On Optimal Packet Routing in Deterministic DTNs

Giovanni Neglia, Xiaolan Zhang, Jim Kurose, Don Towsley, Haixiang Wang

On Optimal Packet Routing in Deterministic DTNs*

Giovanni Neglia,
INRIA Sophia-Antipolis Méditerranée,
Sophia Antipolis FR-06902, France,
Email: giovanni.neglia@inria.fr

Xiaolan Zhang,
Dept. of Computer & Information Science
Fordham University, Bronx, NY,
Email: xzhang@fordham.edu

James F. Kurose, Don Towsley,
Dept. of Computer Science
University of Massachusetts, Amherst, MA 01003
Email: {kurose,towsley}@cs.umass.edu

Haixiang Wang,
Dept. of Computer & Information Science
Fordham University, Bronx, NY,
Email: ahawang@fordham.edu

Abstract

In this paper, we investigate the problem of determining the routing that minimizes the maximum/average delivery time or the maximum/average delivery delay for a set of packets in a deterministic Delay Tolerant Network, i.e. in a network for which all the nodes' transmission opportunities are known in advance. While the general problem with multiple sources and multiple destinations is NP-hard, we present a polynomial time algorithm that can efficiently compute the optimal routing in the case of a single destination or of a single packet that needs to be routed to multiple destinations.

1 Introduction

Deployed in areas with limited infrastructure support, many vehicular networks [5, 2, 9], rely on peer-to-peer connectivity between wireless radios to

*This paper is published in the proceedings of the IEEE 77th Vehicular Technology Conference: (VTC2013-Spring). The paper got one of the best paper awards at the conference.

support data communication. Due to limited transmission power, fast node mobility, sparse node density and frequent equipment failures, many such networks exhibit only intermittent connectivity, and can be characterized as *Disruption Tolerant Network* (DTN, or *Delay Tolerant Network*). End-to-end communication in DTNs adopts a so-called "store-carry-forward" paradigm [16]: a node receiving a packet buffers and carries the packet as it moves, passing the packet on to other nodes that it encounters. The packet is delivered to the destination when the destination meets a node carrying the packet.

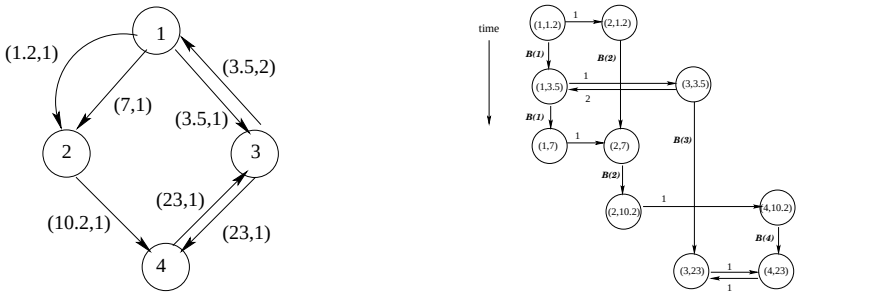
A plethora of works have proposed DTN routing schemes to operate in the case of zero or partial knowledge about future transmission opportunities (e.g. [15, 7, 14, 1]): some explored the trade-off between routing performance and resource consumption, while others attempted to optimize routing performance under certain resource constraints. On the other hand, other works took a different approach and studied the routing under the assumption of a full knowledge of network contacts, i.e. in *deterministic DTNs*. While such assumption is realistic in some specific scenarios like interplanetary networks [3] or transport networks [2], these works have also a theoretical interest, because they shed light on the fundamental hardness of the problem [1], and on best case performances [17, 4], and can be used in the learning phase of a practical scheme [6].

This paper falls within the second category and studies the packet routing problem in deterministic DTNs. Given a group of packets to be routed in such a network, we consider as performance metric both the delivery time, i.e. the absolute time at which a packet is delivered to the destination, and the delivery delay, i.e. the difference between the packet delivery time and the packet generation time at the source. For both metrics, we consider as goal both the minimization of the maximum value across all the packets and the minimization of the average value. This leads to four different optimization problems that we study in a common framework.

Our work build upon the seminal work by Hay and Giaccone [8], which uses the event-driven graph[1] reduction to map several DTN routing problems to flow problems in static graph, including the minimum delay path and the maximum bandwidth routing problems for a single source-destination pair. As the authors noted, the case of DTN routing with multiple source/destination pairs can be mapped to *multi-commodity flow problem*, which can be solved using linear programming if *fractional flows* are allowed. In contrast, in this paper we focus on the routing of unsplittable packets leading to *integral multi-commodity flow problems* which are much harder problems.

The main contributions of this paper are summarized as below:

[1] The space-time graph [12] is another approach to represent DTNs as static graphs that also captures both temporal and resource constraints of the contacts. Even though our discussion is focused on the event-driven graph, the results and algorithms also apply if space-time graph model is adopted.



(a) Temporal network representation, $\mathcal{T}(\mathcal{L})$

(b) Event-driven graph representation, $\mathcal{G}(\mathcal{L}, \mathcal{B})$

Figure 1: Graph representations of a DTN contact trace.

- We extend existing results to prove the NP hardness of the four optimization problems in the general case with multiple sources and multiple destinations.
- For the case of a single destination or a single packet to be delivered to multiple destinations, we show that a polynomial-time algorithm exists that can provide an optimal routing for three of the problems (all but minimizing the maximum delivery delay).
- We show how our algorithm can be used to efficiently characterize a real mobility trace collected from DieselNet.

The algorithm presented in this paper was originally developed in the framework of our research on network coding to calculate the minimum delivery time for a set of packets. For this reason a sketch of the algorithm appears in our work [18]. In comparison to [18], this paper investigates the complexity of a general class of problems, presents a more complete description of the algorithm, showing how it can be used to determine the actual scheduling that minimizes not only the maximum delivery time, but also the average delivery time and the average delivery delay.

The remainder of this paper is structured as follows. In Sec. 2, we introduce the network model and performance metrics considered in this paper. Sec. 3 presents the problems we study, the complexity results for the different cases and our algorithm. In Sec. 4 we show some characteristics of DieselNet traces obtained through our algorithm. Finally, Sec. 5 concludes this paper. Due to space constraints, the proofs can be found in the companion technical report [13].

2 Background

We consider a set of mobile nodes, denoted as $\mathcal{V}$, moving independently in a closed area. Each node is equipped with a wireless radio with a com-

mon transmission range so that when two nodes come within transmission range of each other—they have a *contact*—they can exchange packets. We refer to the list of node-to-node contacts, sorted in temporal order, as *a DTN contact trace*, denoted as $\mathcal{L} = l_1, l_2, l_3, ...$ Each contact, l_i, is a tuple $(t(l_i), s(l_i), r(l_i), b(l_i))$ where $t(l_i)$ denotes the time of the contact, $s(l_i)$ and $r(l_i)$ denote respectively the sending and the receiving node of the contact, and $b(l_i)$ denotes the number of packets that can be transmitted during the contact[2]. We assume each node can store an unlimited number of packets destined for itself, but can only carry a limited number of packets for other nodes. We represent the buffer constraint as a function, $\mathcal{B} : \mathcal{V} \to \mathbb{N}$ where $\mathcal{B}(u)$ is the number of relay packets that node u can carry.

A contact trace can be depicted as a *temporal network* [10], a multi-graph $\mathcal{T}(\mathcal{L}) = \langle \mathcal{V}, \mathcal{E} \rangle$ where each edge in $\mathcal{E}$ represents a contact $l \in \mathcal{L}$. The edge is directed and labeled with a pair, $(t(l), b(l))$, i.e., the time of the contact, and the number of packets that can be exchanged using the contact. For example, Fig. 1(a) illustrates the temporal network model for a contact trace of a DTN with four nodes during the time interval $[0, 24]$.

We use the contact trace in Fig. 1(a) to illustrate the construction of the event-driven graph [8] $\mathcal{G}(\mathcal{L}, \mathcal{B})$ given a contact trace $\mathcal{L}$ and buffer constraints $\mathcal{B}(\cdot)$ (the graph is shown in Fig. 1(b)). For each contact $l = (t, u, v, b) \in \mathcal{L}$, two nodes (u, t) and (v, t) are added to the graph $\mathcal{G}$, respectively denoting the sending and receiving event of the contact. A directed *inter-node edge* (depicted as a horizontal line in Fig. 1(b)), labeled with b, connects node (u, t) to node (v, t), denoting that up to b packets can be transmitted from node u to v at time t. If two consecutive contacts involving node u occur at t_1 and $t_2 (> t_1)$, a directed *intra-node edge* connecting node (u, t_1) to node (u, t_2) is added to graph $\mathcal{G}$, with a capacity equal to $\mathcal{B}(u)$, i.e., the maximum number of relay packets node u can store (this edge is depicted as a vertical line in the figure).

The following proposition is a restatement of Theorem 4 in [8]:

Proposition 2.1 *There is a feasible routing schedule for delivering K packets originated at node u immediately before t_1 to node v by time $t_2 (t_2 \geq t_1)$ under contact trace $\mathcal{L}$ and buffer constraint $\mathcal{B}(\cdot)$ if and only if there is a flow of value K from node (u, t_1) to node (v, t_2) in the event-driven graph $\mathcal{G}(\mathcal{L}, \mathcal{B})$.*

To see this, we note that the value of a flow on an inter-node edge equals the number of packets sent during the corresponding contact whereas the value of a flow on an intra-node edge corresponds to the number of packets being carried by the node during the corresponding time interval.

[2] Contacts can be *directed*, if two independent wireless channels are used for transmissions in the two directions, or *undirected*, if the same wireless channel is used for transmissions in both directions and the total capacity can be arbitrarily divided between them. We focus on the first case in this paper.

We assume that a set of packets $\mathcal{M} = \{m_1, m_2, ..., m_k\}$ propagates in the network. Each packet can be denoted as $m_i = (s_i, d_i, t_i)$, where s_i, d_i and t_i denote respectively the source, the destination, and the generation time of packet m_i[3].

In this paper, we focus on routing performance in terms of delivery time or delivery delay. Let the delivery time of packet m_i be T_i for $i = 1, 2, \ldots k$. We consider the following four metrics defined over the whole set of packets:

- The *Maximum Delivery Time* (MDT) is defined as the time instant at which the last packet in the set is delivered to its destination, i.e. $T_M = \max_{1 \leq i \leq k} T_i$.
- The *Maximum Delivery Delay* (MDD) is defined as $D_M = \max_{1 \leq i \leq k}(T_i - t_i)$.
- The *Average Delivery Time* (ADT) is defined as $T_a = \sum_{i=1}^{k} T_i/k$.
- The *Average Delivery Delay* (ADD) is defined as $D_a = \sum_{i=1}^{k}(T_i - t_i)/k$.

Depending on the specific application one or the other of these metrics may be more relevant.

3 Minimal Time/Delay Routing in DTNs

We want to determine the optimal routing for each of the four metrics defined above. We observe that minimizing ADT and minimizing ADD are equivalent problems because the two metrics differ only by a constant value $(\sum_i t_i/k)$. Apart from this, the three optimization problems (min MDT, min MDD, min ADT/ADD) are different (some examples are shown in [13]).

We start our study by establishing the NP-hardness of these problems in the general scenario with different sources and different destinations.

Proposition 3.1 *Given a contact trace $\mathcal{L}$, and a set of packets $\mathcal{M}$, the following problems are all NP-hard:*

1. *finding a feasible routing that achieves minimal MDT,*
2. *finding a feasible routing that achieves minimal ADT/ADD,*
3. *finding a feasible routing that achieves minimal MDD.*

The NP-hardness of MDT minimization has already been proven in [1] (Lemma 1). In the companion technical report [13], we use mainly the same technique to prove the other results.

In what follows we first present a polynomial-time algorithm (Sec. 3.1) to determine the routing that minimizes MDT in the special case where all

[3] A node can be source or destination for different packets.

packets are destined to the same destination. We then shows in Sec. 3.2 how this routing also minimizes ADT (or ADD), even if the two problems are not equivalent. We conclude this section by discussing a few extensions in Sec. 3.3.

3.1 MDT Routing Algorithm for Common Destination

Although finding the MDT for a set of packets is generally a NP-hard problem, this problem is solvable in polynomial time under the special case where all packets in the set are destined to the same node. We discuss this special case and present our algorithm in this section.

3.1.1 Preliminary

Under the assumption that every packet $m_i \in \mathcal{M}$ is destined to a common destination, we can denote packet m_i as (s_i, d, t_i), where d is the common destination.

We first discuss how to determine whether the set of packets $\mathcal{M}$ can be delivered given contact trace $\mathcal{L}$ and under buffer constraints $\mathcal{B}(\cdot)$ using the event-driven graph model. In order to answer this question, we extend the event-driven graph $\mathcal{G}(\mathcal{L}, \mathcal{B})$ as follows. First, we add a super source node *src* and a super destination node *dest* into the graph. Then, for each packet $m_i = (s_i, d, s_i) \in \mathcal{M}$, a *packet generation node* (s_i, t_i) is added into the graph, and connected to node $(s_i, t_{i,0})$ using an intra-node edge, where $t_{i,0}$ is the time of the first contact after t_i involving node s_i. Next, we connect node *src* to each *packet generation node* using an intra-node edge with capacity 1, denoting that one packet is generated at each source node[4]. We connect all nodes involving node d to node *dest* with an intra-node edge with a capacity of k. Finally, we also change to k the capacity of intra-node edges for the destination node d, as we assume nodes have sufficient buffer space to store packets destined for them. We denote the resulting graph as $\mathcal{G}(\mathcal{L}, \mathcal{B}, \mathcal{M})$. For example, Fig. 2 plots the event-driven graph for the set of packets $\{(1, 4, 0.5), (2, 4, 0.3)\}$ for the DTN trace depicted in Fig. 1.

Based on Proposition 2.1, the set of packets $\mathcal{M}$ can be delivered given contact trace $\mathcal{L}$ and buffer constraints $\mathcal{B}(\cdot)$ if and only if there is a flow of value k from *src* to *dest* in $\mathcal{G}(\mathcal{L}, \mathcal{B}, \mathcal{M})$. We therefore have the following proposition:

Proposition 3.2 *To determine the minimum MDT for the set of packets $\mathcal{M}$ given contact trace $\mathcal{L}$ and buffer constraints $\mathcal{B}(\cdot)$, it suffices to find $\mathcal{L}_{\min}$, the shortest left subsequence of $\mathcal{L}$ such that the event-driven graph $\mathcal{G}(\mathcal{L}_{\min}, \mathcal{B}, \mathcal{M})$ can support a flow of value k from src to dest. The time of*

[4]If within $\mathcal{M}$, there are multiple packets that are generated by the same source at the same time, we merge the identical *packet generation nodes* as one, and sum up the capacity of the corresponding edges.

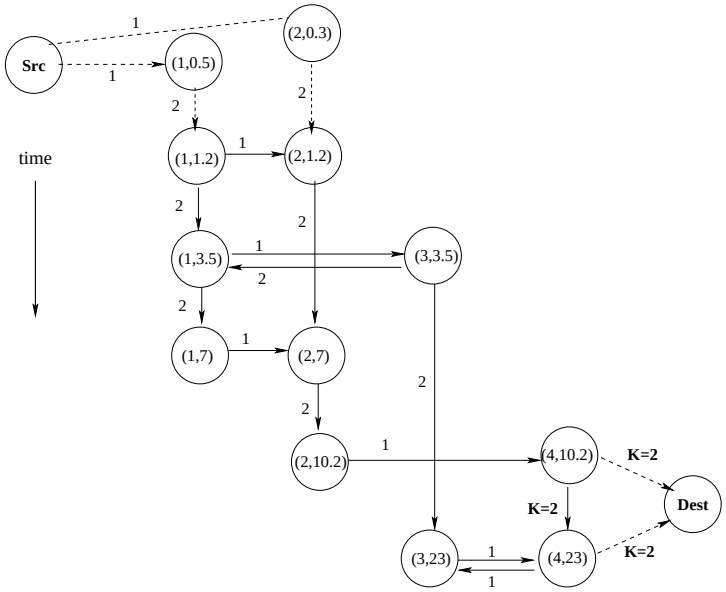


Figure 2: Event-driven graph $\mathcal{G}(\mathcal{L}, \mathcal{B}, \{(1, 4, 0.5), (2, 4, 0.3)\})$ for calculating minimum MDT schedule for packets $(1, 4, 0.5), (2, 4, 0.3)$, with $\mathcal{B}(u) = 2, u \in \mathcal{V}$. The newly added edges are drawn with dashed lines.

the last contact in $\mathcal{L}_{\min}$ is the minimum MDT, and this flow in $\mathcal{G}$ corresponds to a schedule that achieves the minimum MDT.

3.1.2 Algorithm Description

Our minimum MDT routing algorithm (Alg.1) intertwines the steps of searching for $\mathcal{L}_{\min}$ with the iterations of the Ford-Fulkerson algorithm for the maximum-flow problem [11].

Initially, no contact has been added into the event-driven graph, and $\mathcal{G}_f = \mathcal{G}(\emptyset, \mathcal{B}, \mathcal{M})$ contains a super source node *src*, k packet generation nodes $(s_1, t_1), ..., (s_k, t_k)$, and a super destination node *dest*, with the node *src* connected to each packet generation node with an edge of capacity 1. Subsequently, the algorithm iterates the *expand graph phase* and the *find max-flow phase* until the value of the flow reaches k or all contacts in $\mathcal{L}$ have been processed.

In the *expand graph phase*, the graph $\mathcal{G}_f$ is expanded by considering events from L in time order, until *FindPath* $(\mathcal{G}_f, src, dest)$ finds a new path with a non-zero residual capacity[5] (via breadth-first search) from node *src* to node *dest*. Here $Grow(\mathcal{G}_f, \mathcal{B}, l)$ expands $\mathcal{G}_f$ by adding the contact $l \in \mathcal{L}$, following the procedure described in Sec. 2.

Once a path is found, the algorithm enters the *find max-flow phase* where the flow is augmented until the max-flow from node *src* to *dest* in $\mathcal{G}_f$ is determined. The procedure *UpdateResidualGraph* $(\mathcal{G}_f, f, P)$ implements the following two steps of the Ford-Fulkerson algorithm: augmenting the flow f along path P and updating the residual graph $\mathcal{G}_f$. The return value b is the increment of the flow value due to path P. While the Ford-Fulkerson algorithm used here is not the most efficient max-flow algorithm, it allows us to incrementally augment the flow instead of starting the maximum flow calculation from scratch every time the graph is expanded.

When the outer while loop terminates, either the flow value reaches k or all contacts have been processed. The algorithm calls the procedure *ConstructScheduleFromFlow*(f) to construct a packet routing schedule from the flow f, and determines the MDT as the time of the last contact considered (if all k packets are delivered). The above two values are returned, together with the flow value supported (i.e., the number of packets that can be delivered at the end of the contact trace).

Let $\mathcal{L}'$ be the subsequence of the contact trace $\mathcal{L}$ considered up to termination, the computational complexity of Alg. 1 is $O(k|\mathcal{L}'|)$ (see [13] for the derivation).

[5] The residual capacity of an edge is the difference between its capacity and its current flow value, i.e. how much the flow can still be increased on that edge. The residual capacity of a path is defined as the minimum of the residual capacities of all edges in the path.

3.2 Minimum ADD/ADT Routing

In this section we prove that the routing determined by Alg. 1 also guarantees minimum ADD (or ADT). This is a consequence of the following result (the proof is in [13]):

Lemma 1 *Let $F(t)$ denote the fraction of packets delivered by time t with the routing returned by Alg. 1. Consider any other routing and denote as $G(t)$ the corresponding function. We have $F(t) \geq G(t)$ for $t > 0$.*

It follows from this lemma that the routing returned by Alg. 1 also guarantees minimum ADT (or ADD). In fact the average delivery time can be calculated as $\int_0^\infty 1 - F(t)dt$ and $\int_0^\infty 1 - G(t)dt$ respectively for the two routing. As $F(t) \geq G(t)$, we have $\int_0^\infty (1 - F(t))dt \leq \int_0^\infty (1 - G(t))dt$.

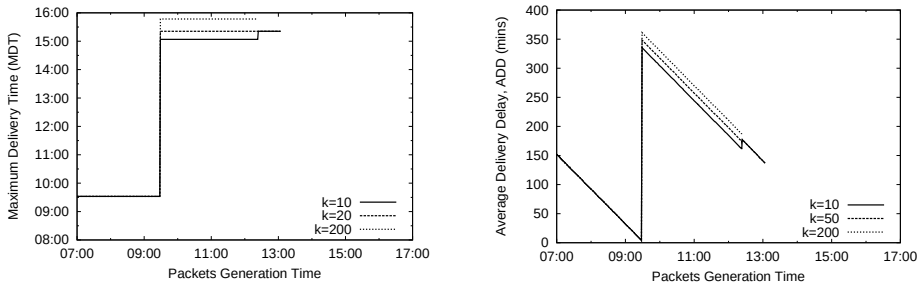
3.3 Extensions

We briefly mention a few extensions that are detailed in [13]. First, the same algorithm can be adapted to deal also with the case where a single packet has to be delivered to multiple destinations. Second, in these specific cases (single destination or single packet) other different optimization problems can be formulated on the event-driven graph as integer programming problems, that can still be solved in polynomial time through standard linear programming (because the solution of the relaxed problem can be guaranteed to be integer). Finally, similar NP-hardness results can be proven for the case of multiple multicast sessions.

4 Simulation Studies

In this section, we demonstrate how our algorithm allows us to characterize a contact trace. For our case study, we use the trace collected from the UMass DieselNet [2] 30-bus testbed in 2006. This trace details for each bus-to-bus contact the time of the contact, the transmitting bus, the receiving bus, and the number of bytes transferred. The number of packets that can be transmitted during the meeting has been obtained by dividing the total number of bytes by 1400.

We investigate the following question, given a file generated at a bus and destined to another bus, what is the minimum delivery time taking advantage of opportunistic transmission opportunities among the different buses? We answer this question by considering the minimal delivery time for a set of k packets with common source and destination, generated at the same time instant. Each packet has size 1400 bytes and the number of packets ranges between 10 and 200. We assume that the set of packets is not delivered by the end of day (i.e., 7 pm each day) will be dropped, or delivered by an alternative way (e.g. when the bus returns to the garage).



(a) Maximum delivery time

(b) Average delivery delay

Figure 3: Delivery time and delay for a set of packets sent from bus 3029 to bus 3038 at different time on 4/4/2006

In Fig. 3 we plot minimal MDT and ADD for a set of packets generated at bus 3029 and destined to bus 3038 at different time instants of the day. Due to the piece-wise linear property of these metrics in terms of packet generation time[6], we only need to use our MDT-routing algorithm on a finite number of generation time instants. As expected, the MDT exhibits a step-curve: later packet generation does not affect the MDT until a critical contact is lost and the routing has to change leading to a later delivery. In terms of delay the curve first decreases (the delivery time does not change but the packet generation time increases), and then jumps to a larger value when the routing changes. We observe that no set of packets generated after 1pm can be delivered by 7pm. The numerical results show how even optimal routing in DieselNet testbed exhibits large delays, that vary significantly according to the generation time.

Our algorithm can also be used to find out the minimal time to collect sensor data from multiple buses to a single sink. Due to the page limit, the result is not shown here.

5 Conclusions

In this paper, we have presented a polynomial time algorithm that can compute optimal routing for a group of packets in a deterministic Delay Tolerant Network in terms of maximum delivery time, average delivery time and average delivery delay. As future work we plan to compare this algorithm to other possible approaches (like more efficient maximum flow algorithms coupled with a binary search of the minimal length contact sequence) and to extend it to more general performance metrics that take into account also energy consumption.

[6]The piece-wise linear property in this case can be proved in a similar way to the single packet case as shown in [17].

References

[1] A. Balasubramanian, B. N. Levine, and A. Venkataramani. Replication Routing in DTNs: A Resource Allocation Approach. *IEEE/ACM Transactions on Networking*, 18(2):596–609, 2010.

[2] J. Burgess, B. Gallagher, D. Jensen, and B. N. Levine. MaxProp: Routing for Vehicle-Based Disruption-Tolerant Networks. In *IEEE International Conference on Computer Communications (INFOCOM)*, 2006.

[3] S. Burleigh, A. Hooke, L. Torgerson, K. Fall, V. Cerf, B. Durst, and K. Scott. Delay-Tolerant Networking: an Approach to Interplanetary Internet. In *IEEE Communications Magazine*, 2003.

[4] A. Chaintreau, A. Mtibaa, L. Massoulie, and C. Diot. The Diameter of Opportunistic Mobile Networks. In *CoNext*, 2007.

[5] A. Doria, M. Ud'en, and D. P. Pandey. Providing Connectivity to the Saami Normadic Community. In *International Conference on Open Collaborative Design for Sustainable Innovation (dyd02)*, Dec 2002.

[6] Z. Guo, B. Wang, and J.-H. Cui. Prediction Assisted Single-copy Routing in Underwater Delay Tolerant Networks. In *IEEE GLOBECOM*, 2010.

[7] Z. J. Haas and T. Small. A New Networking Model for Biological Applications of Ad Hoc Sensor Networks. *IEEE/ACM Transactions on Networking*, 14:27–40, February 2006.

[8] D. Hay and P. Giaccone. Optimal routing and scheduling for deterministic delay tolerant networks. In *WONS 2009 (International Conference on Wireless On-Demand Network Systems and Services)*, pages 27–34, 2009.

[9] B. Hull, V. Bychkovsky, Y. Zhang, K. Chen, M. Goraczko, A. K. Miu, E. Shih, H. Balakrishnan, and S. Madden. CarTel: A Distributed Mobile Sensor Computing System. In *ACM Conference on Embedded Networked Sensor Systems (SenSys)*, November 2006.

[10] D. Kempe, J. Kleinberg, and A. Kumar. Connectivity and Inference Problems for Temporal Networks. In *Journal of Computer and System Sciences, Special issue on STOC*, volume 64, 2002.

[11] J. Kleinberg and E. Tardos. *Algorithm Design*. Addison Wesley, united states ed edition, March 2005.

[12] S. Merugu, M. Ammar, and E. Zegura. Routing in Space and Time in Networks with Preictable Mobility. Technical Report GIT-CC-04-07, College of Computing, Georgia Institute of Technology, March 2004.

[13] G. Neglia, X. Zhang, J. Kurose, D. Towsley, and H. Wang. On Optimal Packet Scheduling in Deterministic DTNs. http://www-sop.inria.fr/members/Giovanni.Neglia/publications/neglia12algorithm.pdf.

[14] T. Spyropoulos, K. Psounis, and C. Raghavendra. Efficient Routing in Intermittently Connected Mobile Networks: The Multi-copy Case. In *ACM/IEEE Transactions on Networking*, volume 16, pages 130–143, 2007.

[15] T. Spyropoulos, K. Psounis, and C. S. Raghavendra. Spray and wait: an efficient routing scheme for intermittently connected mobile networks. In *SIGCOMM Workshop on Delay Tolerant Networking (WDTN)*, 2005.

[16] A. Vahdat and D. Becker. Epidemic Routing for Partially Connected Ad Hoc Networks. Technical Report CS-200006, Duke University, April 2000.

[17] X. Zhang, J. F. Kurose, B. Levine, D. Towsley, and H. Zhang. Study of a bus-based disruption tolerant network: Mobility modeling and impact on routing. In *ACM Conference on Mobile Computing and Networking (MOBICOM)*, 2007.

[18] X. Zhang, G. Neglia, J. Kurose, D. Towsley, and H. Wang. Benefits of Network Coding for Unicast Application in Disruption Tolerant Networks. *to appear in IEEE/ACM Trans. on Networking*, 2012.

Algorithm 1 MIN_MDT_ROUTING $(\mathcal{L}, \mathcal{B}, M)$, find minimum MDT routing for the set of packets, $\mathcal{M} = \{(s_1, d, t_1), (s_2, d, t_2), ..., (s_k, d, t_k)\}$, under contact trace $\mathcal{L}$ and buffer constraints $\mathcal{B}(\cdot)$

Input: $\mathcal{L}, \mathcal{B}, \mathcal{M}$
Output: S, k_d, T_M // *schedule, num. of pkts delivered, minimum MDT*

$\mathcal{L}_r \leftarrow \mathcal{L}$, $x \leftarrow 0$, $\mathcal{P} \leftarrow \emptyset$,
$f(e) \leftarrow 0$ for every edge e in $\mathcal{G}(\emptyset, \mathcal{B}, \mathcal{M})$
$\mathcal{G}_f \leftarrow \mathcal{G}(\emptyset, \mathcal{B}, \mathcal{M})$, // *initial residual network*
while $x < k$ and $\mathcal{L}_r \neq \emptyset$ **do**
  // *Expand Graph Phase*
  repeat
    // Expand graph until a contact to node d is found
    repeat
      $l \leftarrow$pop$(\mathcal{L}_r)$ // *extract next contact from* $\mathcal{L}_r$
      $\mathcal{G}_f \leftarrow$Grow$(\mathcal{G}_f, l, \mathcal{B})$, $\mathcal{G}_f \leftarrow \mathcal{G}'_f$
    until $r(l) = d$ // *until the receiving node of contact* l *is* d
    $P \leftarrow$FindPath$(\mathcal{G}_f, src, dest)$
  until $P \neq null$
  // *Find Max-Flow Phase*
  while $P \neq null$ and $x < k$ **do**
    $(\mathcal{G}'_f, f', b) \leftarrow$ UpdateResidualGraph $(\mathcal{G}_f, f, P)$
    $\mathcal{G}_f \leftarrow \mathcal{G}'_f$, $x \leftarrow x + b$
    $f \leftarrow f'$ // *update the flow*
    $P \leftarrow$FindPath$(\mathcal{G}_f, src, dest)$
  end while
end while
$S \leftarrow$ ConstructScheduleFromFlow (f)
$k_d \leftarrow x$ // *the number of pkts delivered*
$T_M \leftarrow t(l)$ // *time of last contact considered*
return S, k_d, T_M