2014 IEEE 11th International Conference on Mobile Ad Hoc and Sensor Systems

Low Expected Latency Routing in Dynamic Networks

Prithwish Basu
Raytheon BBN Technologies

Feng Yu Matthew P. Johnson Amotz Bar-Noy
City University of New York (CUNY)

***Abstract*—Timely and efficient message transmission through intermittently and sparsely connected networks is a problem of significant interest to the mobile networking community. Although the long-term statistics describing the time-varying connectivity in such networks can be characterized systematically and can be used for selecting good routes, it may be possible to achieve better performance by intelligently using the actual link states at the time of routing, in conjunction with these statistical dynamics models. In this paper, we investigate a family of minimum expected latency routing methods for such dynamic networks, spanning purely model-based and state-oblivious source routing; state-based source routing; and various flavors of dynamic (or hop-by-hop) routing, with increasing amounts of current link state knowledge around the source.**

First, we give a heuristic and an approximation scheme for the model-assisted source routing problem, as well as heuristics for the dynamic routing problem. Then we show using extensive simulations on both synthetic and real time-varying connectivity traces that although dynamically sampling link states helps to improve expected routing latency compared to source routing, the marginal improvements decline rapidly for knowledge of current link states beyond 2 hops. To the best of our knowledge, this is the first thorough characterization of the performance of the entire spectrum of model-assisted routing algorithms ranging from little knowledge to complete knowledge of link dynamics.

I. Introduction

Routing in dynamic wireless networks such as mobile ad hoc networks (MANETs), mesh / sensor networks (WSNs) and disruption tolerant networks (DTNs) has been a popular topic of research for decades in the research community. In MANETs and WSNs, network topology could vary over time due to link outages caused by node mobility, jamming, or interference, or due to node outages caused by duty cycling or, in the extreme case, battery exhaustion. In MANETs, shortest paths can be computed through networks after collecting topology information periodically or in an event-triggered manner. This can be done *on demand* by flooding followed by source routing [9] or reverse path forwarding [16], or *proactively* by flooding topology updates followed by each source computing a shortest path to destinations using standard shortest path algorithms [5]. A chosen path can then be used until a link failure prompts its revision. Alternatively, routing can be done dynamically, through local broadcast [4] or repetition [17]. In this paper, we focus on "single-copy" store-and-advance routing under the latency cost metric, which often applies to lightly loaded DTNs and transportation networks, and study a number of problem variants of this form.

a) Source vs. dynamic routing: Routing in dynamic networks is typically done hop-by-hop, the key benefit of which is responsiveness to topology changes. In our setting, this means either choosing a next-hop link or waiting for a better link to become available. An alternative approach is (store-and-advance) *source routing*, in which a complete, immutable path is chosen for a packet to travel, from source to destination, within a network. Thus, once the path is chosen, no routing decisions need be made online. This is especially advantageous for networks with resource-rich end-hosts but resource-poor nodes in the intermediate core.

b) Different levels of available knowledge of link state information: At one extreme, the current states of all links may be known at all times. At the other extreme, only *local* link state information is available at the *source* (in source routing) or the *current* node (in dynamic routing). (All other links are assumed to be in steady-state.) There are two sorts of intermediate settings: a) global knowledge of all link states but only in the first time step, and b) local knowledge, after each hop, of the links within some k-hop neighborhood of the current node. It may be feasible to inform the source (and intermediate nodes) about changes in network topology over alternate communication channels. An example of this is a "hybrid" network where each node is equipped with two transceivers: a high data rate / low transmission range, intermittently connected transceiver to carry "data" (e.g., WiFi) and a lower data rate / high transmission range, always connected transceiver for "control" (e.g., cellular data).

When the future topology changes known, latency can be minimized by running a modified Dijkstra algorithm [8]. Alternately, one can minimize *estimated expected delay* based on observed contact schedules [10].

c) Link dynamics models: In addition to current or historical link state information, routing decisions can be based on a suitable *model* of the underlying link dynamics. This allows probabilistically projecting the state of a link forward in time, thus alleviating the need to flood current topology state information every time a link's state changes. The dynamic Erdős-Rényi (ER) random graph model[1] is a basic model for studying dynamic graphs, but it does not capture temporal correlation in link dynamics. A more powerful and accurate

Corresponding author email: pbasu@bbn.com. Research was sponsored by the Army Research Laboratory and was accomplished under Cooperative Agreement Number W911NF-09-2-0053 (ARL Network Science CTA). The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

[1] At any given time, a link is up independently with a certain probability p, which is assumed to be known in advance or learnt; and the current state of a link is observable by sampling or exchanging messages between endpoints of the said link.

978-1-4799-6036-1/14 $31.00 © 2014 IEEE
DOI 10.1109/MASS.2014.107

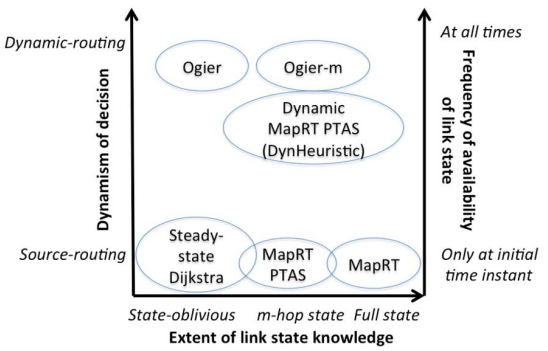


Fig. 1. A collection of model-assisted link state routing schemes.

model, and the focus in this paper, is the (p, q) dynamic discrete-time Markovian model. At each discrete time step in this model, a link is either *on* or *off*; between one time step and the next, the link may transition between these states, with known probabilities of transitioning from off to on (p) and from on to off (q).

In this paper, we consider the problem of *model-assisted* routing through dynamic graphs so as to minimize *expected routing time* (ERT) between a given source and a destination.

Related Work. Stochastic, time-dependent networks (STDNs) have been studied in the transportation literature [7], where edge delays are time-dependent random variables (r.v.), and a standard routing goal is to minimize expected travel time. Because of the dynamic nature of the network, a dynamic routing *policy* has been considered instead of a fixed path.

Stochastic routing in MANETs has been studied in [12]. Though somewhat similar in spirit to our work, the models are quite different – instead of transitions on link states, they consider Markovian transitions on the network state specified by which nodes have received the message so far. Additionally, there has been significant work on DTN routing (see [1], [2] and the references therein). The work closest to ours is by Ogier and Rutenburg [15]. They study the scenario where each dynamic link l is assumed to obey a two-state Markov chain with known parameters (p_l, q_l), and a node can observe the on/off *state* of its neighboring links; links beyond that are assumed to be in steady-state. This problem is #P-complete, even in the special case of $p_l = 1-q_l$ with zero-latency edges. However, for nonzero link latencies or if the graph is directed and acyclic (DAG), then Dijkstra-style dynamic programming yields an optimal solution. Nikolova and Karger [14] gave similar results for the "stochastic edge-latency" version of the Canadian Traveler Problem (CTP) [3]. The "stochastic edge-state" version of CTP is PSPACE-hard [6]. We do not restrict ourselves to having only the current state of neighboring links to be available; e.g., in multi-radio hybrid networks, it is possible to acquire link states beyond the one-hop neighborhood.

Contributions. To the best of our knowledge, this sort of systematic exploration of minimum latency link state routing schemes (see Fig. 1) has not previously appeared in the literature. Our main contributions are as follows: For Markovian networks with global or m-hop state info, we give an efficient heuristic and a polynomial time approximation scheme (PTAS) for source routing, and two heuristics for dynamic routing. We also show that for a number of interesting special cases of (p, q), the routing problems can be solved optimally. We perform an extensive evaluation of these algorithms on both synthetically generated time-varying networks and real networks, investigating questions such as: How beneficial is having link state info beyond the current/source neighborhood, and how far? How much advantage does the flexibility of dynamic routing offer over source routing? We find that both provide benefits, but the state info benefits rapidly diminish beyond 2 hops. The results of this paper are applicable to lightly loaded (unsaturated) multi-radio and WiFi/Cellular hybrid networks mentioned earlier. Our goal is not to provide a complete routing protocol but to investigate the aforementioned fundamental tradeoffs, hence we do not study actual link state dissemination issues in this work.

II. Markovian Networks

We begin with basic assumptions and concepts. Time is measured in discrete time steps. *Time* τ refers to the beginning of the time step τ, for $\tau = 0, ..., T$.

Definition 1 (Dynamic graph): $G[0,T] \equiv \{G_\tau = (V_\tau, E_\tau)\}_{\tau=0}^{T}$ is a dynamic (or time-varying) graph indexed by discrete time step τ if each "graphlet" G_τ is a valid graph. V_τ and E_τ denote the vertex and link sets, respectively, at time τ. A dynamic graph may have a finite or an infinite number of graphlets.

Definition 2 (Underlying graph): $G_\cup = (V_\cup, E_\cup)$ is the underlying graph of a dynamic graph $G[0,T]$ if $V_\cup = \bigcup_{\tau=0}^{T} V_\tau$ and $E_\cup = \bigcup_{\tau=0}^{T} E_\tau$.

Definition 3 (Markovian (p, q) graphs): At time 0, each link is in some state (which we may or may not assumed to be known). The state of a given link i in subsequent time steps is governed by a known two-state Markov chain specified by the probability transition matrix P_i: $P_i = \begin{pmatrix} 1-p_i & p_i \\ q_i & 1-q_i \end{pmatrix}$, with p_i (resp. q_i) the probability that link i transitions from state 0 (resp. state 1) into state 1 (resp. state 0) in one time step. Note that the nontrivial eigenvalue of P_i is $\beta_i = 1 - p_i - q_i \in [-1, 1]$, a quantity which will be very useful later.

Definition 4 (Steady-State probabilities): $\pi_0 = q_i/(p_i + q_i)$ and $\pi_1 = p_i/(p_i + q_i)$ are the stationary probabilities that link i is in states *off* or *on*, respectively.

We note a few important special cases of (p, q) and their resulting steady state probabilities: (i) $(1, 1)$**:** Link states alternate deterministically in every time step, with $\pi_0 = .5$; (ii) $(p, 1-p)$**:** The Erdös-Renyi setting, where a link's state is independent of its previous state, with $\pi_0 = p$; (iii) $(p, 0)$**:** A "densifying network" where edges appear but never disappear; (iv) $(0, 0)$**:** Link states never change, which is consistent (in the unknown initial state setting) with *any* specified steady-state (i.e., initial) probability π_0.

Definition 5 (Transient probabilities): Let $P_{a,b}^{\ell}(\tau) = \Pr(X_\ell(\tau) = b \,|\, X_\ell(0) = a)$ be the probability that link ℓ is in state $b \in \{0, 1\}$ at time τ given that it was in state $a \in \{0, 1\}$ at time $\tau = 0$ for initial state X. In cases where all links have the same parameters p, q, we omit the superscript ℓ.

For any $t \geq 0$, we have [11]: $P_{1,0}(t) = \pi_0(1-\beta^t); P_{1,1}(t) = \pi_1 + \pi_0\beta^t; P_{0,1}(t) = \pi_1(1-\beta^t); P_{0,0}(t) = \pi_0 + \pi_1\beta^t$.

Transmission assumptions. If the message reaches a link's entry node *when the link is on*, then it immediately starts traversing the link; if the link is off, the message waits there until the link appears. For each link ℓ its delay $d(\ell)$ is the number of time steps it takes to traverse ℓ (when on). We also assume that transmission finishes within a single time step and is not interrupted by a link-down event.

Notation. The *global state* of all links at time t is indicated by $X(t)$, which represents an assignment of each edge of the underlying graph to a bit. The state (0 or 1) of a particular link i at time t is indicated by $X_i(t)$, sometimes written X_i^t when notationally convenient.

III. Source Routing in Markovian Networks

We are given an underlying graph $G_\cup$, initial link state configuration G_0, source node s, target node t, and dynamics model parameters (p, q). Let $\mathcal{P}_{s,t}$ be the set of all simple paths from s to t in $G_\cup$. Then the objective of source routing is: $\arg\min_{P_{s,t}\in\mathcal{P}_{s,t}} ETT(P_{s,t}|G_0)$, where $ETT(P_{s,t}|G_0)$ is the expected traversal time in the (p, q) model of a path $P_{s,t}$ with the initial link states given by G_0. This value can be computed for an n-hop path in $O(n^2)$ time[2] [13]. From $G_\cup$, we construct a weighted graph G_w: $\forall \ell = (u, v) \in G_\cup, d(u, v, G_w) = d(u, v, G_u) + \frac{q_\ell}{p_\ell(p_\ell+q_\ell)}$ (Expected latency on ℓ is $\frac{q_\ell}{p_\ell(p_\ell+q_\ell)}$).

A modified-Dijkstra heuristic: We first show that the "subpath optimality property" does not hold in this setting, which precludes using Dijkstra-style dynamic programming techniques to optimally solve the problem. Nonetheless, we give a Dijkstra-based heuristic and analyze its approximation factor.

Definition 6: A Markovian dynamic graph G with its underlying graph $G_\cup$ satisfies the subpath property if for any three different nodes $u, v, w \in G$, assuming there are two distinct paths connecting u, v, namely $X(u, v)$ and $Y(u, v)$, and one path connecting v, w, namely $Z(v, w)$, if $ETT(X) \geq ETT(Y)$, then $ETT(XZ) \geq ETT(YZ)$, or vice versa.

As illustrated in Figure 2, consider two paths in $G_\cup$ with initial states denoted by bit strings X, Y. Let $\Delta = ETT(X) - ETT(Y) = \mathbb{E}_X[T] - \mathbb{E}_Y[T]$ and $\Delta_1 = ETT(X1) - ETT(Y1) = \mathbb{E}_{X1}[T] - \mathbb{E}_{Y1}[T]$. Examples can be found (see table in Figure 2) of strings X, Y violating the subpath property, i.e., $\Delta\Delta_1 < 0$. In fact, the difference Δ_1 can grow arbitrarily large. Nonetheless, we obtain the following lemma[3] and a subsequent observation.

Lemma 1: If $|X| = n$ and $\Delta \leq 0$, then $\frac{\Delta_1}{\mathbb{E}_{X1}[T]} \leq \alpha$, where $\alpha = \min\{\frac{q}{p}, \frac{q}{np(p+q)}\}$.

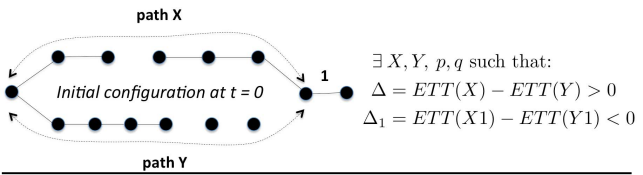
Observation 1: The ratio $\frac{\Delta_1}{ETT(X1)}$ remains small and is upper bounded by α as seen in Figure 2.

Using this insight, we propose Algorithm 1.

Claim 2: The approximation factor of any k-hop solution found by Algorithm 1 is at most $(1 + \alpha)^k$.

[2]The complexity remains $O(n^2)$ when (p_ℓ, q_ℓ) vary by link ℓ, as long as there is only a constant number of distinct (p_ℓ, q_ℓ) pairs. Our algorithms also generalize in this way, but we assume uniform (p, q) values for simplicity.

[3]Proofs of lemmas and theorems have been omitted for brevity.



p	q	X	Y	$\Delta_1/ETT(X1)$	α
.00065	.00098	110111	1111000	$\frac{87.61}{4532} = .019$	1.507
.9964	.9713	100100	1010101	$\frac{0.58}{9.27} = .063$	.071

Fig. 2. Counterexample to the subpath optimality property

Algorithm 1 MapRTAppx$(G_\cup, G_0, s, t)$

1: Run Dijkstra's algorithm on $G_\cup$, except with the following modification: each time a neighbor v of a current node u is being considered, evaluate the tentative distance to v not as $ETT(G_0, path(u)) + d(u, v)$ but instead as $ETT(G_0, path(u) + v)$.

Algorithm 2 SourceRoutingPTAS$(G_\cup, G_w, G_0, s, t, \epsilon)$

1: choose ϵ' satisfying $\frac{1+\epsilon'}{1-\epsilon'} \leq 1 + \epsilon$
2: $D \leftarrow \log_{|1-p-q|}(\frac{p(p+q)(1-|1-p-q|)}{q}\epsilon')$
3: Path set $\mathcal{P} \leftarrow BFS(G_\cup, s, d < D)$
4: **for** each $P_i \in \mathcal{P}$ **do**
5: let $P_i = (s \rightsquigarrow v_i)$
6: $delay(P_i) \leftarrow ETT(G_0, P_i) + Dijkstra(G_w, t)[v_i]$
7: let $path$ be the one with minimum $delay(P_i)$
8: let $path'$ be the path w.r.t $Dijkstra(G_w, t)[v]$
9: **return** $path + path'$

Approximation Algorithms: Now we give a Polynomial-time Approximation Scheme (PTAS) for the source routing problem, based on some bounds on the ETT of a path. For any $\epsilon > 0$, this algorithm finds a path whose cost is within a factor $1 + \epsilon$ of the optimal, and has polynomial time complexity.

First we need some bounds on the expected time to traverse a path of known state.

Theorem 3 (General bounds for ETT): Let x be an an initial state of an n-hop path, and let $B_x(k)$ be the indicator function for the set of bit positions in x equaling 1. Let $SS = n + n\pi_0/p$ be the expected traversal time in steady-state. Then ETT(x) is bound by the following:

$$\Big|SS - \text{ETT}(x)\Big| \leq \frac{1}{p+q}\sum_{k=0}^{n-1}\Big(\frac{q}{p}\Big)^{B_x(k)}|1-p-q|^k \quad (1)$$

Lemma 2: For any $b \in (0, 1)$ and $e \in (0, b]$, we have that $\forall x \geq \log_b e$, $b^x \geq ex$.

Armed with the above bound and lemma, we now derive the PTAS. We assume an error parameter $\epsilon > 0$ that is sufficiently small, specifically $\epsilon \leq |1 - p - q| = |\beta|$.

Theorem 4: Let $X = (x_1, x_2..., x_n)$ be the initial state of a path and SS be the path's steady-state ETT, and let $ETT(X^d)$ and SS^d be the corresponding values restricted to the edges $d+1, ..., n$. Finally, let $ETT_d(X) = ETT(x_1, ..., x_d) + SS^d$. Then for any $\epsilon \in (0, |1 - p - q|]$, there exists a constant $d = d(p, q, n, \epsilon)$ such that $\frac{|ETT(X)-ETT_d(X)|}{ETT(X)} \leq \epsilon$.

Corollary 5: There exists a PTAS (Algorithm 2) for source-routing in the (p, q) model, with running time $O(n^\gamma \cdot f(n))$,

Algorithm 3 Ogier-$m(G_\cup, G_w, G_\tau, m, s, t)$

for each $v \in G_\cup$ **do**
  $e(v) \leftarrow Dijkstra(G_w, t)[v]$
$u \leftarrow s$
while $u \neq t$ **do**
  for each $v \in N(u)$ **do**
    Path set $\mathcal{P}_v \leftarrow BFS(G_\cup, v, d < m)$
      ▷ If $m = 1$, $\mathcal{P}_v = \emptyset$. Set $e(u,v) \leftarrow d(u,v) + e(v)$
    for each $P_i \in \mathcal{P}_v$ **do**
      let $P_i \leftarrow v \rightsquigarrow w$
      $delay(P_i) \leftarrow ETT(G_\tau, P_i) + Dijkstra(G_w, t)[w]$
    $e(u,v) \leftarrow d(u,v) + \min(delay(P_i))$
  $k \leftarrow$Policy$\{e(u,v) | \forall v \in N(u)\}$
  if one of the k best links $\in G_\tau$ **then**
    $e(u,v) \leftarrow \min(e(u,v_i) | (u,v_i) \in G_\tau)$
    $u \leftarrow v$, $\tau \leftarrow \tau + e(u,v)$
  else
    wait for next time slot, $\tau \leftarrow \tau + 1$

where $\gamma = \log_\beta \frac{p(p+q)(1-\beta)}{q}\epsilon'$, $\epsilon' = \epsilon/(2+\epsilon)$, and $f(n)$ is the running time of Dijkstra.

IV. Dynamic routing in Markovian Networks

In dynamic routing, the decision is not made at the source node, but instead at each intermediate node as the message passes through it. Thus, a solution to the dynamic ERT minimization problem takes the form of a *policy* instead of a path. Because the dynamic routing problem subsumes a PSPACE-complete problem [6], we focus on heuristics. We assume local knowledge of the graph at the current time instant τ – the full graph is denoted by G_τ.

Ogier's algorithm: For Markovian networks, Ogier and Rutenburg's algorithm [15] (Algorithm 3 with $m = 1$) assumes that the current node can observe the state of its neighboring links. Links beyond the one-hop neighborhood are assumed to be in steady-state. For each neighbor v of the current node u, it first computes the expected delay if (u, v) is up and is selected; then it finds the maximum k such that waiting at u for one more time slot due to the k lowest expected-delay links being currently all down is preferable to experiencing the expected delay due to the $k+1$-st link. For such k, if at least one of the best k links is up, then take the best among them; otherwise wait a time slot and check again. This policy provides an optimal expected routing time under certain restrictions, such as nonzero-weight edges and a directed acyclic graph (DAG).

Multi-hop extensions to Ogier: Ogier's algorithm could perform poorly in sparsely connected networks where the link states beyond one hop are far from steady-state. If current link state information beyond the one-hop neighborhood can be gathered (e.g., in hybrid networks), we make a natural extension. The most important and difficult part of the multi-hop Ogier (Algorithm 3) is calculating the expected delay through each link with m-hop knowledge of current link states. Here we divide the expected delay *through* a link (u, v) into three parts: 1) the latency of link $d(u, v)$, 2) the minimum ETT of each simple path starting from v and inside the induced m-hop subgraph, and 3) the expected delay from the end of path to the destination. Then the sum of the three will be the expected delay through link (u, v). Note that in a DAG, the minimum ETT of all simple paths could guarantee the minimum expected delay through the link, which yield the optimality of routing.

Algorithm 4 DynHeuristic$(G_\cup, G_\tau, s, t, \epsilon)$

$u \leftarrow s$
while $u \neq t$ **do**
  $path \leftarrow$ SourceRoutingPTAS$(G_\cup, G_\tau, s, t, \epsilon)$
  $h \leftarrow$ next-hop$(path)$
  $\tau \leftarrow \tau + d(u, h)$
  $u \leftarrow h$

A PTAS-based Heuristic: A second algorithm we give using multihop state info is a heuristic (Algorithm 4) that calls the source-routing PTAS as a subroutine. At each current node u, the algorithm runs the PTAS starting from u and then chooses the first link of the resulting path to be its next hop.

We note that unlike Ogier's algorithm which imposes a DAG on the underlying graph to avoid loops, this algorithm works on the undirected underlying graph. There is a danger, therefore, of cycling in loops. Since the graph is dynamic, though, the loops are likely to be transient, not persistent. In fact, we have not encountered any persistent loops during our performance evaluation. Enforcing loop freedom may require maintaining some additional state during route computation, and that is a topic of future research.

V. Performance Evaluation

We use numerical simulations[4] to explore several tradeoffs between the source routing and dynamic routing algorithms proposed in Sections III and IV in representative dynamic Markovian networks.

Random Geometric Graphs: First, we evaluate the different routing schemes in an idealized network with uniform (p, q) parameters for each link. We vary the structure and density of underlying graph, $G_\cup$, which is a random geometric graph (RGG), modeling mesh network topologies where links can be intermittent due to outages; and the dynamics parameters (p, q) between 0 and 1, with special emphasis on the left end of the spectrum. From Fig. 3 (left), we observe that even for networks with low p and q (stable but dynamic networks), source routing performs poorly compared to dynamic routing. Also, we can observe a decisive advantage towards using m hop knowledge as seen by the better performance of Ogier-m algorithms compared to Ogier. However, there is no advantage in using $m > 2$ in routing decisions. The best performing algorithm is the dynamic PTAS DynHeuristic which in this case ends up performing a lookahead of up to 3 hops but uses a different policy than Ogier-3. Its performance is not much worse than (multi-copy) Epidemic Flooding. Fig. 3 (right) plots the performance of the schemes as $p = q$ varies. As p=$q \to 0$, the network becomes more stable. We find that for very low p, source routing schemes perform much worse than dynamic. Note that the SSDijkstra curve has a shape $\propto 1/p$. This is not surprising since each link is in steady-state

[4]Since we are only studying the flow of a single message through the dynamic network, we did not use a discrete event simulation for our simulations.

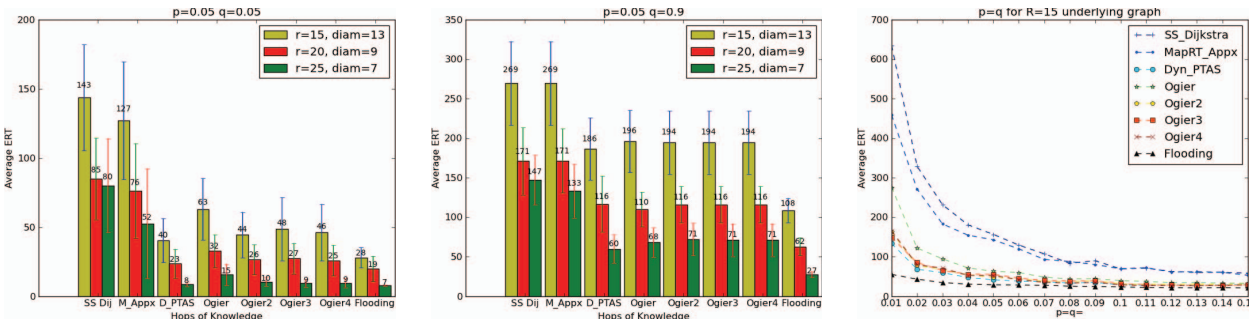


Fig. 3. Algorithm performance for low p values (left and middle) and for $p = q$ (right) (100 nodes are dropped randomly in a square; transmission radius r)

with expected delay $1 + 1/p$. MapRTAppx performs slightly better but is outperformed by the dynamic schemes. Among these, Ogier does the worst, probably because the underlying graph is sparse (close to the critical radius of connectivity), which limits the chances for course correction. The schemes using two or more hops of link state perform better, with DynHeuristic best among them.

MIT Reality Mining Data: The MIT Reality Mining data set is an intermittent connectivity trace for 106 mobile nodes. Fig. 4 illustrates the routing performance between nodes belonging to 3 different classes of node degrees in the underlying graph (low (L), medium (M), and high (H)). We plot two different metrics: a) delivery failure rate and b) expected latency of delivered packets. We observe that all flows going towards high-degree nodes get delivered while a large fraction of flows going towards low-degree nodes do not arrive within the simulation time. In the extreme L-L case, about 20% flows do not make it due to the lack of a non-contemporaneous path. However, Fig. 4(b) indicates that the latency of the delivered messages is not too bad even in the L-L case. This means for some cases, source routing is not a bad choice if one is prepared to tolerate a certain fraction of lost messages. If not, then dynamic routing is preferable.

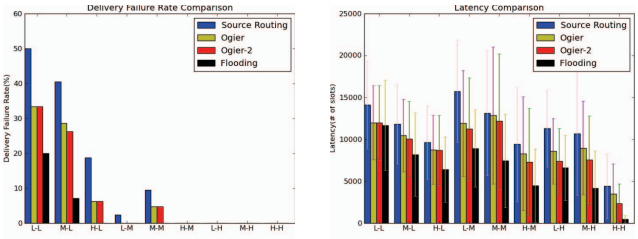


Fig. 4. Relative performance in MIT Reality Mining Data Set. DynHeuristic has high time complexity, hence its performance is not plotted here.

We observe asymmetries in the results. For example, a node with low degree (in $G_\cup$) has lower failure rate as a source than as a destination. This is because as a source, a low-degree node can avail any of the few opportunities to push the message out, which then gets routed easily through better connected neighborhoods. In contrast, as a destination, a low-degree node with the same number of contact opportunities is less likely to receive the message before the opportunity passes. We also observe that Ogier-2 outperforms Ogier in all situations. The fractional gains are the highest in H-H, M-H, H-M and L-H scenarios, where the overall latency is low to begin with.

VI. Conclusion and Future Work

This paper constitutes the first step towards a theory of routing in time-correlated Markovian networks. We found that model-assisted dynamic routing generally outperforms source routing significantly in highly intermittently connected networks. We also found that lookahead beyond 1-hop is useful, particularly in sparse networks where mistakes can be hard to correct. Much more lookahead, however, did not buy much further improvement. The proposed dynamic heuristic outperformed source routing as well as Ogier's dynamic routing algorithm. Reducing the heuristic's time complexity and seeking performance guarantees are topics of future research.

References

[1] U. G. Acer, S. Kalyanaraman, and A. A. Abouzeid. DTN routing using explicit and probabilistic routing table states. *Wireless Networks*, 17:1305–1321, 2011.

[2] A. Balasubramanian, B. N. Levine, and A. Venkataramani. DTN routing as a resource allocation problem. In *SIGCOMM*, 2007.

[3] A. Bar-Noy and B. Schieber. The Canadian Traveller Problem. In *SODA*, 1991.

[4] S. Biswas and R. Morris. Exor: opportunistic multi-hop routing for wireless networks. In *SIGCOMM*, 2005.

[5] T. Clausen, P. Jacquet, A. Laouiti, P. Muhlethaler, A. Qayyum, and L. Viennot. Optimized Link State Routing Protocol. In *INMIC*, 2001.

[6] D. Fried, S. E. Shimony, A. Benbassat, and C. Wenner. Complexity of Canadian traveler problem variants. *Theor. Comput. Sci.*, 487:1–16, 2013.

[7] R. W. Hall. The fastest path through a network with random time-dependent travel times. *Transport. Science*, 20(3):182–188, 1986.

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

[9] D. B. Johnson, D. A. Maltz, and J. Broch. DSR: the dynamic source routing protocol for multihop wireless ad hoc networks. In *Ad Hoc Networking*, pages 139–172. Addison-Wesley, 2001.

[10] E. P. C. Jones, L. Li, and P. A. S. Ward. Practical routing in delay-tolerant networks. In *SIGCOMM Workshop: WDTN*, 2005.

[11] D. A. Levin, Y. Peres, and E. L. Wilmer. *Markov Chains and Mixing Times*. American Mathematical Society, 2008.

[12] C. Lott and D. Teneketzis. Stochastic routing in ad-hoc networks. *IEEE Trans. Automat. Contr.*, 51(1), 2006.

[13] P. Nain, D. Towsley, M. P. Johnson, P. Basu, A. Bar-Noy, and F. Yu. Computing Traversal Times on Dynamic Markovian Paths. In *MobiCom Workshop: CHANTS*, 2013.

[14] E. Nikolova and D. R. Karger. Route planning under uncertainty: the canadian traveller problem. In *AAAI*, 2008.

[15] R. G. Ogier and V. Rutenburg. Minimum-expected-delay alternate routing. In *INFOCOM*, 1992.

[16] C. E. Perkins and E. M. Royer. Ad-hoc on-demand distance vector routing. In *IEEE WMCSA*, 1997.

[17] A. Vahdat, D. Becker, et al. Epidemic routing for partially connected ad hoc networks. Technical report, CS-200006, Duke University, 2000.