Robust Routing in Networks With Time-Varying Links

Victoria Manfredi
Dept. Computer Science
University of Massachusetts
Amherst MA USA
{vmanfred}@cs.umass.edu

1. INTRODUCTION

In this work we consider the problem of routing in networks with time-varying links. We focus on wireless and mobile ad-hoc networks where link changes occur at the timescale of seconds to minutes. Since links are changing, to maintain network connectivity paths must be periodically recomputed, incurring increased control overhead. Suppose, however, we can perform routing in such a way that paths are "robust" to link changes in the network: i.e., link changes may occur, but paths still perform well enough. Then paths can be recomputed less frequently and the control overhead decreased. In this work we specifically examine the problem of how to perform such "robust" routing.

As a motivating example, consider the effect on throughput when paths are updated every T timesteps rather than every link change. We examine a 6×6 torus network where each link has capacity 1.0 and links are up according to a 2-state process (a link stays up with probability 0.95 and stays down with probability 0.5). To select a set of paths we randomly choose 10 source-destination pairs; for each pair we use the two shortest disjoint paths (computed greedily). We then use the multi-path rate controller described in [1] to distribute flow over the paths. While we update the paths between the sources and destinations only every T timesteps, we update the rates every timestep. We record the total throughputs for 40 timesteps for a given set of sources and destinations and a given update interval T, and we average the throughputs over 81 sets of randomly chosen sources and destinations. This average throughput over time is shown in Figure 1. As T increases, the average throughput decreases, but when T is sufficiently large, the average throughput begins to level out. Since $T = 1$ is the maximum throughput, we see the loss in optimality as paths are recomputed less frequently. Figure 1 also shows how the transient and steady-state behaviours of the set of paths differ. We now use these observations to develop a routing algorithm.

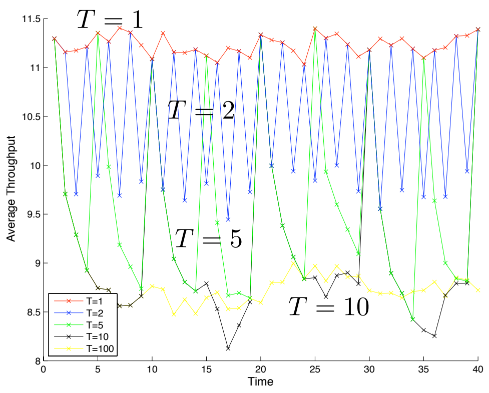

Figure 1: Throughput as a function of time.

2. ROUTING ALGORITHM

Our routing algorithm is summarized in Table 1. Let N be the graph of the entire network and let the graph G initially equal N. In line 3 we select the best path P from G. In line 4 we construct a braid around P and set the subgraph formed by this braid to be the graph G'. In line 5 we select an additional path P' from N and reset $G = G' \cup P'$. Finally, in line 6 we perform local forwarding over the graph G for T timesteps. Note that G does not grow without bound since it is rebuilt every T, always starting from only path P. We next discuss the important features of our algorithm.

Recomputing routes periodically. We select, not the set of paths that is optimal at the time at which paths were recomputed, but the set that will be most "robust" over an interval T until paths are next recomputed.

Building a braid of paths over which to perform routing. Rather than selecting a single path between a source and destination, we select a braid of non-disjoint paths. We describe here two possible metrics to use to select the braid: connectivity and throughput.

Suppose connectivity is our metric. We can compute the probability $p_{up}(t)$ that there is at least one path up between a source and destination at time t for a braid of paths using a time-dependent version of the reliability polynomial from [2]. Rather than use the steady-state probability that a link is up we use the transient probability that a link that is initially down (up) is up at time t based on known results for the M/M/1/1 queueing model. We then define the random variable U to be the probability that there is at least one path up between each pair of nodes at a uniformly randomly chosen time during the interval T. Then $E[U] = \frac{1}{T}\int_0^T p_{up}(t)dt$ is the metric with which we evaluate braids.

Suppose throughput is our metric. Using backpressure routing [3] we can compute the throughput of a path or braid in a decentralized way. In backpressure routing, each node keeps a separate queue for each outgoing link and destination pair. Then upon receipt of a packet for a particular destination, a node computes for each outgoing link the current difference in length of the node's queue and the queue at the other end of the link for the given destination. The packet is then forwarded over the link with the largest queue length difference. We use these differences in queue length to obtain an estimate of the throughput of the braid.

We then construct the braid as follows. The best path P is the path for which $E[U]$ is maximized (computed in a centralized way) or the maximum throughput path (computed in a decentralized way) between the source and destination, in the previous graph G. We then build a new braid G' around P by adding any node (and its edges) within k hops of any node in P. We will keep track of the throughput or $E[U]$ values obtained for different values of k, and use this information when choosing a value of k with which to build the braid.

Selection of an alternate path P'. We use an additional path P' whose purpose is to ensure that if there is a better best path than the one currently in use, it is eventually found. P' is an uniform randomly chosen path from the entire network with probability ϵ. With probability $1-\epsilon$, P' is a "promising" path in the network. Note that P' may overlap with nodes or links in the braid. What constitutes a promising path? We know that any path must use at least one link from the source and one link into the destination. To select a promising path then, we first give each node a weight based on its degree, the probability with which its links are up, and its novelty (e.g., how recently has it been tried and how many link changes has it undergone since it was last tried). The source then greedily chooses to use the outgoing link with the highest weight, with successive links similarly chosen greedily. Path branching is constrained by the current best path length.

Local forwarding over G. Given the sub-graph G,

Table 1: Robust routing algorithm.

```
1   Let $G$ initially be graph of entire network
2   Loop every $T$:
3       Select best path $P$ from graph $G$
4       Build braid around $P$ to obtain graph $G'$
5       Choose additional path $P'$; set $G = G' \cup P'$
6       Perform local forwarding on $G$
```

rather than forwarding packets over a path, we consider all of G: i.e., we make local decisions to select the next hop out of all possible next hops within G. While the sub-graph G changes every T timesteps, local forwarding decisions are computed by nodes every timestep. To perform local forwarding, we again use backpressure routing [3], since it reflects the current and past downstream drain rates of the braid (implicitly incorporating links being up or down, and therefore more suitable to a changing environment than [1]), is throughput optimal (in a stationary environment). and links together the behaviour of multiple sessions (rather than just a single source and destination).

3. FUTURE WORK

We are interested in other ways to select the braid and perform local forwarding, and in the impact of the update interval T. Finally, we are currently evaluating the decrease in performance as less control overhead is used, or as paths are updated less frequently.

4. ACKNOWLEDGMENTS

The author thanks Majid Ghaderi, Matt Grossglauser, Robert Hancock, Jim Kurose, Patrick Thiran, and Don Towsley for their help. This research was supported in part under the International Technology Alliance sponsored by the U.S. Army Research Laboratory and the U.K. Ministry of Defence under Agreement Number W911NF-06-3-0001, and by the the National Science Foundation via an International Research in Engineering Education supplement to Engineering Research Centers Program award number EEC-0313747. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and should not be interpreted as representing the official policies, either expressed or implied, of the US Army Research Laboratory, the US National Science Foundation, the US Government, the UK Ministry of Defense, or the UK Government.

5. REFERENCES

[1] H. Han, S. Shakkottai, C. V. Hollot, R. Srikant, and D. Towsley, "Multi-path tcp: A joint congestion control and routing scheme to exploit path diversity on the internet," *IEEE/ACM Transactions on Networking*, vol. 14:6, 2006.

[2] C. J. Colbourn, *The Combinatorics of Network Reliabiilty.* New York: Oxford University Press, 1987.

[3] L. Tassiulas and A. Ephremides, "Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks," *IEEE Transactions on Automatic Control*, vol. 37:12, 1992.