

Cluster-based Forwarding in Delay Tolerant Public Transport Networks

Shabbir Ahmed and Salil S. Kanhere

School of Computer Science and Engineering, The University of New South Wales
Sydney, Australia

Email: {shabbira,salilk}@cse.unsw.edu.au

Abstract—Packet forwarding in Public Transport Networks is particularly challenging due to the high mobility, rapidly changing topology and intermittent connectivity observed in these networks. Though clustering of nodes can aid forwarding decision in these Delay Tolerant Networks (DTNs), the clustering process is extremely costly in a large network. In this paper, we introduce a generic efficient clustering method which is suitable for grouping the nodes of large networks. We also demonstrated how encounter frequencies of public transport networks can be fed to that clustering algorithm in order to build clusters of nodes. And finally, our large scale extensive simulation study on real bus traces shows the efficacy of clustering in packet forwarding.

I. INTRODUCTION

Delay Tolerant Networks (DTN) [8] are an emerging class of networks wherein the underlying network between the communicating devices is not fully connected. Consequently, it is very rare that an explicit end-to-end path is available between the source and destination. Rather the topology is made up of intermittently and partially connected node clusters. DTN rely on the inherent mobility of the participating nodes to *store-carry-and forward* [4] the messages for delivery to the destination. DTN are particularly attractive for use in areas where there is no communication infrastructure due to harsh environments (battlefields, forests, space) or economic conditions (rural and remote areas, developing countries). Of course, such networks are suitable for running applications, which can tolerate large delays such as transferring files and non-interactive messaging. Two particularly attractive instances of DTN include (i) *Pocket Switched Networks* [10], wherein personal communication devices carried by humans self-organise to form an intermittently connected network, enabling a new class of social networking applications (e.g.: PeopleNet [16]) and (ii) Vehicular DTN, which can leverage the large data storage and energy capabilities offered by vehicles to create a large-scale powerful DTN. Examples include the use of vehicle-based DTN to provide low cost digital communication to remote villages [17] and vehicular sensing platforms such as CarTel [7] for urban monitoring.

Public Transport Networks (PTN) such as trains, buses and ferries offer a particularly suitable platform for deploying a large-scale DTN, primarily because their mobility patterns are repetitive and known *a priori*. Note that all public transport systems follow a pre-defined timetable (with some minor delays) and travel along planned routes. Further, these schedules are repeated with a certain periodicity (usually daily). This

information about the mobility behavior of the vehicles can be exploited to instrument effective packet forwarding algorithms in such networks. In addition to engineering a mobile sensing platform, the PTN can also extend the connectivity of PSN, for example, two nearby buses in the radio range of one another, allow the individual PSN in each bus to exchange messages with each other.

Routing is one of the most challenging aspects of DTN. Since there are no end to end path between source and destination in a DTN, the conventional ad-hoc routing protocols do not work effectively. In recent years, there has been significant research in DTN routing, a detailed overview of which can be found in [15]. A particularly effective approach involves exploiting any available knowledge about the node mobility patterns [12]. In this paper, we present one approach, wherein the known mobility of PTNs is used to cluster the buses into groups that share some common features. The forwarding process then uses the cluster labels to find an effective next hop node. We will focus on a bus network as a representative example of a PTN.

Clustering can be done based on various shared properties of the nodes. In a bus network, some of the potential shared features of the nodes (buses) that be used in the classification purpose are:

- a) Buses which meet each other frequently (i.e. a high encounter frequency). Additional knowledge as contact duration and intermeeting duration can be used to guess the link quality between two neighbors.
- b) Buses which travel along the same physical routes [3].
- c) Buses which move frequently in high density areas (hotspots such as) where potential data transfer can occur.

In this paper, we use the encounter frequencies (i.e. the number of contact opportunities) of buses to find similarities among them. The rationale is that buses which meet each other more frequently (i.e. higher encounter frequency) are probably suitable forwarders for each other. Based on the encounter frequencies among the buses, an abstract cluster can be formed. A node in this cluster has better delivery predictability than nodes in other clusters when the destination node also belongs to this cluster. However, using traditional graph theoretic approaches to form clusters based on encounter frequencies for a large network such as a PTN is not trivial and computationally intensive. In this paper, we have also

shown how dynamic programming methods can be applied to reduce computational complexity in cluster formation. Our cluster formation algorithm is tunable enabling variable size and number of clusters and has the ability produce overlapping clusters (i.e. a single bus may belong to multiple clusters). We evaluate the effectiveness of our clustering technique and the gains achieved in improving packet forwarding using empirical bus traces from a real-world public transport system. It should be noted that Hui et al. [11] have recently demonstrated the effectiveness of cluster-based forwarding in DTN using data from a real-world human mobility scenario. However, the clustering technique used in their work is fairly trivial. As discussed in this paper, large-scale DTN such as a PTN require significantly advanced techniques for node classification.

To summarize, we make the following contributions:

- As a first step to forming clusters based on encounter frequencies, we formulate a generic graph model of the DTN. Graph theoretic algorithms to find network covers can readily be applied to build clusters from our generic graph model.
- Next, we propose a novel technique to reduce the complexity of forming clusters by an order of magnitude using dynamic programming methods.
- And finally, we demonstrate the cluster formation techniques and the resulting improvements in packet forwarding using real-world bus mobility traces.

We shall use the term node and bus interchangeably throughout the paper. Also we shall use the term packet and message synonymously. The rest of the paper is organized as follows: Section II provides an overview of related works. Section III briefly introduces the bus traces that we have used in our evaluation. Section IV presents our grouping algorithm and problem mapping. Section V briefly discusses our evaluation methodology. Experimental results are provided in Section VI. Finally, Section VII concludes the paper.

II. RELATED WORKS

In this section we provide a brief overview of related work in classifying and analyzing node movement patterns in DTN.

In recent years there have been several attempts at collecting and characterizing human mobility data using small mobile devices such as iMotes and PDAs [10]. Su et al. [19] have collected traces of pair-wise connections by providing PDAs to a group of students. They show that it is feasible to build delay-tolerant networks based on human mobility. They also share their experience in collecting mobility traces and in solving some challenges regarding power management. The authors in [10] have gathered human mobility data in a conference setting by providing Bluetooth enabled devices to 54 participants. They introduce the notion of Pocket Switched Networks that makes use of both human mobility and wireless connectivity among other devices in order to route messages. In [5], based on the findings from [10], the researchers have demonstrated that current mobility models (such as: random way-point, uniformly distributed locations) do not exhibit characteristics

that correspond to real life mobility patterns. They also suggest that the empirically collected movement patterns can be utilized to develop knowledge-based opportunistic forwarding algorithms.

The frequency and duration of encounters between mobile nodes can play an important role in effectively choosing the next hop forwarding node in DTN. Encounter duration determines the opportunity to transfer messages between two nodes. A neighbor whose encounter duration is small should not be chosen as a forwarder since the data transfer opportunity might be too small to transfer complete message. Further, nodes which have a higher encounter frequency with the destination node are particularly attractive choices for forwarding. In [6], [9], the researchers use the information about encounter ages and last encounter to forward a packet towards destination.

In [11], the authors use small labels to aid packet forwarding in pocket switched networks. The idea is based on a simple intuition that people belonging to the same community are likely to meet frequently, thus acting as suitable forwarders for messages destined to members belonging to the same community in a pocket switched network. People belonging to the same community are simply identified by a unique small community label, which is then used as the basis for forwarding packets. The authors have demonstrated the improvements achieved by using small label identifiers using empirical results from a conference, wherein a sub-set of the participants were naturally formed into subgroups according to academic affiliations. Their work is the closest to our research. In [3], the authors classify public transport buses according to bus routes and used this information for packet forwarding. The experiments demonstrate that even such a basic classification process results in a significant reduction in routing overheads. Even though the concept of clustering has been shown to improve forwarding performance in a DTN, the classification of nodes into groups is quite trivial (based on affiliation of humans in [11] and bus routes in [3]). In this paper we present an efficient clustering technique for a large-scale DTN. The ability to form overlapping clusters, variable cluster size and number are some of the features of our cluster formation algorithm.

In [4], the authors propose a technique which attempts to learn the mobility pattern of mobile nodes and use this information for effective forwarding. The effectiveness of their scheme was demonstrated by implementing the protocol in a small-scale bus network consisting of 30 buses. In this paper, we focus on a much larger DTN consisting of over 1000 buses.

III. DETAILS OF BUS MOVEMENT TRACES

Before discussing our clustering algorithm and packet forwarding mechanism, we first elaborate on the empirical public transport movement traces that we have used in our work.

We have used the mobility traces of buses in the King County Metro bus system in Seattle, USA [13]. This public transport system consists of close to 1163 buses plying over 236 distinct bus routes covering an area of 5100 square kilometers. The traces were collected over a two week period

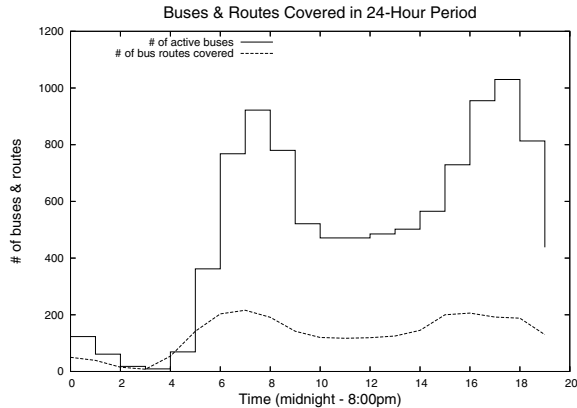


Fig. 1. Seattle bus numbers and routes during 24-hours period

in November 2001. The traces are based on location update messages sent by each bus. Each bus logs its current location (using GPS), its bus id and route id along with a timestamp. The usual frequency at which an individual bus sends an update is once in every 30 seconds. However there are several instances where we found that some of the updates were missing.

One of the problems we have faced is the coarse granularity of the logged data (one update every 30 seconds or more). Assuming an average speed of 40 km/hr (for city driving), a bus can cover a distance of 330 meters in 30 seconds. This implies that on average we only know of the location of a bus after it has traveled 330 meters. If we use the traces *as-is*, assuming a radio range of 250m (consistent with current 802.11 radios), it is highly likely that several inter-bus contacts are excluded from our analysis. Further, those inter-bus contact durations that can be measured will only be represented at a granularity of 30 seconds. This will lead to severe over-estimation of these contact durations especially given that these durations are fairly short. To generate fine-grained location information, we fed the traces into ns-2 and extrapolated the location of the buses every 5 seconds. For this we assumed that a bus moves at a constant speed along a straight line between two consecutive logged entries. Though this may not be an entirely accurate representation of the bus motion, we believe that this assumption is a reasonable approximation especially for urban areas. We have used these modified fine-grained movement traces in our analysis. We assume that every 5 seconds each bus broadcasts a beacon indicating its location coordinates. This ensures that the buses have an up-to-date view of their neighborhood. We use a small random back-off interval to avoid collisions.

From the Seattle bus traces, we observe that the number of buses follow a highly repeatable daily pattern over all weekdays (Fig. 1). This is expected since the weekday timetables for public-transport vehicles are identical. Fig. 1 shows the average bus density and routes covered over a 24 hour period. As seen, one can broadly identify four regions (i.e. time-slots) of interest: a) from midnight to 6am, b) 6am to 9am, c) 9am to

3pm and d) 3pm to 8pm. There are no buses plying from 8 pm to midnight. The time slots 6am-9am and 3pm-8pm exhibit peak operation behavior with a large number of buses and routes being operational. The time slot 9am-3pm indicates off-peak behavior while the midnight-6am duration can be labeled as super off-peak with the lowest number of buses being active. In our analysis, we have found that this behavior is repeated over all weekdays. We have excluded the weekend traces from our analysis.

IV. PROBLEM FORMULATION AND GROUPING ALGORITHM

Before we discuss the details of the clustering algorithm, we briefly discuss our network model.

A. Modeling the Problem

Let us assume that we want to classify the buses based on their meeting frequencies. Let $G = (V, E, W)$ be a weighted undirected graph with vertex set V , edge set E and weight set W . A straight forward way to map ad-hoc bus network into a static graph is to consider buses as vertex set V , direct reachability with other nodes as E and the number of encounters (or meetings) as W . Let the objective function be f which indicates the similarity between two clusters. In our problem (where W is the set of encounter frequencies), the objective function is $\max\{M_1, M_2, \dots, M_k\}$ where M_i is the maximum weight of an edge whose endpoints are vertices in cluster C_i . Intuitively, we put two nodes in a group if they have maximum similarity between them (indicated by the encounter frequency between them). Unfortunately, using graph theoretic approach (e.g. finding network cover) to build clusters in public transport networks is practically infeasible because of the large number of buses and the exponential nature of the complexity of the algorithms. Later in this section we have shown how to use dynamic programming approach to tackle the scalability problem in finding clusters. We use the term objective function and distance function synonymously throughout the paper.

B. Hierarchical approach

Let us begin by depicting the generic bottom-up hierarchical clustering algorithm defined by S. C. Johnson in 1967 [14]. We have chosen this greedy clustering algorithm as the basis to classify buses in public transport networks because it is straightforward to control the cluster size and cluster number which is needed in our bus classification problem.

Algorithm 1 Generic Hierarchical Clustering Algorithm

Start by assigning each item to a cluster

repeat

 Merge best pair of clusters

 Compute the distance between new cluster and each of the old clusters

until all items are clustered

Initially, each leaf node of the tree is considered a cluster consisting of a single node. At each stage of the algorithm,

two best nodes (based on distance function) are merged to form a new cluster. Merging and the updating of the distance matrix define the behavior of the algorithm. We can control the number of clusters by restricting the iteration of the algorithm. However, the main problem of hierarchical approach for cluster formation is its mutually exclusiveness in other words, the clusters formed in this way are non-overlapping. In our problem of classifying buses, a bus can be a member of two or more clusters; so we need to modify the generic hierarchical approach to suit our purpose.

C. Solving mutually exclusive partitioning problem

Instead of forming a single-node cluster at a leaf, we form a two-node cluster there. Each node is paired with its neighbors to form a two-node cluster at leaf level. Let, $V = \{v_1, v_2, \dots, v_n\}$ be the set of nodes (buses) in our network and B_i (where $B_i \in V$) be the set of neighbors of a_i . Then, $\forall j = 1 \dots |B_i|$, we make groups consisting of $\forall i \{v_i, B_{ij}\}$ at leaf level. So at leaf, a node that has a large number of neighbors appears in many leaf nodes. In this way, a node has the opportunity to appear in other clusters as well.

D. Calculation of Distance function

An important component of a clustering algorithm is measuring the distance or similarity between data points (nodes in our case). One of the most common distance measures is Euclidean distance. However, Euclidean distance does not make any sense to indicate the closeness between two clusters because encounter frequencies rather than nodes coordinates have been used to calculate similarity between two nodes in our network. In our case, coordinate space does not reflect real locations. We use the encounter frequency between two buses during a period as a distance measure. Let, D be the adjacency matrix which is initially populated from W (set of weights of edges of the graph). For each $v_i \rightarrow v_j$ ($i \neq j$), D_{ij} contains the weight of the edges W_{ij} (which is in this case the encounter frequencies). The distance between any two clusters M and N can be approximated from the following relation:

$$\frac{1}{|M| \times |N|} \sum_{i=1, i \neq j}^{|M|} \sum_{j=1, j \neq i}^{|N|} D_{ij} \quad (1)$$

Intuitively, this is the average of many-to-many relationship among the members between the two groups.

E. Scalability Issues

From relation (1), we can see that it takes $O(n^2)$ operations to compute the distance value between a pair of clusters and it takes $O(n^3)$ operations to complete an iteration of the hierarchical algorithm (since the distance value must be computed between the newly merged node and all the other nodes). So, the complete execution of the algorithm will require $O(n^4)$ operations. Clearly, for a network with a large number of nodes, relation (1) is overwhelming and practically infeasible. In the next subsection, we show how the complexity of the hierarchical clustering algorithm can be reduced by an order by using dynamic programming method.

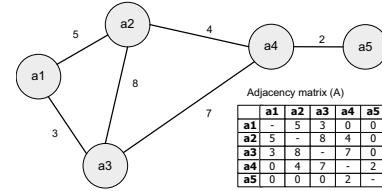


Fig. 2. Example network graph model

F. Solution using dynamic programming method

Fortunately the hierarchical partitioning problem exhibits the properties of *overlapping subproblems* and *optimal substructure* which make it suitable to be tackled by dynamic programming method. Dynamic programming makes use of *memoization* to avoid re-computing of optimal solutions to the subproblems it has already solved.

So, if we save the solutions of the subproblems and reuse it to solve bigger problems, then the computation complexity can be reduced significantly. We merge two groups at a time. Let P and Q be two groups to be merged to produce group R . The distance between R and any other group X ($X \neq P, X \neq Q$) will be:

$$D_{R,X} = \frac{1}{2} (D_{P,X} + D_{Q,X}) \quad (2)$$

Since the distance between $P \rightarrow X$ (denoted by $D_{P,X}$) and $Q \rightarrow X$ (denoted by $D_{Q,X}$) have already been calculated previously, we can directly use their value in equation (2) to calculate the distance between $R \rightarrow X$ and save the result for future use. In this way, we can calculate distance values with a complexity of $O(n)$ instead of $O(n^2)$ as in relation (1).

The following example illustrates the operation of our clustering algorithm.

G. Example

Fig. 2 shows an example graph model of five buses ($a1, a2, \dots, a5$). The weights of the edges represent the encounter frequencies which can be calculated from the previous statistics of bus movement pattern and the adjacency matrix A_{ij} is built.

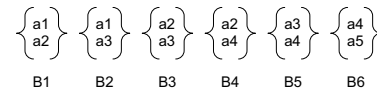


Fig. 3. Grouping at leaf level

1) *Step 1: Pair-wise grouping at leaf nodes:* Let, the new pair-wise groups among neighbors be $\{B1, B2, \dots, B6\}$ (Fig. 3).

2) *Step 2: Build distance matrix:* Build distance matrix D for $\{B1, B2, \dots, B6\}$ (Table I). For example, distance between $B1$ and $B5$ is (using relation (1)):

$$D_{B1,B5} = \frac{1}{4} \{A(a1, a3) + A(a1, a4) + A(a2, a3) + A(a2, a4)\}$$

And the distance between $B1$ and $B2$ is:

$$D_{B1,B2} = \frac{1}{3} \{A(a1, a3) + A(a2, a1) + A(a2, a3)\}$$

This ends the initialization part of our algorithm.

TABLE I
DISTANCE MATRIX D FOR $\{B1, B2, \dots, B6\}$

	B1	B2	B3	B4	B5	B6
B1	-	5.33	5.33	3	3.75	1
B2	5.33	-	5.33	5	3.33	1.75
B3	5.33	5.33	-	6.33	6.33	2.75
B4	3	5	6.33	-	6.33	2
B5	3.75	3.33	6.33	6.33	-	3
B6	1	1.75	2.75	2	3	-

3) Step 3: Looping: Find the $\max\{D_{ij}\}$ and merge B_i and B_j . In this case, either $\{B3, B4\}$ or $\{B3, B5\}$ can be merged because they both have maximum distance value (it is 6.33 in this case). Let us merge $\{B3, B4\}$ and label the new group as C1 (Fig. 4). The new distance matrix D that includes C1 is shown in Table II.

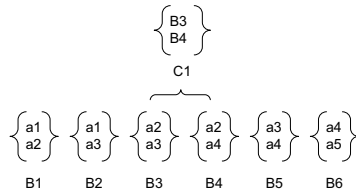


Fig. 4. Groups B3 and B4 are merged together and form group C1

TABLE II
UPDATED DISTANCE MATRIX D THAT INCLUDES C1

	B1	B2	B5	B6	C1
B1	-	5.33	3.75	1	4.16
B2	5.33	-	3.33	1.75	5.16
B5	3.75	3.33	-	3	6.33
B6	1	1.75	3	-	2.37
C1	4.16	5.16	6.33	2.37	-

Now the $\max\{D_{ij}\}$ is 6.33 and it is for $\{B5, C1\}$. So merge $\{B5, C1\}$ and label the new group as E1 (Fig. 5). Re-calculating the distance matrix D that includes E1, is shown in Table III.

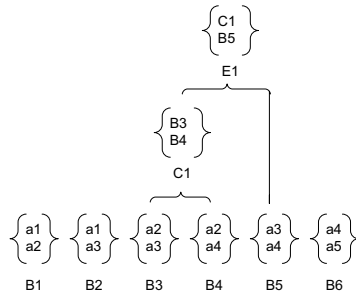


Fig. 5. Groups C1 and B5 are merged together and form group E1

Now the $\max\{D_{ij}\}$ is for B1 and B2. So we merge $\{B1, B2\}$ and give the new group to a new label F1 (Fig. 6). The updated distance matrix D is shown in Table IV.

TABLE III
UPDATED DISTANCE MATRIX D THAT INCLUDES E1

	B1	B2	B6	E1
B1	-	5.33	1	3.95
B2	5.33	-	1.75	4.24
B6	1	1.75	-	2.68
E1	3.95	4.24	2.68	-

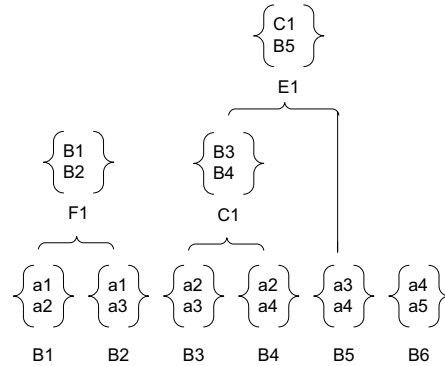


Fig. 6. Groups B1 and B2 are merged together and form group F1

From the distance matrix D , it can be clearly seen that the next best pairs to be merged are $\{E1, F1\}$. In this way, the process can be continued. The final dendrogram is shown in Fig. 7.

By controlling the iteration of the algorithm, we can control group size and group number. At any stage, the root nodes of the disjoint trees indicate the formed groups.

H. Forwarding Decision

Once we build group ids for each node, we can use these group ids to aid forwarding decision in several ways. A naive forwarding approach is that a packet is forwarded to a

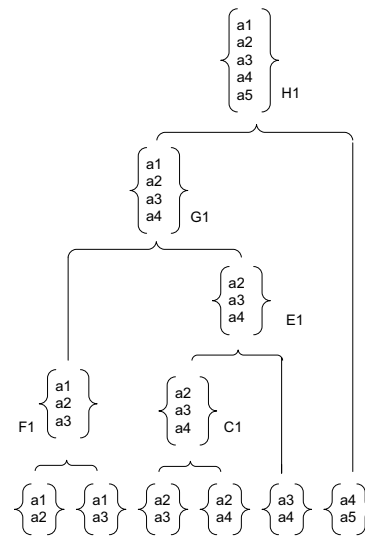


Fig. 7. Final dendrogram

TABLE IV
UPDATED DISTANCE MATRIX D THAT INCLUDES $F1$

	B6	E1	F1
B6	-	2.68	1.37
E1	2.68	-	4.09
F1	1.37	4.09	-

neighbor only if that neighbor belongs to the destination nodes group. Packet forwarding is done in epidemic fashion once the destination belongs to the same group of the forwarder.

V. EVALUATION

We use NS-2 [1] as our simulation tool. In order to deal with huge data efficiently, we have used PostgreSQL [2] as a backend database server. We have compared our forwarding by grouping approach with epidemic [20], wait [15] and spray-and-wait [18] forwarding methods. Epidemic and Wait are two extreme packet forwarding protocols. In epidemic routing, a forwarder forwards a packet to all unique neighbors it encounters whereas in Wait, a forwarder waits until it meets with the destination. Epidemic routing exhibits the best case delivery ratio with minimum delay whereas Wait shows minimum possible cost to deliver a packet. In a variation of Spray and Wait (which is a multicopy forwarding scheme), when a node makes a copy, it keeps half of its current copies for itself and instructs the next hop to distribute the rest [15]. When a forwarder receives a packet with only single copy instruction, it degenerates into Wait protocol.

Contrast to various other research papers, we use real bus movement traces of a large (spanning an area of 58km x 88km) public transport network [13]. Some details of these real bus movement traces have been discussed in section III.

A. Comparison metrics

The most important metric for a delay tolerant routing protocol is the delivery ratio which is the ratio of the messages delivered to the messages created. Though a large packet delay is common in delay tolerant networks, it is always desirable to receive a packet as early as possible. Delay and delivery cost are two other important metrics which we have used in comparing various delay tolerant routing protocols. Since these routing protocols are multi-copy schemes, the receiver may receive the same message more than once. While calculating delay of a received message, we have considered the received message which arrived first. For hop-count calculation, we use the received message which has the lowest hop-count. The message that arrives first at the destination and the message that takes lowest hop to reach the destination may not be the same message. A hop-count metric is also an indication of the delivery cost of a routing protocol.

We measure the delivery cost by counting the total number of messages transmitted and normalize it by dividing it by the total number of unique message created. If $\{Mt_1, Mt_2, \dots, Mt_n\}$ be the set of messages forwarded by the nodes during a period and $\{Mc_1, Mc_2, \dots, Mc_m\}$ be the set of message created by the

TABLE V
SIMULATION PARAMETERS

Parameter	Value
Beacon Interval	5sec
Radio Range	250m
Area Covered	58000m x 88000m
# of Buses	284
# of Groups	16
# of Source/Dest. pairs	50
MAC protocol	802.11
Antenna Type	Omni-directional
Radio Propagation Model	Two-Ray Ground

nodes, then the delivery cost d_c is defined as (3):

$$d_c = \frac{\sum_{i=1}^n Mt_i}{\sum_{j=1}^m Mc_j} \quad (3)$$

B. Methodology and Simulation parameters

We choose the bus traces of one of the peak working periods (from 6am-9am) (Fig. 1) for our simulation. We use 50 random source-destination pairs and start sending packets at $1000 + t_{rand}$ ($0 < t_{rand} < 3600$) after the simulation starts in order to ensure that all the buses in our trace file become active by this time. The whole simulation runs for 10800 seconds. We run the whole simulation five times with different set of source-destination pairs in order to get statistical significant result. We allocate sufficient number of packet buffers to each node in order to prevent packet losses due to packet queue overflow. The other simulation parameters can be found in Table I.

As the simulation time passes, we calculate delivery ratio and delivery cost. The plot of these indicates the growth rate of these metrics.

VI. RESULT

A. Delivery Ratio

Epidemic routing exhibits the upper bound on delivery ratio and delivery delay. The delivery ratio of our cluster-based forwarding approach (CBF) comes next. The Spray and Wait also performs well because of the dense nature of our network. Please note that growth rate of delivery ratios during the warm up period (0-4600 seconds) may not reflect actual ratios because the packets are generated throughout this period.

B. Delivery Cost

The Epidemic routing protocols achieves its higher delivery ratio with a very high cost (the cost is calculated using Eq. 3). The delivery cost is about 30 times less in our cluster-based forwarding method. The cost of Spray and Wait is slightly higher than that of our forwarding technique. The Spray and Wait achieves considerably low cost in delivery because it reduces a packets TTL exponentially at each hop and thus reduces a packets lifetime in a network. In order to highlight our forwarding by grouping mechanism and to be fair in comparing with multicopy routing protocols, we have

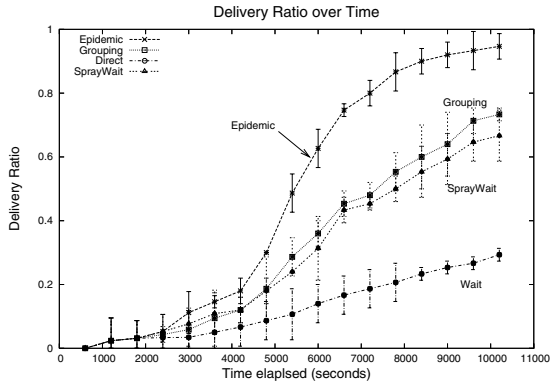


Fig. 8. Comparison of delivery ratios of various DTN routing protocols

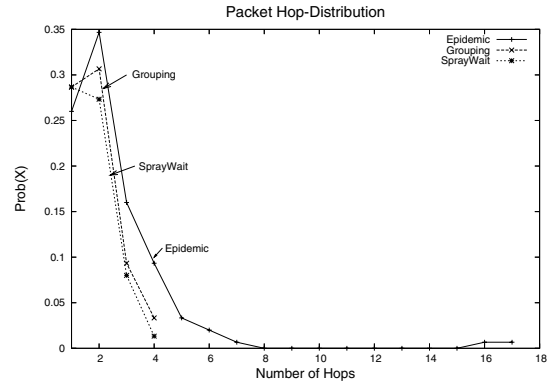


Fig. 10. Hop-count distribution of delivered packets

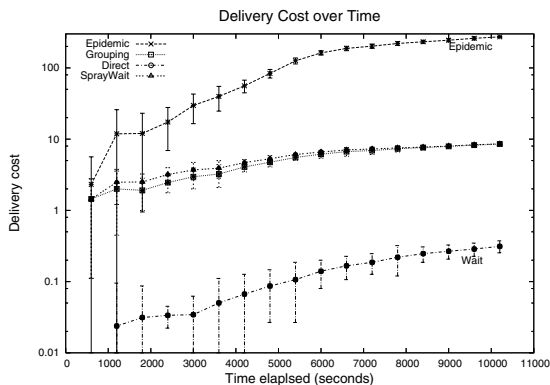


Fig. 9. Comparison of delivery costs of various DTN routing protocols

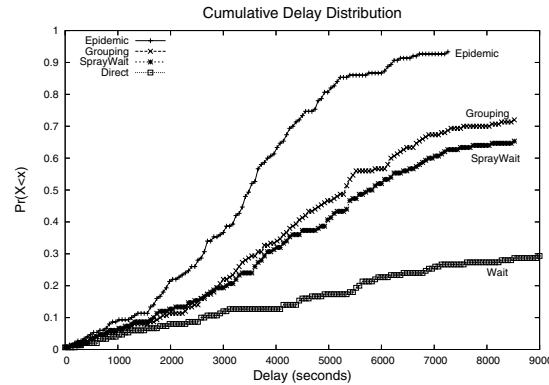


Fig. 11. Cumulative delay distribution

not reduced packets lifetime by modifying TTL field at each hop. The graphs (Fig. 9) clearly shows that even the naive forwarding by grouping (without TTL modification) performs better than the Spray and Wait. It is expected that adding TTL modification similar to Spray and Wait in will reduce the cost of our forwarding mechanism even further.

C. Hop Distribution

The hop distribution gives us an idea about how a particular routing protocol chooses a neighbor. It also indicates approximate cost of a routing protocol since more hops means more cost. As it can be seen from Fig. 10, both our forwarding approach and Spray and Wait delivers most of the packets (if delivered) within four hops whereas the Epidemic routing protocol uses as many as seventeen hops to deliver some packets. These gives us an indication of the relative cost and resource overhead of the routing protocols. The Epidemic routing protocol has the highest cost and overhead compared to other routing approaches.

D. Delay Distribution

A low hop count does not necessarily mean lower delay or vice versa. In order to get an idea of packet delivery delay, we plot the cumulative delay distribution of the routing protocols (Fig. 11).

Epidemic routing achieves the minimum possible packet delivery delay (receives 50% of the packets during first 4000 seconds) whereas the Direct delivery approach shows worst case delivery delay (receives only 12% packets during the first 4000 seconds). The packet delivery delay of our forwarding by grouping approach comes after the Epidemic routing.

It is evident from the above mentioned graphs that the delivery ratio of our forwarding method (CBF) approaches about 85% of the best case costing 30 times less than that of Epidemic routing. In terms of delivery ratio and delay, CBF performs next to Epidemic routing.

E. Effect of varying group numbers

Using our algorithm to form groups, we can control the number of groups that can be formed. It is intuitive that the number of groups in a network influences the packet delivery cost and rate. At one extreme, when all the nodes become members of a single group, our forwarding rule behaves similar to epidemic routing. Whereas at the other extreme, each node becomes a member of a separate group, our forwarding rule degenerates into direct delivery (wait) routing scheme. Thus, by tuning the group numbers, we can achieve higher delivery ratio (as in epidemic routing) with lower cost (as in direct delivery method). Fig. 12 shows the effect of varying group numbers (8, 16 and 32) in delivery cost. The delivery cost increases as the group number decreases

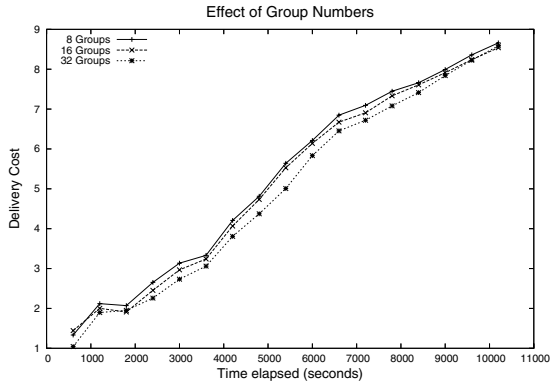


Fig. 12. Delivery cost over time with 8, 16 and 32 groups

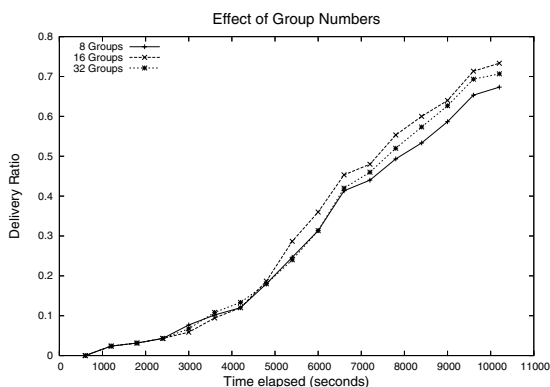


Fig. 13. Delivery ratio over time with 8, 16 and 32 groups

and it should approach to that of epidemic routing if group number becomes one. However, as time elapses, the delivery cost becomes similar in all three cases. This is because as time passes, more and more nodes forward packets and hence increase cost (since there is no control over the number of packets forwarding as in Spray and Wait mechanism).

Fig. 13 shows the delivery ratio of the packets with settings of 8, 16 and 32 groups. From the figure, we can see that the delivery ratio does not have a linear relationship with group numbers. Among the three cases, the delivery ratio is lowest in case of 8 groups. If we increase the group numbers to 16, the delivery ratio also increases but if we increase the group numbers even further (to 32 groups), then the delivery ratio again decreases. In our bottom-up clustering method, members of the groups suffer from loose bonding among themselves as the group number is approaching to unity. Besides, as the group number is reducing to unity, the groups are becoming skewed (i.e. some groups have very large number of members and some have few). These two effects denigrated grouping concepts and are the primary causes for a decrease in delivery ratio in case of 8 groups. So, there exists an optimal value of group number to achieve maximum delivery ratio without incurring much cost. We intend to delve into this matter in our future work.

VII. CONCLUSION

In this paper, we have demonstrated how clustering of nodes in a DTN can enhance the packet forwarding performance. By the application of dynamic programming method, we have presented a generic hierarchical grouping algorithm which is capable of grouping a large number of nodes using the node encounter frequencies as a parameter for grouping. With rigorous experiments using real-world bus traces we have demonstrated the efficacy of our approach. Grouping using other metrics, such as encounter duration, inter-meeting duration, etc. and the study of the effect of group numbers and group size is a topic for future research.

REFERENCES

- [1] Network simulator, <http://www.isi.edu/nsnam/ns>.
- [2] Postgresql, <http://www.postgresql.org>.
- [3] S. Ahmed and S. S. Kanere. Skvr: Scalable knowledge-based routing architecture for public transport networks. In *VANET'06, MobiCom workshop*, Los Angeles, California, September 2006.
- [4] J. Burgess, B. Gallagher, D. Jensen, and B. N. Levine. Maxprop: Routing for vehicle-based disruption tolerant networks. In *IEEE INFOCOM*, Barcelona, Spain, April 2006.
- [5] 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 IEEE INFOCOM*, Barcelona, Spain, April 2006.
- [6] H. Dubois-Ferriere, M. Grossglauser, and M. Vetterli. Age matters: Efficient route discovery in mobile ad hoc networks using encounter ages. In *Proceedings of ACM MobiHoc*, pages 257–266, 2003.
- [7] B. H. et al. Cartel: A distributed mobile sensor computing platform. In *Proceedings of ACM SenSys*, October 2006.
- [8] K. Fall. A delay-tolerant network architecture for challenged internets. In *Intel Research Technical Report, IRB-TR-03-003*, February 2003.
- [9] M. Grossglauser and M. Vetterli. Locating nodes with ease: Last encounter routing in ad hoc networks through mobility diffusion. In *The Proceedings of IEEE INFOCOM*, San Francisco, USA, March 2003.
- [10] P. Hui, A. Chaintreau, J. Scott, R. Gass, and C. Diot. Pocket switched networks and the consequences of human mobility in conference environments. In *Workshop on Delay Tolerant Networking*, August 2005.
- [11] P. Hui and J. Crowcroft. How small labels create big improvements. In *Proceedings of International Workshop on Intermittently Connected Mobile Ad hoc Networks*, March 2007.
- [12] S. Jain, K. Fall, and R. Patra. Routing in delay tolerant network. In *Proceedings of ACM SIGCOMM*, August 2005.
- [13] J. G. Jetcheva, Y.-C. Hu, S. PalChaudhuri, A. K. Saha, and D. B. Johnson. Design and evaluation of a metropolitan area multitier wireless ad hoc network architecture. In *Proceedings of 5th IEEE Workshop on Mobile Communications*.
- [14] S. C. Johnson. Hierarchical clustering schemes. In *Psychometrika*, volume 2, pages 241–254, 1967.
- [15] E. P. C. Jones and P. A. S. Ward. Routing strategies for delay-tolerant networks. In *Proceedings of ACM SIGCOMM*, 2004.
- [16] V. S. M. Notani and P. Nuggehalli. Peoplenet: Engineering a wireless virtual social network. In *Proceedings of ACM MOBICOM*, 2004.
- [17] A. Pentland, R. Fletcher, and A. Hasson. Daknet: Rethinking connectivity in developing nations. In *IEEE Computer*, volume 37(1), pages 78–83, January 2004.
- [18] T. Spyropoulos, K. Psounis, and C. S. Raghavendra. Spray and wait: An efficient routing scheme for intermittently connected mobile networks. In *Proceedings of ACM SIGCOMM*, 2005.
- [19] J. Su, A. Goel, and E. de Lara. An empirical evaluation of the student-net delay tolerant network. In *Proceedings of ACM MOBIQUITOUS*, July 2006.
- [20] A. Vahdat and D. Becker. Epidemic routing for partially connected ad hoc networks. In *Technical Report CS-200006*, Duke University, 2000.