Robust Routing in Dynamic MANETs

Victoria Manfredi[†], Robert Hancock[‡], Jim Kurose[†]

Technical Report 2008-25
August 12, 2008

[†]Department of Computer Science
140 Governors Drive
University of Massachusetts
Amherst, Massachusetts 01003-4601
{vmanfred, kurose}@cs.umass.edu

[‡]Roke Manor Research Ltd
Romsey, Hampshire, U.K.
robert.hancock@roke.co.uk

Abstract

Wireless networks, and mobile ad-hoc networks (MANETs) in particular, are characterized by time-varying link characteristics and network topology. In such an environment, the network must accommodate the changes, providing end-end packet delivery while at the same time incurring low control overhead. This paper proposes a routing algorithm for MANETs with the primary goal of maximizing connectivity while limiting overhead. Rather than using one or more disjoint paths between a source and destination, a set of non-disjoint paths, or braid, is selected. To adapt to link changes, local rerouting is performed within the braid, thus avoiding network-wide recalculations. We analytically characterize the source-destination connectivity of the braid. Through simulation, we compare the reliability of braided routing and various other MANET routing protocols, including Ad-hoc On-Demand Distance Vector (AODV) routing, and quantify the relative amounts of control overhead incurred by braided routing and AODV.

1 Introduction

Wireless networks, and mobile ad-hoc networks (MANETs) in particular, are characterized by time-varying link characteristics and network topology. In such an environment, the network must accommodate the changes, providing end-end packet delivery while still achieving low control overhead. Yet this ideal is difficult to meet in practice: end-end delivery requires some form of end-end (potentially global) coordination, and frequent changes make adaptation to each and every change costly. Link and mobility characteristics may also be difficult to estimate *a priori*, making proactive or predictive routing approaches difficult to implement in practice.

In this paper, we investigate *robust* routing in MANETs. By "robust" we mean that although a particular routing configuration (in our case, a set of multipath routes) may not be optimal for a single *specific* configuration (e.g., specific network topology and link characteristics), it will perform well over a larger set of likely network configurations: i.e., it is robust to changes without requiring global recomputation. The issues of local versus global adaptation to link/topology changes, and the timescale(s) at which this adaptation occurs (and the concomitant overhead incurred) are central to the MANET routing problem. There is a growing recognition [2, 3, 4] that the scale and dynamic nature of MANETS present severe challenges for classical MANET protocols, which are conceptually based on maintaining a consistent network-wide topological viewpoint. In contrast, the approach to MANET routing explored in this paper is based on the intuition that a global routing configuration should be determined at a coarse time-scale (e.g., periodically, every T time units), with local adaptation to link or topology changes occurring at a finer time-scale within the current global configuration.

This paper specifically investigates an approach towards MANET routing, which we refer to as "braided routing," that is robust to changes in link characteristics and network topology. Informally, braided routing operates at two timescales. At the longer time-scale, a routing subgraph (i.e., a braid, defined formally in Section 3) is constructed that connects a source and destination. At the shorter time-scale, local forwarding decisions are made to select the "best" next hop out of all possible next hops. Unlike many existing "backup routing" approaches that pre-compute disjoint paths, e.g., [12], or partially disjoint paths, e.g., [7], a braid does not impose such requirements on the subgraph. Like approaches such as [7], braided routing performs local adaptation in response to link and topology changes. But unlike approaches that route packets over the entire network topology to achieve robustness (e.g., [26]), the braid subgraph over which packets are forwarded is purposefully limited to limit control overhead (e.g., for braid construction and state maintenance). The tradeoff between the control overhead incurred (which depends in turn on the size of the braid and the interval at which the braid is re-computed), and packet delivery performance will be of principal concern to us. Our results show that braided routing can indeed achieve a performance gain over a traditional MANET routing algorithm such as Ad-hoc On-Demand Distance Vector (AODV) routing, and other approaches such as disjoint path routing, without significantly increasing overhead.

We analyze braided routing from several different viewpoints in order to fully explore and understand its properties. We analytically characterize the reliability (the probability that the source and destination nodes have an instantaneous path) of a class of braids, their optimality properties, and counter-examples to conjectured optimality properties in a well-structured (grid) network. Through simulation, we compare the reliability of braided, disjoint-path, and full-network routing in both torus and random networks. Finally, we investigate the control overhead of braided routing and AODV from implementations of these two protocols in a GloMoSim simulation of a mobile scenario. In addition to quantifying the gains and overheads of braided routing, our simulations also illustrate the impact and subtleties involved with using different underlying network models.

The remainder of this paper is structured as follows. In Section 2, we discuss related work on backup routing. In Section 3, we describe the reliability metric we use as a measure of robustness. In Section 4, we present analytic results evaluating our braid structure in terms of reliability, while in Section 5. we present analytic results comparing braids and disjoint paths. Section 6 gives experimental results evaluating our braid structure. Then in Section 7 we propose a simple algorithm for constructing and maintaining a braid and present simulation results evaluating its performance. Finally, Section 8 summarizes our results and outlines future work.

2 Related Work

A variety of other work has considered the use of disjoint routes in ad hoc networks, including [13, 16, 20, 23]. In addition to the overhead cost of finding disjoint paths, if any link in a path breaks then the path itself breaks. Detection and recovery from failures is also expensive since it cannot be carried out locally. These considerations have thus motivated research on the use of non-disjoint paths.

Considering non-disjoint paths, backup routing [14] reinforces the path selected by AODV [21] by allowing nodes that overhear AODV control messages to become part of the routing subgraph, to be used only when links on the AODV path break. [24] proposes duct routing in mobile packet radio networks, allowing nodes neighbouring the primary path to be used. When sending packets to the ith hop node along the primary path, one of either the ith hop node or one of its neighbours will hear the transmission first. The first node that hears the transmission will forward the packet to the $(i+1)$st hop node; the other nodes will overhear the forwarding transmission and refrain from transmitting. For underwater networks, [19] proposes a geo-routing mesh using only nodes within a given distance from the vector from the source or current forwarding node to the sink. Finally, braided multipaths are proposed in [7] to protect against node failure. A braided multipath corresponds to selecting a primary path and then adding an additional path for each node i on the primary path that does not use node i, possibly reusing parts of the primary path. We note that [24] (when all nodes neighbouring the primary path are used) and [14, 19] build routing subgraphs which structurally correspond to what we will describe in Section IV as a 1-hop braid. Our work generalizes that of [24, 14, 19] since we consider k-hop braids, with $k \geq 1$. Our work differs from [7] in the construction and structure of the routing subgraph.

For changing network topology, [27] show for a class of graphs that it is possible to maintain paths whose lengths are within a constant factor of the shortest path while limiting overhead. Considering the Internet, [18] propose splicing together paths identified using different parameter settings of the routing protocol (e.g., such as different perturbations of the link weights in the network); the increased path diversity then gives increased reliability when links fail. Focusing on reliability, [17] argues for the reliability benefits of using non-disjoint paths in wireless mesh networks, showing gains over disjoint paths. Also focusing on reliability, [8] considers the problem of finding the most reliable subgraph for routing. Due to the #P-hardness of this problem, they propose a method to approximately compute reliability and leverage known contact probabilities between node pairs to select a routing subgraph. Finally, [25] propose a routing algorithm that first selects the most reliable path and then locally reinforces those links whose probability of being up is lower than a threshold. Unlike [8, 25], our work focuses on identifying those properties of a routing subgraph which make it reliable, and then efficiently identifying such a subgraph without actually computing reliability.

3 What do we mean by robust?

Informally, we consider a routing subgraph to be *robust* if there is at least one path up between the source and destination with high probability. If a subgraph is robust, then even if a link or path between the source and destination breaks, an alternative link or path is available with (high) probability. In reliability theory [5], the probability that there is at least one path up between a source and destination is known as 2-terminal reliability, a metric we will use for evaluating the robustness of different routing subgraphs and providing intuition about what types of graphs are "highly" reliable.

Following Colbourn [5], the 2-terminal reliability of a graph $G = (V, E)$ with IID edges up with probability p is given by,

$$R(G,p) \quad = \quad \sum_{i=0}^{m} N_i p^i (1-p)^{m-i} \tag{1}$$

where $m = |E|$, N_i is the number of pathsets with i edges, and $p^i(1-p)^{m-i}$ is the probability that a pathset with i edges is up. A pathset is defined as a subset of edges for which there is a route between the two terminals.

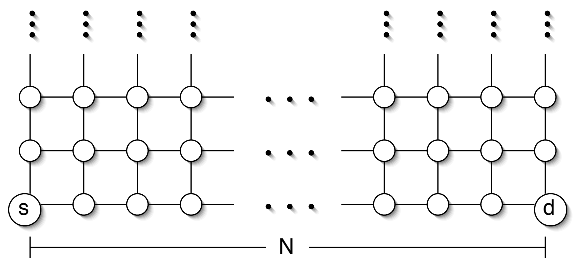


Figure 1: Model used in Section 4, comprising source (s) and destination (d) on a line in a bounded half-plane grid.

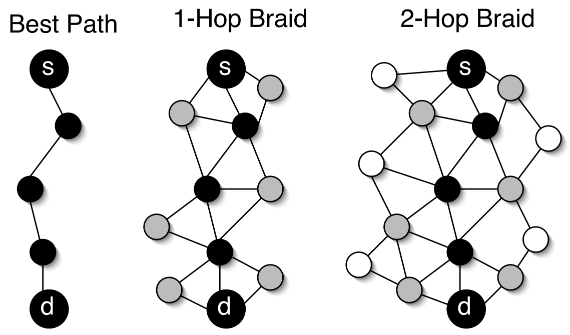


Figure 2: Example best path, 1-hop braid, and 2-hop braid between a source (s) and destination (d).

Ideally, for a given source and destination, and specified number of edges or nodes, we would select the subgraph that has maximum 2-terminal reliability while using at most the specified number of links or nodes. Computing reliability exactly, however, is generally $\#P$-complete [5], as is solving the corresponding optimization problem [8]. For all-terminal reliability (the probability that a graph is connected), [11] gives a randomized fully polynomial time approximation scheme. For very reliable graphs, [11] shows that only small cuts are likely to fail and that there are only a polynomial number of such cuts, otherwise Monte Carlo simulation may be used. The approach in [11] could presumably be used to approximate 2-terminal reliability, although this does not efficiently solve the optimization problem, nor lend itself easily to theoretical comparisons of the reliability of different subgraphs.

Given the difficulty of exactly computing reliability, except in the case of relatively simple networks, we also use simulation to estimate reliability. In a discrete-time simulation of a time-varying network, we can check whether there is a path from the source to the destination at each time-step. The ratio of the number of time-steps where there was a path and the total number of time-steps simulated is then an estimate of the probability of there being at least one path from source to destination. We refer to computing the reliability in this way as "computing the reliability experimentally."

4 Braided Graphs

In this section, we characterize the properties of braids, concentrating on well-structured grid networks. Our goal is to analyze how well a braided routing subgraph performs with respect to 2-terminal reliability given a fixed number of nodes or links in the subgraph. These results then provide insight into more general network topologies, which are analytically intractable.

4.1 The k-hop Braid

Our goal here is to explore the use of a k-hop braid built around the best (most reliable) path. A k-hop braid consists of the path itself, and all nodes and links within k hops of nodes on the best path. Figure 2 shows an example best path, 1-hop braid, and 2-hop braid between a source s and destination d.

In the small p limit, the reliability, see Equation (1), is dominated by terms from shorter paths; this indicates that in this limit, the most reliable path is an appropriate part of the braid. Conversely, [5, 22] gives an alternative expression for reliability as a polynomial in $q = 1-p$ with source, s, and destination, d, as follows,

$$\begin{aligned} R(G,p) &= 1 - \sum_{C_i \in \mathcal{C}} P(E_i) \\ P(E_i) &= q^{|C_i|}\left[1 - \sum_{C_j \in L(C_i)} \frac{P(E_j)}{q^{|C_i \cap C_j|}}\right] \end{aligned} \tag{2}$$

where C_i is the set of edges in minimal cut i (partitioning s and d), $\mathcal{C}$ is the set of all C_i, and, informally, $L(C_i)$ is the set of minimal cuts lying entirely between node s and edges in cut C_i. In the small q limit, the unreliability $1 - R(G,p)$ is dominated by the smallest cuts, so we observe that a good braid will have large minimal cut: i.e., the braid should widen uniformly along the shortest path. In Lemma 1, we show this for arbitrary p and $k = 1$ for the idealised network model shown in Figure 1. Informally, Lemma 1 says that when incrementally adding nodes (one or two at a time), adding all nodes one hop away from the shortest path before adding any nodes that are two hops away maximizes reliability.

Lemma 1: *Assume the network structure in Figure 1: the source, s, and destination, d, are connected by a shortest path, P, comprising N nodes; links are IID and up with probability $0 < p < 1$. Let G be the sub-graph formed by P plus $0 < n < N$ additional 1-hop nodes, (where a k-hop node is a node k hops away from P). Using one additional 1-hop node (and its associated edges) that is also adjacent to another 1-hop node, increases the reliability of G strictly more than does using any two additional 2-hop nodes (and their associated edges),*

Proof: Figure 3(a) shows the general structure of the graphs we consider. Suppose we can add either one of the grey nodes or the black node. Adding only one of the grey nodes will not affect the reliability from s to d, as no links will be reinforced, while adding the black node will increase the probability of getting from nodes d_0 and d_1 to node d.

Now consider adding two nodes at a time. Again consider the topologies in Figure 3. We partition the top graph in Figure 3 into the sub-graphs shown in the bottom graph; note that each edge appears only once, although nodes may be repeated (which will not affect the reliability). Using sub-graph decompositions we decompose the reliability by conditioning on the intermediate nodes as follows. We first condition on intermediate nodes s_0 and s_1 to obtain,

$$P(d|s) = P(d|s_0 s_1)P(s_0 s_1|s) + P(d|s_0 \bar{s}_1)P(s_0 \bar{s}_1|s) + P(d|\bar{s}_0 s_1)P(\bar{s}_0 s_1|s) \tag{3}$$

where e.g., $P(d|s_0\bar{s}_1)$ is the probability that node d can be reached given that node s_0 but not s_1 can be reached, and $P(s_0\bar{s}_1|s)$ is the probability that node s_0 but not s_1 can be reached given that node s can be reached. We recursively condition on nodes $\{d_0, d_1\}$ and $\{q_0, q_1\}$ to obtain an equation for $P(d|s)$ as the sum of 27 terms, shown in Table 1.

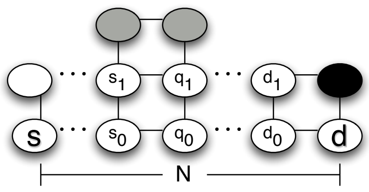
We use the resulting equation to compute both the reliability when adding both of the grey nodes in Figure3 and when adding the black node. Ignoring those terms that correspond to subgraphs that are identical for both we need only compute the reliability for the $\{s_0, s_1\} \rightarrow \{q_0, q_1\}$ and $\{d_0, d_1\} \rightarrow d$ subgraphs. These calculations are shown in Table 2. Examining Table 2 shows that for each $P(\{q_0, q_1\}|\{s_0, s_1\})P(d|\{d_0, d_1\})$

$$
\begin{aligned}
P(d|s) &= P(d|s_0s_1)P(s_0s_1|s) + P(d|s_0\bar{s}_1)P(s_0\bar{s}_1|s) + P(d|\bar{s}_0s_1)P(\bar{s}_0s_1|s) \\
\\
&= [P(d|d_0d_1)P(d_0d_1|s_0s_1) + P(d|d_0\bar{d}_1)P(d_0\bar{d}_1|s_0s_1) + P(d|\bar{d}_0d_1)P(\bar{d}_0d_1|s_0s_1)]P(s_0s_1|s) + \\
&\quad [P(d|d_0d_1)P(d_0d_1|s_0\bar{s}_1) + P(d|d_0\bar{d}_1)P(d_0\bar{d}_1|s_0\bar{s}_1) + P(d|\bar{d}_0d_1)P(\bar{d}_0d_1|s_0\bar{s}_1)]P(s_0\bar{s}_1|s) + \\
&\quad [P(d|d_0d_1)P(d_0d_1|\bar{s}_0s_1) + P(d|d_0\bar{d}_1)P(d_0\bar{d}_1|\bar{s}_0s_1) + P(d|\bar{d}_0d_1)P(\bar{d}_0d_1|\bar{s}_0s_1)]P(\bar{s}_0s_1|s) \\
\\
&= [P(d_0d_1|q_0q_1)P(q_0q_1|s_0s_1) + P(d_0d_1|q_0\bar{q}_1)P(q_0\bar{q}_1|s_0s_1) + P(d_0d_1|\bar{q}_0q_1)P(\bar{q}_0q_1|s_0s_1)]P(d|d_0d_1)P(s_0s_1|s) + \\
&\quad [P(d_0\bar{d}_1|q_0q_1)P(q_0q_1|s_0s_1) + P(d_0\bar{d}_1|q_0\bar{q}_1)P(q_0\bar{q}_1|s_0s_1) + P(d_0\bar{d}_1|\bar{q}_0q_1)P(\bar{q}_0q_1|s_0s_1)]P(d|d_0\bar{d}_1)P(s_0s_1|s) + \\
&\quad [P(\bar{d}_0d_1|q_0q_1)P(q_0q_1|s_0s_1) + P(\bar{d}_0d_1|q_0\bar{q}_1)P(q_0\bar{q}_1|s_0s_1) + P(\bar{d}_0d_1|\bar{q}_0q_1)P(\bar{q}_0q_1|s_0s_1)]P(d|\bar{d}_0d_1)P(s_0s_1|s) + \\
&\quad [P(d_0d_1|q_0q_1)P(q_0q_1|s_0\bar{s}_1) + P(d_0d_1|q_0\bar{q}_1)P(q_0\bar{q}_1|s_0\bar{s}_1) + P(d_0d_1|\bar{q}_0q_1)P(\bar{q}_0q_1|s_0\bar{s}_1)]P(d|d_0d_1)P(s_0\bar{s}_1|s) + \\
&\quad [P(d_0\bar{d}_1|q_0q_1)P(q_0q_1|s_0\bar{s}_1) + P(d_0\bar{d}_1|q_0\bar{q}_1)P(q_0\bar{q}_1|s_0\bar{s}_1) + P(d_0\bar{d}_1|\bar{q}_0q_1)P(\bar{q}_0q_1|s_0\bar{s}_1)]P(d|d_0\bar{d}_1)P(s_0\bar{s}_1|s) + \\
&\quad [P(\bar{d}_0d_1|q_0q_1)P(q_0q_1|s_0\bar{s}_1) + P(\bar{d}_0d_1|q_0\bar{q}_1)P(q_0\bar{q}_1|s_0\bar{s}_1) + P(\bar{d}_0d_1|\bar{q}_0q_1)P(\bar{q}_0q_1|s_0\bar{s}_1)]P(d|\bar{d}_0d_1)P(s_0\bar{s}_1|s) + \\
&\quad [P(d_0d_1|q_0q_1)P(q_0q_1|\bar{s}_0s_1) + P(d_0d_1|q_0\bar{q}_1)P(q_0\bar{q}_1|\bar{s}_0s_1) + P(d_0d_1|\bar{q}_0q_1)P(\bar{q}_0q_1|\bar{s}_0s_1)]P(d|d_0d_1)P(\bar{s}_0s_1|s) + \\
&\quad [P(d_0\bar{d}_1|q_0q_1)P(q_0q_1|\bar{s}_0s_1) + P(d_0\bar{d}_1|q_0\bar{q}_1)P(q_0\bar{q}_1|\bar{s}_0s_1) + P(d_0\bar{d}_1|\bar{q}_0q_1)P(\bar{q}_0q_1|\bar{s}_0s_1)]P(d|d_0\bar{d}_1)P(\bar{s}_0s_1|s) + \\
&\quad [P(\bar{d}_0d_1|q_0q_1)P(q_0q_1|\bar{s}_0s_1) + P(\bar{d}_0d_1|q_0\bar{q}_1)P(q_0\bar{q}_1|\bar{s}_0s_1) + P(\bar{d}_0d_1|\bar{q}_0q_1)P(\bar{q}_0q_1|\bar{s}_0s_1)]P(d|\bar{d}_0d_1)P(\bar{s}_0s_1|s)
\end{aligned}
$$

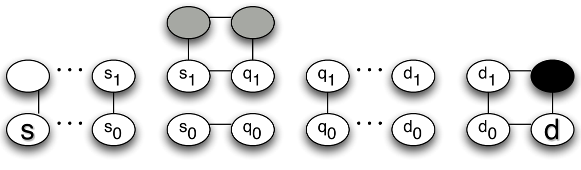
Table 1: Reliability computation for the top graph in Figure 3.

	Two Nodes on Top	One Node at End
$P(q_0q_1\|s_0s_1) = P(q_0\|s_0)P(q_1\|s_1)$	$p(p + p^3 - p^4)$	$p \cdot p$
$P(q_0\bar{q}_1\|s_0s_1) = P(q_0\|s_0)P(\bar{q}_1\|s_1)$	$p(1 - p - p^3 + p^4)$	$p(1-p)$
$P(\bar{q}_0q_1\|s_0s_1) = P(\bar{q}_0\|s_0)P(q_1\|s_1)$	$(1-p)(p + p^3 - p^4)$	$(1-p)p$
$P(q_0q_1\|s_0\bar{s}_1) = P(q_0\|s_0)P(q_1\|\bar{s}_1)$	0	0
$P(q_0\bar{q}_1\|s_0\bar{s}_1) = P(q_0\|s_0)P(\bar{q}_1\|\bar{s}_1)$	p	p
$P(\bar{q}_0q_1\|s_0\bar{s}_1) = P(\bar{q}_0\|s_0)P(q_1\|\bar{s}_1)$	0	0
$P(q_0q_1\|\bar{s}_0s_1) = P(q_0\|\bar{s}_0)P(q_1\|s_1)$	0	0
$P(q_0\bar{q}_1\|\bar{s}_0s_1) = P(q_0\|\bar{s}_0)P(\bar{q}_1\|s_1)$	0	0
$P(\bar{q}_0q_1\|\bar{s}_0s_1) = P(\bar{q}_0\|\bar{s}_0)P(q_1\|s_1)$	$p + p^3 - p^4$	p
$P(d\|d_0d_1)$	p	$\leq p + p^3 - p^4$
$P(d\|d_0\bar{d}_1)$	p	$p + p^3 - p^4$
$P(d\|\bar{d}_0d_1)$	p^2	$2p^2 - p^4$

Table 2: Reliability computations for the bottom subgraphs in Figure 3.

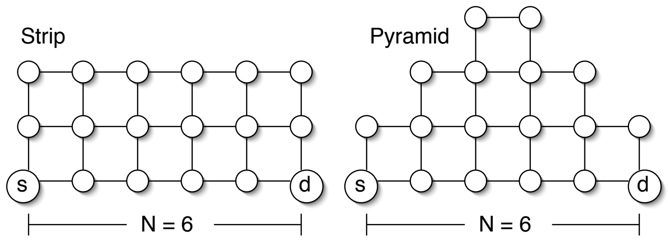


(a)

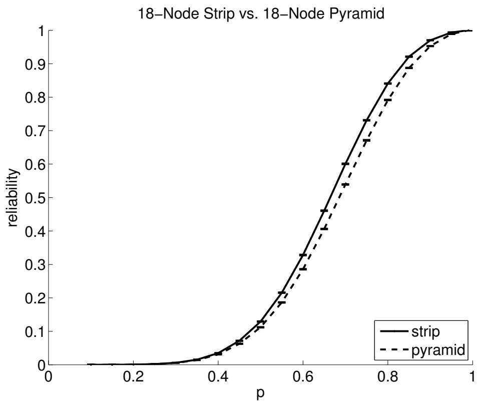


(b)

Figure 3: Graphs used in Lemma 1. We decompose the graph shown in (a) into the subgraphs shown in (b) so we need only compute the reliability for the subgraphs of interest.



(a)



(b)

Figure 4: (a) Two topologies, both using 18 nodes. (b) Reliability is averaged over 100 runs of 10,000 time-steps each. 95% bootstrap confidence intervals over the runs are shown.

product, adding the black node to the end of the $2 \times N$ node strip gives the same or higher reliability as compared with adding the two grey nodes anywhere on top of the strip. $\diamondsuit$

Lemma 1 is more general than stated as it does not depend on the form of the subgraphs between s and $\{s_0, s_1\}$, or $\{q_0, q_1\}$ and $\{d_0, d_1\}$. These results suggest that k-hop braids have several desirable reliability properties, at least in this well-structured environment and with uniform p, giving confidence in studying k-hop braids in scenarios where the optimum subgraph cannot be determined. In Figure 1, the $k \times N$ node strip from s to d is a $(k-1)$-hop braid. We close with two conjectures:

Conjecture 1: *Given N additional nodes (and their associated edges) plus the shortest path, the* $2 \times N$ *node strip is the most reliable subgraph.* It follows from Lemma 1 that for $N \leq 5$ additional nodes, reliability is maximized by the $2 \times N$ node strip,

Conjecture 2: *Given* $2N$ *additional nodes (and their associated edges) plus the shortest path, the* $3 \times N$

node strip is more reliable than the corresponding pyramid. Comparing the $3 \times N$ node strip for $N = 6$ versus the pyramid that can be built using 18 nodes, see Figure 4 (a), we find experimentally that the strip has higher reliability than the pyramid, as shown in Figure 4(b).

4.2 Exact Results for $2 \times N$ Node Strip

4.2.1 Definitions

For the $2 \times N$ node strip, shown in Figure 5(a), we determine an exact expression for reliability (see Appendix B for a discussion of the $3 \times N$ node strip). Define

$$\begin{aligned} R_N &\equiv P(s \text{ is connected to } d) \\ S_N &\equiv P(s \text{ is connected to } d') \\ T_N &\equiv P(s \text{ is connected to both } d \text{ and } d') \end{aligned}$$

where s and d are the source and destination respectively, and d' is the node diagonally opposite to s. All of these expressions are simple connectedness problems, in that for any pathset which contributes to an expression, all supersets of the pathset contribute to the same expression. For small graphs (low values of N), the expressions can be calculated by brute force, by enumerating all the minpaths (pathsets from which no links can be removed) and then using the inclusion-exclusion principle. The first few values of R_N, S_N, and T_N can be calculated as:

$$\begin{aligned} R_1 &= p + p^3(1-p) \\ R_2 &= p^2 + p^4(1-p)(3 + p - 2p^2) \\ R_3 &= p^3 + p^5(1-p)(6 + 3p - 4p^2 - 6p^3 + 4p^4) \\ R_4 &= p^4 + p^6(1-p)(10 + 6p - 4p^2 - 18p^3 - 2p^4 + 20p^5 - 8p^6) \\ \\ S_1 &= p^2 + p^2(1-p)(1+p) \\ S_2 &= p^3 + p^3(1-p)(2 + 2p + p^2 - 2p^3) \\ \\ T_1 &= p^2 + p^3(1-p)(2) \\ T_2 &= p^3 + p^4(1-p)(3 + 4p - 4p^2) \end{aligned}$$

The expressions R_N, S_N and T_N are not disjoint. We therefore define

$$\begin{aligned} r_N &= P(s \text{ is connected to } d \text{ and not to } d') \\ s_N &= P(s \text{ is connected to } d' \text{ and not to } d) \end{aligned}$$

in terms of which

$$\begin{aligned} r_N &= R_N - T_N && (4) \\ s_N &= S_N - T_N && (5) \end{aligned}$$

4.2.2 Recursion Relationships

We now develop recursion relationships for the reliability expressions for the $2 \times (N+1)$ node strip, shown in Figure 5(b), in terms of the expressions for the $2 \times N$ node strip. We consider cutting the strip in the middle

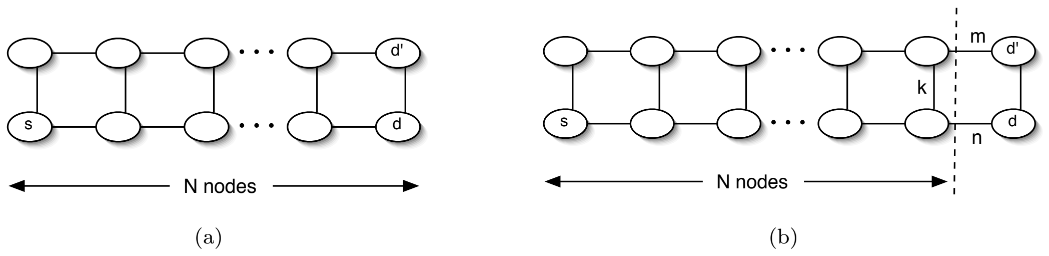


Figure 5: (a) The $2 \times N$ node strip. (b) The $2 \times (N+1)$ node strip.

of the last block, and partition the expressions depending on whether only the lower cut link, the upper cut link, or both cut links are reachable from s, see Figure 5(b). Note that for the conditions represented by r_N and s_N, the vertical link before the cut, marked k in Figure 5(a), must be unavailable, because otherwise either both the cut links would be simultaneously accessible or neither of them would be. Then the recurrence relationships as the strip grows in length by 1 are,

$$\begin{aligned} R_{N+1} &= r_N p + s_N p^2 + T_N(p + p^2 - p^3) \\ &= R_N p + S_N p^2 - T_N p^3 \qquad (6) \\ S_{N+1} &= r_N p^2 + s_N p + T_N(p + p^2 - p^3) \\ &= R_N p^2 + S_N p - T_N p^3 \qquad (7) \\ T_{N+1} &= (r_N + s_N)p^2 + T_N(3p^2 - 2p^3) \\ &= (R_N + S_N)p^2 + T_N(p^2 - 2p^3) \qquad (8) \end{aligned}$$

These equations have been checked to reproduce the first few values for R_N listed above. These equations hold for $N = 0$, provided we make the reasonable definitions,

$$R_0 = 1, \quad S_0 = p, \quad T_0 = p$$

We now define new variables $a_N = R_N - S_N$ and $b_N = R_N + S_N$. Then,

$$\begin{aligned} a_{N+1} &= a_N p(1-p) \qquad (9) \\ b_{N+1} &= b_N p(1+p) - 2T_N p^3 \qquad (10) \\ T_{N+1} &= b_N p^2 + T_N p^2(1-2p) \qquad (11) \end{aligned}$$

Solution of the first equation is trivial, with $a_N = (1-p)[p(1-p)]^N$. The other equations can be written in matrix form for $\mathbf{x} = (b, T)$:

$$\mathbf{x}_{N+1} = A\mathbf{x}_N \qquad (12)$$

where

$$A = p \begin{pmatrix} 1+p & -2p^2 \\ p & p(1-2p) \end{pmatrix} \qquad (13)$$

4.2.3 Solving the Coupled Equations

To solve the coupled equations, we write A in terms of its eigendecomposition, $A = pQ\Lambda Q^{-1}$. The characteristic equation for the eigenvalues is

$$\lambda^2 - \lambda(1 + 2p - 2p^2) + p - p^2 = 0 \qquad (14)$$

which leads to

$$\lambda_{0,1} = \frac{1}{2}\left(1 + 2p(1-p) \pm \sqrt{1 + [2p(1-p)]^2}\right) \tag{15}$$

The eigenvalues $p\lambda$ are always distinct and in the range (0,1) except in the trivial cases $p = 0, 1$. Conventionally we take $\lambda_0 > \lambda_1$ in the following. λ_0 is greater than 1, while λ_1 is always close to zero; indeed, $\lambda_0/\lambda_1 \geq (7 + 3\sqrt{5})/2$. Note also that $\lambda_0\lambda_1 = p(1-p)$. Q can then be constructed from the corresponding eigenvectors of A, finally to give

$$A = \frac{1}{2p^2(\lambda_0 - \lambda_1)} \begin{pmatrix} 2p^2 & 2p^2 \\ 1 + p - \lambda_0 & 1 + p - \lambda_1 \end{pmatrix} \begin{pmatrix} p\lambda_0 & 0 \\ 0 & p\lambda_1 \end{pmatrix} \begin{pmatrix} 1 + p - \lambda_1 & -2p^2 \\ -(1 + p - \lambda_0) & 2p^2 \end{pmatrix} \tag{16}$$

The coupled equations can now be solved explicitly, in the form

$$\begin{pmatrix} b_L \\ T_L \end{pmatrix} = \frac{1}{2p^2(\lambda_0 - \lambda_1)} \begin{pmatrix} 2p^2 & 2p^2 \\ 1 + p - \lambda_0 & 1 + p - \lambda_1 \end{pmatrix} \begin{pmatrix} p\lambda_0 & 0 \\ 0 & p\lambda_1 \end{pmatrix}^L \begin{pmatrix} 1 + p - \lambda_1 & -2p^2 \\ -(1 + p - \lambda_0) & 2p^2 \end{pmatrix} \begin{pmatrix} 1 + p \\ p \end{pmatrix} \tag{17}$$

which in turn gives explicit equations for R_N, S_N. Unfortunately, these expressions do not simplify in any particularly attractive way.

4.2.4 Growth of R_N

The full expression for R_N can be written as follows:

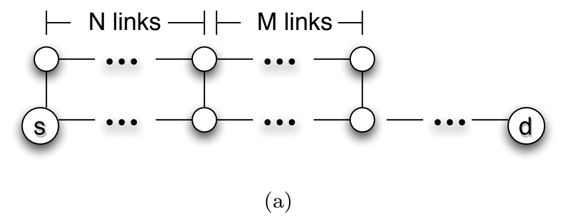
$$\begin{aligned} R_N &= \frac{1}{2}(C_0(p\lambda_0)^N + C_1(p\lambda_1)^N + (1-p)(\lambda_0\lambda_1)^N) \\ \lambda_{0,1} &= \frac{1}{2}\left(1 + 2p(1-p) \pm \sqrt{1 + [2p(1-p)]^2}\right) \\ C_0 &= \frac{(1+p)^2 - 2p^3 - \lambda_1(1+p)}{\lambda_0 - \lambda_1} \\ C_1 &= -\frac{(1+p)^2 - 2p^3 - \lambda_0(1+p)}{\lambda_0 - \lambda_1} \end{aligned} \tag{18}$$

Since $\lambda_1 < \lambda_0$ for all p, which can be seen by graphing them, then $C_1(p\lambda_1)^N$ is always smaller than $C_0(p\lambda_0)^N$. Since $\lambda_1 < p$ for all p, then $(1-p)(\lambda_0\lambda_1)^N$ is always smaller than $C_0(p\lambda_0)^N$. Consequently, the largest eigenvalue (λ_0) controls the reliability when N is large. The evolution of R_N with N indicates that as s and d get one hop further apart, the reliability of the strip decreases by a factor of $p\lambda_0 > p$, compared to a factor of p for the single path or a pair of disjoint paths. As $p \to 1$,

$$p\lambda_0 \quad \to \quad 1 - (1-p)^2 + O[(1-p)^3] \tag{19}$$

which is a much slower degradation of robustness. We return to this theme in a more general setting in Section 5.2.

We now use (18) to establish a subsidiary result about the growth of the $2 \times N$ node strip: that it is optimal to add nodes contiguously, regardless of how many are added at a time. We consider adding nodes to give a total of $N + M$ links 1 hop away from the shortest path, either in a single group of $N + M + 1$ nodes as in Figure 6(a), or separate groups of $N + 1$ and $M + 1$ as in Figure 6(b). Formally, we show that $R_{N+M-1} \geq R_N R_M$ for arbitrary N and M, despite the use of fewer nodes and links. Referring to (18), C_0 varies approximately linearly from 1 to 2, whereas C_1 is small and negative throughout. Recall that $R_0 = 1$ and so $C_0 + C_1 = 1 + p$. The expression for R_N is then in the appropriate form for the second inequality of



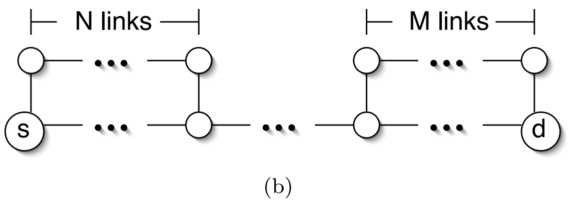


Figure 6: Adding nodes (a) contiguously or (b) not contiguously.

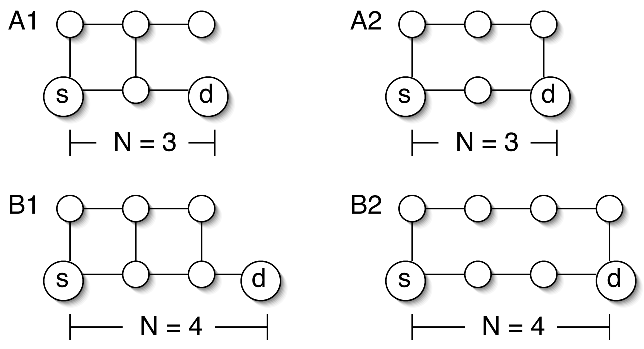


Figure 7: Counterexamples when adding links rather than nodes.

Appendix C provided we make the identifications

$$R_L \longrightarrow g(x)$$

$$\frac{C_0(p)}{2}, \frac{1-p}{2} \longrightarrow a_1, a_2 \qquad\qquad \frac{C_1(p)}{2} \longrightarrow b_1$$

$$p\lambda_0, \lambda_0\lambda_1 \longrightarrow \lambda_1, \lambda_2 \qquad\qquad p\lambda_1 \longrightarrow \mu_1$$

The condition $g(0) = 1$ follows directly; that $g''(0) > 0 \forall p$ can be seen by plotting it graphically. That it is optimal to add nodes contiguously then follows from the inequality of Appendix C.

5 Braid and Disjoint Path Comparison

A degenerate case of a 1-hop braid, where all internal links are missing, is a pair of disjoint paths which use neighbouring nodes. Does the optimal braid with a constraint on the number of links contain holes of this type? The answer depends on the measure of overhead and the value of p. Consider the examples in Figure 7 of a partial braid and a pair of disjoint paths. Graphs A_1 and A_2 both use six links total, however $R(A_1) = p^2 + p^4 - p^5$ while $R(A_2) = p^2 + p^4 - p^6$. Hence, graph A_2 is more reliable than graph A_1 for all values of p. Similarly, graphs B_1 and B_2 both use eight links total, however $R(B_1) = p^3 + 3p^5 - 2p^6 - 3p^7 + 2p^8$

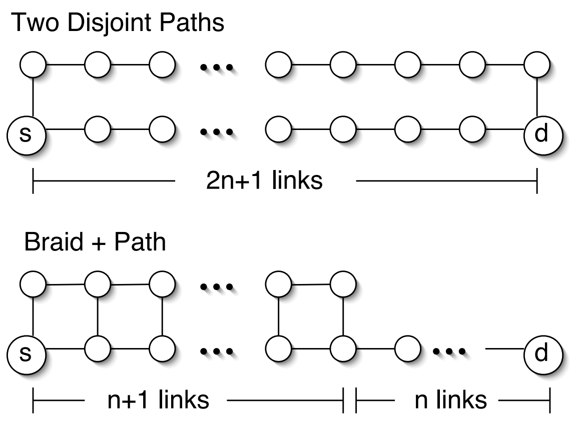


Figure 8: Two disjoint paths versus a 1-hop braid plus a path.

while $R(B_2) = p^3 + p^5 - p^8$. Now, however, graph B_2 is more reliable than graph B_1 when $p > \sqrt{2/3}$, i.e., the braid is only more reliable for low values of p. More generally, a pair of disjoint paths, containing $2n+1$ and $2n+3$ links respectively, see Figure 8, has reliability $P_{disjoint} = p^{2n+1}(1 + p^2 - p^{2n+3})$. In comparison, a braided path of length $n+1$ links, combined with a path of single links of length n, see Figure 8, has reliability $P_{braided} = p^n R_{n+2}$. The question is whether $P_{braided}$ is greater or less than $P_{disjoint}$. The answer depends on both n and p:

- $n = 0$: $P_{braided} = P_{disjoint}$ (the graphs are identical).
- $n = 1$: $P_{braided} - P_{disjoint} = p^5(1-p^2)(2-3p^2)$. The braided graph is more reliable if $p < \sqrt{2/3} \approx 0.816$.
- $n = 2$: $P_{braided} - P_{disjoint} = p^7(1-p)(5 + 2p - 5p^2 - 7p^3 + 3p^4)$. The braided graph is more reliable if $p <\approx 0.875$.
- $n = 3$: $P_{braided} - P_{disjoint} = p^9(1-p)(9 + 5p - 5p^2 - 19p^3 - 3p^4 + 19p^5 - 9p^6)$. The braided graph is more reliable if $p <\approx 0.902$.

5.1 Cross-over Point

The sequence suggests expanding looking for a cross-over value around $q \equiv 1 - p \approx 0$ for large n. For large N and small q, only the first term in expression (18) survives since the others are of order q^N. The other approximations are:

$$\begin{aligned}
\lambda_0 &= 1 + q + O(q^3) \\
\lambda_1 &= q - 2q^2 + O(q^3) \\
C_0 &= 2(1 - 2q^2) + O(q^3) \\
R_{n+1} &= (1 - 2q^2 + O(q^3))p^{n+1}\lambda_0^{n+1} \\
\frac{P_{braided} - P_{disjoint}}{p^{2n+1}} &= (1 - 2q^2 + O(q^3))(1 + q + O(q^3))^{n+1} \\
&\quad -(1 + (1-q)^2 - (1-q)^{2n+3})
\end{aligned} \tag{20}$$

We approximate the powers of $1 \pm q$ using the expression

$$(1 + a)^N = e^{aN}(1 + O(a)) \tag{21}$$

which is good for large N and $aN = O(1)$. We end up with

$$(1 + q - 2q^2 + O(q^3))(e^{q(n+1)} + O(q)) = 1 + (1 - q)^2 - (1 - q)(e^{-2q(n+1)} + O(q)) \tag{22}$$

or

$$e^{3(n+1)q} - 2e^{2(n+1)q} + 1 = (e^{(n+1)q} - 1)(e^{2(n+1)q} - e^{(n+1)q} - 1) = O(q) \tag{23}$$

Vanishing of the first term corresponds to the trivial case $q = 0$, while the second is a quadratic which can be solved to give

$$q \approx \frac{1}{n+1} \log\left(\frac{1+\sqrt{5}}{2}\right) \tag{24}$$

where correction terms will be $O(1/n^2)$. In other words, the braid is more reliable than the disjoint paths unless the reliability is within $O(1/n)$ of 1; this is despite the fact that that braid is "in series" with a sequence of n single links, each of which is a single point of failure. The reliabilities thus match at a critical link up probability, $P_{critical}$. As the number of nodes in the shortest path, $N = 2n+2$, increases, $1 - P_{critical}$ increases as follows,

$$1 - P_{critical} \quad \rightarrow \quad \frac{2}{N} \log\left(\frac{1+\sqrt{5}}{2}\right)$$

I.e., the regime in which the braid is more reliable becomes larger for larger networks. Note that this analysis assumes that "links used" is the appropriate overhead metric; an alternative metric is "nodes used," for which the appropriate comparison is between the disjoint paths and the full 1-hop braid, and the latter is always more reliable.

5.2 Scaling Behaviour

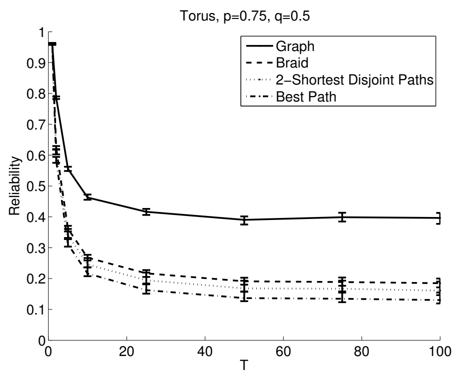
We extend this analysis to k-hop braids and k'-disjoint paths, studying how the reliability scales with increasing N by one hop, for large values of N. For disjoint paths, the reliability decreases by factor p for each additional hop regardless of the value of k' (which only affects a fixed overall coefficient). For the k-hop braid an exact expression is not available (except for $k = 1$ as given above); however, we can use the Provan-Ball equation (2) to find the leading terms in the reliability in the limit of small q. As discussed at the start of Section 4.1, these leading terms correspond to the minimum cuts in the network; as the source-destination distance increases by 1, there is one additional minimum cut of length $k + 1$. Thus the corresponding reliability decrease is just $1 - (1 - p)^{k+1}$ (compare this with Equation 19 for $k = 1$), which can be made arbitrarily small by increasing k. Even for moderate values of p, it is possible to grow to large networks without compromising robustness. Thus, in terms of reliability, braids will be most effective for large diameter networks with relatively unreliable links.

6 Braid Simulation Results

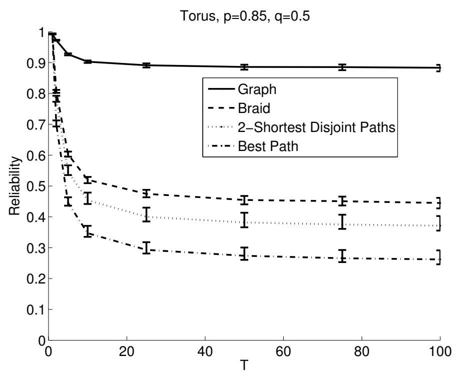
In this section we compare the reliability of a 1-hop braid with that of the shortest path, the two shortest disjoint paths, and the entire graph, using a time-varying network simulated in Matlab. We first describe the model and then present results.

6.1 Network model

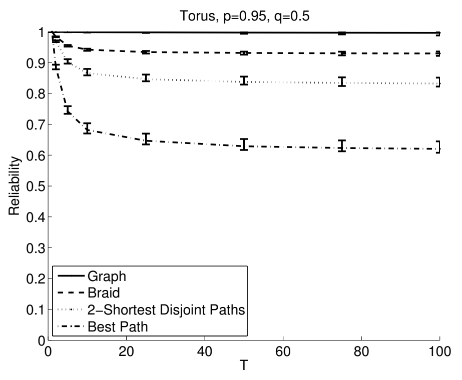
Consider a graph $G = (V, E)$ with nodes V and edges E. We examine (i) an $\sqrt{|V|} \times \sqrt{|V|}$ torus where $|E|$ comprises the set of all edges in the torus and (ii) a random model, where $|V|$ nodes are placed uniformly



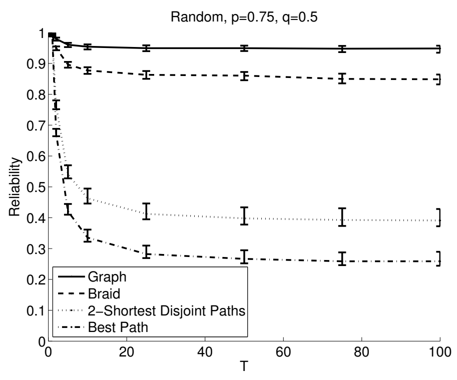
(a) Torus



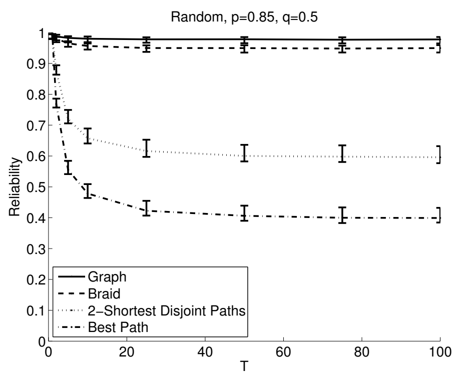
(b) Torus



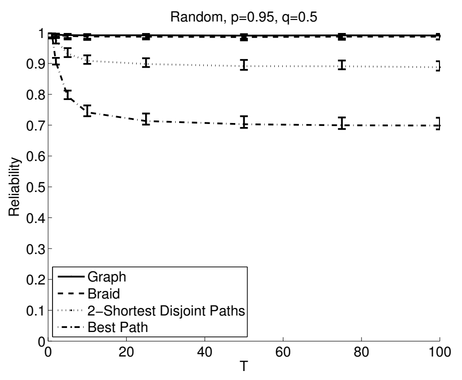
(c) Torus



(d) Random



(e) Random



(f) Random

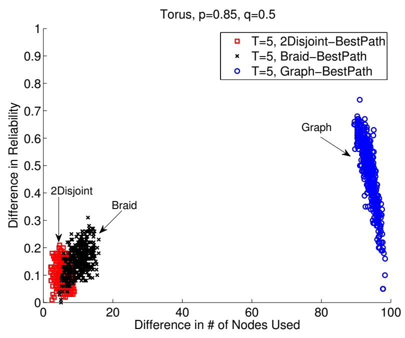
Figure 9: Reliability of different routing subgraphs. Reliability is averaged over 500 runs of 100 time-steps. 95% confidence intervals over the runs are shown. As not all sets of samples were normally distributed, bootstrap confidence intervals were computed using Matlab (hence the error bars are not symmetric).

randomly and independently in the plane, and edges exist between those nodes within a communication radius L of each other. We assume links are IID; to model link changes, we use a two-state Markov model where links stay up with probability p and stay down with probability q at each time-step. Unlike in the previous section, in this section we do not assume that $q = 1 - p$.

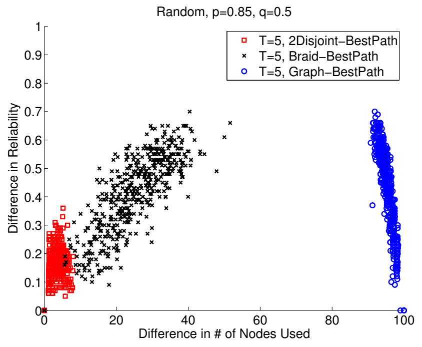
In our experiments, we use (i) a 10×10 torus and (ii) 100 nodes distributed randomly in an area of size 10×10 using a communication radius $L = 2$. We perform 500 simulation runs, each comprising 100 timesteps. In each run, a random source-destination pair is selected. For each time-step, we check whether each link is up. For the two-state Markov model we use $p = \{0.75, 0.85, 0.95\}$ and $q = 0.5$. We use the steady-state probability that a link is up to initially select which links are up or down. The routing sub-graph for each algorithm is recomputed every T timesteps, using only links that are up in the graph at the time of re-computation. All algorithms were evaluated on identical network topologies, and we estimate the reliability experimentally as discussed in Section 3.

6.2 Results

Figure 9 shows that for all p, that as the update interval T is increased, the reliability of the selected routing subgraph decreases and eventually reaches steady-state. For the torus, Figure 9(a) shows that for $p = 0.75$, the reliability of the 1-hop braid, 2-shortest-disjoint paths, and shortest path are all within a range of 0.1.



(a) Torus, T=5



(b) Random, T=5

Figure 10: Overhead of 1-hop braid vs. that of the shortest path, the two shortest disjoint paths, and the entire graph. Reliability was estimated experimentally.

Increasing p to 0.85 in Figure 9(b) shows a larger gap in reliability between the 1-hop braid and the 2-shortest disjoint paths, and also a larger gap between the braid and the full graph. Using $p = 0.95$ in Figure 9(c) shows an even larger gap in reliability between the 1-hop braid and the 2-shortest disjoint paths, but now a much smaller gap between the braid and the full graph. For the random model, Figures 9(d), (e), and (f) again show that for all p, the 1-hop braid has consistently higher reliability than the shortest path or 2-shortest disjoint paths, now as much as 0.4 greater than the 2-shortest disjoint paths when $p = 0.75$ or $p = 0.85$. This is in part a consequence of there not always being 2 disjoint paths in the graph (unlike in the torus). When $p = 0.95$, in Figure 9(f), the reliability achieved by the 1-hop braid is almost identical to that achieved by the full graph.

Figure 10 plots the reliability gain and number of additional nodes used over the shortest path by the 2-shortest disjoint paths, 1-hop braid, and full graph. Each point represents a simulation run (i.e., a selected source destination pair); for clarity we show only results for when $T = 5$. For the torus, Figure 10(a) indicates that the 1-hop braid provides an increase in reliability while using fewer than 20 extra nodes. For the random model, Figure 10(b) indicates that while the braid provides consistent and significant (up to about 0.4) gains in reliability, it also uses around 40 more nodes than the shortest path, but fewer than half the nodes used by the full graph.

In summary, the torus results indicate that the 1-hop braid can achieve reliability greater than that of the shortest path and the 2-shortest disjoint paths, and that the gains increase as p increases. We expect, however, that using a 2-hop braid would increase the reliability gain of the braid for small p. The results from the random model indicate that while using more nodes, the 1-hop braid can achieve reliability close to that of the full graph, and that the gain increases as p decreases.

7 Robust Routing Algorithm

In this section we outline our robust routing algorithm and evaluate it using GloMoSim [28].

7.1 Algorithm

Our robust routing algorithm is summarized in Table 3. The important features are as follows.

Recomputing routes periodically. We select a routing sub-graph that can be found efficiently, and that is

Table 3: Robust routing algorithm.

```
1   Let $G$ be graph of entire network
2   Loop every $T$:
3       Select "best" path $P$ from graph $G$
4       Build $k$-hop braid around $P$ to obtain graph $B$
5       Perform local forwarding on $B$
```

expected to perform "well" over the time period T during which it is not updated. Based on our analysis in the previous section, we choose this subgraph to be a k-hop braid.

Local forwarding within braid. Given the braid sub-graph B, rather than forwarding packets over a path, we consider all of B: i.e., we make local forwarding decisions to select the next hop out of all possible next hops within B. While the sub-graph B changes every T timesteps, local forwarding decisions are computed by nodes every timestep. A simple approach to perform local forwarding (which we use to obtain simulation results in the next section) is to have a node select its next hop based on which of its outgoing links have dropped packets; we describe this approach further in the next sub-section.

7.2 Simulation Results

In this section we compare the performance of the braided routing algorithm using a 1-hop braid, with that of Ad-hoc On-Demand Distance Vector (AODV) routing [21]. We first describe our implementation in GloMoSim [28], and then present experimental results.

7.3 Algorithm Implementation

AODV is used to construct the best path for the braid algorithm (but any other single path routing algorithm could be used). The 1-hop braid around this best path is then constructed as follows. When a node receives data to forward along the AODV path, it sends a braid request for the associated destination (if one has not yet been sent). When a node receives a braid request for a destination, it groups the request with other requests for that destination. If it finds it can hear at least two nodes on the path, it sends a braid reply to all nodes it can hear (except the node closest to the destination).

To tear the braid down, a braid node sends error messages to nodes it can hear on the AODV path when either one of its links to the AODV path breaks (i.e., drops a packet) and T has elapsed, or when it receives a more recent braid request for the destination (indicating that the current AODV path has been replaced). A node deletes its next hop braid for a destination when either (i) its next hop or later link on its AODV best path has dropped a packet for that destination, or (ii) a node for that destination is updated in its AODV routing table. A node marks a link as "bad" whenever the node attempts to use a link and has a packet dropped. The AODV path and/or braid will be recomputed only when T has elapsed. Whenever routes are recomputed, links are marked as "good."

To summarize, the braid overhead comprises, (1) braid requests by nodes on the AODV path to identify 1-hop neighbours, (2) braid replies by 1-hop neighbours to nodes on the AODV path, and (3) braid errors by 1-hop neighbours to nodes on the AODV path.

Nodes perform local forwarding within the braid as follows. Nodes on the AODV path select their AODV next hop with probability 1 if it is "good" or if there is no next hop braid node, and with probability 0.1 if it is "bad." If the AODV next hop was not selected, then the node iterates through its braid links. A braid link is selected with probability 1 if it is good or probability 0.1 if it is bad. If the node iterates through all of its braid links without selecting a next hop, then by default the AODV next hop is returned. If the node is a braid node, then it iterates through the nodes it can hear on the AODV path, selecting the AODV path node that is currently both closest to the destination and good. To ensure that bad links are also attempted, any AODV path node can be selected with probability 0.1. If the node iterates through all of its AODV path nodes without selecting a next hop, then by default the first AODV path node in its list is returned.

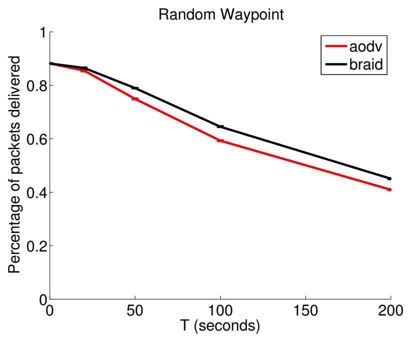
7.4 Simulation Setup

Our GloMoSim [28] simulation uses 60 nodes, moving according to the following mobility models. *(1) Random waypoint*: the pause time was 0 sec and node speeds were uniformly chosen between 4km/hr and 10km/hr. *(2) Gauss-Markov* [15]: average node speed was 7.2km/hr with standard deviation of 1.08km/hr and we use $\alpha = 0.2$ and $\Delta t = 100$. We use a 1.5km x 1.5km area for the random waypoint experiments and a 1km x 1km area for the Gauss-Markov experiments. Traces of node mobility were generated using BonnMotion [10] and fed into GloMoSim, letting us evaluate braid routing and AODV on identical mobility scenarios. We use a constant bit rate flow between two nodes for which data was generated every 0.5 sec and a total of 5 million packets were generated. We performed 10 simulation runs, each for the duration of the flow (about 29 simulated days). To address the problem of a long transient phase, the length of the flow was selected by examining the packet drop rate for progressively longer flows; when the change in % of packets dropped was sufficiently small ($< 0.05\%$), we assumed that steady-state had been reached. A better method would be to e.g., implement the "perfect simulation" method of Le Boudec and Vojnovic [1]; we leave this for future work. The MAC protocol used was 802.11 and the transmission radius was about 250 meters (from setting the radio transmit power to 7.9dBM).

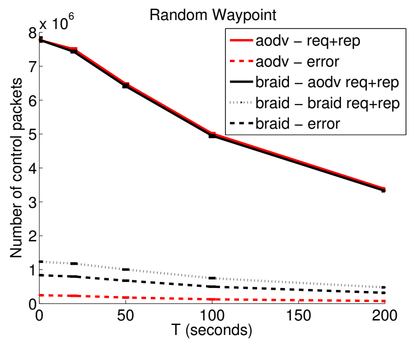
7.4.1 Results

Figure 11 compares AODV and braid routing with respect to throughput, overhead, and links used. Figures 11(a) and (d) show for both mobility models that the braid achieves a maximum of about 5% higher throughput than AODV for $T = 50, 100, 200$. Figures 11(b) and (e) show for both mobility models that the braid uses about the same amount of AODV overhead when building its best path as AODV (as measured by the number of path requests and replies transmitted by AODV); under Gauss-Markov mobility, however, this overhead is about 2.7×10^6 fewer packets, likely due in part to the smaller, 1km x 1km, area used. Figures 11(b) and (d) also show that while the braid incurs overhead from braid requests and replies, this overhead is about 1/4 of the AODV overhead under random waypoint, and about 1/2 of the AODV overhead under Gauss-Markov; the total braid overhead, however, for both mobility models is similar. Figures 11(b) and (d) also show that the total number of error packets transmitted for braid routing (aggregating error packets for both AODV and the braid) is perhaps five times greater than AODV error packets, in part because the braid involves more nodes in routing. As in Figure 11, other work, e.g., [6], has also observed that AODV can use as much (or more) control overhead as data transmitted, so we focus here on the additional overhead incurred by the braid. Since the braid construction is independent of the "best" path algorithm, another routing algorithm besides AODV could be used. Finally, Figures 11(c) and (f) show for both mobility models that the braid algorithm attempts to use more links than AODV (where "attempt" indicates that the routing algorithm attempted to transmit a packet over a link, but may not have been successful), in part because it may use a longer path. The braid, however, also has fewer links broken on average than does AODV.

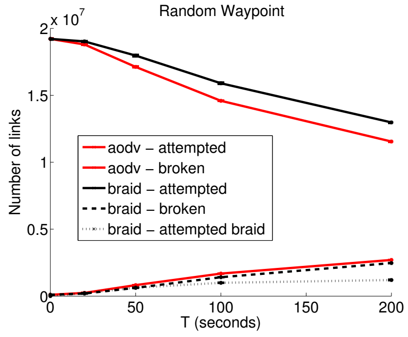
In summary, Figure 11 indicates that the 1-hop braid gains about 5% more throughput while using significantly less overhead than, for instance, would be needed to construct a second disjoint AODV path. The gains in throughput, however, are not as significant as the gains in reliability shown in the Matlab experiments in Section 6. We conjecture that this discrepancy is in part a consequence of (1) building the



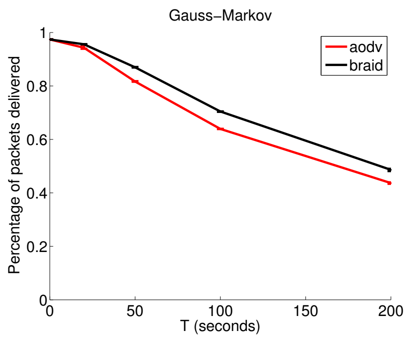
(a)



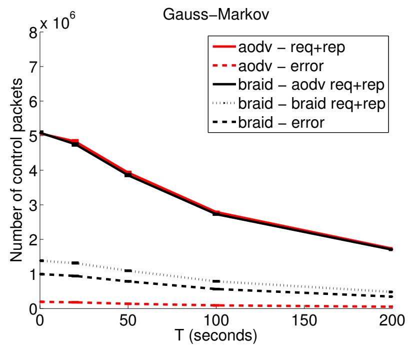
(b)



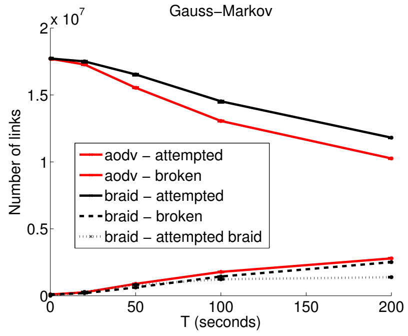
(c)



(d)



(e)



(f)

Figure 11: Comparison of 1-hop braid with AODV under (a)-(c) random waypoint and (d)-(f) Gauss-Markov mobility. 95% bootstrap confidence intervals over 10 simulation runs are shown.

braid around the shortest path rather than the most reliable path, and (2) the different network models, particularly in how they differ with respect to the rate at which links appear/disappear, and the temporal and spatial correlations among links changes. Note that since the braid construction is independent of the "best" path algorithm, a routing algorithm that identifies the most reliable path could be used rather than AODV.

Comparing the different network models, consider first the rate at which links appear/disappear. Results from [9] indicate that the inter-meeting times for two nodes using the random waypoint model are "well-approximated by an exponential distribution, at least for small to moderate transmission radii (with respect to the size of the area)." Using Lemma 1 in [9], we compute that for the transmission radius and random waypoint model considered here, the expected inter-meeting time for two nodes is given by $1/\lambda$ with $\lambda = 2.65$/hr. Hence, on average, two nodes will meet once every 22.7 minutes. Thus, in our random waypoint GloMoSim experiments, when a link breaks (due to mobility) it likely stays down for an interval significantly longer than T. Conversely, the probability of transitioning from down to up during T was 0.5 in the models used in the Matlab experiments in Section 6. Long inter-meeting times limit the throughput gains achieved by the braid since when braid links fail it is unlikely that they will re-appear before the remaining time in the interval T has elapsed.

Next consider correlations among links. In the Matlab experiments we assumed links failed independently. Conversely, we would expect that outgoing links of a given node would tend to have correlated failures when links break due to mobility. We would also expect that since all link failures are varying functions of how much time t of the interval T has elapsed, that link failures among different nodes would also be dependent due to the shared dependence on t. Correlated link failures limit the throughput gains achieved by the braid since if a link on the AODV path fails, it is also more likely that one of the links routing around the failed link will also fail soon (if it has not already).

8 Conclusions

This paper described a braided routing algorithm to improve the robustness of dynamic MANETs. We proposed a general method for braid construction and presented analytic optimality results in a restricted class of networks. We developed scaling laws for robustness as a function of path length and braid width. We validated the theoretical results through simulation, finding additional effects due to link failure correlations.

For future work we are interested in triggering route updates as a result of changes in end-to-end network performance, rather than using a fixed update interval. Similarly, rather than using a fixed braid width, we are interested in techniques to locally widen the braid to meet a robustness target. Addressing the issue of correlated links, we would like to explicitly consider correlations as well as more realistic radio link models. Finally, we would like to explore rate control mechanisms such as backpressure routing [26] for local forwarding to achieve a solution which is robust in throughput as well as connectivity.

Acknowledgments

The authors would like to thank Prithwish Basu, Majid Ghaderi, Matt Grossglauser, Andy Lam, Bruno Ribeiro, John Spicer, Patrick Thiran, Don Towsley, and Andy Twigg for their input. 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 under award number CNS-0519998 and 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. The U.S. and U.K. Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

References

[1] J.-Y. L. Boudec and M. Vojnovic. The random trip model: Stability, stationary regime, and perfect simulation. *IEEE/ACM Transactions on Networking*, pages 1153–1166, 2006.

[2] J. Burbank, P. Chimento, B. Haberman, and W. Kasch. Key challenges of military tactical networking and the elusive promise of MANET technology. *Communications Magazine, IEEE*, 44(11):39–45, 2006.

[3] K.-W. Chin. The behaviour of MANET routing protocols in realistic environments. In *IEEE 11th Asia-Pacific Conference on Communications*, 2005.

[4] K.-W. Chin, J. Judge, A. Williams, and R. Kermode. Implementation experience with MANET routing protocols. *Computer Communication Review*, 32(5):49–59, 2002.

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

[6] S. Das, C. Perkins, and E. Royer. Performance comparison of two on-demand routing protocols for ad hoc networks. In *Infocom*, 2000.

[7] D. Ganesan, R. Govindan, S. Shenker, and D. Estrin. Highly-resilient, energy-efficient multipath routing in wireless sensor networks. *Mobile Computing and Communications Review*, 4(5), 2001.

[8] J. Ghosh, H. Ngo, S. Yoon, and C. Qiao. On a routing problem within probabilistic graphs and its application to intermittently connected networks. In *Infocom*, 2007.

[9] R. Groenevelt, P. Nain, and G. Koole. The message delay in mobile ad hoc networks. Technical Report RR-5372, INRIA Sophia Antipolis, Nov. 2004.

[10] Institute of Computer Science IV, University of Bonn. BonnMotion: A mobility scenario generation and analysis tool. *http://web.informatik.uni-bonn.de/IV/Mitarbeiter/dewaal/BonnMotion/*, 2005.

[11] D. Karger. A randomized fully polynomial time approximation scheme for the all-terminal network reliability problem. *SIAM Review*, 43:3, 2001.

[12] M. Kodialam and T. V. Lakshman. Dynamic routing of restorable bandwidth-guaranteeed tunnels using aggregated network resource usage information. *Transactions on Networking*, 11:3, 2003.

[13] S. Lee and M. Gerla. Split multipath routing with maximally disjoint paths in ad hoc networks. In *IEEE ICC*, 2002.

[14] S.-J. Lee and M. Gerla. AODV-BR: Backup routing in ad hoc networks. In *IEEE Wireless Communications and Networking Conference*, 2000.

[15] B. Liang and Z. Haas. Predictive distance-based mobility management for multidimensional pcs networks. *Transactions on Networking*, 11(5), 2003.

[16] M. Marina and S. Das. On-demand multipath distance vector routing in ad hoc networks. In *IEEE ICNP*, pages 14–23, 2001.

[17] M. Mosko and J. J. Garcia-Luna-Aceves. Multipath routing in wireless mesh networks. In *IEEE Workshop on Wireless Mesh Networks*, 2005.

[18] M. Motiwala, N. Feamster, and S. Vempala. Path splicing: reliable connectivity with rapid recovery. In *ACM SIGCOMM HotNets VI*, 2007.

[19] N. Nicolaou, A. See, P. Xie, J.-H. Cui, and D. Maggiorini. Improving the robustness of location-based routing for underwater sensor networks. In *MTS/IEEE OCEANS conference*, 2007.

[20] P. Papadimitratos, Z. Haas, and E. G. Sirer. Path set selection in mobile ad hoc networks. In *MOBIHOC*, 2002.

[21] C. E. Perkins and E. M. Royer. Ad-hoc on-demand distance vector routing. In *IEEE Workshop on Mobile Computing Systems and Applications*, 1999.

[22] J. Provan and M. O. Ball. Computing network reliability in time polynomial in the number of cuts. *Operations Research*, 32:3, 1984.

[23] A. Reeve. Resilient real-time communications across meshed networks under adverse conditions. In *1st SEAS DTC Technical Conference*, 2005.

[24] N. Shacham, E. Craighill, and A. Poggio. Speech transport in packet-radio networks with mobile nodes. *IEEE Journal on Selected Areas in Communications*, SAC-1:6, 1983.

[25] X. Su, S. Chan, and K.-S. Chan. RLAR: Robust link availability routing protocol for mobile ad hoc networks. In *ICC*, 2007.

[26] 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*, 37:12, 1992.

[27] D. Tschopp, S. Diggavi, and M. Grossglauser. Hierarchical routing over dynamic wireless networks. In *Sigmetrics*, 2008.

[28] UCLA Computer Science Department Parallel Computing Laboratory and Wireless Adaptive Mobility Laboratory. GloMoSim: A scalable simulation environment for wireless and wired network systems. *http://pcl.cs.ucla.edu/projects/glomosim/*.

A The Provan-Ball Equation

The following material is summarised from Section 2.7 of Colbourn [5]. Define a minpath as a set of edges forming a path from the source to the destination, such that removing any edge on the path would disconnect the path. Define a mincut as a cut which separates the source and destination but for which any (strict) subset leaves the graph connected. The inclusion-exclusion method gives a way to calculate reliability by enumerating minpaths, and lends itself to examination of the regime $p \approx 0$, since only the shortest minpaths need be considered. The opposite extreme, $q = 1 - p \approx 0$, can likewise be analysed in terms of mincuts. The Provan-Ball equation, see Equations (25) and (26), is an expression for the (2-terminal) reliability of a network in terms of its mincuts. Since the Provan-Ball equation gives a polynomial time expression in terms of the mincuts, the method is economical if only the shortest mincuts are needed.

For any graph G, denote the set of mincuts as $\mathcal{C} = \{C_i\}$; each edge e in C_i has failure probability q_e. Then define for each C_i,

$$\begin{aligned} S(C_i) &= \{\text{nodes between the source node and } C_i\} \\ L(C_i) &= \{C_j : j \neq i, S(i) \subset S(i)\} \end{aligned}$$

Informally, $L(C_i)$ is the set of cuts that lie to the left of C_i, assuming that the source and destination are read left to right. While some of the C_j might intersect with C_i, there must be at least some nodes of G between the two, otherwise one cut would be a subset of the other, which is impossible, since they are mincuts. The technique partitions the set of failed states into corresponding (disjoint) events E_i for each C_i, whose probability is $P(E_i)$ computed as follows.

$$P(E_i) = \left[\prod_{e \in C_i} q_e\right] \left[1 - \sum_{C_j \in L(C_i)} \frac{P(E_j)}{\prod_{e \in C_i \cap C_j} q_e}\right] \tag{25}$$

where an empty product in the denominator has the value 1. The 'leftness' relationship between cuts given by $L(C_i)$ defines a partial ordering which arranges $\mathcal{C}$ into a directed acyclic graph. Each cut C_i is a vertex on this graph, with the cut through the links of the source node corresponding to the root vertex, and the cut through the links of the destination node corresponding to the sink. The $P(E_i)$ typically have to be calculated sequentially, starting with the root vertex and working forwards along this directed acyclic graph, since the sum in the second term is over all vertices upstream of the E_i being calculated. The probability of network failure is then just the sum of these $P(E_i)$, and the network reliability is

$$R(G) = 1 - \sum_{\mathcal{C}} P(E_i) \tag{26}$$

where together, Equations 25 and 26 define the Provan-Ball equation.

B The $3 \times N$ Node Strip

Can we extend the recurrence approach of Section 4 to the $3 \times N$ node strip? The number of different intermediate states increases compared to $2 \times N$ node strip, even when symmetries are taken into account. In terms of Figure 12 we define:

$$\begin{aligned} R_N^{(1)} &= P(s \text{ is connected to } d_1) \\ R_N^{(2a)} &= P(s \text{ is connected to } d_0) \\ R_N^{(2b)} &= P(s \text{ is connected to } d_0\text{, and } d_1 \text{ is connected to } d_2) \\ R_n^{(3)} &= P(s \text{ is connected to } d_0 \text{ and } d_1) \\ R_N^{(4)} &= P(s \text{ is connected to } d_0 \text{ and } d_2) \\ R_N^{(5)} &= P(s \text{ is connected to } d_0,\ d_1 \text{ and } d_2) \end{aligned}$$

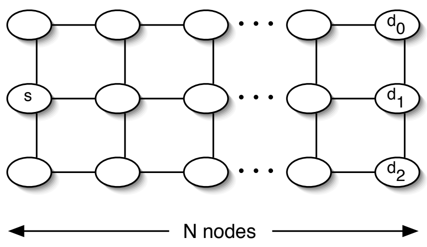


Figure 12: The $3 \times N$ node strip.

Again we also have to define quantities which are mutually exclusive, namely

$$\begin{aligned}
r_N^{(1)} &= P(s \text{ is connected only to } d_1) \\
r_N^{(2a)} &= P(s \text{ is connected only to } d_0, \text{ and } d_1 \text{ is not connected to } d_2) \\
r_N^{(2b)} &= P(s \text{ is connected only to } d_0, \text{ and } d_1 \text{ is connected to } d_2) \\
r_N^{(3)} &= P(s \text{ is connected only to } d_0 \text{ and } d_1) \\
r_N^{(4)} &= P(s \text{ is connected only to } d_0 \text{ and } d_2) \\
r_N^{(5)} &= P(s \text{ is connected to } d_0,\, d_1 \text{ and } d_2)
\end{aligned}$$

in terms of which we then have

$$\begin{aligned}
R_N^{(1)} &= r_N^{(1)} + 2r_N^{(3)} + r_N^{(5)} \\
R_N^{(2a)} &= r_N^{(2a)} + r_N^{(2b)} + r_N^{(3)} + r_N^{(4)} + r_N^{(5)} \\
R_N^{(2b)} &= r_N^{(2b)} + r_N^{(5)} \\
R_N^{(3)} &= r_N^{(3)} + r_N^{(5)} \\
R_N^{(4)} &= r_N^{(4)} + r_N^{(5)} \\
R_N^{(5)} &= r_N^{(5)}
\end{aligned}$$

We conjecture initial ($N = 0$) values for these as

$$\begin{array}{ll}
R_0^{(1)} = 1 & r_0^{(1)} = (1-p)^2 \\
R_0^{(2a)} = p & r_0^{(2a)} = 0 \\
R_0^{(2b)} = p^2 & r_0^{(2b)} = 0 \\
R_0^{(3)} = p & r_0^{(3)} = p(1-p) \\
R_0^{(4)} = p^2 & r_0^{(4)} = 0 \\
R_0^{(5)} = p^2 & r_0^{(5)} = p^2
\end{array}$$

We can now, with some effort, develop recurrence relationships for the $R_N^{(x)}$, (these have been checked for relating $R_1^{(x)}$ to $R_0^{(x)}$, and also for some cases of $R_2^{(x)}$):

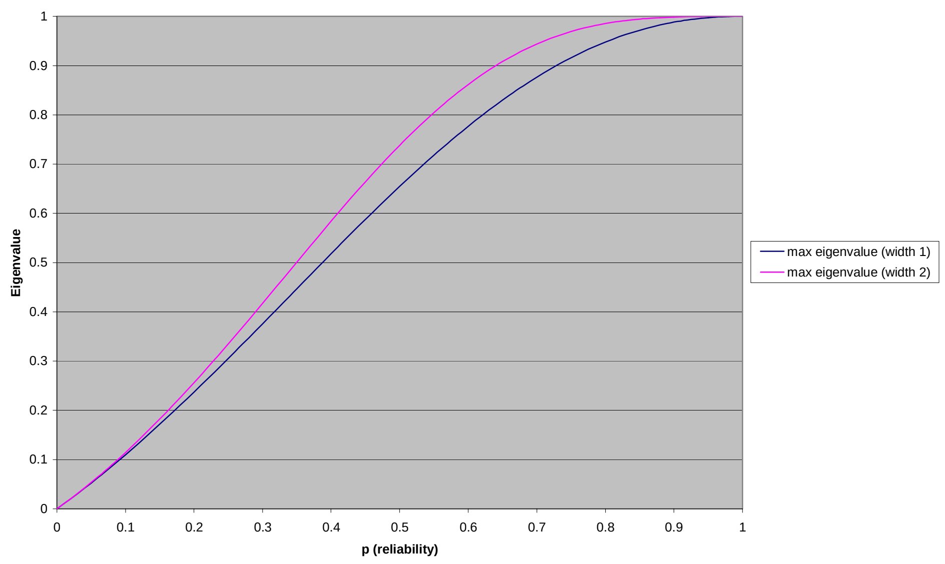


Figure 13:

$$\begin{aligned}
R_{L+1}^{(1)} &= pr_N^{(1)} + 2p^2(r_N^{(2a)} + r_N^{(2b)}) + 2(p + p^2 - p^3)r_N^{(3)} + (2p^2 - p^4)r_N^{(4)} + (p + 2p^2 - 2p^3 - p^4 + p^5)r_N^{(5)} \\
R_{L+1}^{(2a)} &= p^2r_N^{(1)} + (p + p^3)r_N^{(2a)} + (p + p^3 + p^4 - p^5)r_N^{(2b)} + (p + 2p^2 - p^4)r_N^{(3)} + (p + p^3 - p^4)r_N^{(4)} + (p + p^2 - 2p^4 + p^5)r_N^{(5)} \\
R_{L+1}^{(2b)} &= p^3r_N^{(1)} + (p^2 + p^3)r_N^{(2a)} + (p^2 + 2p^3 - p^5)r_N^{(2b)} + (p^2 + 4p^3 - 3p^4)r_N^{(3)} + (p^2 + 2p^3 - 2p^4)r_N^{(4)} + (p^2 + 5p^3 - 8p^4 + 3p^5)r_N^{(5)} \\
R_{L+1}^{(3)} &= p^2r_N^{(1)} + (p^2 + p^3)r_N^{(2a)} + (p^2 + p^3 + p^4 - p^5)r_N^{(2b)} + (4p^2 - p^3 - p^4)r_N^{(3)} + (p^2 + 2p^3 - 2p^4)r_N^{(4)} + (3p^2 - 4p^4 + 2p^5)r_N^{(5)} \\
R_{L+1}^{(4)} &= p^3r_N^{(1)} + 2p^3r_N^{(2a)} + 2(p^3 + p^4 - p^5)r_N^{(2b)} + 2(3p^3 - 2p^4)r_N^{(3)} + (p^2 + 2p^3 - 2p^4)r_N^{(4)} + (p^2 + 5p^3 - 8p^4 + 3p^5)r_N^{(5)} \\
R_{L+1}^{(5)} &= p^3r_N^{(1)} + 2p^3r_N^{(2a)} + 2(p^3 + p^4 - p^5)r_N^{(2b)} + 2(3p^3 - 2p^4)r_N^{(3)} + (4p^3 - 3p^4)r_N^{(4)} + (8p^3 - 11p^4 + 4p^5)r_N^{(5)}
\end{aligned}$$

which can be rearranged to give

$$\begin{aligned}
R_{L+1}^{(1)} &= pR_N^{(1)} + 2p^2R_N^{(2a)} - 2p^3R_N^{(3)} - p^4R_N^{(4)} + p^5R_N^{(5)} \\
R_{L+1}^{(2a)} &= p^2R_N^{(1)} + (p + p^3)R_N^{(2a)} + (p^4 - p^5)R_N^{(2b)} - (p^3 + p^4)R_N^{(3)} - p^4R_N^{(4)} + (2p^5 - p^4)R_N^{(5)} \\
R_{L+1}^{(2b)} &= p^3R_N^{(1)} + (p^2 + p^3)R_N^{(2a)} + (p^3 - p^5)R_N^{(2b)} + (p^3 - 3p^4)R_N^{(3)} + (p^3 - 2p^4)R_N^{(4)} + (4p^5 - 3p^4)R_N^{(5)} \\
R_{L+1}^{(3)} &= p^2R_N^{(1)} + (p^2 + p^3)R_N^{(2a)} + (p^4 - p^5)R_N^{(2b)} + (p^2 - 2p^3 - p^4)R_N^{(3)} + (p^3 - 2p^4)R_N^{(4)} + (3p^5 - 2p^4)R_N^{(5)} \\
R_{L+1}^{(4)} &= p^3R_N^{(1)} + 2p^3R_N^{(2a)} + 2(p^4 - p^5)R_N^{(2b)} + 2(p^3 - 2p^4)R_N^{(3)} + (p^2 - 2p^4)R_N^{(4)} + (5p^5 - 4p^4)R_N^{(5)} \\
R_{L+1}^{(5)} &= p^3R_N^{(1)} + 2p^3R_N^{(2a)} + 2(p^4 - p^5)R_N^{(2b)} + 2(p^3 - 2p^4)R_N^{(3)} + (2p^3 - 3p^4)R_N^{(4)} + (p^3 - 6p^4 + 6p^5)R_N^{(5)}
\end{aligned}$$

These equations appear analytically intractable - there are no obvious simplifications (e.g. eigenvectors visible by inspection). However, the eigenvalue spectrum can be extracted numerically. For some values of p there are only two real eigenvalues, with the others coming in conjugate pairs. We expect the behaviour of the reliability for large N will be dominated by the largest eigenvalue, and this is plotted in Figure 13, along with the analytic solution for the strip of width 1. Note that at $p = 1$ the matrix degenerates (all rows become equal), and this provides the leading eigenvalue $\lambda = 1$ and corresponding eigenvector (1,1,1,1,1,1). It might be possible to obtain the leading eigenvalue around $q \approx 0$ by perturbation; however, the Provan-Ball method seems more convincing (and much easier to generalize).

C Two Inequalities

The inequality needed for the R_L growth discussion is proved here.

First consider $f(x)$, where $f''(x) > 0$ (f is concave) and $f(0) \leq 0$.

$$\begin{aligned} f''(x) &> 0 && (27)\\ \Leftrightarrow xf''(x) + f'(x) &> f'(x) && (28)\\ \Leftrightarrow \frac{d}{dx}(xf'(x) - f(x)) &> 0 && (29) \end{aligned}$$

Since $f(0) \leq 0$, integrating we find that

$$xf'(x) - f(x) > 0 \Leftrightarrow \frac{xf'(x) - f(x)}{x^2} > 0 \Leftrightarrow \frac{d}{dx}\frac{f(x)}{x} > 0 \tag{30}$$

In other words, $f(x)/x$ is an increasing function. We therefore have

$$f(x+y) = (x+y)\frac{f(x+y)}{x+y} = x\frac{f(x+y)}{x+y} + y\frac{f(x+y)}{x+y} > x\frac{f(x)}{x} + y\frac{f(y)}{y} = f(x) + f(y) \tag{31}$$

Now consider $g(x)$, where

$$g(x) = \sum_{i=1}^{n} a_i \lambda_i^x \tag{32}$$

where $a_i > 0$ and $g(0) \leq 1$. We want to show $g(x+y) > g(x)g(y)$ for $x \geq 0$. First we show that $\log g(x)$ is concave:

$$\begin{aligned} \frac{d}{dx}\log g(x) &= \frac{\sum_{i=1}^{n} a_i\lambda_i^x \log\lambda_i}{g(x)} && (33)\\ \frac{d^2}{dx^2}\log g(x) &= \frac{\sum_{i=1}^{n} a_i\lambda_i^x \sum_{j=1}^{n} a_j\lambda_j^x \log^2\lambda_j - \left(\sum_{i=1}^{n} a_i\lambda_i^x \log\lambda_i\right)^2}{g(x)^2} && (34)\\ &= \frac{\sum_{i,j=1;i\neq j}^{n} a_i a_j \lambda_i^x\lambda_j^x(\log^2\lambda_i - \log\lambda_i\log\lambda_j)}{g(x)^2} && (35)\\ &= \frac{\sum_{i,j>1}^{n} a_i a_j \lambda_i^x\lambda_j^x \log^2\lambda_i/\lambda_j}{g(x)^2} && (36) \end{aligned}$$

All the terms in the summation are positive, and so $\log g(x)$ is concave with respect to x. Using the previous result with $f(x) = \log g(x)$ we get $g(x+y) > g(x)g(y)$ which is the result we wanted.

Unfortunately, the actual expression for R_L does not fit exactly this form, since one of the coefficients, $C_1(p)$, is negative. We keep definition (32) and additionally impose (without loss of generality) that $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$. Defining $h(x)$ by

$$\frac{d^2}{dx^2}\log g(x) = \frac{h(x)}{\alpha^{2x} g^2(x)} \tag{37}$$

(where for the moment α is a free variable) we are interested in the sign of

$$h(x) = \sum_{i,j>1}^{n} a_i a_j (\alpha\lambda_i)^x (\alpha\lambda_j)^x \log^2 \alpha\lambda_i/\alpha\lambda_j \tag{38}$$

for which we require $h(0) = g''(0)g^2(0) \geq 0$ and $h'(x) \geq 0$ for $x = 0$. But

$$h'(x) = \sum_{i,j>1}^{n} a_i a_j \log(\alpha\lambda_i\alpha\lambda_j)(\alpha\lambda_i)^x (\alpha\lambda_j)^x \log^2 \alpha\lambda_i/\alpha\lambda_j \tag{39}$$

so to show that $h'(x) \geq 0$ it is sufficient (though, maybe, not necessary). to show

$$\forall i : (\alpha\lambda_i)^{a_i} \geq 0 \tag{40}$$

and so

$$\begin{array}{ll} \alpha < 1/\lambda_i & a_i < 0 \\ \alpha > 1/\lambda_i & a_i > 0 \end{array} \tag{41}$$

Given that the λ_i are in decreasing order of size, this is equivalent to the condition that all the negative a_i correspond to the smallest λ_i (since we can choose α freely). Putting it all together, we have two expressions for functions $g(x)$ which satisfy $g(x+y) \geq g(x)g(y)$:

$$g(x) = \sum_{i=1}^{n} a_i \lambda_i^x \quad \text{with} \quad a_i > 0,\ g(0) \leq 1 \tag{42}$$

$$g(x) = \sum_{i=1}^{n} a_i \lambda_i^x + \sum_{j=1}^{m} b_j \mu_j^x \quad \text{with} \quad a_i > 0,\ b_j < 0,\ \lambda_i > \mu_j,\ g(0) \leq 1,\ g''(0) \geq 0$$