

Optimal Routing and Scheduling for Deterministic Delay Tolerant Networks

David Hay
Dipartimento di Elettronica
Politecnico di Torino, Italy
Email: hay@tlc.polito.it

Paolo Giaccone
Dipartimento di Elettronica
Politecnico di Torino, Italy
Email: paolo.giaccone@polito.it

Abstract—Delay Tolerant Networks (DTNs), in which contacts between nodes come and go over time, is a promising approach to model communications in mobile ad-hoc networks, where scenarios of network partitioning and node disconnections are likely to happen. One of the most important challenges in such networks is how to route information and schedule transmissions, coping with the continuously changing network topology.

In this paper, we focus on a deterministic and centralized DTN in which the contact times between nodes are known in advance or can be predicted; this model is applicable for various real-life scenarios.

We provide a general framework for devising optimal routing algorithms to such networks under different objective functions and various real-life constraints (such as the available buffer and energy). The key insight is to model the DTN as an equivalent time-independent graph; this allows the usage of well-known algorithms and techniques to achieve optimal results. These algorithms can be also used as approximation for less certain settings or as benchmarks to evaluate other routing algorithms. In addition, we extended our framework to deal with long-lived DTNs in which contacts are periodic.

Our algorithms are demonstrated by simulations, based directly on real-life traces, showing capacity-delay tradeoffs and the influence of the constraints and periodicity on the achievable throughput of the network.

I. INTRODUCTION

In Delay (or Disruption) Tolerant Networks (DTN), end-to-end communications are not always possible on a direct routing path through the network, since the network connectivity is intermittent and at any instant the network may become partitioned. Thus, routing is realized by multi-hop transmissions, exploiting spontaneous connections among the nodes.

The DTN model is especially appealing in large mobile ad-hoc wireless networks (MANETs) that have no fixed infrastructure. In such networks, performance scalability is achieved by exploiting the mobility of the nodes in order to transmit data in a “store, carry and forward” manner [1]: data is sent by a node to a close-by node, which stores it in its buffer; then, the intermediate mobile node carries the data from one physical location to another and finally forwards it to its destination. This transmission paradigm is easily modeled through a DTN, in which contacts between two nodes appear in the DTN when the nodes are physically close-by.

The promising performance results and concise modeling have led to an extensive research on DTN during the last few years (e.g. [2]–[5]). At the core of this research line are routing and scheduling algorithms: at any given time, each node should find when and where to forward the data stored in its buffer so that it reaches the destination in a timely manner.

It is important to notice that the performance of the DTN are directly dictated by the inter-contact times between the nodes (which, in turn, are the result of the mobility pattern of the network). In particular, packet delays become comparable with these inter-contact times, implying that application running on DTNs should be tolerant to delays of order of minutes/hours.

Prime examples in which DTN networks are used is when “mobility comes for free”: vehicular networks [6] (in which data is carried over cars and buses), “pocket area networks” [7] (in which data is carried by people carrying small devices like PDAs), mixed ground/satellite networks and sensor networks. Note that the DTN model captures also scenarios in which some nodes are mobile and some nodes are fixed (e.g. mobile devices with fixed gateways).

This paper focuses on deterministic DTNs in which contact times are known (or can be accurately predicted) in advance. We note that this is not always the case, and we consider the problem of building the DTN from a specific mobility pattern as orthogonal to this paper. However, in most of the contexts, the mobility, even if random or not perfectly known, is quite constrained in the space (like, following some specific routes, or having some spatial limitations like roads) and in the time (like, following some time schedule or some habit rule). Very recently, [8] has shown that also human mobility, considered traditionally almost unpredictable, shows on the contrary simple and predictable time/space patterns, most of the time. In general, these spatial/temporal constraints are the key issues to consider when designing practical DTN networks, providing some level of performance guarantees (QoS) to the end users. Thus, investigating a deterministic DTN can serve as approximation and/or reference model for less certain scenarios.

A. Our contribution

In this paper we address the routing and scheduling problem in deterministic DTNs; namely, finding the sequence of nodes on which data should follow from the source to the destination,

and computing the contact times in which this data should be transmitted.

Due to the deterministic nature of our problem, we were able to devise a centralized optimization framework which computes polynomially the optimal performance of the network, in the spirit of evaluating the achievable level of QoS. When contact times are not known in advance or when centralized algorithms are not possible, our results can be considered as a reference benchmark for performance comparison. Taking a different perspective, given some QoS constraints, if our optimization framework is not able to find a solution compatible with them, based on full contact time knowledge and complete scheduling control, it is not possible to devise any scheme to satisfy such constraints.

In general, the core of our framework consists in constructing a time-independent graph expressing the chronological order between contacts. Then, optimization problems are solved on this graph using classic algorithms from Graph Theory. It is important to note that previous works (e.g. [2]–[5]) suggested tailored-made algorithms for each objective function, which works directly on a particular time-dependent graph, a.k.a. *DTN graph*. Both the time and space requirements of our construction are *linear* in the number of contacts, thus do not introduce extra complexity on the classical algorithms.

Our approach is preferable since it provides a general reduction to known problems and therefore can be easily applied to many objective functions and constraints as the need arises. For example, we are able to model easily the following performance costs: minimum delay (work-span), maximum bandwidth, minimum number of hops; in addition, we consider the following constraints: minimum bandwidth per flow, maximum buffer occupancy per node, and maximum available energy per node. Note that the last two constraints appear only for several scenarios in which DTN is usually considered: energy is very important for PDAs, cell-phones, but irrelevant for road vehicles; on the other hand, buffer sizes are not usually a constraint in wireless networks, but in some particular sensor devices it can be very small (like in iMotes [9]), or in cooperative networks with selfish users, there might be some limitations on the size of the buffer devoted to *altruistic* forwarding.

Moreover, our constructions can easily model the problem of multicasting and simultaneous source-destination flows as classical *maximum multi-commodity flow* and *minimum Steiner tree* problems. Thus, we can use the extensive research tackling these problems (and their many variants) in Graph Theory and apply it directly on DTN.

Finally, it is important to note that in this paper we deal only with *atomic contacts*, assuming that the contact durations are negligible with respect to the inter-contact times. This assumption is realistic in many scenarios, in which simultaneous contacts do not exist.

B. Previous work

Routing problems in DTNs have been extensively investigated in recent years (see [5] for a reasonable set of references

on these problems and other DTN-related design issues). In general, following the taxonomy in [10], routing algorithms can be classified either as *replication schemes*, which send many copies of the same data packet across the network, or *forwarding schemes*, which send only a single copy of each packet across the network. This paper deals with such routing algorithms and therefore in the sequel we will focus only on relevant results regarding forwarding schemes. It is important to notice that since in these schemes only a single copy is sent, their efficiency strongly relies on the *knowledge* of future contacts across the entire network.

Seminal paper [2] has defined the routing problem in DTN and addressed many design issues highlighting different approaches based on several levels of knowledge within each node. The original definition of the DTN graph, based on the same assumptions as in our work, appears in [2]; as we have already observed, such construction is very useful to define the routing problem, but, since it is based on a time-dependent graph, it requires the design of ad-hoc algorithms; indeed, to compute the minimum delay path in the DTN graph, [2] proposes a modified version of the Dijkstra algorithm.

An alternative modeling of the routing problem was proposed in [11], which defines a particular space-time graph on which the minimum delay path can be found exploiting a classic shortest path algorithm. Specifically, the space-time graph holds a layer of the original DTN graph for each time-slot, such that two nodes are connected at some layer t if and only if there was a contact between them at time t ; furthermore, copies of the same node are connected across layers. It is important to notice the size of the space-time graph (as well as the time it takes to run algorithms on it) depends on the overall time period T in which the DTN is active. This results, taking the example in [11], in a *weakly-polynomial* all-pairs shortest-path algorithm with complexity $O(n^3T)$, where n is the number of physical nodes. On the other hand, our construction is linear with the number of contacts and in fact does not introduce extra complexity to the classical (strongly-polynomial) algorithms. We note that a similar (weakly-polynomial) construction was considered also in Graph Theory research in the context of *network flows over time* (a.k.a. *dynamic networks*). Unlike the classical *static* network flow model, this model assumes flows requires predefined amount of time to traverse over an edge in the graph. As a consequence, the flow value over each edge can change over time in order to maximize the total flow. The model (and that construction) was first considered by Ford and Fulkerson [12]; references [13], [14] provide good surveys on the advances in this topic over the last decades.

More recently, [3] has considered the scenario in which the mobility pattern of the nodes is deterministic and periodic. As in [2], the authors computed the minimum delay (work-span) path using an extended version of Dijkstra's algorithm. Note that [3] is more focused on a novel hierarchical routing scheme, based on multilevel clustering, than on the routing algorithms.

An interesting approach, extending the model of periodic

mobility, has later been proposed in [4], which considers the minimum delay routing problem in the case of periodic but not deterministic mobility. Specifically, in this case each contact is characterized not only by a contact-time but also by a probability that the contact event will indeed occur. The solution of the problem is obtained using a construction that partially resembles ours. Indeed, [4] starts from a probabilistic space-time graph and then builds a probabilistic space-space graph which is time-invariant; on this new graph, the optimal routing is evaluated as a solution of a Markov decision process. Note that this construction has been proposed for the minimum delay path only.

Finally, [15] has considered the same deterministic mobility of our paper and has proposed some new algorithms to solve the routing problem (also in the multi-commodity flow scenario) taking into account similar constraints and performance costs as in this work. The algorithms in [15], which were derived from the algorithm in [2], run specifically on the DTN graph and their performance are not guaranteed to be optimal. On the contrary, our optimization framework works on a time-invariant graph, on which well-known algorithms are used to find optimal solutions for the problem and all its variants.

II. MODEL DEFINITIONS

Let $D = \langle V, E \rangle$ be a delay-tolerant network multi-graph (a.k.a. *DTN Graph*) in which V denotes the participating nodes and E denotes the timed-contact edges: each edge $e \in E$ has a label $t(e) \in \mathbb{R}^+$ which specifies its contact time; in addition, let $bw(e) \in \mathbb{R}^+ \cup \{+\infty\}$ denotes the bandwidth of the contact (namely, the number of bytes that can be transmitted between the nodes exploiting the contact). We assume that transmissions do not overlap in time or, equivalently, that $t : E \mapsto \mathbb{R}^+$ is an injective function and that transmissions are instantaneous.

In addition, we consider some resource constraints on the nodes of the DTN graph: for each $v \in V$, let $buf(v) \in \mathbb{R}^+ \cup \{+\infty\}$ denote the buffer size residing at node V and let $en(v) \in \mathbb{R}^+ \cup \{+\infty\}$ be the total amount of energy (battery) available at the node for transmissions.

The goal of the DTN is to deliver a certain amount of bytes from one node $s \in V$ to another node $d \in V$. The bytes are delivered along (possibly many) paths $P = \{p_1, \dots, p_m\}$ between s and d , which is the decomposition of the *flow* on D to paths. Each path is denoted by alternating sequence of nodes and edges: $p_i = (s, e_0, v_0, e_1, v_1, \dots, e_{i_n-1}, v_{i_n-1}, e_{i_n}, d)$. The time associate with the routing scheme is the maximum time associated with a link in one of its paths, namely $t(P) = \max\{t(e_j) \mid e_j \in p_i, p_i \in P\}$; this quantity is sometimes called also the *work-span* of the routing scheme. It is important to notice that since each edge is associated with a specific time, a path p_i dictates not only the routing flow of the data but also the *scheduling decisions* within each node. We call P *the routing scheme* of D and when it is clear from the context, with a slight abuse of notations, we also use P to denote its corresponding *flow*.

A *feasible* routing scheme P of x bytes between s and d must satisfy the following constraints:

- **Chronological Order:** For each $p \in P$ and for each $j \in \{0, i_{n-1}\}$, $t(e_j) \leq t(e_{j+1})$.
- **Flow Feasibility:** The corresponding flow of P is a legal flow on D , whose value is at least x .
- **Buffer Constraints:** Let $occ(v, t, P)$ be the number of bytes stored at node v at time t under routing scheme P . For all v, t , $occ(v, t, P) \leq buf(v)$.
- **Energy constraints:** Let $recv(v, P)$ and $sent(v, P)$ be the number of bytes received by node v under routing scheme P and the number of bytes sent by node v under routing scheme P . For every node v , $\alpha \cdot recv(v, P) + \beta \cdot sent(v, P) \leq en(v)$, where α and β are parameters of the problem instance. For simplicity, we normalize α and β such that $\alpha + \beta = 1$. In addition, since P is a flow, if $v \notin \{s, d\}$ then $recv(v, P) = sent(v, P)$, thus the energy constraint can be rephrased as $recv(v, P) \leq en(v)$ (unless $v = s$ or $v = d$).

III. EVENT-DRIVEN GRAPH CONSTRUCTION

Our primary method in order to deal with problems on DTN graphs is to reduce them into *time-independent* graphs which capture the original constraints of the DTN. Then, the routing and scheduling problems on the DTN can be easily solved on these graphs, using well-known algorithms and techniques (such as augmenting path algorithms and linear programming).

We first introduce the notation of an *event* in the DTN. In directed DTN, a node $v \in V$ is associated with a *sending event* at time t if there is an outgoing contact from v at time t . Similarly, a node $u \in V$ is associated with a *receiving event* at time t if there is an incoming contact to u at time t . Thus, each contact in a directed DTN will result in two events, one in its sending end and the other on the receiving end. In an undirected DTN, each contact will result in four events, since each end can both receive and send data.

The key concept of the construction of the time-independent graph is to represent each *event* in the system as a node. Then, two such event-nodes are connected if either they are the sending and receiving event of the same contact (we call this type of connections *inter-edges*) or if both events occur on the same DTN node and no event in that node occurs between them (we call this type of connections *intra-edges*).

Our construction is illustrated by a simple example in Fig. 1. We next given a formal definition of constructing the graph for a directed DTN; generalizing the construction for a undirected DTN will follow.

Given an directed DTN graph $D = \langle V, E \rangle$, let *event-driven* graph $G(D) = \langle V', E' \rangle$ be constructed as follows:

- **Nodes:** For each node $v \in V$, let $E_v \subseteq E$ be the set of edges touching node v and define the following set of nodes $V'_v = \{\langle v, t \rangle \mid e \in E_v, t = t(e)\}$. The set of nodes in the event-driven graph is defined as $V' = \bigcup_{v \in V} V'_v$. Furthermore, in the sequel, we say that a node $\langle v, t \rangle$ *belongs to a super-node* v if $\langle v, t \rangle \in V'_v$.

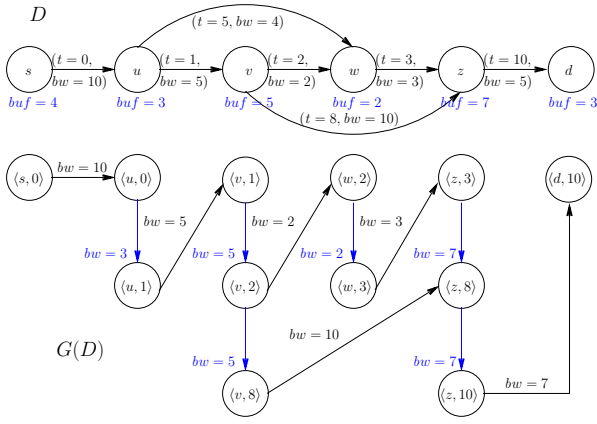


Fig. 1. Example of constructing event-driven graph $G(D)$ from a DTN graph D . In each edge, t denotes the time of the edge, and bw its bandwidth. For each node, buf denotes the buffer size. In $G(D)$, the vertical edges are intra-node edges, while other edges are inter-node edges. Note that the edge (u, w) is pruned.

- **Intra-node Edges:** For each set $V'_v \subseteq V \setminus \{s, d\}$, sort the nodes in ascending order according to their corresponding times; namely $V'_v = (\langle v, t_0 \rangle, \langle v, t_1 \rangle, \dots, \langle v, t_\ell \rangle)$, where $t_0 \leq t_1 \leq \dots, t_\ell$. For each $i \in \{0, \dots, \ell - 1\}$, add the edge $(\langle v, t_i \rangle, \langle v, t_{i+1} \rangle)$ bandwidth $buf(v)$ to E' .
- **Inter-node Edges:** For every $e = (u, v) \in E$, if $\langle u, t(e) \rangle \in V'$ and $\langle v, t(e) \rangle \in V'$, add the edge $(\langle u, t(e) \rangle, \langle v, t(e) \rangle)$ to E' . The bandwidth of the edge is $bw(e)$.

The event-driven graph is not a time-dependent graph, so all its edges “exist” all the time. Furthermore, a simple optimization can prune nodes of $\langle v, t \rangle \in V'$ for which there is no outgoing contact from super-node v after time t .

We next extend the construction to undirected DTN graphs. A simple approach to handle undirected DTN graph could have been treating each undirected contact edge as two anti-symmetric directed edges occurring at the same time; however, this will result in a DTN graph for which the time function $t : V \mapsto E$ is not injective as required. We take a more refined approach in which we characterize each event not only by its node-time pair, but also by its *role* (namely, sending or receiving event) and treating a sending event at time t at node v as if it occurs *just before* the receiving event at time t at node v . This implies that we assume that when a node u sends data to a node v , it immediately evacuates its buffer and therefore can simultaneously receive and store the same amount of data from v . Fig. 2 illustrates how to convert a single undirected contact to its corresponding event-driven sub-graph; notice that each node in the event-driven graph is represented by a triplet corresponding to the super-node, time and role (0 for sending event and 1 for receiving event). The formal definitions are omitted from this paper due to space considerations.

We proceed by providing the following properties of event-driven graphs:

Property 1: If $D = \langle V, E \rangle$ is a directed DTN graph, then the number of nodes of its corresponding event-driven graph

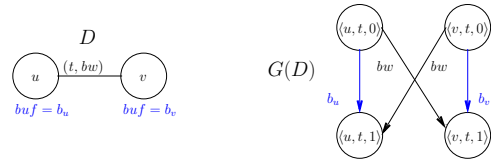


Fig. 2. Example of constructing event-driven graph $G(D)$ from a delay-tolerant network D with a single contact.

is at most $2|E|$ and the number of edges is bounded by $3|E|$. If $D = \langle V, E \rangle$ is undirected DTN graph, then the number of nodes is bounded by $4|E|$ and the number of edges is at most $6|E|$.

Property 2: For any DTN graph D (directed or undirected), the event driven graph $G(D)$ is a *directed acyclic graph* with maximum in-degree 2, out-degree of 2 and total-degree of 3.

Property 3: The construction of an event-driven graph $G(D)$ from a DTN graph $D = \langle V, E \rangle$ takes $O(|E|)$ time and requires $O(|E|)$ space.

Finally, we note that the above-mentioned construction does not deal with the *energy* constraints of the DTN. A discussion on these constraints and how to incorporate them in the event-driven graph will be presented in Section IV-C.

IV. ROUTING AND SCHEDULING ALGORITHMS

We demonstrate how to use the event-driven graph in order to derive routing algorithms for DTNs. The algorithms will differ from each other in their objective function.

A. Minimum Delay (Work-span) Path

In this section, the goal is to find a feasible routing scheme path P with strictly positive bandwidth for which $t(P)$ is minimal (that is, its last contact e has the minimal possible time value of associated with it). Since the only requirement is that the bandwidth will be strictly positive, it is easy to show that it is sufficient to look at routing schemes which consist on only a single path $P = \{p\}$. In [2], the authors gave a tailored-made modification to Dijkstra’s algorithm in order to solve this problem directly on DTN graph. We will take a different approach and solve the problem using the event-driven graph.

The path can be found by adding a global source node s' to $G(D)$ and connect it with all nodes belonging to super-node s . Then, by performing any graph traversal algorithm—e.g., breadth first search (BFS)—from node s' , one can easily compute which nodes are reachable from s . Let $R_d \subseteq V'$ be the nodes that are reachable from s' and belongs to super-node d , and let t be the minimum time associated with nodes in R_d . Thus, the minimum delay is t and the minimum delay path is the path from s' to $\langle d, t \rangle$ as was discovered in the graph traversal (for example, the path from s' to $\langle d, t \rangle$ in the BFS tree). The time and space complexity of this algorithm is $O(|E|)$.

Minimum delay and minimum number of hops: Sometimes, in order to reduce the number of transmissions, it is desirable to choose, among all minimum delay paths, the one with the minimum number of hops. This can easily achieved by first

computing the minimum delay t , then pruning $G(D)$ so that all edges and nodes with time more than t are deleted, and finally running Dijkstra's shortest path algorithm on the pruned $G(D)$ giving each inter-edge a length of 1 and each intra-edge a length of 0.

B. Maximum Bandwidth

When the goal is to find a feasible routing scheme that maximizes the total amount of data that can be sent from s to d , one can use a *maximum flow algorithm* on the event-driven graph. This stems from the following theorem, showing the correlation between a feasible routing scheme on D and a flow on $G(D)$ (proof omitted); in the sequel we will assume that there is a global source s' in $G(D)$ which is connected with infinite bandwidth edges to all the nodes that belongs to super-node s and in addition there is a global sink d' connected with infinite bandwidth edges from all the nodes that belong to super-node d .

Theorem 4: Assume that a DTN graph $D = \langle V, E \rangle$ has no energy constraints (that is, for every $v \in V$, $en(v) = +\infty$). Then, there is an $s'-d'$ flow on $G(D) = \langle V', E' \rangle$ with value x if and only if there is a feasible routing scheme on D with the same value x . Moreover, the routing scheme on G can be efficiently constructed from the flow on $G(D)$.

The gist of the proof is an (efficient) construction of a legal flow on $G(D)$ from each routing scheme on D , and vice versa. Basically, the flow on the inter-edges of $G(D)$ is equal to the amount of data sent on the contacts of G , whereas the flow on the intra-edges of $G(D)$ corresponds to the occupancy of D 's buffer in a specific time interval. Moreover, it is important to notice that since Theorem 4 provides a method to translate a given flow on $G(D)$ to a routing scheme in D , the following result immediately follows:

Corollary 5: By applying a maximum-flow algorithm on $G(D)$, one can efficiently find a routing scheme on D with the maximum value.

Maximum bandwidth with minimum delay: In many cases, it is desirable to choose, among all routing scheme with maximum bandwidth, the one that occurs the minimal end-to-end delay and/or the one that requires the the least possible transmissions per byte of data. Computing maximum flows on the event-driven graph, or subgraphs of it, is very useful for solving these problems as well.

For example, Algorithm 1 shows how, by computing several maximum flows, one can find a routing scheme with maximum bandwidth and minimum delay. The algorithm performs a binary search on the maximum flow values of subgraphs of $G(D)$, pruned according to the possible end times, until the maximum flow with minimum delay is found; the number of iterations is bounded by $O(\log |E_d^{in}|)$ where E_d^{in} is the set of incoming contacts (in the DTN graph) to the target node d .

Other extensions: In order to find the routing scheme with maximum bandwidth and minimum number of transmission per byte, one can solve the *maximum flow minimum cost* problem, assigning a cost of 0 to each intra-edge and a cost of 1 to each inter-edge. These two extensions can be combined

noend 1 Finding the maximum bandwidth routing scheme with minimum delay

```

1:  $G(D) \leftarrow \text{CONSTRUCT-EVENT-DRIVEN-GRAPH}(D)$ 
2:  $f \leftarrow \text{MAX-FLOW}(G(D))$ 
    $\triangleright f$  is a function from the edges of  $G(D)$  to their assigned
   flow
3:  $x \leftarrow \sum_{e \in E_d^{in}} f(e)$ 
    $\triangleright E_d^{in} \subseteq E'$  is the set of incoming edges in  $G(D)$  to the
   global sink  $d'$ ; hence,  $x$  is the value of (max) flow  $f$ 
4:  $T \leftarrow E_d^{in}$ 
5:  $\text{SORT}(T)$  by ascending order of edges times
6:  $l \leftarrow 0, h \leftarrow |T| - 1$ 
7: while  $l \leq h$  do
8:    $m \leftarrow \lfloor (l + h)/2 \rfloor$ 
9:    $G'(D) \leftarrow \text{PRUNE}(G(D))$  according to time  $T[m]$ 
10:   $f' \leftarrow \text{MAX-FLOW}(G(D'))$ ,  $x' \leftarrow \sum_{e \in E_d^{in}} f'(e)$ 
11:  if  $x' < x$  then
12:     $l \leftarrow m + 1$ 
13:  else
14:     $\text{min\_delay} \leftarrow T[m], h \leftarrow m - 1$ 
15:   $G'(D) \leftarrow \text{PRUNE}(G(D))$  according to time  $\text{min\_delay}$ 
16:   $f' \leftarrow \text{MAX-FLOW}(G(D'))$ 
17:   $P \leftarrow \text{CONSTRUCT-ROUTING-SCHEME-FROM-FLOW}(f')$ 
18: return  $P$ 

```

in order to choose among the routing scheme with maximum bandwidth and minimum delay the one with minimum number of transmission per byte: we first apply Algorithm 1 to find the minimum delay, then prune the graph accordingly and apply the maximum flow minimum cost algorithm. Alternatively, if the contact bandwidths are integral (and therefore the flow is integral as well) one can scale the cost of the edges of E_d^{in} according to their times, so that each minimum cost solution will immediately yield minimum delay; in this case the cost that should be assigned to inter-edge corresponds to $e \in E_d^{in}$ is $t(e) \cdot n \cdot x$ where n is the number of inter-edges in $G(D)$ and x is the maximum flow on $G(D)$.

Another simple extension to the following algorithms is to find routing schemes that achieve a certain bandwidth demand y , with minimum delay, minimum number of transmissions (hops), or both. This can be done, for example, by replacing the comparison in Line 11 of Algorithm 1 to be with y instead of x .

Finally, when there are different sources that should send data to different destinations along the DTN, the problem can be modeled as a *multi-commodity flow problem* on the corresponding event-driven graph. It is well-known that this problem can be solved using linear programming (in case fractional flows are allowed); many other results on this problem and its many variants exist in literature and can be directly applied in this case as well (c.f. [16], [17] for more comprehensive surveys). In addition, one may use our construction to model *multicast* scenarios in DTN (that is, when a single source should send data to many destinations) and tackle them using off-the-shelf algorithms available on time-independent networks (e.g. approximation algorithms for the minimum Steiner tree problem).

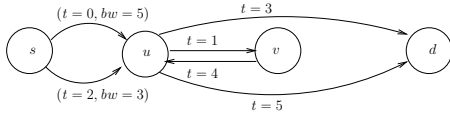


Fig. 3. Example of using a node u as a transient buffer for another node v . The bandwidth bw of all contact is ∞ unless otherwise specified.

C. Dealing with Energy Constraints

In this section we introduce the energy constraints on nodes of the DTN graph D and discuss how to devise algorithms taking into account these constraints as well.

Recall that since each intermediate node (that is, a node other than s and d) forwards all the data it gets, the energy constraint translates to the amount of data a node receives over the entire time span.

A key phenomenon, which makes handling energy constraints more difficult, is the usage of nodes as *transient buffers*, such that the flow of data may form cycles on the DTN graph (but never on the event driven graph). These cycles, in turn, yield that, in order to transmit x bytes from s and d that go y times through super-node v (i.e. in y cycles), the amount of energy needed at super node v is $x \cdot y$. For comparison, in a (traditional) max-flow problem there is no incentive to employ cycles in the graph, and therefore such a problem does not exist. Fig. 3 depicts a scenario in which such a phenomenon happens: suppose that all nodes but u do not have any buffer or energy constraints, and, on the other hand, $buf(u) = 3$ and $en(u) = 7$; furthermore denote by e_i the contact for which $t(e_i) = i$. We first compute the maximum bandwidth routing scheme without taking into account the energy constraints: the resulting scheme is $P = \{p_1, p_2\}$ of total bandwidth 6, where $p_1 = (s, e_0, u, e_1, v, e_4, u, e_5, d)$ of bandwidth 3 and $p_2 = (s, e_2, u, e_3, d)$ of bandwidth 3. Note that p_1 contains a cycle (u is visited twice), and in fact node v was used as a transient buffer for the data of p_1 between time 1 and 3. The total amount of energy spent at node u is 9, thus exceeding $en(u)$. To correct this one should decrease the bandwidth that goes through u ; since each byte along p_1 is “counted” twice, it is sufficient to reduce the bandwidth of p_1 to 2, resulting in total bandwidth of 5.

Going back to the *event-driven graph* $G(D)$, recall that the energy constraints were not realized in its construction. The reason behind this is that *energy* is counted over the entire time span and not only between events; thus, it is common to all nodes that belongs the same super-node, as illustrated in Fig. 4.

Unlike Section IV-B, where we apply combinatorial algorithms to solve maximum-flow related problems, here we formalize the problem on the event-driven graph as a *linear program*. It is important to notice that since the resulting flow need not be integral, solving the linear program yields an efficient algorithm to find the maximum flow. For example, the following linear program describes the problem of finding a maximum flow on the graph, respecting both the buffer and energy constraints:

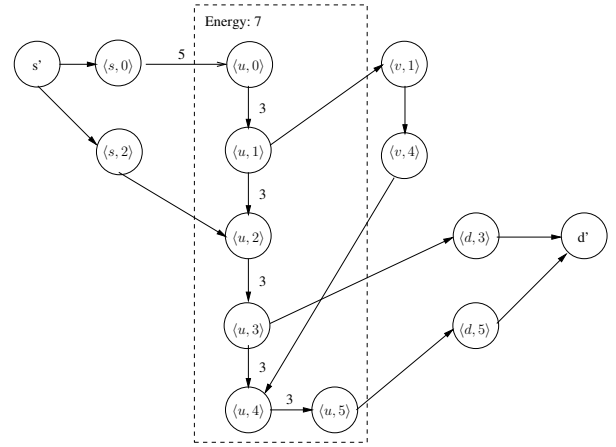


Fig. 4. The event-driven graph of the DTN depicted in Fig. 3. Non-labeled edges have infinite bandwidth.

$$\begin{array}{ll}
 \text{maximize} & f_{d's'} \\
 \text{such that} & f_e \leq bw(e) \quad \forall e \in E' \\
 & \sum_{j:(j,i) \in E'} f_{ji} - \sum_{j:(i,j) \in E'} f_{ij} \leq 0 \quad \forall i \in V' \\
 & \sum_{k \in V'_i} \sum_{j:(j,k) \in E'} f_{jk} I_{jk} \leq en(v) \quad \forall v \in V
 \end{array}$$

where I_{jk} is an 0–1 indicator variable that equals 1 if and only if (j, k) is an inter-edge.

Note that the dual of this program is

$$\begin{array}{ll}
 \text{minimize} & \sum_{e \in E'} bw(e) d_e + \sum_{v \in V} en(v) r_v \\
 \text{such that} & d_{ij} - p_i + p_j + r_{v(j)} I_{ij} \geq 0 \quad \forall (i, j) \in E' \\
 & p_{d'} - p_{s'} \geq 1 \\
 & d_e, r_v \geq 0
 \end{array}$$

where $v(j)$ is the super-node $v \in V$ that the node $j \in V'$ belongs to.

It is important to notice that unlike a traditional MAXIMUM FLOW problem, the solution of the dual problem is not necessarily integral, implying that a *max-flow min-cut* theorem is unlikely to hold. In fact, the example depicted in Fig. 3 shows also that a simple augmentation-path based algorithm cannot solve this problem, since one can first choose p_1 with bandwidth 3 without violating the constraints.

V. PERIODIC DTN

In some cases the DTN graph describes a periodic pattern of contact between the nodes. Specifically, suppose the DTN period length is given by $\tau > \max_{e \in E} t(e)$, then if a contact between some node u and another node v has time $t(e)$, it implies that u can send data to v at times $t(e) + k\tau$ for any $k \in \mathbb{N}$. Moreover, since the system is now “*long-lived*”, we treat the energy constraints as the energy that can be spent by a node over one period.

We focus on a maximum bandwidth problems where the goal is to deliver as much data as possible *per period*. Since we measure the bandwidth per period, one can mistakenly assume that the problem can be solved by looking only on a single period; however, as the example depicted in Fig. 5 shows, this is not the case: when looking on a single-period

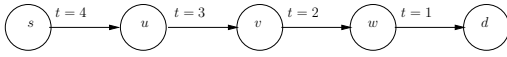


Fig. 5. Example of a periodic DTN which can deliver bandwidth of 1, while during a single period the DTN cannot deliver any data. The bandwidth of all contact $bw = 1$.

DTN, it is clear that no data can be transferred from the source to the destination, however for a periodic DTN the amount of bandwidth per period goes to 1 as data is sent over the links at times $k\tau + 4$, $(k+1)\tau + 3$, $(k+2)\tau + 2$, $(k+3)\tau + 1$ for any period $k \in \mathbb{N}$. This scenario somehow resembles the *pipeline* paradigm.

Therefore, we model the problem by assuming a periodic routing scheme, which treats traffic transmitted from source s at time $k\tau + t$ as the traffic transmitted at time t . Then, we can solve the maximum bandwidth per period problem by looking only on the traffic transmitted from the source at the first period, while respecting the following additional constraints:

- let e be a contact, and let $f(e, t, P)$ be the flow assigned to contact e at time t under routing scheme P . Thus, $\sum_{k=0}^{\infty} f(e, t(e) + k\tau, P) \leq bw(e)$; namely, the total amount of flow on a contact over all periods does not exceed its bandwidth.
- recall that $occ(v, t, P)$ is the number of bytes stored at node v at time t under routing scheme P . Thus, we require for all $v \in V$ and $t < \tau$, $\sum_{k=0}^{\infty} occ(v, t + k\tau, P) \leq buf(v)$.
- let $recv(v, k, P)$ to be the number of bytes received by node v during period k under routing scheme P . Thus, we require for all $v \in V$ that, $\sum_{k=0}^{\infty} recv(v, k, P) \leq en(v)$.

Hence, periodic DTN graph can be conceptually reduced to a cyclic event-driven graph $\tilde{G}(D) = \langle V', E' \rangle$ in which for each super-node $v \in V$ there is an *extra edge* of bandwidth $buf(v)$ connecting the last node (ordered by time) of V'_v with the first node of V'_v . Running a maximum-flow algorithm on this event-driven graph will give the maximum bandwidth per period that can be achieved. Moreover, given the maximum flow f , one can construct a routing scheme P (using some *path splitting* algorithm) and compute the corresponding delay of P by counting the number of extra edges in each path. Let k_{\max} be the maximum number of periods needed to complete any path in P (that is, the number of extra edges in the path plus one).

It is important to notice that $k_{\max}\tau$ is only an upper bound on the minimum delay needed to achieve the maximum bandwidth. In fact, computing the routing scheme with minimum bandwidth and maximum delay is more complicated in case the DTN is periodic. This is done by considering an event-driven graph $\tilde{G}(D) = \langle \tilde{V}, \tilde{E} \rangle$ which holds k_{\max} copies of the original event-driven graph $G(D) = \langle V', E' \rangle$. Nodes corresponding to super-node s appears in $\tilde{G}(D)$ only in the first copy (implying that we consider only traffic transmitted from the source at the first period) and the last node (ordered by time) of some super-node v at k -th copy is connected by

an intra-edge of bandwidth $buf(v)$ to the first node of v at $k+1$ -th copy; the set of all newly-added edges between copies k and $k+1$ is denoted by $E''_{k,k+1}$. Furthermore, let the set of all nodes that belongs to super node $v \in V$ in the k -th copy be $\tilde{V}_{v,k}$, while the set of all k_{\max} copies of an edge $e \in E' \cup E''_{0,1}$ is denoted by \tilde{E}_e . Finally, we assume that the global source node s' is connected to all nodes that belongs to super node s while the global sink node d' is connected to all nodes that belongs to super node d in all k_{\max} copies.

The additional constraints of the periodic DTN can then be realized by adding the corresponding linear inequality constraints, resulting in the following linear program that computed the maximum-flow on $\tilde{G}(D)$:

$$\begin{array}{ll}
 \text{maximize} & f_{d's'} \\
 \text{such that} & \sum_{\tilde{e} \in \tilde{E}_e} f_{\tilde{e}} \leq bw(e) \quad \forall e \in E' \cup E''_{0,1} \\
 & \sum_{j:(j,i) \in \tilde{E}} f_{ji} - \sum_{j:(i,j) \in \tilde{E}} f_{ij} \leq 0 \quad \forall i \in \tilde{V} \\
 & \sum_{k=0}^{k_{\max}} \sum_{i \in \tilde{V}_{v,k}} \sum_{j:(j,i) \in \tilde{E}} f_{ji} I_{ji} \leq en(v) \quad \forall v \in V
 \end{array}$$

where I_{ji} is an 0–1 indicator variable that equals 1 if and only if (j, i) is an inter-edge.

Finally, this procedure can be plugged into Algorithm 1, thus computing the maximum flow with minimum delay.

VI. EXPERIMENTAL RESULTS

Many experimental traces, publicly available in [18], store the contacts observed in different mobility scenarios (like, public transportation systems, humans); they allow to build directly the DTN graph, which is the input of our optimization framework.

For lack of space, we have chosen to show just some results referred to the mobility of a set of 37 buses running routes for 16 weeks in the UMass campus [3]. These buses operated daily as dictated by their time schedules but also by their failures and maintenance stops. The trace provides basically the sequence of all the contacts in the following format: time, source bus, destination bus, capacity of the contact (measured in bytes). We have considered just the contacts referred to the third week of the experiment.

We have run the optimal algorithm to maximize the bandwidth in a data flow between two given buses. Fig. 6 shows the curves of the minimum delay necessary to transfer a given amount of data for two cases, corresponding to two specific pairs of source and destination buses (similar results were observed for all the other possible pairs). Both curves are step functions. Indeed, assume that, for a specific value of bandwidth x and delay y , the last contact in the routing (with contact time y) has not been yet completely saturated. When x increases, then the delay remains the same until the bandwidth of the last contact is fully exploited. At that point, the delay jumps to the time of the next contact exploited.

The graph can be seen also as the representation of the *capacity-delay region* achievable in the two cases. Indeed, there is no algorithm that is able to achieve any point below and to the right of each curve, since the routing and scheduling scheme are optimal from both the capacity and the delay

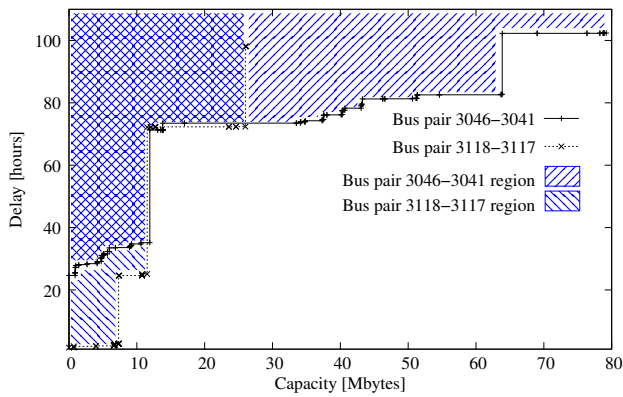


Fig. 6. Minimum delay for two specific pairs of buses. Curves define also the optimal delay-capacity regions for the two cases.

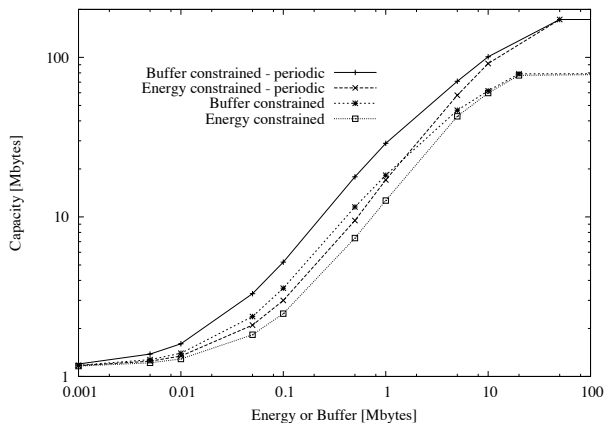


Fig. 7. Maximum bandwidth achievable for the bus pair 3046-3041 as function of the buffer and energy available at each bus

point of view. Note that this region shows some performance limitations of the DTN considered in the experiment; this is coherent with previous work [15] and due to the fact that time schedule for public transportation is inherently designed to reduce contacts among the buses.

Fig. 7 considers only the data flowing from bus 3046 to 3041, and shows the effect of the constraints on the resources, i.e. the energy and the buffer available at each node; for simplicity, we assume an homogeneous network in which all the nodes share the same amount of buffer and have the same amount of energy available (measured as the maximum number of bytes that can be transmitted by a node other than the source). As expected, capacity is a monotonic function of the energy and the buffer, and its maximum value (80 Mbytes, coherently with Fig. 6) is obtained for sufficiently large energy and buffer. In addition, the constraint on the energy tends to affect more severely the performance, since the buffer, as a resource, can be re-utilized in different times, whereas the energy can be exploited only once. When the buffer is very small, the routing is so constrained that the buffer tends to be exploited just once and the effects of the energy and the buffer

are equivalent.

Furthermore, Fig. 7 shows the effect of considering periodic DTNs. The qualitatively behavior is similar as before. Periodic DTNs cannot benefit from small value of resources, especially from small values of energy, which can be exploited only once. Larger values of resources allow to achieve larger capacity, and the maximum possible capacity corresponds, in the original DTN graph, to the minimum between the cut around the source node and the cut around the destination node. The capacity gain by considering periodic DTNs is more than 50% for buffer/energy larger than 1 Mbytes, and reaches 122% when resources are unbounded.

VII. CONCLUSIONS

We have proposed an optimization framework to find the optimal routing for deterministic DTNs. Our novel approach is very general and based on a construction of an event-driven graph from the classical DTN graph; this allows us to exploit standard algorithms from Graph Theory. We have demonstrated our approach by showing solutions to different cost functions (maximum bandwidth, minimum delay) and constraints (maximum buffer and energy per node). We have also discussed some numerical results referring to a real mobile network, whose DTN graph is known through traces.

REFERENCES

- [1] M. Grossglauser and D. Tse, "Mobility increases the capacity of ad hoc wireless networks," *IEEE/ACM Trans. Netw.*, vol. 10, no. 4, pp. 477–486, 2002.
- [2] S. Jain, K. R. Fall, and R. K. Patra, "Routing in a delay tolerant network," in *ACM SIGCOMM*, 2004, pp. 145–158.
- [3] C. Liu and J. Wu, "Scalable routing in delay tolerant networks," in *ACM MobiHoc*, 2007, pp. 51–60.
- [4] —, "Routing in a cyclic mobispace," in *ACM MobiHoc*, 2008, pp. 351–360.
- [5] "Delay tolerant networking research group." [Online]. Available: <http://www.dtnrg.org>
- [6] J. Burgess, B. Gallagher, D. Jensen, and B. N. Levine, "Maxprop: Routing for vehicle-based disruption-tolerant networks," in *INFOCOM*, 2006.
- [7] P. Hui, A. Chaintreau, J. Scott, R. Gass, J. Crowcroft, and C. Diot, "Pocket switched networks and human mobility in conference environments," in *ACM WDTN*, 2005, pp. 244–251.
- [8] M. C. Gonzalez, C. A. Hidalgo, and A.-L. Barabasi, "Understanding individual human mobility patterns," *Nature*, vol. 453, no. 7196, pp. 779–782, 2008.
- [9] "Crossbow technology, inc." [Online]. Available: <http://www.xbow.com>
- [10] A. Balasubramani, B. N. Levine, and A. Venkataramani, "DTN Routing as a Resource Allocation Problem," in *ACM SIGCOMM*, 2007.
- [11] S. Merugu, M. Ammar, and E. Zegura, "Routing in space and time in networks with predictable mobility," Georgia Institute of Technology, Tech. Rep., 2004.
- [12] L. R. Ford and D. R. Fulkerson, "Constructing maximal dynamic flow from static flows," *Operation Research*, vol. 6, pp. 419–433, 1958.
- [13] B. Hoppe, "Efficient dynamic network flow algorithms," Ph.D. dissertation, Cornell University, 1995.
- [14] M. Skutella, "An introduction to network flows over time," in *Research Trends in Combinatorial Optimization*. Springer, 2008, pp. 451–482.
- [15] A. Di Nicolò and P. Giaccone, "Performance limits of real delay tolerant networks," in *IEEE WONS*, 2008, pp. 149–155.
- [16] B. Awerbuch and F. T. Leighton, "Multicommodity flows: A survey of recent research," in *ISAAC*, 1993, pp. 297–302.
- [17] G. Karakostas, "Faster approximation schemes for fractional multicommodity flow problems," in *ACM-SIAM SODA*, 2002, pp. 166–173.
- [18] "Crawdad repository." [Online]. Available: <http://crawdad.cs.dartmouth.edu>