Understanding the Tolerance of Dynamic Networks: A Routing-Oriented Approach [*]

Yifeng Shao and Jie Wu
Department of Computer Science and Engineering
Florida Atlantic University
Boca Raton, FL 33431, USA
{yshao1@, jie@cse}.fau.edu

Abstract

Research on delay tolerant networks (DTNs) has brought about a plethora of routing algorithms targeted at networks with different mobility patterns. However, few research works have been done towards developing methods to understand certain characteristics of DTNs, especially those with predictable characteristics (e.g., predictable mobility). As a step towards closing this gap, we devise an efficient scheme to study the tolerance of a DTN-like dynamic network in terms of delay. Our work is from the viewpoint of routing, and is based on theories in evolving graphs [1]. In this paper, we define the solvability of a DTN in terms of delay and show that it is non-trivial to examine it in the scope of a DTN through examples and analysis. We then propose both centralized and distributed solutions to this problem. Through extensive simulations based on real traces from UMass DieselNet [2] we study the solvability of both kinds of networks using approaches proposed in this paper. Finally, we briefly discuss the scenarios to which our work could be applied.

Keywords: *delay tolerant networks (DTNs), contact, evolving graphs, predictable mobility*

1 INTRODUCTION

Recently, a fundamental research work [4] has been done towards understanding routing problems in DTNs. In [4], Ramanathan et al. developed a formal framework to answer the following question: given a dynamic network which can be eventually connected, eventually routable, or eventually transportable, and a specified routing mechanism based on whether an end-to-end path is required, whether replication is allowed and whether schedule is known, is the network solvable by the routing mechanism? However, the solvability of a dynamic network corresponding to a routing mechanism in an *eventual* manner is not sufficient. More specifically, to answer whether a dynamic network is solvable during a time period by considering some specific metric (e.g. within a given delay) is sometimes more meaningful but more challenging. This paper focuses on this issue.

In this paper, we study the following problem: given a dynamic network and an upper bound of routing metric such as delay, determine whether the network is solvable within that upper bound. To simplify, we assume that problems are considered in some given period of time $[0, Te]$. To solve this problem in the scope of MANET may be trivial. However, to solve it in the scope of a DTN is definitely nontrivial. As aforementioned, the routing paradigm has changed from store-forward to store-carry-forward, and optimal routes satisfying a certain metric change from a function of source and destination to a function of source, destination, and *start time*. Furthermore, enumerating all the start times in $[0, Te]$ is costly and even impossible. We will illustrate the problem in more detail in Section 2.3.

We propose our framework to study the aforementioned problem by taking into account both space and time. Our framework is based on evolving graph theories [1]. Due to the fact that the problem is closely related to node mobility patterns, and also to make our research results more practical, we limit the networks we study to those with predictable contacts (contact definition will be given in Section 2.1) over either finite time horizons or infinite time horizons due to periodicity. To address the case over finite time horizons, we propose a centralized method which is based on the result in [1]. Subsequently, we propose a decentralized method which is a modification of the distributed Bellman-Ford algorithm to address the case over infinite time horizons where nodes are either static or have a strict repetitive motion. [2]. The contributions of our research work

[*] This work was supported in part by NSF grants CCR 0329741, CNS 0422762, CNS 0434533, CNS 0531410, and CNS 0626240.

are summarized as follows: (1) To the best of our knowledge, we are the first to examine the solvability of a dynamic network with predictable mobility during a time period by considering some specific metric. The problem we raise here may lay a foundation for future research such as routing scheduling and routing optimization. (2) By modeling a dynamic network with predictable contacts as an evolving graph, we devise both centralized and decentralized solutions to the nontrivial problem. We study the solvability of a real network.

The remainder of this paper is organized as follows. In Section 2, we review the concept of evolving graphs, model the network, and illustrate the problem. Section 3 presents the algorithms. Section 4 presents the simulation results. Some related works will be reviewed in Section 5 and this paper is concluded in Section 6.

2 PRELIMINARIES

In this section, we first review the evolving graph model [1] which is a good tool to analyze the characteristics of time-varying dynamic networks. Then we describe the network model in our work. Finally we formalize our problems.

2.1 Evolving Graph

An evolving graph is a graph which captures the dynamic characteristics of each node and link.

Definition 1 ($Evolving\ Graph$ [1]). *Let be given a graph $G(V, E)$ and an ordered sequence of its subgraphs, $S_G = G_1, G_2, ..., G_T$ such that $\cup_{i=1}^{T} G_i = G$. Let $S_T = t_0, t_1, t_2, ..., t_T$ be an ordered sequence of time instants. Then the system $\mathcal{G} = (G, S_G, S_T)$, where each G_i is the subgraph in place during $[t_{i-1}, t_i)$, is called an evolving graph. Let $V_{\mathcal{G}} = V$, and $E_{\mathcal{G}} = E$.*

The following concepts will be used in the remainder of this paper. We include them here for completeness.

Definition 2 ($Journey$ [1]). *A journey $\mathcal{J} = (R, R_\sigma)$ in an evolving graph $\mathcal{G}$ is comprised of $R = e_1, e_2, ..., e_k$ the sequence of edges it traverses, and $R_\sigma = \sigma_1, \sigma_2, ..., \sigma_k$ the corresponding start time instants of edge traversal.*

Definition 3 ($Relevant\ Journey\ (in\ terms\ of\ delay)$ [1]). *Let $\mathcal{G}$ be a timed evolving graph, and $u, v \in V$. Let $\mathcal{J}$ be a journey from u to v with departure time $t_{departure}$ and arrival time $t_{arrival}$. If there is another journey $\mathcal{J}'$ from u to v with departure time $t'_{departure}$ and arrival time $t'_{arrival}$ such that $[t_{departure}, t_{arrival}] \supset [t'_{departure}, t'_{arrival}]$, $\mathcal{J}'$ is called relevant journey and $\mathcal{J}$ is not.*

Definition 4 ($Journey\ Class$ [1]). *Given a journey $\mathcal{J} = (R, R\sigma)$ with $R = e_1, e_2, ..., e_k$ and $R_\sigma = \sigma_1, \sigma_2, ..., \sigma_k$, if each edge e_i is valid during interval $[\sigma_i, \sigma_i + \delta + \zeta(e_i)]$ for some δ, we introduce $I(R_\sigma, \delta) = [\sigma_1, \sigma_1 + \delta], [\sigma_2, \sigma_2 + \delta], ..., [\sigma_k, \sigma_k + \delta]$ and we define this class of journeys $(R, I(R_\sigma, \delta))$ a journey class.*

A tuple $(t_{first\_departure}, t_{first\_arrival}, \delta_{width}, T_{cycle})$ can also be used to represent a journey class. Here $t_{first\_departure} = \sigma_1, t_{first\_arrival} = \sigma_k + \zeta(e_k), \delta_{width} = \delta$ and T_{cycle} is the cycle of the journey class.

A journey class is called a relevant journey class if it contains only relevant journeys. A relevant journey class that is not contained in any other relevant journey class is called *maximal relevant journey class (MRJC).*

2.2 Network Model

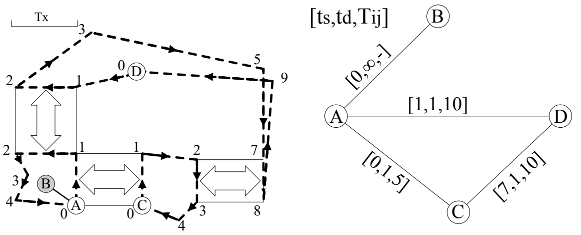
We model the dynamic network we study as an evolving graph where vertices are nodes and edges are sets of contacts. A *contact* is a period of time during which two neighboring nodes can communicate with each other. A contact can be represented as a tuple $(i, j, t_s, t_d, d_{ij}, T_{ij})$, where i and j are the two nodes associated with this contact, t_s and t_d are the time when the contact becomes available and the duration during which the contact is alive respectively, d_{ij} is the link capacity (and to a lesser extent, the average propagation delay) during t_d, and T_{ij} is the cycle of the contact. We extend the basic evolving graph model by using contacts instead of a link presence schedule, and incorporate more information in the graph. We denote $\delta_E = max\{number\ of\ contacts\ on\ edge\ e, e \in E\}$. See the example network in Figure 1(a) whose corresponding evolving graph model is in Figure 1(b).

To make our model practical, we assume that the set of contacts corresponding to each edge is a countable set. We also assume that the interval of a contact presence is long enough for a message to traverse the corresponding edge. Routing occurs in a store, carry, and forward fashion. We assume that the cost (in time) for a message to traverse an edge equals the length of time for which it waits at the source node plus the propagation delay. Contacts are predictable in our model.

2.3 Problem Definition

Here we formalize the problems and explain in detail why they are non-trivial in DTN-like dynamic networks.

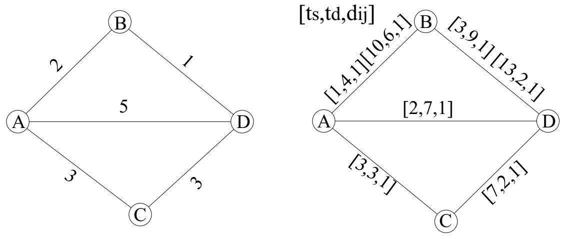
Problem 1 ($(t, d) - Problem$). *Given an evolving graph $\mathcal{G}$, $(u, v) \in V \times V$, and $t \in [0, Te]$, evaluate whether a message can be generated by u at t and routed to v at $t + d$ through a journey $\mathcal{J}$.*



(a) A dynamic network consisting of either static nodes or nodes with cyclic motion.

(b) The evolving graph corresponding to the network in Figure 1(a).

Figure 1. An example of a cyclic dynamic network and its corresponding infinite evolving graph.



(a) The topology of a traditional MANET.

(b) The evolving graph of a DTN with the same topology as Figure 2(a) but different link presence schedule.

Figure 2. An example of a MANET and a DTN.

Here, d is given. The total costs (in time) for this message includes the time this message spends waiting at u for $\mathcal{J}$ plus the delay of $\mathcal{J}$.

Problem 2 ($(*, d) - Problem$)**.** *Given an evolving graph $\mathcal{G}$, $(u, v) \in V \times V$, $\forall t \in [0, Te]$, evaluate the probability p that Problem 1 is true.*

Problem 2 is on the node level. By replacing "given $(u, v) \in V \times V$" with $\forall(u, v) \in V \times V$, we scale it to the network level and get the new definition of the solvability of a dynamic network.

Definition 5 ($d - Solvable$)**.** *On the network level, if Problem 2 returns 1, the corresponding dynamic network is d-solvable. Otherwise it is d-unsolvable.*

Take Figure 2 as an example. Figure 2(a) is a traditional MANET where the number represents the propagation delay corresponding to that link. Due to the fact that traditional mobile ad hoc networks (MANETs) are assumed to be static or lowly dynamic, using Dijkstra's algorithm or the Bellman-Ford algorithm can easily solve Problem 1 and Problem 2. In Figure 2(a), considering node pair (A, D), we assert that $\forall d \geq 3$, Problem 1 and Problem 2 return true and 1 respectively and that $\forall d < 3$, Problem 1 and Problem 2 return false and 0 respectively. We can say the network is 4-solvable but 2-unsolvable.

However, in the scope of DTN-like dynamic networks, solving Problem 2 is non-trivial. Consider Figure 2(b) which is an evolving graph of a DTN. To solve Problem 1, we use a simple modification of Dijkstra's algorithm as in [5] by calculating the shortest journey (in terms of delay) starting at or after a given time t. Consider node pair (A, B). During time interval $[1, 4]$ and $[10, 15]$, the shortest journey takes the form of $A \rightarrow B$ and holds a delay of 1. During $(4, 8]$, the shortest journey takes the form of $A \rightarrow D \rightarrow B$ and holds a delay of 2. During $[0, 1)$, the shortest journey takes the form of waiting at A until time instant 1 and $A \rightarrow B$ and holds a delay including the waiting time plus the propagation delay 1. During $(8, 10)$, the shortest journey takes the form of waiting at A until time instant 10 and $A \rightarrow B$ and holds a delay including the waiting time plus the propagation delay 1. After time instant 15, no journey exists from A to B. Given time instant 0.5 and $d = 1$, Problem 1 returns false. Given time instant 9 and $d = 2.5$, Problem 1 returns true.

The methods used in the case of MANETs cannot be applied directly here to solve Problem 2 due to the highly time-varying nature of the network and the store-carry-forward routing paradigm. Still, consider node pair (A, B). Given d, one way to answer Problem 2 is to enumerate all the time instants in $[0, Te]$ with Problem 1 and sum up the results. However, this can be impossible to implement, let alone to determine the d-solvability of the network.

3 Methodology

In this section, we illustrate our solutions to Problem 2 in DTN-like networks. First, we describe our method based on Maximal Relevant Journey Classes (MRJCs). Then we introduce the centralized and the decentralized implementations to compute MRJCs respectively.

3.1 MRJCs-based Solution

Recall the definition of *maximal relevant journey class* in Section 2.1. Given an evolving graph $\mathcal{G}$ of a dynamic network, $(u, v) \in V \times V$, if we can find all the maximal relevant journey classes from u to v, intuitively, Problem 2 can be calculated based on them with reduced problem size.

Still consider the node pair (A, B) in Figure 2(b). Figure 3 shows all three of the maximal relevant journey classes from A to B. The delay of journey classes I is $|I|_d = 1$. The delay of journey classes II is $|II|_d = 2$ and the delay

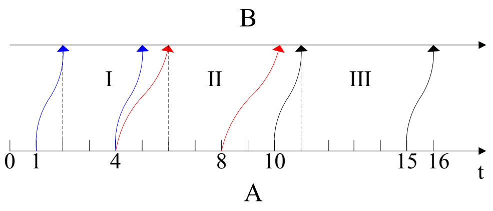


Figure 3. The maximal relevant journey classes from A to B in Figure 2(b).

of journey classes III is $|III|_d = 1$. If A generates a message at $t = 0$ and wants it to be routed to B, it can choose to (1) wait until I is available and use I; (2) wait until II is available and use II; or (3) wait until III is available and use III. A chooses the first option because it takes the smallest amount of time which is $d_0 = 2$. Furthermore, if the case is $t = 1$, A may choose to (1) directly uses I; (2) wait until II is available and use II; or (3) wait until III is available and use III. A chooses the first option because it takes the smallest amount time which is $d_1 = 1$. Similarly, $d_4 = 1$, $d_8 = 2$, $d_{10} = 1$, $d_{15} = 1$ and $d_{16} = \infty$.

Now we consider interval $[0, 4]$ which consists of two sub intervals $[0, 1)$ and $[1, 4]$. If Problem 1 returns true at $t = 0$, it returns true for the entire interval $[0, 4]$. If Problem 1 returns false at $t = 0$ but returns true at $t = 1$, it returns true for the interval $[1, 4]$ and interval $[1 - (d - d_1), 1]$. If Problem 1 returns false at both $t = 0$ and $t = 1$ but returns true at $t = 4$, it returns true for the interval $[4 - (d - d_4), 4]$. Iteratively, $(4, 8]$, $(8, 15]$ and $(15, 16]$ are considered in the same way. Note that interval $(4, 8]$ consists of sub intervals $(4, 4)$ and $(4, 8]$ according to the above way we partition an interval. $(15, 16]$ consists of sub intervals $(15, 16)$ and $[16, 16]$. Summing up all the results solves Problem 2. In our example, given $d = 0.9$, Problem 2 returns 0 (assuming $T_e = 16$); given $d = 1$, it returns $\frac{(4-1)+(15-10)}{16} = 0.5$; and given $d \geq 3$, it returns $\frac{15}{16}$.

Generally, given an evolving graph $\mathcal{G}$, a node pair $(u, v) \in V \times V$, maximal relevant journey classes $L[u, v]$ from u to v, $[0, Te]$ is divided into several consecutive intervals $[t_i, t_{i+1}]$ where $i \geq 0$ by $L[u, v]$. We prove that the problem size of Problem 2 in terms of the number of $[t_i, t_{i+1}]$ is bounded by $2|E|\delta_E + 1$.

Theorem 1. *Given an evolving graph, $\mathcal{G}$, a node pair $(u, v) \in V \times V$, maximal relevant journey classes $L[u, v]$, the problem size of Problem 2 is reduced to $O(2|E|\delta_E + 1)$ by applying $L[u, v]$.*

Proof. According to [1], $|L[u, v]|$ is bounded by $2|E|\delta_E$, hence the problem size of Problem 2 is bounded by $2|E|\delta_E + 1$. □

3.2 Centralized Implementation

In this section, we present the centralized algorithm to calculate maximal relevant journey classes (MRJCs). This algorithm is based on the augmenting journey classes algorithm proposed in [1]. Given an edge (u, v) and a list of journey classes $L[s, u]$ from source s to u, the augmenting journey classes algorithm computes a list of journey classes $L[s, v]$ from s to v based on $L[s, u]$. More details about the augmenting journey classes algorithm can be found in [1]. This algorithm will also be used with a slight modification in our distributed implementation to calculate MRJCs (an edge is a list of one-hop journey classes). The centralized implementation is described in Algorithm 1. In Algorithm 1, the "merge" operation may eliminate irrelevant journey classes.

Algorithm 1 Centralized Algorithm to Calculate MRJCs

Input: an evolving graph $\mathcal{G}$; a vertex $s \in V$.
Output: $L[s, v]$ which gives for each vertex $v \in V$ the maximal relevant journey classes from s to v. $\mathcal{L}[s, v]$ is initially set to $\emptyset$.

```
for $i = 1$ to $|V|$ do
    for all $v \in V$ do
        $L'[s, v] \leftarrow L[s, v]$
    end for
    for all $(u, v) \in E$ do
        $L_{s \to v} \leftarrow AugmentJourneyClasses((u, v), L[s, u])$
        $L'[s, v] \leftarrow$ merge $L'[s, v]$ and $L_{s \to v}$
    end for
    for all $v \in V$ do
        $L[s, v] \leftarrow L'[s, v]$
    end for
end for
```

3.3 Decentralized Implementation

In this section, we extend our centralized algorithm to calculate maximal relevant journey classes to a distributed version the basic idea behind which is similar to the one of the distributed Bellman-Ford algorithm. The decentralized method is easy to implement and applies well to the network where contacts between nodes are either cyclic or persistent.

The algorithm is a proactive table-driven protocol such that every node maintains an MRJCs table. Each entry of this table at node u is a 2-tuple $(d, L[u, d])$ where d represents another node in the network and $L[u, d]$ represents the up-to-date MRJCs from u to d computed based on u's knowledge about the network. Whenever a node u detects that a contact with a node v comes up, u checks whether its local table has changed since the last contact with v. If so, u sends the entire table to v. To reduce communication cost, u can choose to send to v only the entries that have changed. It requires additional storage on u to map changes to nodes.

Algorithm 2 Distributed Algorithm to Calculate MRJCs(at each node u)

```
if a certain contact between u and v is just available then
    for all $d \in V$ do
        send $L[u,d]$ to $v$, where $L[u,d]$ is the current MRJCs from
        $u$ to $d$ computed at $u$
    end for
end if
if receives $L[v,d]$ from $v$ then
    $L'[u,d] \leftarrow AugmentJourneyClasses(L[v,d], L[u,v])$
    $L[u,d] \leftarrow$ merge $L'[u,d]$ and $L[u,d]$
end if
```

In this paper, we assume u sends the entire table to v just for simplicity. Upon receiving the table from u, v updates the up-to-date MRJCs table using the information in u's table. This update uses the modified augmenting journey classes algorithm. Algorithm 2 illustrates the details.

4 SIMULATION

In this section, we study the d-solvable problem on the UMass DieselNet Spring 2006 [2].

Traces preprocessing. We assume that when two buses are within transmission range, data flow in both directions, i.e., all contacts are symmetric. Note that there are many contacts between the same pair of nodes that overlap with one another. Technically, for each pair of buses, we combine any two subsequent contacts that overlap with each other. We use the earlier contact's start time as the merged contact's start time and the later contact's end time as the merged contact's end time. The average propagation delay associated with each contact is designated as the time to transmit a 500 bytes message across that contact. By doing this, each contact is assigned a time-varying average propagation delay. In our simulation, there are totally 37 buses and 78164 contacts and time is indexed by second. Also Te is designated to 10231965.

Results evaluation. The results are included in Figure 4. Figure 4(a) depicts buses and the corresponding number of MRJCs to the other buses. For each candlestick, the circle denotes the mean value, the bar across each box denotes the median value, the lower and the upper bounds of each box show the 25th and the 75th percentiles respectively, and the lower and the upper "whiskers" show the rest of the values. Figure 4(a) indicates a high variance in the number of MRJCs from one bus to the rest of the buses in UMass DieselNet. We also note that some of the buses have small chances to route to the other buses in UMass DieselNet. Figure 4(c) depicts Problem 2 on the node level. We pick four source-destination pairs which capture the characteristics of the problem to some extent. We note that the probability that the problem is true increases as the parameter $delay$ of the problem increases and converges to a certain value while the parameter $delay$ is sufficiently large which indicates the "eventual manner" in [4]. The converged value of each source-destination pair may differ from each other due to several factors such as node mobility, node degree and the like. Figure 4(d) depicts Problem 2 on the network value. The notation of each candlestick is the same as that in Figure 4(a). In Figure 4(d), the percentage with which the network is d-solvable with the increase of d and converges to a certain value while d is sufficiently large. If we denote the mean value as the d-solvability of the network, we note that UMass DieselNet is d-solvable with probability around 70% even in an eventual manner.

5 RELATED WORK

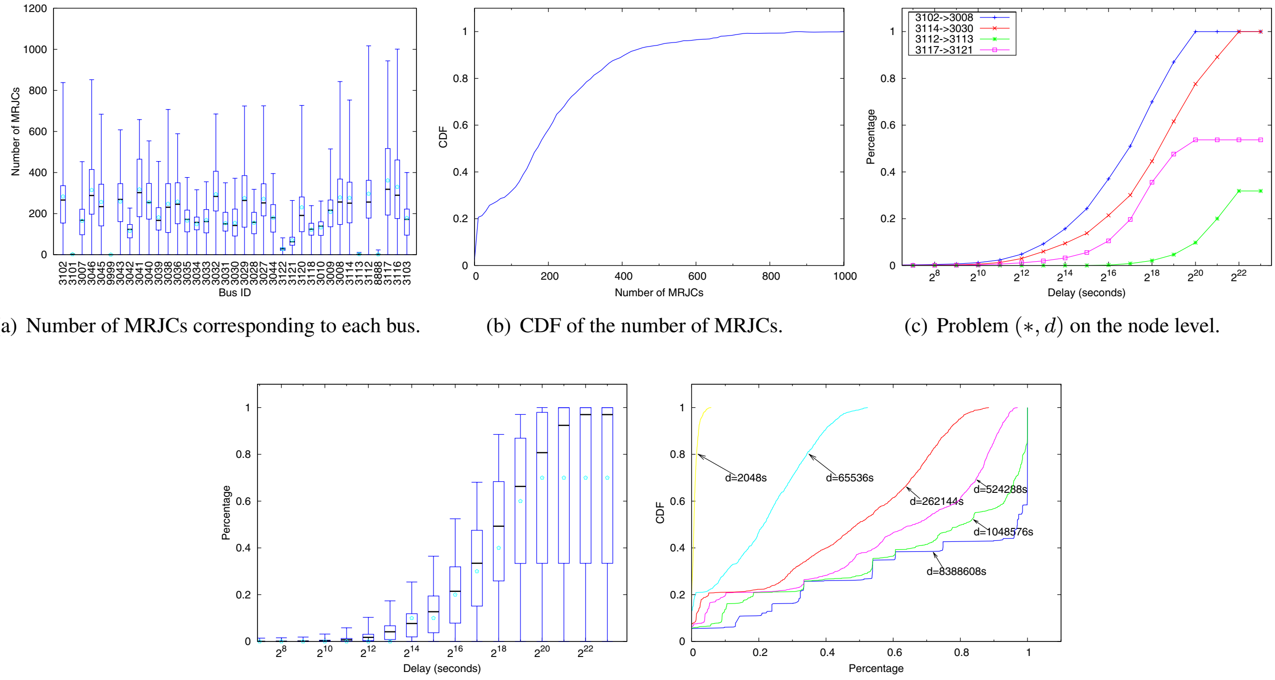
The fact that no end-to-end connections may exist at any time instant in DTNs gives rise to the difficulty of the routing problems. Previous works have focused on the routing algorithm design. They include replication based methods such as Epidemic Routing [6], probability-based forwarding and purging [7], and future contact prediction based approaches [8]. [9] studied the relationship between the routing problem and the use of knowledge about the underlying network, which laid a groundwork for future study.

Instead of proposing a new algorithm to the problem space, we aim to study the characteristics of DTNs from the viewpoint of routing. In [5], Borrel et al. classified the dynamic networks using a routing-aware approach. The classification is based on the evolving graph theory [1]. In [4], in addition to classifying the dynamic networks (problem space) using connectivity as the main discriminator, Ramanathan et al. classified the algorithms (solution space) by constraints of path requiring, schedule unavailability, and single copy. Also, the author studied the solvability (bundle delivery) by various classes of algorithms. However, neither examined the solvability of a dynamic network in terms of delay.

Theoretical models for networks in general have been developed. In [10], the Merugu et al. modeled a dynamic network as a space-time graph which aims to capture both the space and temporal information of a network. In [1], Ferreira modeled a dynamic network as an evolving graph and studied the routing problem over network evolution.

6 CONCLUSION

In this paper, we study the solvability of a DTN in terms of delay. Through analytical study, we show that it is nontrivial to solve the d-solvable problem in the environment of DTN-like networks. We propose an MRJCs based approach which aims to solve the problem by reducing the problem size to a reasonable one.



(a) Number of MRJCs corresponding to each bus.

(b) CDF of the number of MRJCs.

(c) Problem $(*, d)$ on the node level.

(d) Problem $(*, d)$ on the network level.

(e) Problem $(*, d)$ on the network level.

Figure 4. UMass DieselNet traces.

Our solutions are mainly based on MRJCs. However, this may lead to contention and load unbalancing to some extent while routing strategies are based on our results. [11] found that many near-optimal journeys exist in a DTN-like network. Hence one of our future directions is to extend our solutions by considering near-optimal journeys in addition to MRJCs. Due to the "path explosion" phenomenon [11], our algorithms suffer from relatively high computation cost, which inspires us to find good approximation methods to answer the problems we raise in this paper with low cost.

References

[1] A. Ferreira. Building a reference combinatorial model for manets. *IEEE Network*, 18(5):24–29, 2004.

[2] X. Zhang, J. Kurose, B. N. Levine, D. Towsley, and H. Zhang. Study of a bus-based disruption tolerant network: Mobility modeling and impact on routing. In *Proc. of ACM MOBICOM*, 2007.

[3] C. Liu and J. Wu. Scalable routing in delay tolerant networks. In *Proc. of ACM MobiHoc*, 2007.

[4] R. Ramanathan, P. Basu, and R. Krishnan. Towards a formalism for routing in challenged networks. In *Proc. of CHANTS*, 2007.

[5] V. Borrel, M. H. Ammar, and E. W. Zegura. Understanding the wireless and mobile network space: a routing-centered classification. In *Proc. of CHANTS*, 2007.

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

[7] T. Spyropoulos, K. Psounis, and C. S. Raghavendra. Spray and wait: an efficient routing scheme for intermittently connected mobile networks. In *Proc. of WDTN*, 2005.

[8] J. Burgess, B. Gallagher, D. Jensen, and B. N. Levine. Maxprop: Routing for vehicle-based disruption-tolerant networks. In *Proc. of IEEE INFOCOM*, 2006.

[9] S. Jain, K. Fall, and R. Patra. Routing in a delay tolerant network. In *Proc. of ACM SIGCOMM*, 2004.

[10] S. Merugu, M. Ammar, and E. Zegura. Routing in space and time in networks with predictable mobility. *Technical Report GIT-CC-04-7, Georgia Institute of Technology*, 2004.

[11] V. Erramilli, A. Chaintreau, M. Crovella, and C. Diot. Diversity of forwarding paths in pocket switched networks. In *Proc. of IMC*, 2007.