Capacity Enhancement using Throw-Boxes in Mobile Delay Tolerant Networks

Wenrui Zhao[†] Yang Chen[†] Mostafa Ammar[†]
[†]College of Computing
Georgia Institute of Technology
Atlanta, Georgia 30332
{wrzhao, yangchen, ammar, ewz}@cc.gatech.edu

Mark Corner[‡] Brian Levine[‡] Ellen Zegura[†]
[‡]Department of Computer Science
UMass Amherst
Amherst, MA 01003
{mcorner,brian}@cs.umass.edu

***Abstract*— Delay tolerant networks (DTNs) are a class of emerging networks that are subject to frequent and long-duration partitions. Due to intermittent connectivity, DTNs might be significantly limited in supporting application needs, for example, leading to low throughput or high delay. To address this problem, we propose the use of *throw-boxes* to improve data delivery performance. Throw-boxes are small, inexpensive devices equipped with wireless interfaces and deployed to relay data between mobile nodes. Being small and inexpensive, throw-boxes represent a flexible and cost-effective approach to enhance network capacity.**

In this paper, we systematically study two inter-related issues, namely deployment and routing, in using throw-boxes for throughput enhancement. Specifically, we develop algorithms for throw-box deployment and data forwarding under various routing strategies, including single path, multi-path and epidemic routing. Using extensive *ns* simulations, we evaluate the utility of throw-boxes and the impact of various routing and deployment strategies on network performance. Our objective is to guide the design and operations of throw-box-enhanced DTNs. We find that throw-boxes are very effective in improving both data delivery ratio and delay, especially for multi-path routing and environments with regular node movement.

I. Introduction

Delay tolerant networks (DTNs) [1] are a class of emerging networks that are subject to frequent and long-duration partitions. Due to intermittent connectivity between nodes, a DTN might be significantly limited in its capability for data delivery. For example, consider a hypothetical disaster relief scenario where communication among rescuers is essential for exchanging vital information about victims and hazards, and coordinating rescue efforts. However, it is challenging to provide communication services in this environment because no stable infrastructure is available. Given the geographic span of the affected area and the limited range of radios, rescuers may not be able to form a connected network. Instead, data communication relies on the intermittent connectivity between nodes, e.g., when nodes come close to each other. However, node movement is general dictated by non-communication purposes. As a result, contacts between nodes may occur infrequently can be insufficient to support communication needs.

To address this problem, various approaches have been proposed to enhance network capacity. These approaches fall into two categories, depending on whether the resulting network is connected or disconnected. Approaches in the first category seek to form and maintain a connected network, either using radios with longer transmission ranges or deploying a mesh network as an infrastructure to cover the affected area [2]. However, since many mobile nodes use batteries for power supply, the use of a long range radio may lead to excessive energy consumption. In addition, the availability of such devices in many scenarios, such as disaster relief, would be questionable. Further, the deployment of a mesh network might be costly for a large area. It is also not trivial to configure and maintain a mesh network in hostile environments, e.g., nodes may fail or be damaged.

On the other hand, approaches in the second category try to enhance contact opportunities between nodes without maintaining a connected network[3], [4], [5], [6], [7]. That is, data are forwarded in a *store-and-forward* manner to overcome network partitions. Message ferrying is an example of this approach, which utilizes a set of special nodes called *message ferries* to enhance connectivity between nodes [6], [7]. By exploiting ferry mobility to relay data, nodes can communicate with others that otherwise would not be possible. The use of ferries, however, may raise issues of robustness as ferries play a critical role in data delivery. Ferry failures or compromises may pose significant degradation in the performance or security of the whole network [8].

In this paper, we propose the use of *throw-boxes* to improve data delivery performance in DTNs that consist of mobile nodes. Throw-boxes are small and inexpensive devices equipped with wireless interfaces and storage. Throw-boxes are deployed to relay data between mobile nodes in a store-and-forward manner, and can operate without communication with other throw-boxes. As compared to previous approaches, the use of throw-boxes has the following advantages. First, throw-boxes can be deployed dynamically and easily, which would be important in critical environments. For example, rescuers in a disaster relief scene can physically throw a bunch of throw-boxes into the area. Throw-boxes can also be deployed airborne via airplanes or attached to moving vehicles. Second, throw-boxes are designed to operate in the present of network partitions. In addition, throw-boxes can operate without coordination among themselves, which further eases the deployment and management of throw-boxes. Third,

due to the relatively low cost, throw-boxes can be deployed in a larger number, enhancing reliability and robustness. In summary, throw-boxes represent a flexible and cost-effective approach to enhance network capacity.

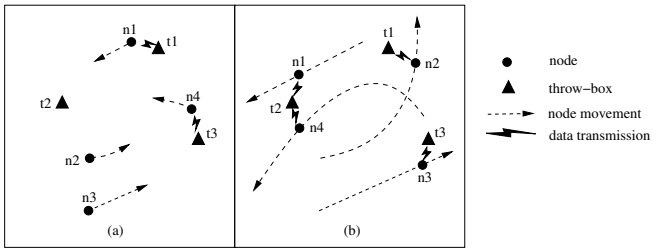


Fig. 1. An example of DTNs using throw-boxes.

The use of throw-boxes enhances network capacity by increasing the opportunities for nodes to communicate with each other. This is because with throw-boxes, nodes can communicate with each other via throw-boxes by visiting the same locations (i.e., where throw-boxes are located) even at different times. Fig. 1 shows an example of using throw-boxes in a DTN. With throw-boxes, nodes are able to communicate with each other. For example, node *n4* can send data to node *n3* using throw-box *t3* over time; throw-box *t2* bridges node *n1* and *n4*, which can communicate using multi-hop routing (see Fig. 1(b)).

In fact, when node mobility are less random, which holds for many real-life scenarios, throw-boxes can improve data delivery significantly. Let us consider the UMassDiesel DTN test-bed which operates on buses of a public transport system[9]. In this test-bed, buses of 9 routes are equipped with wireless cards and form a DTN. Using simulations based on the traces of real bus movement, we study the communication opportunities between two buses on different routes. We compare the communication opportunities and the effective capacity between these buses before and after the deployment of a single throw-box. The simulations are run for a period of 100 hours in simulation time. The communication range of both buses and the throw-box is 141 meters, and the radio bandwidth is 2 Mbps. Table I shows the simulation results. We observe an increase by a factor of 20 in the effective capacity with the deployment of even a single throw-box. In addition, the deployment of the throw-box significantly reduces the data delivery delay.

	Before	After
Total Contact Duration (sec)	631	11927
Effective Capacity (Kbps)	3.5	66.3
Delay (sec)	63012	3120

TABLE I

PERFORMANCE BEFORE AND AFTER THROW-BOX DEPLOYMENT

The above example clearly demonstrates the potential of performance improvement using throw-boxes. To best utilize throw-boxes, one should consider both throw-box deployment and routing. Throw-box deployment determines where to place throw-boxes, which significantly affects contacts between throw-boxes and nodes, and data delivery performance. Routing also impacts the achieved performance by specifying how data are forwarded among nodes and throw-boxes.

In this paper, we present a framework to systematically study these two issues. Specifically, we develop algorithms for throw-box deployment and data forwarding under various routing strategies, including single path, multi-path and epidemic routing. We also consider different deployment strategies, depending on how information about contacts and traffic is used to determine throw-box locations. These deployment and routing strategies cover a variety of situations. Using extensive *ns* simulations, we evaluate the utility of throw-boxes and the impact of various routing and deployment strategies on network performance. Our objective is to guide the design and operations of throw-box-enhanced DTNs. Based on the simulation results, we find the following:

- Throw-boxes are very effective in improving throughput. The use of throw-boxes can also reduce data delivery delay.
- Throw-boxes are more useful when nodes follow regular movement patterns rather than random movement.
- Multi-path routing achieves larger performance improvement with the use of throw-boxes than single path and epidemic routing.
- Throw-box deployment based on contact information has good performance. The availability of traffic information can lead to better performance, especially for single path routing. In addition, single path routing is more sensitive to the deployment scheme used than multi-path routing. Epidemic routing is least sensitive to deployment.

The rest of this paper is structured as follows. In Section II, we describe throw-box characteristics and the network model considered in this paper. In Section III, we present a framework to systematically study the issues of throw-box deployment and data routing. We study the various routing approaches in Section IV to VI. The simulation results are presented in Section VII and summarized in Section VIII. We review related work in Section IX and conclude the paper in Section X.

II. NETWORK ARCHITECTURE USING THROW-BOXES

In the previous section, we introduce the concept of throw-boxes. We now describe the characteristics of throw-boxes and then present the network model we consider in this paper.

A. Throw-box

As illustrated in the previous section, throw-boxes are used to relay data between nodes in DTNs, where contacts among nodes occur intermittently and often irregularly. To overcome network partitions, data are forwarded in a store-and-forward fashion. That is, nodes buffer data when there is no path available for the destinations and forward data when a contact with other nodes occurs. To support such intermittent and opportunistic transmission that is typical in DTNs, throw-boxes should have the following characteristics.

- Ready for communication. Since contact opportunities between throw-boxes and nodes may be very limited, throw-boxes should be highly available for them to be effective.
- Supporting bursty operations. In DTNs, contacts between nodes and throw-boxes may last for a very short period of time. To fully utilize available contact opportunities, throw-boxes should be able to transmit data at a relatively high data rate, e.g., on the order of Mbps, and have sufficient processing power to handle such data rate as well as plenty storage for buffering the data.
- High energy efficiency. When throw-boxes have limited power supplies, throw-boxes should operate in an energy efficient manner to prolong the lifetime.

Processor	Intel PXA255 400MHz
Memory	64MB SDRAM, 32MB FLASH
Power consumption	< 500mA
Size	3.5" x 2.5"
Weight	47g

TABLE II
STARGATE DATASHEET.

Devices with the above characteristics have been developed in experimental prototypes or are commercially available, e.g., StarGate [10] and Turducken [11]. Table II lists a brief technical description of StarGate devices. Given the current trend of technology advances in processor and storage, it is expected that the cost and size of such devices will continuously decrease. In addition, they will be more energy efficient in the future. Therefore, large scale deployment of throw-boxes becomes feasible.

B. Network Model

Throw-boxes can be used in a variety of scenarios, each with different requirements and characteristics. In this paper, we focus on the use of throw-boxes in mobile DTNs where mobile nodes are sparsely distributed in an area. Nodes communicate with each other via wireless interfaces and are equipped with storage for data buffering. Due to the sparse distribution, nodes are not able to form a connected network. Instead, nodes communicate intermittently when nodes move into the transmission range of each other. To discover other nodes for communication, nodes broadcast beacon messages periodically. Upon receiving beacon messages, nodes initiate data exchange. This way, data are forwarded along a series of contacts over time, forming a path from the source to the destination. In DTNs, data are communicated in application data units called *messages* (or bundles [12]), which can be of varied sizes. Each message carries with it a timeout value which specifies when this message should be dropped if not delivered.

Nodes are sources and destinations of data communication. Suppose that the number of nodes is n. We characterize the traffic demand by a matrix $\mathbf{b_{n\times n}}$ where b_{ij} specifies the long term relative traffic demand from node i to node j, $1 \leq i, j \leq n$, and $\sum_{1\leq i,j\leq n} b_{ij} = 1$. This traffic model is general in that it can model different traffic patterns, i.e., uniform traffic among nodes that is typical in ad hoc networks or concentrated traffic to a single destination in sensor networks.

Given the inherent capacity limitation of DTNs, we consider the use of throw-boxes to enhance network capacity. Throw-boxes are equipped with the same wireless interfaces as nodes and have storage for data buffering. Throw-boxes are not sources or destinations of data communication. In this paper, we assume that throw-boxes are deployed to fixed locations[1]. In addition, we assume that throw-boxes have sufficient energy supplies. For example, throw-boxes might be equipped with renewable (solar) energy sources, or throw-boxes may be used for a relatively short period of time such that energy supplies become of less concern. We further assume that throw-boxes do not interact with each other[2]. That is, throw-boxes only communicate with nodes.

Once deployed, throw-boxes are involved in relaying data between nodes. Specifically, when a throw-box receives a beacon message from a node, it will exchange buffered messages with the node. To avoid duplicate message transmission, both throw-boxes and nodes first exchange meta data to negotiate which messages should be sent[13].

III. THROW-BOX DEPLOYMENT AND ROUTING FRAMEWORK

Throw-box deployment determines where to place throw-boxes, which affects contacts between throw-boxes and nodes. Routing specifies how data are forwarded among nodes and throw-boxes. Both throw-box deployment and routing can significantly affect the data delivery performance. Thus one needs to consider both issues to effectively enhance DTN capacity. To systematically study these problems, we present a throw-box deployment and routing framework in the paper. In this framework, we study the use of throw-boxes under various deployment and routing strategies. First, depending on the environment of interest, there may be various levels of flexibility in choosing the deployment and routing approaches. On one extreme, throw-box deployment might not be controllable due to constraints in the deployment area or the specific deployment mechanism used. Or routing may be constrained in requiring data to follow certain paths in the network for security reasons. On the other extreme, both routing and deployment can be customized for effective use of throw-boxes. Therefore, there is a need to develop solutions for these different cases. Second, to guide the use of throw-boxes, it is important to understand the performance implication of various routing and deployment approaches.

In the following, we will describe the performance objectives for throw-box usage. Then we present various deployment and routing approaches considered in this paper. Finally

[1]The results and algorithms obtained in this paper can be extended for the more general case where throw-boxes can be deployed to mobile entities.

[2]Throw-boxes may interact with each other and coordinate their actions to improve performance. However, as a starting point, we do not consider coordinated throw-boxes in this paper and defer it as future work.

we present our framework.

A. Performance Objective

Due to intermittent connectivity, DTNs may be significantly limited in supporting application needs because of low throughput and large delay. In this paper, we focus on improving throughput using throw-boxes. Specifically, we would like to maximize the total traffic demand that can be supported for a given number of throw-boxes[3]. Note that b_{ij} specifies the relative traffic demand between node i and node j, and $\sum_{i,j} b_{ij} = 1$. The objective of throw-box usage is to maximize λ such that traffic load of λb_{ij} from node i to node j is supported. In other words, we try to maximize the total traffic load which is λ.

Another important performance metric is message delay. In this paper, we choose to optimize the throughput for the following reasons. Given frequent partitions in DTNs, it would be more important to deliver messages successfully than to minimize the delay, if delivered at all. In addition, applications in DTNs are expected to tolerate relatively large delay. Further, maximizing the throughput might be in accordance with minimizing delay. Intuitively, if two nodes meet a throw-box frequently, both throughput and delay can be improved, as shown by the example in Section I and our simulation results.

B. Deployment Approaches

Throw-box deployment determines the location of throw-boxes in the network and consequently the connectivity between nodes and throw-boxes. We consider various types of deployment strategies, depending on how information about DTNs is used to optimize the placement of throw-boxes. Specifically, we consider the following deployment approaches.

- *Contact-oblivious deployment*. In this approach, throw-boxes are deployed without considering the contact opportunities between nodes and throw-boxes. Two examples are grid deployment in which throw-boxes are placed regularly in an area to form a grid, and random deployment in which throw-boxes are deployed to random locations in an area. Neither scheme requires any information about node contacts or traffic.
- *Contact-based deployment*. In this approach, throw-boxes are deployed to maximize the improvement in contact opportunities between nodes. Intuitively, by increasing contact opportunities between nodes, the data delivery performance would be improved. This approach applies to environments where contact information is available. In this paper, we consider two contact based schemes, depending whether we want to maximize the absolute or relative improvement, which will be described in detail in Section IV.
- *Customized deployment.* In this approach, throw-boxes are deployed specifically to optimize the performance for given traffic demand. Thus it applies to scenarios where both traffic and contact information are available.

[3]An alternative problem is to minimize the number of throw-boxes used to support given traffic demand. Both problems are equivalent in that solutions developed for one problem can be easily extended to solve the other.

C. Routing Approaches

Routing determines both the routing paths and the traffic load on each path. In this paper, we consider the following routing approaches, which are representatives for the various routing algorithms in DTNs.

- *Epidemic routing.* In epidemic routing, nodes try to exchange all buffered messages when nodes are in contact, essentially flooding messages throughout the network. This approach generates a large number of redundant messages, resulting in low utilization of network resources. But epidemic routing is also robust to network partitions and requires no information to operate. There is no explicit way to choose either the paths or the traffic load on each path.
- *Single path routing.* In this approach, messages for a source-destination pair would follow a single path. Thus the traffic load for a source-destination pair determines the traffic load on a specific path. We may have the flexibility to choose the set of nodes on the path.
- *Multi-path routing.* In this approach, messages may be forwarded along multiple paths for a source-destination pair.

D. Routing and Deployment Framework

In this paper, we systematically study how different routing and deployment approaches affect the data delivery performance in DTNs using throw-boxes. We develop algorithms for throw-box deployment and routing, and evaluate their performance.

Fig. 2 shows an overview of the routing and deployment framework. Specifically, we will study the six regions in the figure. In region I-1, we consider multi-path routing with customized deployment. In this case, we solve the joint problem of routing and deployment, which is formulated as a mixed integer linear programming problem. In region I-2, we consider multi-path routing with contact-oblivious and contact-based deployment. In this case, deployment and routing are determined independently. The routing problem can be solved as a linear programming problem. In region II-1 and II-2, we study single path routing, which is similar to multi-path routing. The difference is that data are forwarded along a single path between any pair of nodes. To solve the deployment and routing problem, we adapt the solution for the multi-path routing case by enforcing the single path constraint. Region III focuses on epidemic routing, where routing is fixed. We consider various types of deployment, including customized deployment (region III-1), and contact-based and contact-oblivious deployment (region III-2). Note that both contact-oblivious and contact-based deployment schemes are identical for all routing approaches because these schemes do not utilize traffic or routing information. In contrast, different customized deployment schemes are developed for various routing approaches.

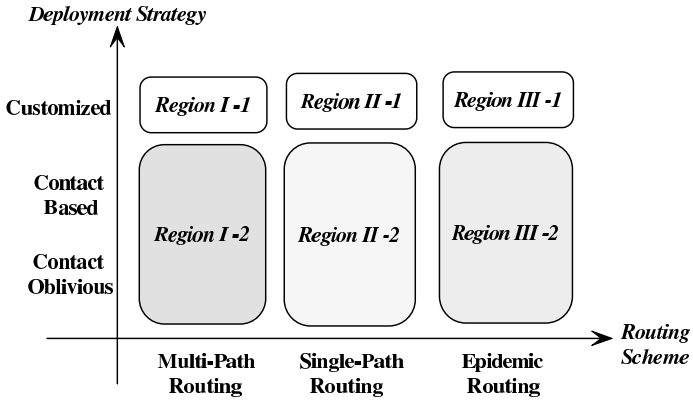


Fig. 2. Deployment and routing framework

In the following sections, we describe algorithms for throw-boxes deployment and routing for three routing approaches and present simulation results in Section VII.

IV. Capacity Enhancement with Multi-path Routing

In this section, we study the use of throw-boxes in multi-path routing. We first consider customized deployment where throw-boxes are deployed specifically to improve performance. We then study contact-based deployment. For contact-oblivious deployment, we have considered in Section III-B two simple schemes, namely random and grid deployment. In all these approaches, throw-box deployment is solved first and then routing is computed based on the deployment vector.

A. Multi-Path Routing with Customized Deployment

With customized deployment, throw-boxes are deployed to maximize data delivery performance. Thus, we need to consider the effect of routing in throw-box deployment. That is, we need to jointly consider the issues of deployment and routing. In the following, we first formulate this joint problem as a mixed integer linear programming (MIP) problem. We then develop a greedy algorithm to solve it.

1) Problem Formulation: Before presenting the formulation, we first describe how to determine the potential locations for throw-box deployment and characterize contacts between nodes. With customized deployment, a throw-box can be placed at any location in the area. However, solving the deployment problem in a continuous domain would be difficult because throw-boxes can be deployed at an infinite number of potential locations. Instead, we consider an approximation by dividing the entire area into a grid of cells and placing throw-boxes at the center of those cells. This approach becomes a good approximation of the arbitrary deployment when the cell size is small.

Nodes in DTNs communicate via intermittent contacts, whose occurrence is a time-varying process. To represent the transmission opportunities between two nodes, we define the *average capacity* as the maximum data rate that can be sent between two nodes in the long term. Let u_{ij} be the average contact duration between node i and j. Denote v_{ij} as the average inter-contact duration between node i and j. The average capacity can be computed as $c_{ij} = \frac{u_{ij}w}{u_{ij}+v_{ij}}$ where w is the average transmission data rate between nodes when they are in contacts. This is because node i and j are only able to communicate with each other for a fraction $\frac{u_{ij}}{u_{ij}+v_{ij}}$ of time. Note that the average capacity is shared by traffic in both directions. In addition, wireless interference is ignored in this model because of the sparse distribution of nodes. Similarly, we define the average capacity between a node and a throw-box.

N	the set of nodes
L	the set of potential throw-box locations
N^+	the set of nodes and potential throw-box locations
λ	total traffic load
b_{ij}	relative traffic demand from i to j, $i, j \in N$
m	number of throw-boxes
c_{ij}	average capacity between i and j, $i, j \in N^+$
x_i	number of throw-boxes deployed at location i
f_{ij}^{sd}	traffic from node s to d from i to j, $i, j \in N^+$

TABLE III
Notations used in the problem formulations

We now formulate the joint deployment and routing problem. Let N be the set of nodes and L be the set of potential throw-box locations. We denote N^+ as the union of N and L. Suppose that λ is the total traffic load between nodes. So $b_{ij}\lambda$ is the traffic load from node i to node j. Throw-box deployment is denoted by a deployment vector $\mathbf{x}$ where x_i is the number of throw-boxes that are deployed at location i, $i \in L$. Since we do not consider energy constraints of throw-boxes in this paper, it is not advantageous to deploy more than one throw-box at the same location. Thus the number of throw-boxes deployed at each location is set to be either 0 and 1. For traffic between a source-destination pair (s, d), the amount of traffic load forwarded from node i to j is denoted as f_{ij}^{sd}. A routing vector is the combination of f_{ij}^{sd}. Given m throw-boxes, the joint problem is to determine a deployment vector and a routing vector that maximize the total traffic load λ, which is shown in Fig. 3. The notations used are summarized in Table III.

Now we briefly explain the constraints in this formulation. Constraint (1) represents flow conservation for traffic between a source-destination pair, i.e., at every node or throw-box, except the sources and the destinations, the amount of incoming traffic is equal to the amount of outgoing traffic. Constraint (2) states that source s has no incoming traffic and λb_{sd} outgoing traffic. Similarly, constraint (3) restricts the amount of traffic to and from destinations. Constraint (4) requires that the total amount of traffic between node i and node j is no greater than the average capacity c_{ij}. Similarly, constraint (5) enforces the average capacity between nodes and throw-boxes. We use a factor of x_j to account for whether there is a throw-box deployed at a location. Constraint (6) states that the total number of throw-boxes is m. Constraint (7) specifies that the

maximize λ

Subject to

$$\sum_{i\in N^+} f_{ij}^{sd} = \sum_{i\in N^+} f_{ji}^{sd}, \quad s,d \in N, j \in N^+ - \{s,d\} \quad (1)$$

$$\sum_{i\in N^+} f_{is}^{sd} = 0, \sum_{i\in N^+} f_{si}^{sd} = \lambda b_{sd}, \quad s,d \in N \quad (2)$$

$$\sum_{i\in N^+} f_{id}^{sd} = \lambda b_{sd}, \sum_{i\in N^+} f_{di}^{sd} = 0, \quad s,d \in N \quad (3)$$

$$\sum_{s,d\in N} (f_{ij}^{sd} + f_{ji}^{sd}) \le c_{ij}, \quad i,j \in N \quad (4)$$

$$\sum_{s,d\in N} (f_{ij}^{sd} + f_{ji}^{sd}) \le c_{ij} x_j, \quad i \in N, j \in N^+ \quad (5)$$

$$\sum_{i\in L} x_i \le m \quad (6)$$

$$x_i \in \{0,1\}, \quad i \in L \quad (7)$$

$$f_{ij}^{sd} \ge 0, \quad i,j \in N^+, s,d \in N \quad (8)$$

Fig. 3. Multi-path routing formulation.

number of throw-boxes deployed at each location is an integer of 0 or 1. The last constraint restricts the amount of traffic load between nodes or between nodes and throw-boxes to be non-negative.

Note that the number of throw-boxes deployed at each location (i.e., x_i in Fig. 3) must be integers. So this formulation is an integer linear programming problem, which is generally NP-hard to obtain exact solutions. We next present a heuristic algorithm to solve it.

2) Greedy Algorithm: We have shown that solving the joint deployment and routing problem is difficult. Instead of finding an optimal solution which is computational expensive, we develop a greedy algorithm to solve this problem. In this algorithm, throw-boxes are deployed iteratively. At each stage, a throw-box is deployed to a location that maximizes the total traffic load. Algorithm 1 shows the sketch of the greedy algorithm. Initially, no throw-box is deployed. That is, $x_i = 0$ for all $i \in N$. Then the algorithm tries to deploy the first throw-box. For each potential location i, the algorithm will compute the achieved traffic load λ if the throw-box is placed at location i. Let h be the location that achieves the maximum λ. The algorithm will deploy the throw-box at location h. The algorithm will continue to deploy the second throw-box and repeat this process until all throw-boxes are deployed. Finally, with all throw-boxes being deployed, the algorithm will compute the routing vector.

In the greedy algorithm, we need to compute the maximum traffic load for a particular deployment. In step (4) in Algo. 1, we need to determine the resulting performance if a throw-box is placed at a potential location. In step (9), we need to compute the routing vector when throw-box deployment is finished. In both cases, the number of throw-boxes at each potential location is known. So the formulation in Fig. 3 becomes a linear programming (LP) problem. In fact, the computation of λ in these cases is a concurrent flow problem [14]. The concurrent flow problem is a classic problem in network flow and can be solved more efficiently than both the MIP and LP formulations. Let C be the computation complexity of solving a concurrent flow problem. The complexity of the greedy algorithm is $O(m|L|C)$ where $|L|$ is the number of potential locations.

Algorithm 1 Greedy algorithm for multi-path routing

1: $x_i = 0, i \in L$;
2: **for** $t = 1$ to m **do**
3: **for all** $i \in L$ **do**
4: Compute λ if a throw-box is deployed at location i;
5: **end for**
6: Let h be the location that achieves the maximum λ;
7: $x_h = x_h + 1$;
8: **end for**
9: Compute λ based on throw-box deployment;

B. Multi-Path Routing with Contact-Based Deployment

We now study multi-path routing with contact-based deployment. In this deployment approach, throw-box locations are chosen to maximize the improvement on contact opportunities between all pairs of nodes. Thus throw-box deployment is determined without consideration of traffic or routing. Depending on whether to maximize the absolute or relative improvement, we develop two contact-based schemes, as described in the following.

1) Absolute Contact Enhancement Algorithm: In this scheme, we try to maximize the absolute enhancement of contact between nodes. We define the *total contact capacity* between node i and j as the data rate that can be transmitted between i and j directly or via a throw-box. So the total contact capacity can be computed as $\sum_{h\in L} \min\{c_{ih}, c_{jh}\}x_h + c_{ij}$. Specifically, $\min\{c_{ih}, c_{jh}\}$ is the contact capacity between i and j via a throw-box at h. The factor x_h specifies whether there is a throw-box at location h. c_{ij} is the contact capacity between i and j without throw-boxes. Thus with throw-box deployment, the absolute contact enhancement for node pair i and j is $\sum_{h\in L} \min\{c_{ih}, c_{jh}\}x_h$. By combing all pairs of nodes, we have the absolute contact enhancement as

$$E_A(\mathbf{x}) = \sum_{i,j\in N} \sum_{h\in L} \min\{c_{ih}, c_{jh}\}x_h. \quad (9)$$

Now the deployment problem is to find a deployment vector $\mathbf{x}$ such that $E_A(\mathbf{x})$ is maximized. We solve this problem using a greedy algorithm. The greedy algorithm is similar to the customized deployment case illustrated in Algo. 1. The only differences are in step (4) and (6). In the absolute contact enhancement scheme, we computes E_A for each potential location (in step (4)) and denote h as the location that achieves the maximum E_A (in step (6)).

Note that for the absolute contact enhancement scheme, the greedy algorithm solves the deployment problem optimally. This is because at each stage, the greedy algorithm chooses the location that maximizes $E_A(\mathbf{x})$. After simple transformation, we have $E_A(\mathbf{x}) = \sum_{h\in L} \sum_{i,j\in N} \min\{c_{ih}, c_{jh}\}x_h$. So the deployment vector computed is optimal.

2) Relative Contact Enhancement Algorithm: In the relative contact enhancement scheme, we try to maximize the relative improvement for contacts between all pairs of nodes. Suppose that we compare the contact opportunities when the deployment vectors are $\mathbf{x_0}$ and $\mathbf{x_1}$ respectively. Consider a pair of nodes i and j. The total contact capacity between i and j is c_{ij}^0 when the deployment vector is $\mathbf{x_0}$. Similarly, let c_{ij}^1 be the contact capacity when the deployment vector is $\mathbf{x_1}$. Then the relative capacity enhancement for node pair i and j is $\frac{c_{ij}^1}{c_{ij}^0}$. By combing all pairs of nodes, we have the *relative capacity enhancement* as

$$E_R = \sum_{i,j \in N} \frac{c_{ij}^1}{c_{ij}^0}. \tag{10}$$

In the relative contact enhancement scheme, we try to deploy throw-boxes one at a time. at each stage, the algorithm tries to find a deployment location such that E_R is maximized. The operation of the algorithm is similar to the absolute contact enhancement algorithm except that the algorithm now maximizes E_R instead of E_A at each stage. Note that in this algorithm, E_R measures the improvement of contact capacity due to the deployment of each single throw-box.

V. Capacity Enhancement with Single-path Routing

In the previous section, we consider how to enhance network capacity with multi-path routing using throw-boxes. Now we consider the case of single-path routing in which messages for a source-destination pair follow a single path. For example, messages might be forwarded along the shortest DTN path [15].

In the following, we focus on customized deployment. We first formulate the the joint deployment and routing problem for single path routing. Then we develop an algorithm to solve this problem by extending the greedy algorithm for multi-path routing in Section IV. For contact-based and contact-oblivious deployment, the computation of routing is the same as in customized deployment.

A. Problem Formulation for Customized Deployment

As in multi-path routing, we consider the effects of routing on throw-box deployment and formulate the joint deployment and routing problem as an optimization problem. In single path routing, there is an additional requirement that a single path is used for messages between each node pair. To account for this single path constraint, we use binary variables z_{ij}^{sd} to record whether data between node pair (s, d) are forwarded from i to j, where i and j can be throw-boxes or nodes. We extend the MIP formulation in Fig. 3 by adding the following constraints.

$$f_{ij}^{sd} \le z_{ij}^{sd} c_{ij}, \qquad i, j \in N^+, s, d \in N \tag{11}$$

$$z_{ij}^{sd} \in \{0, 1\}, \qquad i, j \in N^+, s, d \in N \tag{12}$$

$$\sum_{j \in N^+} z_{ij}^{sd} \le 1, \sum_{j \in N^+} z_{ji}^{sd} \le 1, \qquad i \in N^+, s, d \in N \tag{13}$$

Specifically, constraint (11) states that data are not forwarded from i to j, where i and j are throw-boxes or nodes, if z_{ij}^{sd} is 0. Constraint (12) requires that z_{ij}^{sd} be 1 or 0. Constraint (13) enforces that data from each source-destination pair are forwarded along a single path.

As compared to multi-path routing formulation in Fig. 3, more integer constraints are introduced to enforce the single path requirement. Thus, this formulation is an MIP problem, which is difficult to solve optimally.

B. Greedy Algorithm for Customized Deployment

As compared to multi-path routing, single path routing has an additional constraint on the number of paths that can be used between each source-destination pair. We develop a greedy algorithm for single path routing by extending the counterpart for multi-path routing. Specifically, we modify the computation of total traffic load λ in step (4) or (9) of Algo. 1 as follows. We first compute λ and the routing vector $\mathbf{f}$ as in multi-path routing, i.e., solving a concurrent flow problem. Then we enforce the single path constraint by selecting a single forwarding path for each source-destination pair, as will be described below. By fixing these routing paths, we can compute λ and the routing vector again. This time the single path constraint is enforced in the resulting routing vector.

We now describe how to select a single path for messages between a source-destination pair. Suppose that $\mathbf{f}$ is the routing vector computed in the multi-path routing algorithm. In this paper, we choose the path that has the highest traffic load according to $\mathbf{f}$. That is, under multi-path routing, the selected path would carry the most amount of traffic among all paths between a source-destination pair. Consider the simple example in Fig. 4 where data are forwarded from node S to node D. The number along each edge represents $\mathbf{f}$, the traffic load between nodes. The path that achieves the highest traffic load is $S - 1 - 5 - D$. That is, the highest traffic load is 0.3.

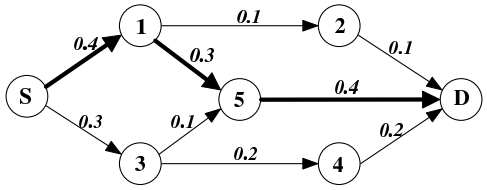


Fig. 4. Single-path aggregation based on multi-path routing

To find the path with highest traffic load, we use a binary search algorithm, which is illustrated in Algo. 2. We denote a path as a α path if all edges of this path have traffic load at least α. So the binary algorithm is to find the highest load α such that a α path exists. To determine whether a α path exists, the algorithm first generates a graph $G(V, E)$ based on the routing vector, where V consists all nodes and throw-boxes, and E consists of edges with traffic load at least α in $\mathbf{f}$. Then the algorithm determines whether s and d is connected in $G(V, E)$. If so, there must exist a path such that all edges of this path have traffic load at least α. That is, a α path exists. The algorithm conducts a binary search for the highest α.

Initially, the upper bound for α is the maximum load between source/destination and other nodes. And the lower bound for α is the minimum positive traffic load among all edges. The algorithm terminates when the gap between the upper and lower bounds is no greater than a predefined parameter α_{th}.

Algorithm 2 Compute single path with highest traffic load

$\alpha_{up} = \max\{f_{si}^{sd}, f_{id}^{sd} | i \in N^+\}$;
$\alpha_{low} = \min\{f_{ij}^{sd} | f_{ij}^{sd} > 0, i, j \in N^+, s, d \in N\}$;
while $(\alpha_{up} - \alpha_{low} > \alpha_{th})$ **do**
  $\alpha_{mid} = (\alpha_{up} + \alpha_{low})/2$;
  Generate $G(V, E)$ where $E = \{(i, j) | f_{ij}^{sd} \geq \alpha_{mid}\}$;
  if s and d is connected in $G(V, E)$ **then**
    Find a path $\mathbf{p}$ between s and d in $G(V, E)$;
    $\alpha_{low} = \alpha_{mid}$;
  else
    $\alpha_{up} = \alpha_{mid}$;
  end if
end while
Return path $\mathbf{p}$;
End

VI. Capacity Enhancement with Epidemic Routing

In this section, we investigate capacity enhancement when epidemic routing is used. Unlike multi-path or single path routing, epidemic routing floods messages throughout the network. This is no flexibility in selecting routing paths or determining traffic load on each path.

In the following, we study the case of customized deployment. To optimize the performance of epidemic routing, we need to take into account of how traffic is propagated under this routing approach. However, given traffic demand and throw-box deployment, it is hard to characterize the traffic load among nodes because the actual traffic load is affected by the contact occurrence between nodes. We observe that epidemic routing is able to exploit all paths available to propagate messages. So if we could improve the performance for data from all source-destination pairs, we could improve the performance of epidemic routing. Based on this observation, we develop two heuristics to compute throw-box deployment using the algorithms developed for multi-path routing.

A. Multi-path Heuristic

In this approach, we deploy throw-boxes to the same locations as in the multi-path routing case. That is, we use Algo. 1 to compute throw-box locations.

B. Proportion Based Heuristic

This heuristic is based on the assumption that messages are flooded throughout the network. Therefore, the contact capacity between nodes would be shared by messages from all source-destination pairs. In this heuristic, we allocate the contact capacity between nodes to data from all source-destination pairs according to their relative traffic demand. Recall that b_{sd} is the relative traffic demand from node s to d. Consider the contact capacity c_{ij} between two node i and j. The amount of traffic from source s to destination d is limited to $c_{ij}b_{sd}$. By replacing constraint (4) in Fig. 3 as follows,

$$(f_{ij}^{sd} + f_{ji}^{sd}) \leq c_{ij}b_{sd}, \qquad i, j \in N^+, s, d \in N$$

we can compute throw-box deployment for epidemic routing using Algo. 1.

VII. Performance Evaluation

In this section, we evaluate the performance of the deployment and routing algorithms presented in this paper using simulations. We aim to evaluate the utility of throw-boxes in enhancing network performance and understand how various deployment and routing approaches impact the performance. In the following, we first describe our simulation methodology and performance metrics in Section VII-A and then present our results in the following sections.

A. Experiment Methodology and Performance Metrics

In this paper, we simulate mobile DTNs with various node mobility patterns. Our simulations include two components: computation of deployment and routing vectors, and packet-level simulation using *ns* simulator [16]. In the computation component, we first calculate the statistics for contacts between nodes. Specifically, we generate the movement of nodes for the entire simulation duration and then compute the contact statistics. The connectivity between nodes in reality depends on many factors such as terrain, obstacles and interference, which are generally scenario specific. Without real measurement data, we adopt a simple approximation that radios have a circular range, i.e., nodes are able to communicate if the distance between them is less than the radio range. In addition, as described in Section IV, the radio data rate in the computation component measures the capabilities of links in transmitting application layer data, which exclude routing or MAC layer overhead. We determine the radio data rate by measuring the rate of application layer data that can be transmitted between two closely located nodes. The application data rate is 835Kbps when the radio bandwidth is 1Mbsp.

We then determine the set of potential locations for throw-box deployment using the cell based approach described in Section IV. Obviously there is a trade-off between the amount of computation time and the granularity of deployment locations. In our simulations, the cell size is 500m. To reduce the computation time in the customized scheme, we remove locations that are less likely to be used. Specifically, we use the absolute contact scheme to select the 50 best locations, which are used as the potential locations for the customized scheme.

With contact statistics, traffic and location information, we compute the deployment and routing vectors using the algorithms proposed in Section IV-VI. We use the GLPK linear programming tool [17] to solve the formulation in Fig. 3.

Based on the computed deployment and routing vectors, we run *ns* simulations. In the simulations, data are forwarded

according to the routes computed by the computation component. Specifically, to enforce the traffic load forwarded on each link, each node or throw-box records g_{ij}^k, the total amount of messages from source-destination pair k that has been forwarded on link (i, j). Let t_0 be the time after throw-box deployment. A new message from source-destination pair k is allowed to forward on link (i, j) if $g_{ij}^k \leq f_{ij}^k(t_0 + \Delta)$ where f_{ij}^k is the computed load and Δ is a system parameter. In other words, the actual load on each link would not be larger than the expected load by $f_{ij}^k\Delta$. In our simulations, we set Δ for link (i, j) to be the average inter-contact time between node i and j.

We evaluate the performance of the various deployment and routing schemes using two metrics, namely the message delivery ratio and delay. The *message delivery ratio* is defined as the ratio between the number of unique messages being delivered and the number of generated messages. We compute the message delivery ratio over the simulation duration. This metric measures how successful each scheme is in delivering messages. The *message delay* is the average time from the generation of a message to the earliest reception of the message at the destination. The message delay considers delivered messages only.

B. Simulation Settings

In our simulations, we use the following default settings unless specified otherwise. Nodes are simulated to move in a 25Km $\times$ 25Km area according to different mobility models, including predictable mobility as well as random mobility, as will be described in the following sections. Each result is averaged over five runs with different random seeds.

Nodes are sources and destinations of data communication. We consider a uniform traffic model where 20 nodes are chosen as sources with random destinations. Each source generates messages at the same data rate according to a Poisson process. Messages are of size 1500 bytes. The buffer size of both nodes and throw-boxes is 50000 messages.

Our simulations use the IEEE 802.11 MAC layer. The radio range and data rate are $250m$ and 1Mbps respectively. To discover other nodes for communication, each node broadcasts a beacon message every 5 seconds. After receiving a beacon message, nodes initiate an exchange of buffered messages. To avoid duplicate message transmissions, nodes first negotiate which messages should be sent. Suppose that node A tries to transmit messages to a neighbor B. Node A will first send an ADV message including information about messages it wants to transmit. Upon reception of the ADV message, node B replies with a REQ message which lists only messages it currently does not have. Then node A will send the messages listed in the REQ message. This way, messages will not be transmitted to a node multiple times. In addition, messages are transmitted from the storage in a FIFO order. When the buffer overflows, a node will drop the first message in the buffers.

C. UMass Mobility Model

We first consider the UMass bus network scenario [9], which consists of 9 buses equipped with radio transceivers to transport data. Each bus follows a fixed route and it takes about 60 to 90 minutes to finish one round of a route. Based on the traces of real bus movement obtained using GPS devices, we generate synthetic traces by adding variation into the bus movement. Specifically, the traces record the bus locations every minute. In our simulations, we assume that buses move at a constant speed during each one minute period. We compute the bus speed based on the traces and vary it by a random factor within [0.95, 1.05]. The buses will follow the same route repeatedly. Each simulation is run for a period of 40000 seconds in simulation time.

The UMass mobility model represents environments where node movement is predictable and constrained in space, i.e., along the roads and highways. Such model would be useful for studying many real life scenarios, where node movement is often not random.

1) Multi-Path Routing: We first consider the case with multi-path routing. Fig. 5 depicts the performance of various deployment schemes with different number of throw-boxes. The total traffic load is high at 334Kbps. The delivery ratio is shown in Fig. 5(a). We make the following observations. First, for the customized and contact based schemes, the delivery ratios improve significantly as the number of throw-boxes increases. For example, when the customized scheme is used, the amount of delivered data increases by a factor of 3 using four throw-boxes as compared to the case without using any throw-box. We also see that the customized scheme is more consistent than the contact based schemes, and achieves the best delivery ratio. This is because the customized scheme can utilize the specific traffic information to deploy throw-boxes. In contrast, the contact based schemes deploy throw-boxes based on the connectivity between nodes. Due to the irregularity of spatial node distribution in the UMass model, traffic between some nodes may not benefit from throw-box deployment. The relative contact scheme is shown to be better than the absolute contact scheme. Second, random or grid deployment does not affect the performance. This is because nodes only move on the roads in the area. Given the large span of the area, random or grid scheme tends to deploy throw-boxes to locations where nodes do not visit.

Fig. 5(b) shows the message delay. We can see that with customized and contact based deployment, the use of throw-boxes can significantly reduce message delay. For example, when the customized scheme is used, the message delay is reduced from 12000s to 6000s using four throw-boxes. This is because nodes are able to communicate with each other more frequently via throw-boxes. Thus messages can be delivered earlier. In addition, the customized scheme achieves the lowest delay, while the random and grid schemes do not affect the message delay.

Fig. 5(c) depicts the message delay when the traffic load is relative low at 83.5Kbps. It can be seen that the use of throw-

boxes can significantly reduce the message delay when the traffic load is low. With four throw-boxes, for example, the message delay is about 3100 seconds under the customized scheme. In contrast, the message delay is about 5800 seconds when no throw-box is used.

2) Single-Path Routing: We now consider the case of single path routing, in which data between a pair of source and destination follow a single path. Fig. 6(a) and (b) show the delivery ratio and delay respectively when the traffic load is high at 334Kbps. The results are similar to those of multi-path routing. For example, the use of throw-boxes can significantly improve the delivery ratio and reduce the message delay. One notable difference is that the customized scheme is able to achieve much better delivery ratios than the contact based schemes. This is because in single path routing, the utility of throw-boxes for data delivery between a source/destination pair depends on those throw-boxes that are able to support the highest data rate. With multi-path routing, however, the utility of throw-boxes depends on *all* throw-boxes that able to forward data for a source/destination pair. Thus, the performance of multi-path routing is less sensitive to the throw-box locations than that of single path routing. Fig. 6(c) shows the message delay when the traffic load is relatively low at 83.5Kbps. As to the case of multi-path routing, we observe a significant decrease of message delay by using throw-boxes for the customized scheme.

3) Epidemic Routing: We next evaluate the performance for epidemic routing. Fig. 7 depicts the results when the total traffic load is 334Kbps. From Fig. 7(a), we observe that the use of throw-boxes has limited benefit for improving the delivery ratio. All deployment schemes achieve similar delivery ratios. That is because of the poor utilization of resources in epidemic routing, which floods messages throughput the network.

Fig. 7(b) shows the result for the message delay. For the customized and contact based schemes, the use of throw-boxes reduces the message delay, e.g., by more than 20% when eight throw-boxes are used. The message delay also decreases with more throw-boxes. This is because throw-boxes provide shorter paths for messages. For random and grid schemes, there is no improvement in the performance. Fig. 7(c) shows the message delay when the traffic load is relatively low at 83.5Kbps. We can see that the use of throw-boxes leads to lower delay.

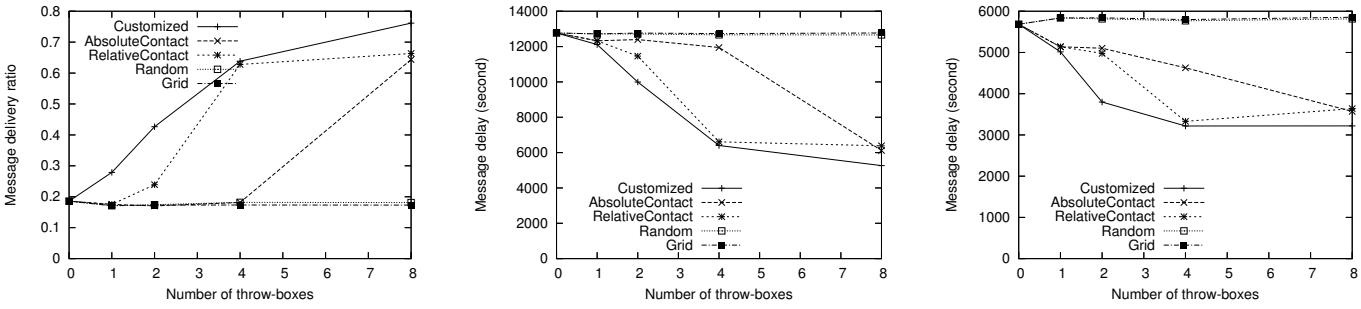
D. Random-Waypoint Mobility Model

In this section, we consider networks where nodes follow the random-waypoint mobility (RWP) model [18]. As compared to the UMass model where node movement is predictable and constrained in space, the RWP model represents the opposite with unconstrained and random node movement. Specifically, we simulate 40 nodes moving in a $25Km \times 25Km$ area. Each node selects a random destination location in the area and moves toward it with a random speed. After reaching the destination, nodes repeats this process. The maximum and minimum node speeds are 7.5m/s and 22.5m/s. Nodes do not pause between movement. We run the simulations for 80000 seconds in simulation time.

1) Multi-Path Routing: Fig. 8 depicts the performance of multi-path routing when the total traffic load is 125Kbps. Fig. 8(a) shows the delivery ratio. We make the following observations. First, both random and grid schemes improve the delivery ratio, which is different from the case of the UMass mobility. This is because of the random movement in the RWP model where nodes tend to visit all locations in the area. So even throw-boxes are deployed randomly into the area, nodes would benefit from the message relaying capability of throw-boxes. Second, the delivery ratio improves much more significantly with the customized and contact based schemes. For example, 3 times more messages are delivered with 16 throw-boxes than without throw-box. Third, the customized and contact based schemes achieve similar performance. With the contact based schemes, throw-boxes tend to be deployed near the center of the area because nodes visit this area more frequently in the RWP model. Due to the random movement of nodes, throw-boxes deployed in these locations would be able to relay messages for most source and destination pairs. So the performance of the contact based and customized schemes is not different.

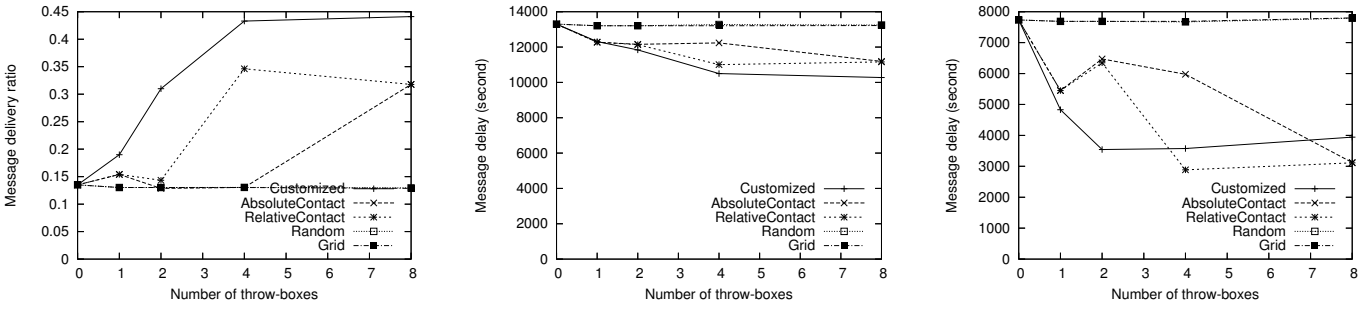
Fig. 8(b) shows the delivery delay. We can see that the use of throw-boxes can reduce the message delay. However, the reduction is much less than in the UMass case. This is because in the UMass model, node movement is much more regular. The use of throw-boxes can significantly increase the contact opportunities between nodes. In the RWP model, in contrast, nodes move randomly in the area. So with the use of throw-boxes, while nodes are able to communicate with more nodes via throw-boxes, the improvement for a specific pair of nodes is less significant. This also explains the large improvement on the delivery ratio and the much less improvement on message delay. Fig. 8(c) depicts the delivery delay when the total traffic load is low at 8.35Kbps. We can see that the use of throw-boxes does not improve the message delay in this case.

2) Single-Path Routing: We now study the case of single path routing. Fig. 9 depicts the performance when the total traffic load is high at 42Kbps. Fig. 9(a) shows the delivery ratio. First, random or grid scheme does not improve the delivery ratio for single path routing, which is different from the multi-path routing case. Second, the customized and contact based schemes improve the delivery ratio as more throw-boxes are used. And the customized scheme achieve the better performance. Both observations can be explained as in Section VII-C. Since data follow a single path in the case of single path routing, the delivery ratio is more sensitive to the choice of throw-box locations. So without taking into account of specific node mobility, random and grid schemes are not able to improve the performance. Similarly, without considering the traffic information, the contact based schemes perform worse than the customized scheme. Fig. 9(b) and (c) show the message delay when the total traffic load is high at 42Kbps and low at 8Kbps respectively. We can see there is no improvement in message delay for all schemes in both cases.



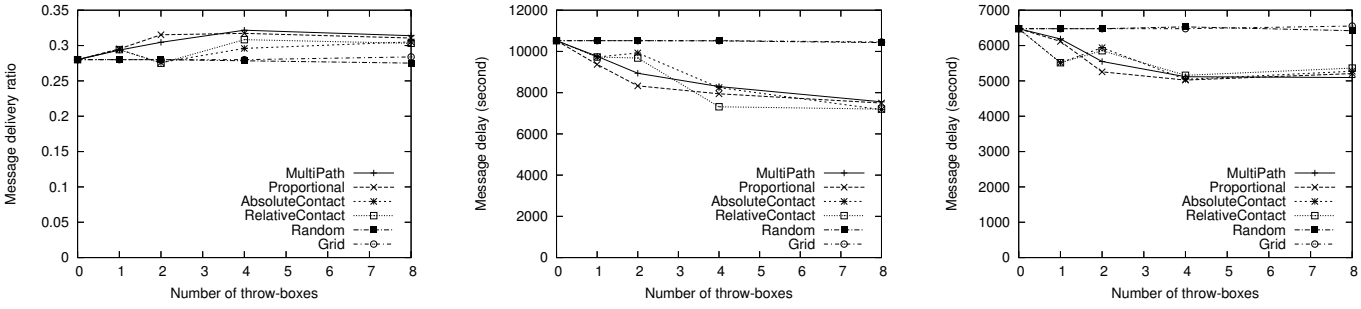
(a) Delivery ratio (b) Delay (high load) (c) Delay (low load)

Fig. 5. Multiple path routing performance with different number of throw-boxes under the UMass mobility.



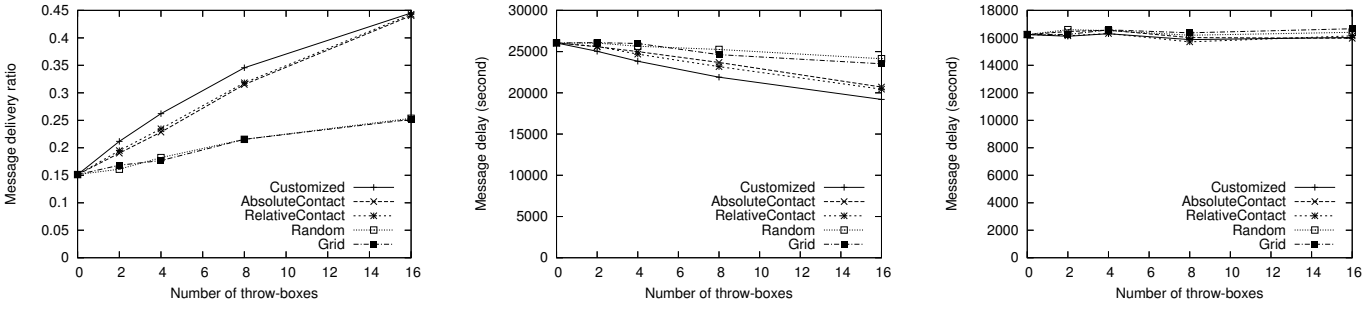
(a) Delivery ratio (b) Delay (high load) (c) Delay (low load)

Fig. 6. Single path routing performance with different number of throw-boxes under the UMass mobility.



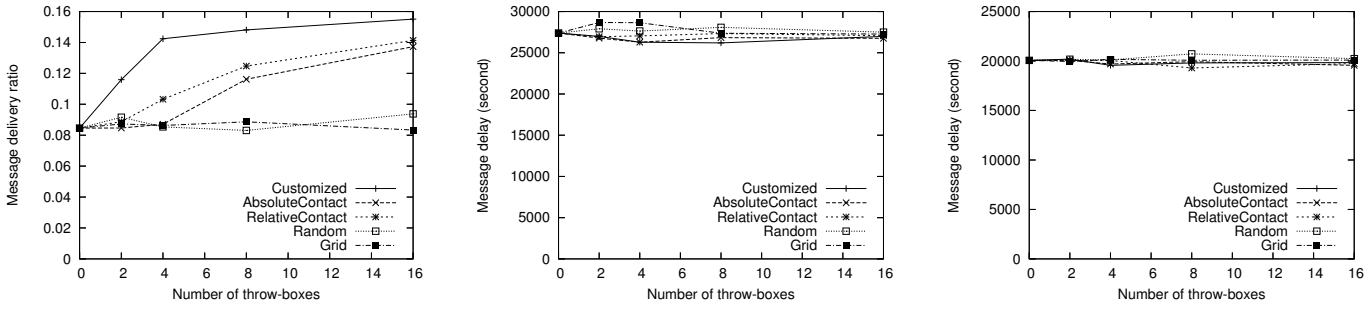
(a) Delivery ratio (b) Delay (high load) (c) Delay (low load)

Fig. 7. Epidemic routing performance with different number of throw-boxes under the UMass mobility.



(a) Delivery ratio (b) Delay (high load) (c) Delay (low load)

Fig. 8. Multiple path routing performance with different number of throw-boxes under the RWP mobility.



(a) Delivery ratio (b) Delay (high load) (c) Delay (low load)

Fig. 9. Single path routing performance with different number of throw-boxes under the RWP mobility.

3) Epidemic Routing: We also evaluate the performance of epidemic routing under the RWP model. Fig. 10 depicts the performance when the total traffic load is 42Kbps. As shown in Fig. 10(a), the delivery ratio increases for all schemes when the number of throw-boxes increases. In addition, random and grid schemes achieve modest improvement in the delivery ratio. As expected, the customized and contact based schemes have similar performance, both significantly improve the delivery ratio. The message delay is shown in Fig. 10(b). Under the relatively high load, we can see that the message delay is not affected by the use of throw-boxes. Fig. 10(c) depicts the message delay when the total traffic load is low at 8.4Kbps. The message delay decreases as more throw-boxes are used. For example, with 16 throw-boxes, the delay is reduced by 25% for the customized scheme. This is because of the low traffic load, epidemic routing is able to forward messages along the shortest paths. Thus with the use of throw-boxes and shortest paths, the message delay is reduced. In addition, the contact based schemes is very similar to the customized scheme in message delay. And both customized schemes, namely the multi-path heuristic and the proportional based heuristic, achieve very similar performance. Thus, we will focus on the multi-path heuristic in the rest of this section.

E. Manhattan Mobility Model

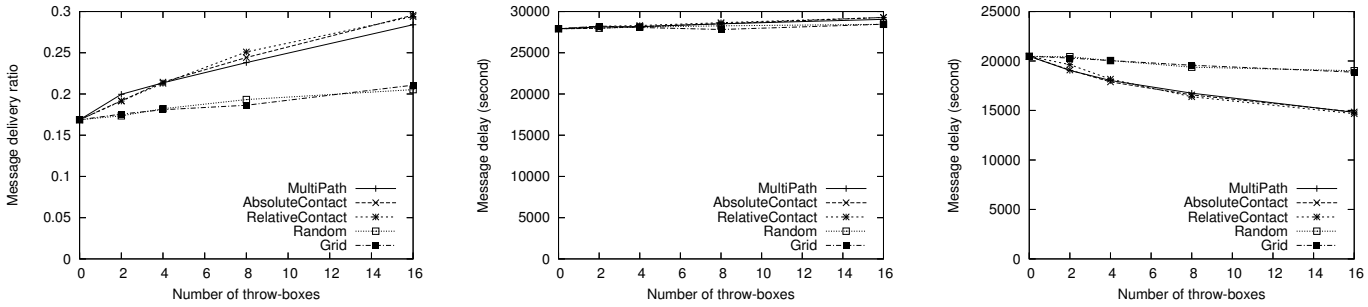
In this section, we consider the third mobility model, the Manhattan mobility model[4] which approximates the vehicle movement within a metropolitan area. In the Manhattan model, 20 nodes move along a grid of 10 $\times$ 10 highways in a 25Km $\times$ 25Km area. On each intersection, nodes will choose to go straight, turn left, or turn right with certain probabilities and move on a random speed. Node movement in this model is restricted to highways, but random in the movement directions, thus representing a middle ground between the predictable UMass model and the random RWP model. In our simulations, the minimum and maximum speeds are 7.5m/s and 22.5m/s respectively. The probability that a node will continue its direction on an intersection is twice the probability that the node turns. When reaching the boundary, nodes will turn left or right with equal probability, i.e., 0.5. The simulation time is 80000 seconds and the message timeout is 40000 seconds.

1) Multi-Path Routing: We first study the case of multi-path routing. Fig. 11 shows the performance when the total traffic load is relatively high at 251Kbps. As shown in Fig. 11(a), with the random or grid scheme, the use of throw-boxes has no effect on the delivery ratio. This is as expected because of the constrained movement of nodes. By deploying throw-boxes randomly or placing throw-boxes into a grid, nodes have few contacts with throw-boxes, thus no improvement on the performance. On the other hand, the delivery ratio increases linearly with the number of throw-boxes when the customized and contact based schemes are used. For example, the use of 16 throw-boxes increases the delivery ratio by a factor of 5 from 11% to 55%, . Such large improvement is achieved because node movement is more constrained in the Manhattan model than in the RWP, i.e., nodes move along the highways instead of in the whole area. Thus, by placing throw-boxes along the highways, the utility of throw-boxes are more evident than in the RWP case. In addition, we observe that the performance of the customized scheme is very similar to that of the contact based schemes, which is a result of the random and uniform node movement.

The message delay is shown in Fig. 11(b). As expected, random or grid deployment does not effect the delay. With the customized and contact based schemes, however, the delay decreases near linearly with the number of throw-boxes used. With 16 throw-boxes, the delay is reduced from about 27000 seconds to 19000 seconds. This is because with the throw-boxes, nodes are able to communicate with each other more frequently via the throw-boxes, which leads to earlier delivery of messages. Similar results are obtained for message delay when the traffic load is relatively low, as shown in Fig. 11(c).

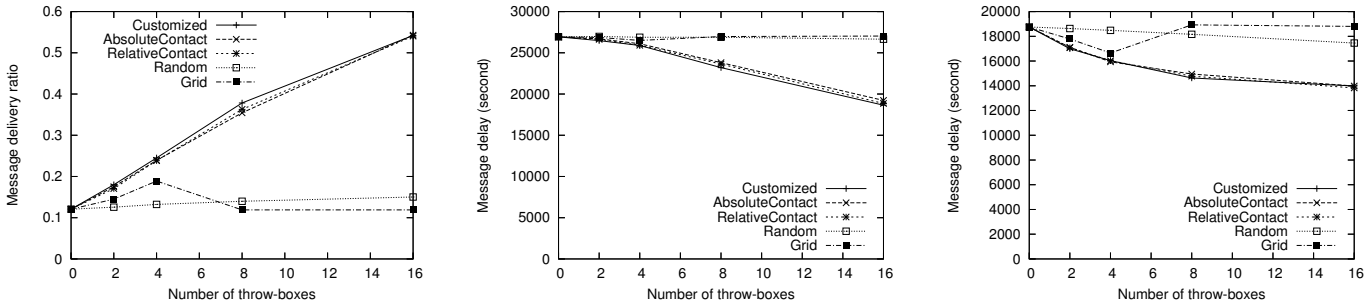
2) Single-Path Routing: We now examine the performance of single path routing in the Manhattan model. The delivery ratio of various schemes are depicted in Fig. 12(a). The total traffic load is 84Kbps. It can be seen that the customized scheme achieves the highest delivery ratio. And the contact based schemes deliver more data than the random or grid schemes. Overall, the improvement in the delivery ratio is less significantly as in the case of multi-path routing. The reason

[4]This model is slightly different from the Manhattan mobility in [19], which focuses on the impact of node mobility characteristics such as group mobility on MANET routing performance.



(a) Delivery ratio (b) Delay (high load) (c) Delay (low load)

Fig. 10. Epidemic routing performance with different number of throw-boxes under the RWP mobility.



(a) Delivery ratio (b) Delay (high load) (c) Delay (high load)

Fig. 11. Multiple path routing performance with different number of throw-boxes under the Manhattan mobility.

is that multi-path routing is better in exploiting the message relaying capabilities of all throw-boxes.

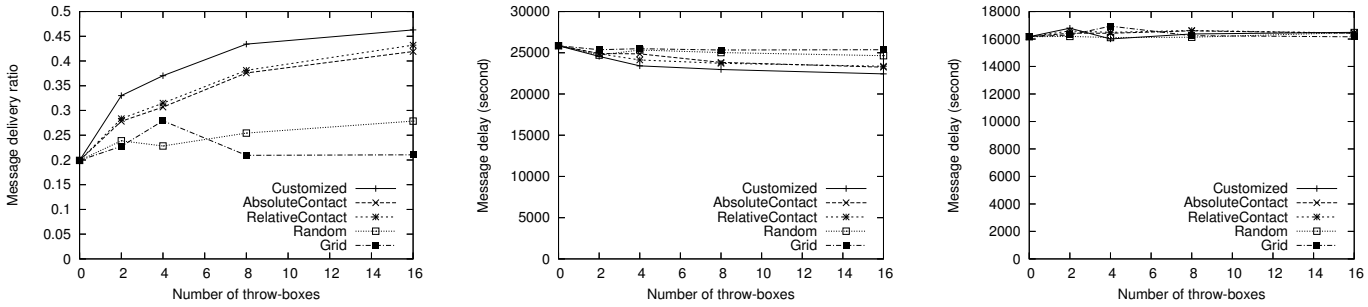
Fig. 12(b) shows the message delay. We can see that the delay is not significantly affected by the use of throw-boxes. This is because to have notable improvement on message delay, the deployment of throw-boxes should be able to shorten the routing paths for messages from most source and destination pairs. This is less likely to achieve for single path routing where data is forwarded along relatively "good" paths, i.e., paths that are able to support higher data rate. Such good paths are less likely to benefit from the deployment of throw-boxes. Similar results are shown for the case when the traffic load is relatively low, at 8Kbps, as depicted in in Fig. 12(c).

3) Epidemic Routing: We now study the performance of epidemic routing in the Manhattan mobility model. Fig. 13(a) shows the delivery ratio when the total traffic load is 42Kbps. We can see that for the customized and contact based schemes, the delivery ratios are similar and increase as more throw-boxes are deployed. However, due to the poor utilization of resources, the improvement is less than that in the multi-path or single path routing cases. And random and grid deployment of throw-boxes does not improve performance. Fig. 13(b) shows the message delay which is not affected by the use of throw-boxes. In contrast, when the traffic load is low, the message delay is reduced with the use of throw-boxes, as shown in Fig. 13(c). This is because when the traffic load is low, there is less contention for transmission opportunities and epidemic routing is able to forward messages along the shortest paths available. With the deployment of throw-boxes, nodes are able to communicate more frequently via throw-boxes, leading to shorter paths between sources and destinations. So the message delay is reduced.

F. Effect of Traffic Load

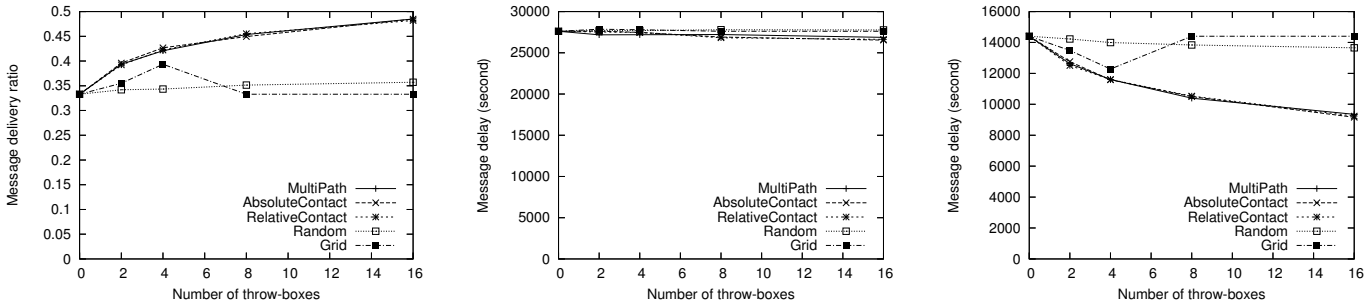
In this section, we examine the effect of traffic load on the network performance. As shown in the previous results, the customized scheme achieves the best performance. So we focus on this scheme in the rest of Section VII.

Fig. 14 depicts the performance of various deployment schemes under the UMass mobility model. The labels "t=4" and "t=0" in the figure represent the number of throw-boxes used. As shown in Fig 14(a), when the traffic load increases, the delivery ratio decreases for all routing approaches. With the use of throw-boxes, both multi-path and single path routing achieve higher delivery ratios, especially when the traffic load is high. This suggests that throw-boxes are very effective in enhancing network capacity. On the other hand, the performance of epidemic routing is insensitive to the use of throw-boxes. This is as expected because of the poor utilization of resources. Fig. 14(b) shows the message delay, which increases as the traffic load becomes higher. This is due to the increased contention for transmission opportunities. With higher load, messages have longer queue delay waiting for being transmitted to the next hop. We can see that the use of throw-boxes also leads to lower delay for all routing schemes, especially for multi-path and single path routing. These results



(a) Delivery ratio (b) Delay (high load) (c) Delay (low load)

Fig. 12. Single path routing performance with different number of throw-boxes under the Manhattan mobility.



(a) Delivery ratio (b) Delay (high load) (c) Delay (low load)

Fig. 13. Epidemic routing performance with different number of throw-boxes under the Manhattan mobility.

are consistent with the observations we make in the previous sections.

Fig. 15(a) and (b) depict the delivery ratio and delay respectively under the RWP model. We compare the performance when 8 or 0 throw-boxes are used. As expected, the use of throw-boxes increases the delivery ratio for all routing schemes. For message delay, multi-path routing is able to benefit from the use of throw-boxes when the traffic load is relatively high.

G. Effect of Transmission Range

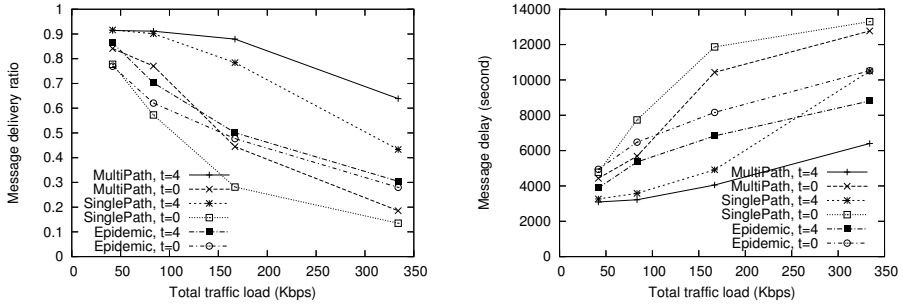
We now study how the radio transmission range affects the network performance. We simulate different radio ranges in *ns*, from 50, 100, 250, to 500 meters. Fig. 16 shows the results under the UMass model when the total traffic load is 334 Kbps. Fig. 16(a) shows the delivery ratio for the three routing schemes. With a larger radio range, nodes will have more opportunities for message transmission, resulting in better delivery ratios. The delivery ratio of epidemic routing increases at a slower pace when the radio range is large. This is due to the limitation of buffer space. Epidemic routing generates a large number of redundant messages and has a higher demand on buffer. In Fig. 16(b), the delivery ratios are normalized to the case with zero throw-box. So it represents the improvement of delivery ratio with the use of throw-boxes. We can see that with throw-box deployment, the delivery ratios for single path and multi-path routing are improved over different radio ranges. Epidemic routing is much less affected by the use of throw-boxes. In addition, the improvement is less significant when the radio range is small. Fig. 16(c) depicts the message delay. As expected, the message delay decreases when the radio range increases. This is because with more transmission opportunities, the queueing delay for messages is reduced. Throw-box deployment also reduces the message delay.

Fig. 17 shows the results when nodes follow the RWP mobility model. We can see that throw-box deployment enhances network performance, especially for multi-path routing. One notable difference is that the use of throw-boxes does not reduce message delay except for multi-path routing.

H. Effect of Link Bandwidth

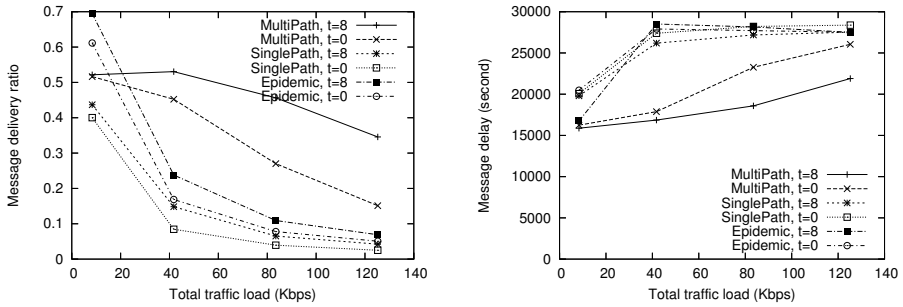
We now study the effect of link bandwidth or radio data rate on the network performance using throw-boxes. We use different radio data rates, from 200Kbps, 500Kbps, 1Mbps to 2Mbps. The application layer data rates are 177, 434, 835, 1552 Kbps. In these simulations, we set the total traffic load to be proportional to the radio data rate, i.e., the traffic load scales with the radio data rate.

Fig. 18(a) shows the performance results for the UMass mobility model. The traffic load is set to be 40% of the radio data rate. We can see that the delivery ratio remains the same under different link bandwidth for single path and multi-path routing. This confirms that the performance enhancement using throw-boxes is proportional to the link bandwidth. As to epidemic routing, the delivery ratio decreases as the radio data



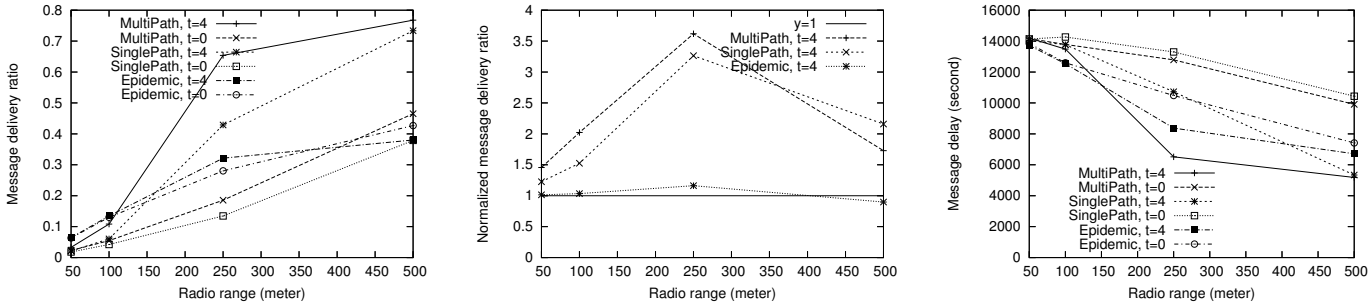
(a) Delivery ratio (b) Delay

Fig. 14. Effect of traffic load under the UMass mobility.



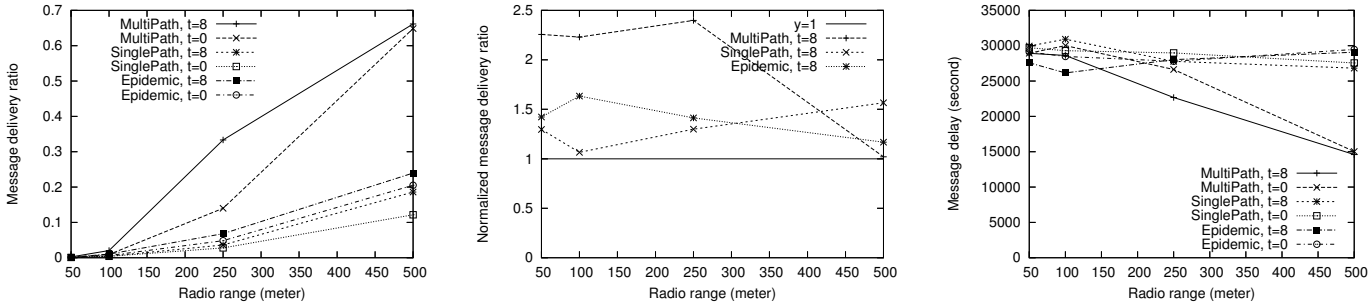
(a) Delivery ratio (b) Delay

Fig. 15. Effect of traffic load under the RWP mobility.



(a) Delivery ratio (b) Normalized delivery ratio (c) Delay

Fig. 16. Effect of transmission range under the UMass mobility.



(a) Delivery ratio (b) Normalized delivery ratio (c) Delay

Fig. 17. Effect of transmission range under the RWP mobility.

rate increases. This is partly due to the contention of buffers in throw-boxes and nodes. Since the traffic load scales with the radio rate in these simulations, the number of generated messages increases for a higher radio data rate, leading to more buffer overflows and message drops. So the delivery ratio is reduced. Fig. 18(b) and (c) show the message delay when the traffic load factor is 40% and 5% of the radio data rate respectively, which represent the scenarios of high and low traffic load in the network. We observe that the message delay is not affected by the radio data rate. This is because the message delay is mainly caused by the intermittent connectivity among nodes. An exception is that the delay for epidemic routing increases for higher radio data rates. As explained before, this is due to buffer contentions which causes message drops and shorter message lifetime in the network. Consequently the delay for delivered messages is decreased.

Fig. 19 depicts the results for the RWP model when the traffic load is 15% of the radio data rate. We can see that the delivery ratio and delay remain the same for all routing approaches, including epidemic routing. This is because the traffic load is less in these simulations. So the contention for buffer space is less severe. Similar results are obtained for the Manhattan model, which are omitted here for the sake of space.

I. Effect of Buffer Size

In this section, we study the effect of buffer size on the network performance. The buffer size varies from 10000, 20000, 50000, to 80000. Fig. 20(a) shows the delivery ratio for the UMass model when the traffic load is 334Kbps. We make the following observations. First, the delivery ratio remains the same for single path routing. This is because of the limited contact opportunities. The number of messages in the network is relatively low, so the availability of more buffer space does not improve performance. On the other hand, epidemic routing benefits from larger buffer sizes since messages are propagated throughout the network. So the delivery ratio increases as the buffer size increases. Multi-path routing performance improves as the buffer size increases from 10000 to 20000 but saturates when the buffer size is more than 20000. In addition, the performance using throw-boxes is better than the case without throw-boxes, except for epidemic routing with a small buffer size. The message delay is shown in Fig. 20(b). We can see that with larger buffer size, the message delay increases. This is because with more buffers, messages tend to stay in the network for a longer time. So the delay for delivered messages increases.

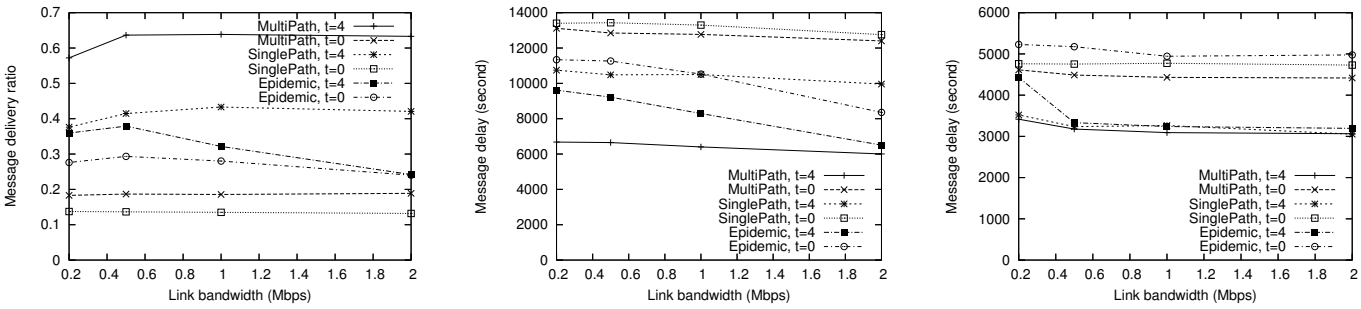
Fig. 20(c) shows the delivery ratio for the RWP model when the traffic load is 125Kbps. We can see that the delivery ratio generally is not affected by the buffer size. The reason is that under the RWP model, contact opportunities between nodes are fewer. The number of messages that nodes need to relay is much less than the case of the UMass mobility. So even the smallest buffer size in our simulations (i.e., 10000) is sufficient because the performance bottleneck is on the contact opportunities. Similar results have been obtained for the Manhattan model, which are omitted in the paper.

VIII. Summary of Results

In this section, we summarize the simulation results obtained in the previous section. Fig. 21(a) shows the conceptual performance enhancement using throw-box under various routing approaches and node mobility models. We consider the improvement in both message delivery ratio and delay. As shown in Fig. 21(a), the deployment of throw-box increases the delivery ratio for all scenarios considered in our study. The improvement is more significant for the cases with multi-path routing or the UMass mobility. We observe an increase in the delivery ratio by more than 3 with the deployment 4 or 8 throw-box. Since messages are forwarded along multiple paths, multi-path routing is more effective in exploiting the availability of extra capacity by the use of throw-box. For epidemic routing, the improvement is least significant because of the poor utilization of resources. Nodes in the UMass mobility model follow fixed routes periodically and visit the same locations frequently. So the deployment of throw-box enable nodes to communicate with each other more frequently via throw-box. In contrast, nodes in the RWP or Manhattan model follow random movement and do not visit the same location frequently, which is more acute in the RWP model as nodes move around the whole area. Thus throw-box provide less utility for relaying messages. This explains why the performance enhancement is generally more significant under the UMass mobility model than the RWP or Manhattan model.

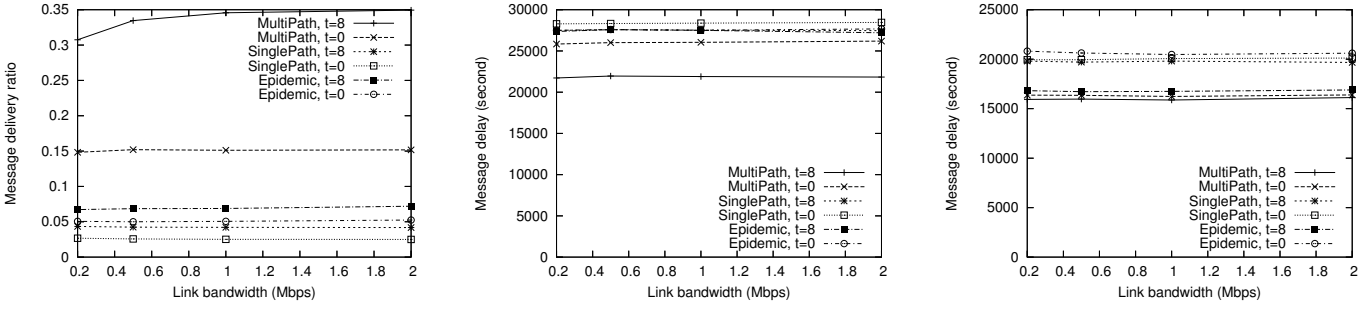
The use of throw-box has also been shown to reduce message delay in our simulations. We consider the cases with both high and low traffic load. When the traffic load is relatively high, the use of throw-box decreases the message delay for the cases with multi-path routing or the UMass model. This is in accord with the improvement in delivery ratio. For example, In the case of multi-path routing and UMass model, the message delay is reduced by 50% using 8 throw-box. The improvement in message delay is due to the fact that nodes are able to communicate with each other more frequently via throw-box, which leads to lower delay. We note that the delay of singe path routing does not improvement for the RWP and Manhattan models, even the delivery ratio is increased by the use of throw-box. In single path routing, the routing algorithm tries to choose paths that can support high data rates. In other words, messages tends to be forwarded along links with high capacity. In the RWP and Manhattan model, due to the random movement of nodes, while the use of throw-box enable more nodes to communicate, it does not increase the link capacity significantly, i.e., creating good links. So even with the use of throw-box, the message delay does not improve. In addition, because of the high traffic load and severe contention, epidemic routing in the RWP and Manhattan models is not able to reduce the message delay.

We obtain similar results on delay improvement when the traffic load is relative low. One exception is that the message delay for epidemic routing decreases for all mobility models.



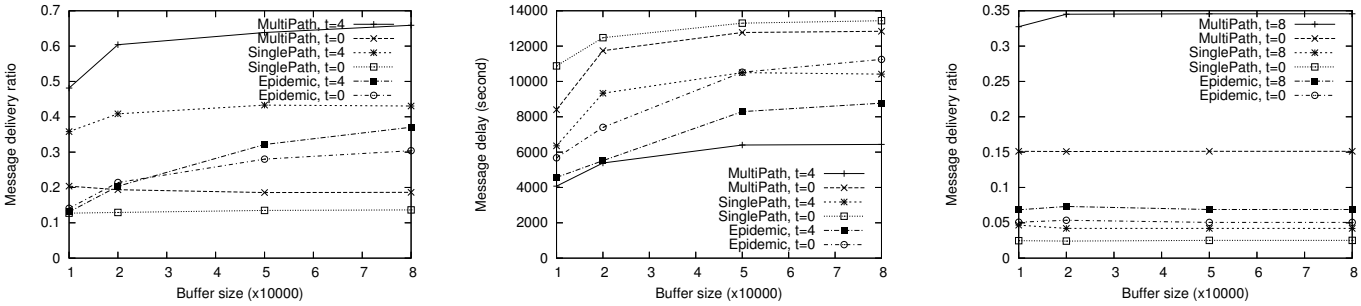
(a) Delivery ratio (b) Delay (b) Delay

Fig. 18. Effect of radio data rate under the UMass mobility.



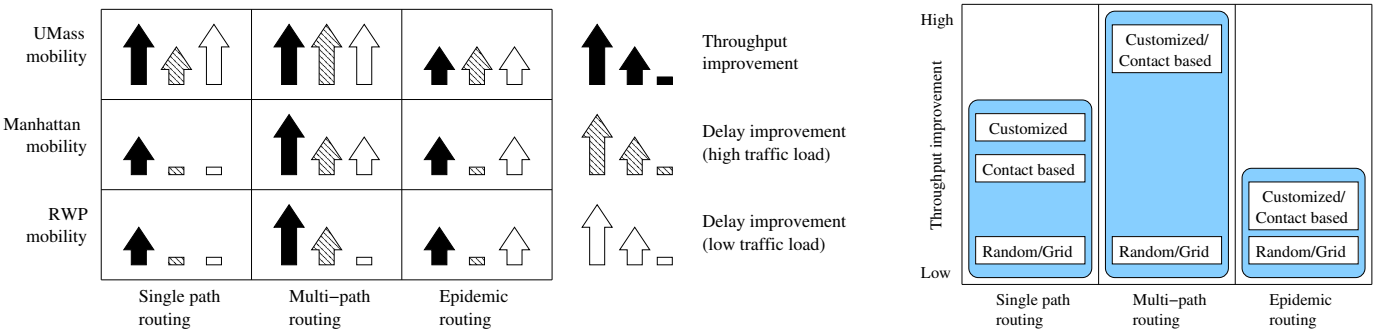
(a) Delivery ratio (b) Delay (b) Delay

Fig. 19. Effect of radio data rate under the RWP mobility.



(a) Delivery ratio (UMass) (b) Delay (UMass) (b) Delivery ratio (RWP)

Fig. 20. Effect of buffer size.



(a) Performance improvement using throw-boxes under customized deployment

(b) Conceptual performance of various deployment schemes

Fig. 21. Summary of simulation results.

This is because under low traffic load, epidemic routing is able to forward messages along the shortest paths via flooding. So the message delay is reduced when throw-box are used.

In our simulations, we also evaluate the various deployment schemes developed in the previous sections. Fig. 21(b) shows the conceptual performance under different node mobility models. First, for single path routing, we observe that random and grid schemes do not improve network performance. This confirms the intuition that given a relatively small number of throw-box, careful placement is critical to achieve good performance. On the other hand, both the contact based and customized schemes are shown to improve the delivery ratio. And the customized schemes achieve the better performance than the contact based schemes because of the use of traffic information. Second, for multi-path routing, the contact based and customized schemes have similar performance, both very effective in enhancing network performance. As expected, the random and grid schemes do not affect network performance except in the case of the RWP model, in which random and grid schemes achieve modest increase in delivery ratio. Third, for epidemic routing, the contact based and customized schemes have similar performance and the improvement is modest.

We can see that epidemic routing is least sensitive to the deployment locations of throw-box because of the poor usage of resources. On the other hand, both single path and multi-path routing require careful placement of throw-box to be effective, as evident in the performance gap between these schemes and the random/grid schemes. And the customized scheme is able to achieve better performance than the contact based schemes with single path routing. That is, single path routing is most sensitive to the deployment of throw-box among the three routing approaches. We also notice that the customized scheme performs better for multi-path routing under the UMass model, where node movement is not random or uniform. We expect that without consideration of traffic information, these schemes is less effective in situations where node mobility or traffic is non-uniform.

IX. Related Work

In this section, we review some related work in DTNs and node placement problems. DTNs are a class of networks that exhibit non-Internet-like characteristics, e.g., intermittent connectivity, large delay and high link loss. Examples of DTNs include military ad hoc networks [20], deep space communication [21], [22] and vehicular communication [23]. To achieve interoperability between various types of DTNs, Fall [1] proposes an architecture that is based on an asynchronous message forwarding paradigm. This architecture operates as an overlay above the transport layers to connect different DTNs. In [15], Jain et al. study unicast routing in general DTNs and develop several routing algorithms for scenarios where different levels of knowledge about network is available. The authors present a framework to evaluate these algorithms and find that efficient routing can be achieved using only limited amount of knowledge.

There have been also studies on mobile DTNs that exploit node mobility to deliver data. The proposed schemes can be generally classified as reactive schemes or proactive schemes. In *reactive* schemes, applications rely on movement that is inherent in the devices themselves to help deliver data. For example, Vahdat and Becker [13] propose Epidemic Routing in which mobile nodes carry data and exchange data when they meet, essentially flooding data throughout the network. In the Data Mules project, Shah et al. [5] propose to exploit mobile entities to transport data from sensors to access points, thus conserving energy in resource-limited sensors. In *proactive* schemes, devices move proactively and specifically in order to communicate with others. That is, these schemes enhance network capacity via proactive movement. Li and Rus [24] consider proactive movement of nodes to deliver messages in a disconnected environment and present an algorithm to compute optimal node trajectories.In the Message Ferrying project, Zhao et al. [7], [6] propose the use of special nodes called *message ferries* to provide communication services and exploiting controlled node mobility to improve routing performance. In recent work [3], Burns et al. study the problem of augmenting the capacity of a DTN through mobile agents which move specifically in the network to increase network performance. The authors present a control-based approach and develop multi-objective controllers to control the mobility of mobile agents. Other work includes [25], [26], [4], [27], [28], [29].

Goodman et al. [30] propose the *Infostation* architecture in which wireless ports called Infostations are deployed to provide high bit-rate connections in the their vicinities. Infostations could be placed at accessible locations such as airport and building entrances.

Node or sensor placement has been studied in wireless networks [31] or sensor networks [32], [33], [34], [35] to improve performance. In [31], the authors study the placement of access points to form a mesh network for Internet access. This work considers Internet traffic which aggregates at several Internet gateways and proposes greedy algorithms to solve it. The work in [32] studies the placement of relaying nodes in a sensor network. The authors focus on minimizing the number of relaying nodes required to maintain global connectivity. In [33], the authors consider the placement of sensor nodes and transmission structure for data gathering. The goals are to minimize energy consumption while satisfying the data distortion constraints. The work in [34] studies the issues of adding relay nodes and provisioning energy to existing nodes in sensor networks to prolong network lifetime. The authors formulate this joint problem and develop heuristics to solve it. Our paper differs significantly from this body of work. The previous work focuses on connected networks with stationary nodes. In contrast, the deployment of throw-boxes into a DTN does not necessarily form a connected network. Instead, throw-boxes are intended to improve performance by storing and relaying data between mobile nodes. In addition, this paper studies various types of routing approaches including epidemic and single path routing. which is seldom addressed

in the previous work. We also study various approaches for throw-box deployment. Furthermore, the transmission capacity between nodes depends on node movement which may vary over time. This is different from the constant link capacity in wireless networks.

Our work on throw-box deployment is also related to facility location problems which have been studied extensively in the operation research community [36]. The facility location problems generally consider the selection of facility locations to minimize cost for specified demand, which is different from our problem of throw-box deployment.

There is also a number of studies on deployment strategies in sensor networks [37], [38], [39]. This body of work focuses on achieving desired coverage in sensor networks instead of communication performance.

X. Conclusion

In this paper, we proposed the use of throw-boxes to improve data delivery performance in mobile DTNs. Throw-boxes are small and inexpensive devices equipped with wireless interfaces, which are deployed to relay data between mobile nodes. The use of throw-boxes enhances network capacity by increasing the opportunities that nodes communicate with each other. Being small and inexpensive, throw-boxes represent a flexible and cost-effective approach to enhance network capacity.

In this paper, we focused on the use of throw-boxes to improve throughput. We considered the issue of both throw-box deployment and routing. We present a framework to systematically study these two issues. In this framework, we studied various routing approaches, including single path, multi-path and epidemic routing. We also considered various deployment approaches, including customized, contact-based and contact-oblivious schemes. These routing and deployment approaches cover a wide range of scenarios. We developed algorithms to compute throw-box deployment and routing. Specifically, we formulated the joint deployment and routing problem for customized deployment as an MIP problem and solved it using a greedy algorithm. We also extended the greedy algorithm to the case of single path and epidemic routing

Using extensive *ns* simulations, we evaluated different routing and deployment approaches under different network conditions. We obtained the following findings. First, throw-boxes are very effective in improving throughput and delay, especially when node movement is regular or multi-path routing is used. With regular node mobility, if two nodes visit a location, they would visit this location periodically. Thus the deployment of throw-boxes can significantly improve communication between these nodes, leading to better performance. Multi-path routing achieves better performance because it forwards data via multiple paths and is able to exploit most or all throw-boxes for data relaying. On the other hand, by limiting data forwarding along a single path, single path routing is less effective in using throw-boxes for data relaying. And epidemic routing benefits less from throw-boxes because of the poor utilization of network resources. Second, throw-boxes are more effective in improving throughput than reducing delay. This is because by deploying throw-boxes to enhance the capacity of bottleneck links, the overall throughput is improved. The improvement on delay, however, is less significant because data delay consists of delay on each hop for all source destination pairs. So the reduction of delay on one or more bottleneck links has less effect on the overall data delay. Third, single path routing is found to be most sensitive to different throw-box deployment, while epidemic routing is least sensitive to throw-box deployment. This can be explained by the way each routing approach forwards data. Multi-path routing can use all paths available for data forwarding while single path routing limits data forwarding to a single path for each source destination pair. Thus throw-boxes placed at sub-optimal locations might provide partial utility to multi-path routing but no utility for single path routing. Epidemic routing is least sensitive to throw-box deployment due to the poor utilization of resources. As a result, the customized deployment schemes, which utilizes both traffic and contact information, achieves better performance than the contact based schemes when single path routing is used.

In the future, we plan to consider energy issues in the use of throw-boxes. When throw-boxes are powered by batteries, the amount of forwarded data is limited for each throw-box. So multiple throw-boxes may be deployed at a single location to support data relaying for a specified period of time. In addition, we will investigate throw-box failures and deployment errors. One interesting issue is to study how these factors affect the robustness of the network. Finally, we are interested in studying coordination among throw-boxes to further improve performance.

References

[1] K. Fall, "A delay-tolerant network architecture for challenged internets," in *ACM Sigcomm 2003*, 2003.

[2] I. Akyildiz, X. Wang, and W. Wang, "Wireless mesh networks: A survey," *Computer Networks Journal (Elsevier)*, March 2005.

[3] B. Burns, O. Brock, and B. N. Levine, "MV routing and capacity building in disruption tolerant networks," in *IEEE Infocom 2005*, March 2005.

[4] A. A. Hasson, R. Fletcher, and A. Pentland, "DakNet: A road to universal broadband connectivity," *Wireless Internet UN ICT Conference Case Study*, 2003.

[5] R. Shah, S. Roy, S. Jain, and W. Brunette, "Data MULEs: Modeling a three-tier architecture for sparse sensor networks," in *IEEE SNPA Workshop*, 2003.

[6] W. Zhao, M. Ammar, and E. Zegura, "A message ferrying approach for data delivery in sparse mobile ad hoc networks," in *ACM MobiHoc 2004*, Tokyo Japan, 2004.

[7] W. Zhao and M. Ammar, "Proactive routing in highly-partitioned wireless ad hoc networks," in *the 9th IEEE International Workshop on Future Trends of Distributed Computing Systems*, May, 2003.

[8] W. Zhao, M. Ammar, and E. Zegura, "Controlling the mobility of multiple data transport ferries in a delay-tolerant network," in *IEEE INFOCOM*, 2005.

[9] UMassDiesel project: http://signl.cs.umass.edu/diesel/.

[10] "http://www.xbow.com/products/xscale.htm."

[11] J. Sorber, N. Banerjee, M. D. Corner, and S. Rollins, "Turducken: Hierarchical power management for mobile devices," in *ACM MobiSys'05*, Seattle, WA, June 2005.

[12] V. Cerf, S. Burleigh, A. Hooke, L. Torgerson, R. Durst, K. Scott, K. Fall, and H. Weiss, "Delay tolerant network architecture," *draft-irtf-dtnrg-arch-03.txt*, July 2005.

[13] A. Vahdat and D. Becker, "Epidemic routing for partially-connected ad hoc networks," *Technical report, Duke University*, 2000.

[14] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall, 1993.

[15] S. Jain, K. Fall, and R. Patra, "Routing in a delay tolerant network," in *ACM Sigcomm 2004*, Portland, OR, 2004.

[16] Network simulator 2, http://www.isi.edu/nsnam/ns/.

[17] GNU Linear Programming Kit, http://www.gnu.org/software/glpk/glpk.html.

[18] D. Johnson and D. Maltz, "Dynamic source routing in ad-hoc wireless networks," in *ACM SIGCOMM*, August 1996.

[19] F. Bai, N. Sadagopan, and A. Helmy, "IMPORTANT: A framework to systematically analyze the impact of mobility on performance of routing protocols for ad hoc networks," in *IEEE INFOCOM*, San Francisco, 2003.

[20] "DARPA Disruption Tolerant Networking Program: http://www.darpa.mil/ato/solicit/dtn."

[21] I. F. Akyildiz, O. B. Akan, C. Chen, J. Fang, and W. Su, "Interplanetary internet: State-of-the-art and research challenges," *Computer Networks Journal (Elsevier)*, vol. 43, no. 2, October 2003.

[22] S. Burleigh, A. Hooke, L. Torgerson, K. Fall, V. Cerf, B. Durst, K. Scott, and H. Weiss, "Delay-tolerant networking – an approach to interplanetary internet," *IEEE Communications Magazine*, June 2003.

[23] H. Wu, R. Fujimoto, and G. Riley, "Analytical models for data dissemination in vehicle-to-vehicle networks," in *IEEE VTC 2004/Fall*.

[24] Q. Li and D. Rus, "Sending messages to mobile users in disconnected ad-hoc wireless networks," in *ACM MOBICOM*, August 2000.

[25] A. Beaufour, M. Leopold, and P. Bonnet, "Smart-tag based data dissemination," in *First ACM WSNA Workshop*, September 2002.

[26] A. Chakrabarti, A. Sabharwal, and B. Aazhang, "Using predictable observer mobility for power efficient design of sensor networks," in *Information Processing in Sensor Networks*, Palo Alto, CA, April 2003.

[27] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. Peh, and D. Rubenstein, "Energy-efficient computing for wildlife tracking: design tradeoffs and early experiences with ZebraNet," in *ASPLOS'02*, October 2002.

[28] Tara Small and Zygmunt Haas, "The Shared Wireless Infostation Model - A New Ad Hoc Networking Paradigm (or Where there is a Whale, there is a Way)," in *ACM MobiHoc*, June 2003.

[29] R. Wang, S. Sobti, N. Garg, E. Ziskind, J. Lai, and A. Krishnamurthy, "Turning the postal system into a generic digital communication mechanism," in *ACM SIGCOMM 2004*, August 2004.

[30] D. Goodman, J. Borras, N. Mandayam, and R. Yates, "INFOSTATIONS: A new system model for data and messaging services," in *IEEE VTC'97*, vol. 2, May 1997, pp. 969–973.

[31] L. Qiu, R. Chandra, K. Jain, and M. Mahdian, "Optimizing the placement of integration points in multi-hop wireless networks," in *IEEE ICNP*, 2004.

[32] X. Cheng, D. Du, L. Wang, and B. Xu, "Relay sensor placement in wireless sensor networks," *IEEE Transactions on Computers*, 2001.

[33] D. Ganesan, R. Cristescu, and B. Beferull-Lozano, "Power-efficient sensor placement and transmission structure for data gathering under distortion constraints," in *third international symposium on Information processing in sensor networks*, 2004.

[34] Y. T. Hou, Y. Shi, H. Sherali, and S. Midkiff, "On energy provisioning and relay node placement for wireless sensor networks," *IEEE Transactions on Wireless Communications*, vol. 4, no. 5, 2005.

[35] J. Pan, Y. Hou, L. Cai, Y. Shi, and S. Shen, "Topology control for wireless sensor networks," in *ACM Mobicom*, San Diego, CA, 2003.

[36] M. L. Brandeau and S. S. Chiu, "An overview of representative problems in location research," *Management Science*, vol. 35, no. 6, pp. 645–674, 1989.

[37] S. Dhillon, K. Chakrabarty, and S. Iyengar., "Sensor placement algorithms for grid coverage," in *International Conference on Information Fusion*, 2002.

[38] Y. Zou and K. Chakrabarty, "Sensor deployment and target localization in distributed sensor networks," *Trans. on Embedded Computing Sys.*, vol. 3, no. 1, pp. 61–91, 2004.

[39] T. Clouqueur, V. Phipatanasuphorn, P. Ramanathan, and K. K. Saluja, "Sensor deployment strategy for target detection," in *ACM WSNA'02*, Atlanta, GA, 2002.