirectional (E_u) and bidirectional (E_b) edges between the nodes. The neighbors of a node i are denoted with N (i) and the edge sets are given as follows:

$$
\begin{aligned}
E &= E_u \cup E_b \\
E_b &= \{(i,j) \mid \forall j \in N(i)\} \text{ where, } w(i,j) = \tau_i(j) = \tau_j(i) \\
E_u &= \{(i,j) \mid \forall j,k \in N(i) \text{ and } j \neq k\} \text{ where,} \\
&\quad w(i,j) = \tau_i(j|k)
\end{aligned}
$$

The above definition of E_u allows for multiple unidirectional edges between any two nodes. However, these edges differ from each other in terms of their weights and the corresponding third node. This third node indicates the previous meeting and is used as a reference point while defining the conditional intermeeting time (weight of the edge). In Figure 3, we illustrate a sample DTN graph with four nodes and nine edges. Of these nine edges, three are bidirectional with weights of standard intermeeting times between nodes, and six are

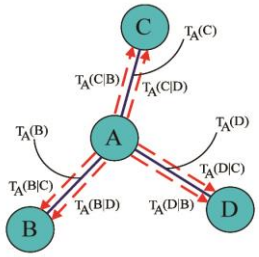


Fig. 3. The graph of a sample DTN with four nodes and nine edges in total.

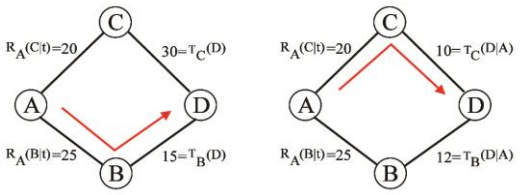


Fig. 4. The shortest path from a source to destination node can be different when conditional intermeeting times are used as the weights of links in the network graph.

Unidirectional edges with weights of conditional intermeeting times.

C. Conditional Shortest Path Routing

Our algorithm basically finds conditional shortest paths (CSP) for each source-destination pair and routes the messages over these paths. We define the CSP from a node n_0 to a node n_d as follows:

$$CSP(n_0, n_d) = \{n_0, n_1, \ldots, n_{d-1}, n_d \mid \Re_{n_0}(n_1|t) + \sum_{i=1}^{d-1} \tau_{n_i}(n_{i+1}|n_{i-1}) \text{ is minimized.}\}$$

Here, t represents the time that has passed since the last meeting of node n_0 with n_1 and $\Re_{n0}$ $(n_1|t)$ is the expected residual time for node n_0 to meet with node n_1 given that they have not met in the last t time units. $\Re_{n0}$ $(n_1|t)$ can be computed as in [13] with parameters of distribution representing the intermeeting time between n_0 and n_1. It can also be computed in a discrete manner from the contact history of n_0 and n_1. Assume that node i observed d intermeeting times with node j in its past. Let τ_i^1 (j), τ_i^2 (j),. . .τ_i^d (j) denote these values. Then:

$$\Re_i(j|t) = \frac{\sum_{k=1}^{d} f_i^k(j)}{|\{\tau_i^k(j) \geq t\}|} \text{ where, } f_i^k(j) = \begin{cases} \tau_i^k(j) - t & \text{if } \tau_i^k(j) \geq t \\ 0 & \text{otherwise} \end{cases}$$

Here, if none of the d observed intermeeting times is bigger than t (this case occurs less likely as the contact history grows), a good approximation can be to assume $\Re_i$ (j|t) = 0.

We will next provide an example to show the benefit of CSP over SP. Consider the DTN illustrated in Figure 4. The weights of edges (A, C) and (A, B) show the expected residual time of node A with nodes C and B respectively in both graphs. But the weights of edges (C, D) and (B, D) are different in both graphs. While in the left graph, they show the average intermeeting times of nodes C and B with D respectively, in the right graph, they show the average conditional intermeeting times of the same nodes with D relative to their meeting with node A. From the left graph, we conclude that SP(A, D) follows (A, B, D). Hence, it is expected that on average a message from node A will be delivered to node D in 40 time units. However this may not be the actual shortest delay path. As the weight of edge (C, D) states in the right graph, node C can have a smaller conditional intermeeting time (than the standard intermeeting time) with node D assuming that it has met node A. This provides node C with a faster transfer of the message to node D after meeting node A. Hence, in the right graph, CSP(A, D) is (A, C, D) with the path cost of 30 time units.

Each node forms the aforementioned network model and collects the standard and conditional intermeeting times of other nodes between each other through epidemic link state protocol as in [10]. However, once the weights are known, it is not as easy to find CSP's as it is to find SP's. Consider Figure 5 where the CSP(A, E) follows path 2 and CSP(A, D) follows (A, B, D). This situation is likely to happen in a DTN, if τ_D (E|B) $\geq \tau_D$ (E|C) is satisfied. Running Dijkstra's or Bellman-ford algorithm on the current graph structure cannot detect such cases and concludes that CSP(A, E) is (A, B, D, E). Therefore, to obtain the correct CSP's for each source destination pair, we propose the following transformation on the current graph structure.

Given a DTN graph G = (V, E), we obtain a new graph G′ = (V ′, E′) where:

$$V' \subseteq V \times V \text{ and } E' \subseteq V' \times V' \text{ where,}$$
$$V' = \{(i_j) \mid \forall j \in N(i)\} \text{ and } E' = \{(i_j, k_l) \mid i = l\}$$
$$\text{where, } w'(i_j, k_l) = \begin{cases} \tau_i(k|j) & \text{if } j \neq k \\ \tau_i(k) & \text{otherwise} \end{cases}$$

Note that the edges in E_b (in G) are made directional in G′ and the edges in E_u between the same pair of nodes are separated in E′. This graph transformation keeps all the historical information that conditional intermeeting times require and

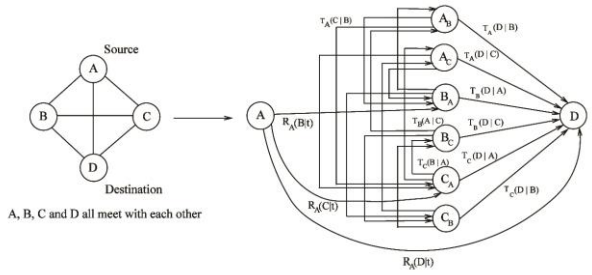


Fig. 6. Graph Transformation to solve CSP with 4 Nodes where A is the source and D the destination node.

also keeps only the paths with a valid history. For example, for a path A, B, C, D in G, an edge like (C_D , D_A) in G′ cannot be chosen because of the edge settings in the graph. Hence, only the correct τ values will be added to the path calculation. To solve the CSP problem however, we add one vertex for source S (apart from its permutations) and one vertex for destination node D. We also add outgoing edges from S to each vertex (i_S) ∈ V ′ with weight $\Re_S$ (i|t). Furthermore, for the destination node, D, we add only incoming edges from each vertex i_j ∈ V ′ with weight τ_i (D|j).

In Figure 6, we show a sample transformation of a clique of four nodes to the new graph structure. In the initial graph, all mobile nodes A to D meet with each other, and we set the source node to A and destination node to D (we did not show the directional edges in the original graph for brevity). It can be seen that we set any path to begin with A on transformed graph G′, but we also put the permutations of A, B and C with each other.

Running Dijkstra's shortest path algorithm on G′ given the source node S and destination D will give CSP. In G′, $|V'| = O(|V|^2)$ and $|E'| = O(|V^3|) = |E|^{3/2}$. Therefore Dijkstra's algorithm will run in $O(|V|^3)$ (with Fibonacci heaps) while computing regular shortest paths (where edge costs are standard intermeeting times) takes $O(|V|^2)$.

The focus of this paper is an improvement of the current design of the Shortest Path (SP) based DTN routing algorithms. Therefore we leave the elaborate discussion of some other issues in SP based routing (complexity, scalability and routing type selection) to the original studies [7] [10]. Using conditional instead of standard intermeeting times requires extra space to store the conditional intermeeting times and additional processing as complexity of running Dijkstra's algorithm increases from $O(|V|^2)$ to $O(|V|^3)$. We believe that in current DTN's, wireless devices have enough storage and processing power not to be unduly taxed with such an increase. Moreover, to lessen the burden of collecting and storing link weights, an asynchronous and distributed version of the Bellman-Ford algorithm can be used, as described in [21].

IV. SIMULATIONS

To evaluate the performance of our algorithm, we have built a discrete event simulator in Java. In this section, we describe the details of our simulations through which we compare the proposed *Conditional Shortest Path Routing* (CSPR) algorithm with standard *Shortest Path Routing* (SPR). Moreover, in our results we also show the performance of upper and lower performance boundaries with Epidemic Routing [4] and Direct Delivery.

We used two different data sets: 1) RollerNet traces [14] which includes the opportunistic contacts of 62 iMotes which are distributed to the rollerbladers during a 3 hour skating tour of Paris on August 20, 2006, 2) Cambridge Dataset [23] which includes the Bluetooth contacts of 36 students from Cambridge University carrying iMotes around the city of Cambridge, UK from October 28, 2005 to December 21, 2005.

To collect several routing statistics, we have generated traffic on the traces of these two data sets. For a simulation run, we generated 5000 messages from a random source node to a random destination node at each t seconds. In RollerNet, since the duration of experiment is short, we set t = 1s, but for Cambridge data set, we set t = 1 min. We assume that the nodes have enough buffer space to store every message they receive, the bandwidth is high and the contact durations of nodes are long enough to allow the exchange of all messages between nodes. These assumptions are reasonable in today's technology and are also used commonly in previous studies [22]. Moreover, we compare all algorithms in the same conditions, and a change in the current assumptions is expected to affect the performance of them in the same manner. We ran each simulation 10 times with different seeds but the same set of messages and collected statistics after each run. The results plotted in Figures 7 and 8 show the average of results

obtained in all runs.

Figure 7 shows the delivery rates achieved in CSPR and SPR algorithms with respect to time (i.e. TTL of messages) in RollerNet traces. Clearly, CSPR algorithm delivers more while it is around 16 in epidemic routing.

essages than SPR algorithm. Moreover, it achieves lower average delivery delay than SPR algorithm. For example, CSPR delivers 80% of all messages after 17 min with an average delay of almost 6 min, while SPR can achieve the same delivery ratio only after 41 min and with an average delay of 12 min. Although we did not show it here for brevity, the average numbers of times the delivered messages are forwarded (number of hops) in SPR and CSPR are very close (1.48 and 1.52 respectively) to each other (and much smaller than the average number of times a message is forwarded in epidemic routing which is around 25).

In Figure 8, we show the delivery rates achieved in Cam-bridge traces. As before, CSPR algorithm achieves higher delivery ratio than SPR algorithm. After 6 days, CSPR delivers 78% of all messages with an average delay of 2.6 days, while SPR can only deliver 62% of all messages with an average delay of 3.2 days. The numbers of times a message is forwarded in SPR and CSPR are 1.73 and 1.78, These results show that the conditional intermeeting time represents link cost better than the standard intermeeting time. Therefore, in CSPR, more effective paths with similar average hop counts are selected to route messages. Consequently, higher delivery rates with lower end-to-end delays are achieved. In SPR and CSPR algorithms here, we used source-routing and let the messages follow the paths which are decided at the source nodes. We also observed similar results in our simulations when other routing types (per-hop and per-contact) are used.

V. CONCLUSION AND FUTURE WORK

In this paper, we introduced a new metric called conditional intermeeting time inspired by the results of the recent studies showing that nodes' intermeeting times are not memoryless and that motion patterns of mobile nodes are frequently repetitive. Then, we looked at the effects of this metric on shortest path based routing in DTN's. For this purpose, we updated the shortest path based routing algorithms using conditional intermeeting times and proposed to route the messages over conditional shortest paths. Finally, we ran

respectively,

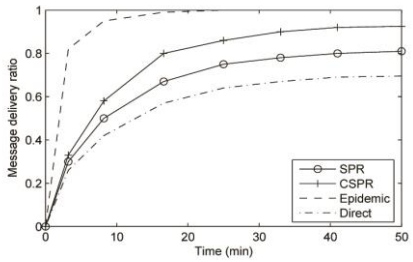


Fig. 7. Message delivery ratio vs. time in RollerNet traces.

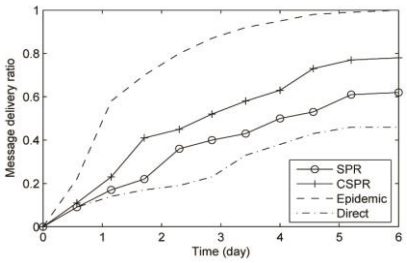


Fig. 8. Message delivery ratio vs. time in Cambridge traces.

simulations to evaluate the proposed algorithm and demonstrated the superiority of CSPR protocol over the existing shortest path routing algorithms.

In future work, we will look at the performance of the proposed algorithm in different data sets to see the effect of conditional intermeeting time in different environments. Moreover, we will consider extending our CSPR algorithm by using more information (more than one known meetings) from the contact history while deciding conditional intermeeting times. For this, we plan to use probabilistic context free gram-mars (PCFG) and utilize the construction algorithm presented in [26]. Such a model will be able to hold history information concisely, and provide further generalizations for unseen data.

REFERENCES

[1] *Delay tolerant networking research group*, http://www.dtnrg.org.

[2] T. Spyropoulos, K. Psounis,C. S. Raghavendra, *Efficient routing in inter-mittently connected mobile networks: The single-copy case*,

IEEE/ACM Transactions on Networking, vol. 16, no. 1, Feb. 2008.

[3] J. Burgess, B. Gallagher, D. Jensen, and B. N. Levine, *MaxProp: Routing for Vehicle-Based Disruption- Tolerant Networks*, In Proc. IEEE Infocom, April 2006.

[4] A. Vahdat and D. Becker, *Epidemic routing for partially connected ad hoc networks*, Duke University, Tech. Rep. CS-200006, 2000.

[5] T. Spyropoulos, K. Psounis,C. S. Raghavendra, *Efficient routing in inter-mittently connected mobile networks: The multi-copy case*, IEEE/ACM Transactions on Networking, 2008.

[6] Y. Wang, S. Jain, M. Martonosi, and K. Fall, *Erasure coding based routing for opportunistic networks*, in Proceedings of ACM SIGCOMM workshop on Delay Tolerant Networking (WDTN), 2005.

[7] S. Jain, K. Fall, and R. Patra, *Routing in a delay tolerant network*, in Proceedings of ACM SIGCOMM, Aug. 2004.

[8] T. Spyropoulos, K. Psounis,C. S. Raghavendra, *Spray and Wait: An Efficient Routing Scheme for Intermittently Connected Mobi le Networks*, ACM SIGCOMM Workshop, 2005.

[9] A. Lindgren, A. Doria, and O. Schelen, *Probabilistic routing in in-termittently connected networks*, SIGMOBILE Mobile Computing and Communication Review, vol. 7, no. 3, 2003.

[10] E. P. C. Jones, L. Li, and P. A. S. Ward, *Practical routing in delay tolerant networks*, in Proceedings of ACM SIGCOMM workshop on Delay Tolerant Networking (WDTN), 2005.

[11] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, and J. Scott, *Impact of Human Mobility on the Design of Opportunistic Forwarding Algorithms*, in Proceedings of INFOCOM, 2006.

[12] X. Zhang, J. F. Kurose, B. Levine, D. Towsley, and H. Zhang, *Study of a Bus-Based Disruption Tolerant Network: Mobility Modeling and Impact on Routing*, In Proceedings of ACM MobiCom, 2007.

[13] S. Srinivasa and S. Krishnamurthy, *CREST: An Opportunistic Forward-ing Protocol Based on Conditional Residual Time*, in Proceedings of IEEE SECON, 2009.

[14] P. U. Tournoux, J. Leguay, F. Benbadis, V. Conan, M. Amorim, J. Whitbeck, *The Accordion Phenomenon: Analysis, Characterization, and Impact on DTN Routing*, in Proceedings of Infocom, 2009.

[15] C. Liu and J. Wu, *Routing in a Cyclic Mobispace*, In Proceedings of ACM Mobihoc, 2008.

[16] H. Dubois-Ferriere, M. Grossglauser, and M. Vetterli, *Age Matters: Efficient Route Discovery in Mobile Ad Hoc Networks Using Enc ounter Ages*, In Proceedings of ACM MobiHoc, 2003.

[17] T. Spyropoulos, K. Psounis, and C. Raghavendra, *Spray and Focus: Efficient Mobility-Assisted Routing for Heterogeneous and Correlated Mobility*, In Proceedings of IEEE PerCom, 2007.

[18] E. Daly and M. Haahr, *Social network analysis for routing in discon-nected delay-tolerant manets*, In Proceedings of ACM MobiHoc, 2007.

[19] P. Hui, J. Crowcroft, and E. Yoneki, *BUBBLE Rap: Social Based Forwarding in Delay Tolerant Networks*, In Proc. of ACM MobiHoc, 2008.

[20] A European Union funded project in Situated and Autonomic Commu-nications, *www.haggleproject.org*.

[21] D. Bertsekas, and R. Gallager, *Data networks (2nd ed.)*, 1992.

[22] C. Liu and J. Wu, *An Optimal Probabilistically Forwarding Protocol in Delay Tolerant Networks*, in Proceedings of MobiHoc, 2009.

[23] J. Leguay, A. Lindgren, J. Scott, T. Friedman, J. Crowcroft and P. Hui, *CRAWDAD data set upmc/content (v. 2006-11-17)*, downloaded from http://crawdad.cs.dartmouth.edu/upmc/content, 2006.

[24] Y. Wang, P. Zhang, T. Liu, C. Sadler and M. Martonosi, *http://crawdad.cs.dartmouth.edu/princeton/zebranet*, CRAWDAD data set princeton/zebranet (v. 2007-02-14), 2007.

[25] E. Bulut, Z. Wang, and B. Szymanski, *Cost-Effective Multi-Period Spraying for Routing in Delay Tolerant Networks*, to appear in IEEE/ACM Transactions on Networking, 2010.

[26] S. Geyik, and B. Szymanski, *Event Recognition in Sensor Networks by Means of Grammatical Inference*, in Proceedings of INFOCOM, 2009.