

Resilient and Low Stretch Routing Through Embedding into Tree Metrics

Jie Gao and Dengpan Zhou

Department of Computer Science,
Stony Brook University,
Stony Brook, NY 11794, USA
{jgao, dpzhou}@cs.sunysb.edu

Abstract. Given a network, the simplest routing scheme is probably routing on a spanning tree. This method however does not provide good stretch — the route between two nodes can be much longer than their shortest distance, nor does it give good resilience — one node failure may disconnect quadratically many pairs. In this paper we use two trees to achieve both constant stretch and good resilience. Given a metric (e.g., as the shortest path metric of a given communication network), we build two *hierarchical well-separated trees* using randomization such that for any two nodes u, v , the shorter path of the two paths in the two respective trees gives a *constant* stretch of the metric distance of u, v , and the removal of any node only disconnect the routes between $O(1/n)$ fraction of all pairs. Both bounds are in expectation and hold true as long as the metric follows certain geometric growth rate (the number of nodes within distance r is a polynomial function of r), which holds for many realistic network settings such as wireless ad hoc networks and Internet backbone graphs. The algorithms have been implemented and tested on real data.

1 Introduction

This paper considers a fundamental problem of designing routing schemes that give low stretch and are resilient to node failures. We consider a metric (P, d) on n nodes (e.g., as the shortest path metric of a given network), in particular, metrics of bounded geometric growth as a popular family of metrics in the real world. The result we present in this paper is a routing structure, constructed in a distributed manner such that each node of P keeps routing information of size $O(\log n)$, the route discovered has constant stretch (e.g., a constant factor longer than the metric distance), and the routing structure is robust to node failures, where a single node failure will only disconnect $O(1/n)$ fraction of the routes between all possible pairs.

The technique we use in this paper is through embedding into tree metrics. Given a metric (P, d) , the simplest way to route is probably by taking a spanning tree to guide message routing. This has a number of benefits, as a tree metric is a much simpler metric with many special features. For example, between any two vertices in a tree, there is a unique simple path connecting them, and the unique path can be found in a local manner by first traversing up the tree towards the root, and traversing down the

tree at the lowest common ancestor. There is a simple labeling scheme such that one can use routing table of $O(\log n)$ bits at each node to support routing on a tree [3].

However, routing on a spanning tree of the metric has a number of problems, in particular, the *poor stretch* and *lack of resilience*. The path on a tree might be much longer than the metric distance. Take the shortest path metric of a cycle of n vertices, any spanning tree will separate some pair of vertices, adjacent on the cycle, by distance $n - 1$. That is, the distortion introduced by routing on a spanning tree is factor of $\Omega(n)$ of their true distance. A more serious problem of routing on a tree is due to the lack of robustness to node failures. If a node fails or decides not to cooperate and stops forwarding messages, the tree is broken into pieces and in the worst case quadratically many pairs have their paths disconnected.

In this paper we use embedding into tree metrics for efficient, scalable routing, but address the shortcomings regarding stretch and resilience. Instead of using one tree, we use simply two trees. The shorter one of the paths from two trees may have better stretch. Regarding node failures, if a node u fails and the path between two nodes x, y is disconnected as it goes through u , the path connecting x, y in the second tree hopefully does not contain u and still remains valid. We briefly elaborate our technical approach and then relate to prior work.

Our Results The tree embedding we use follows from the embedding of a general metric into tree metrics with low distortion. Given a metric (P, d) we embed it to a hierarchically well-separated tree (HST). The leaf nodes of the HST are 1-to-1 mapped to nodes in P and internal nodes of the HST are also mapped to nodes of P although certain nodes may appear multiple times. The embedding of (P, d) into a tree metric necessarily introduces distortions. As discussed earlier, using a fixed tree one cannot avoid the worst case distortion of $\Omega(n)$. But if one build a tree, chosen randomly from a family of tree metrics, the *expected* distortion can be bounded by $O(\log n)$. Thus using this tree for routing one immediately obtains $O(\log n)$ stretch routing with low routing overhead. Approximating a metric with probabilistic hierarchical well-separated trees was first proposed by Bartal [6, 5], with the motivation that many problems are easier to solve on a tree than on a general graph. Later, Fakcharoenphol *et al.* [11] improved the distortion to $O(\log n)$ for any n node metric and this is tight.

The results we prove in this paper are mainly in three pieces

- Using two HSTs, randomly constructed with independent seeds, we show that the stretch can be improved to a constant in expectation. That is, for any two nodes x, y , between the two paths in the two HSTs respectively, one of them is short and is at most a constant factor of the metric distance between x, y .
- Regarding the resilience of using one HST for routing, we show that for any node failure, the number of pairs with their routes on the HST disconnected is at most a fraction of $O(\log \Delta/n)$ of all pairs, where Δ is the aspect ratio of (P, d) , defined as the furthest pair distance versus the closest pair distance. When Δ is polynomial in n the bound is as small as $O(\log n/n)$ but in the worst case when the aspect ratio is exponential the bound can be bad.
- Using two HSTs we substantially improve the routing resilience. We build two HSTs with random, independent seeds. In the case of a node failure, we show that the number of pairs with their routes on both HSTs disconnected is at most a frac-

tion of $O(1/n)$ of all pairs, thus removing the factor of $O(\log \Delta)$ compared with the case of a single HST.

The results hold for metrics with ‘geometric growth’, that is, the number of nodes within distance r from any node grows as a polynomial function of r , not exponential (as in the case of a balanced binary tree). Such a family of metrics appears in many realistic settings, either due to physical constraints such as in wireless networks and VLSI layout networks, or due to geographical constraints such as in peer-to-peer overlay networks [22, 19, 21]. In the next section we introduce the rigorous definitions and elaborate the precise assumptions for each of the results.

Last remark that in the case that (P, d) is the shortest path metric of a given network G , there is a distributed algorithm [12] that constructs the HST with a total number of messages bounded by $O(n \log n)$. In addition, each node is given a label of size $O(\log n)$ such that one can route on an HST using only the node label information. Thus the entire scheme of using one or multiple HSTs for robust, low-stretch and efficient routing can be implemented in a completely de-centralized manner.

Prior Work. There is numerous prior work on routing. We only have the space to review some most relevant ones.

The traditional routing methods as used for the Internet are essentially shortest path routing. Essentially each node keeps a routing table of size $O(n)$ to save the next hop on the shortest path for each destination. This is equivalent to maintaining n shortest path trees, rooted on every node. From this perspective, our approach defines one or two global trees, rather than one tree per node. By doing so we can substantially reduce the size of the routing table from $O(n)$ to $O(\log n)$, while still keeping the routing stretch by a constant.

From a theoretical aspect, compact routing that minimizes the routing table size while achieving low stretch routing has been studied extensively [20]. There are two popular models in the literature, the *labeled routing model* (in which naming and routing schemes are jointly considered) [9, 10, 27] and *name-independent routing* (in which node IDs are independent of the routing schemes) [2, 16]. Generally speaking, the theoretical results in compact routing in a graph whose shortest path metric has a constant doubling dimension are able to obtain, with polylogarithmic routing table size, $1 + \varepsilon$ stretch routing in the labeled routing scheme (see [8] and many others in the reference therein), and constant stretch factor routing in the name-independent routing scheme [16, 1] (getting a stretch factor of $3 - \varepsilon$ will require linear routing table size [1]). The schemes here are all by centralized constructions and aim to get the best asymptotic bounds. Our focus of using tree embedding is to obtain practical routing solutions with theoretical guarantee. Further, the compact routing schemes above have no consideration of robustness to node failures.

Routing methods that can recover from node or link failures receive a lot of interests recently. There are many heuristic methods for Internet routing such as fast re-routing [25], Loop-free alternate (LFA) [4], O2 [7], DIV-R [23] and MARA [28]. But these methods have no theoretical guarantee. Path splicing [18] uses multiple shortest path trees with perturbed edge weights. When routing in one tree metric encounters a problem, the message is quickly routed on a different tree. Using a similar idea we can also use multiple HSTs to recover in-transit failures. The difference is that we do not

keep separate shortest path trees rooted at each node, but rather use two global trees. Thus our storage overhead is substantially better. Our simulation shows that we have roughly the same routing robustness, our stretch is a little higher but we substantially save on routing table size.

2 Preliminaries

Metrics With Geometric Growth. An important family of metrics is the metrics with ‘geometric growth’. There are several related definitions. Given a metric (P, τ) , let $B(p, r) = \{v \mid \tau(p, v) \leq r\}$ denote the radius r ball centered at p . In [15], a metric has bounded *expansion rate* (also called the KR-dimension, counting measure) k_1 if $|B(v, 2r)| \leq k_1 |B(v, r)|$ for a constant k_1 ; and in [14], a metric has bounded *doubling dimension* k_2 if $B(v, 2r)$ is contained in the union of at most k_2 balls with radius r for a constant k ; in [17, 13], a metric has upper bounded growth rate *growth rate* k_3 if for every $p \in V$ and every $r \geq 1$, $|B(p, r)| \leq \rho r^{k_3}$, for a constant ρ and k_3 . A few sensor network papers [24, 29] consider a model when the growth rate is both upper and lower bounded, i.e., $\rho^- r^{k_4} \leq |B(p, r)| \leq \rho^+ r^{k_4}$ for a constant k_4 , where $\rho^- \leq \rho^+$ are two constants. We denote the family of metrics with constant expansion rate, constant doubling dimension, constant upper bounded growth rate, and constant upper and lower bounded growth rate as $\mathcal{M}_{\text{expansion}}$, $\mathcal{M}_{\text{doubling}}$, $\mathcal{M}_{\text{growth}}^+$, $\mathcal{M}_{\text{growth}}$ respectively.

It is not hard to see that $\mathcal{M}_{\text{growth}} \subseteq \mathcal{M}_{\text{expansion}} \subseteq \mathcal{M}_{\text{doubling}} \subseteq \mathcal{M}_{\text{growth}}^+$. See [14, 13] for more discussions. In terms of the results in this paper the detailed definitions actually matter. In the following we will make it clear which definition is needed for each result.

Embedding into Tree Metrics. Given two metric spaces (X, d_X) and (Y, d_Y) , an injective mapping $f : X \rightarrow Y$ is called an *embedding* of X into Y . We can scale up Y to make the embedding to be *non-contractive*, i.e., for any $u \neq v \in X$: $d_Y(f(u), f(v)) \geq d_X(u, v)$. We say Y *dominates* X . The distortion of the pair u, v is $\text{dist}_f(u, v) = \frac{d_Y(f(u), f(v))}{d_X(u, v)}$. The distortion of the embedding f is $\text{dist}(f) = \max_{u, v \in X} \text{dist}_f(u, v)$.

Given a metric (P, d) , we embed it to a tree metric and use the tree metric to guide message routing. Ideally we want the route length to be close to the metric distance. As shown in the introduction, it is not possible to get distortion of $o(n)$ using a single tree. However, it is known that for any metric (P, d) , one can use randomization such that the *expected* distortion is only $O(\log n)$. Such a tree is a type of a *hierarchical well-separated tree* H , as defined below.

Definition 1 (α -HST [5]). A rooted weighted tree H is an α -HST if the weights of all edges between an internal node to its children are the same, all root-to-leaf paths have the same hop-distance, and the edge weights along any such path decrease by a factor of α as we go down the tree.

In this paper we focus on 2-HST. The leaves of T are the vertices in P , and the internal nodes are Steiner nodes. Fakcharoenphol, Rao and Talwar [11] have shown that for any metric (P, d) one can find a family of trees such that a randomly selected metric from the family has expected distortion of $O(\log n)$, which is also tight.

Review of The FRT Algorithm [11]. Without loss of generality, we assume that the smallest distance between any two vertices in P is 1 and the diameter of P is Δ . The aspect ratio is also Δ . Assume $2^{\delta-1} < \Delta \leq 2^\delta$. The FRT algorithm proceeds in a centralized manner by computing a hierarchical cut decomposition $D_0, D_1, \dots, D_\delta$.

Definition 2 (Cut decomposition). For a parameter r , an r -decomposition of a metric (P, d) is a partitioning of P into clusters, each centered at a vertex with radius r .

Definition 3 (Hierarchical cut decomposition). A hierarchical cut decomposition of (P, d) is a sequence of $\delta + 1$ nested cut decompositions $D_0, D_1, \dots, D_\delta$ such that

- $D_\delta = P$, i.e. the trivial partition that puts all vertices in a single cluster.
- D_i is a 2^i -cut decomposition, and a refinement of D_{i+1} . That is, each cluster in D_{i+1} is further partitioned into clusters with radius 2^i .

To find the hierarchical cut decomposition, one first chooses a random permutation $\pi : P \rightarrow \{1, 2, \dots, n\}$ of the nodes. We use $\pi(i)$ to denote the node with rank i in the permutation. We also fix a value β chosen uniformly at random from the interval $[1, 2]$. For each i , compute D_i from D_{i+1} as follows. First set β_i to be $2^{i-1}\beta$. Let S be a cluster in D_{i+1} . Each vertex $u \in S$ is assigned to the first (according to π) vertex v within distance β_i . We also say that u nominates v . Each child cluster of S in D_i then consists of the set of vertices in S assigned to the same center. We denote the center of a cluster C by $\text{center}(C)$. Note that all clusters in D_i have radius $2^{i-1} \leq 2^{i-1}\beta \leq 2^i$. Remark that a node can nominate a center outside of its current cluster in D_{i+1} and one node can be the center for multiple clusters.

An alternative view of the hierarchical cut decomposition is to define for each node u a δ -dimensional *signature vector* $S(u)$. The i -th element in the vector is the lowest rank node within distance $2^i\beta$. $S(u)_i = \arg \min_{v \in B(u, 2^i\beta)} \pi(v)$ where $B(p, r)$ is the collection of nodes within distance r from node p . A cluster at level i contains all the nodes with the same prefix $[1, i]$ of their signature vectors.

To turn the hierarchical cut decomposition to a 2-HST, the points of P are the leaf nodes of the HST and each internal node in the HST corresponds to a cluster of nodes in the hierarchical partitioning. The refined clusters in D_{i-1} of a cluster C in D_i are mapped to children of C . The root corresponds to D_0 . We can also use the center u of a cluster C as the representative node of C in the HST. Thus the root of the HST has $\pi(1)$ as its representative node. Denote by P_i the centers of the clusters in D_i . P_i is the set of node that are ‘nominated’ by others at level i .

The HST has $\delta + 1$ levels, at $0, 1, \dots, \delta$. The level i has a number of internal nodes in the HST corresponding to P_i . The edge weight connecting a cluster C in D_i to its children clusters in D_{i-1} is 2^i , i.e., greater than the radius of the cluster C . Clearly the HST metric dominates (V, d) , as one only relaxes the distances. For any two nodes u, v , suppose that they are first separated in different clusters in the decomposition D_i , i.e., their lowest common ancestor in the HST is at level $i + 1$. In this case we have their distance on the tree to be $d_H(u, v) = 2 \sum_{j=1}^i 2^j = 2^{i+2}$. Fakcharoenphol, Rao and Talwar [11] proved that $d_H(u, v) \leq O(\log n)d(u, v)$, in expectation over all random choices of β and π . A distributed implementation of the algorithm is available in [12].

3 Constant Distortion Routing Using Two HSTs

Starting from this section we examine the properties of routing using *two* HSTs.

Constant Distortion Embedding in Two HSTs For a given metric (P, d) with expansion rate k , we build two HSTs, H_1 and H_2 with independent, random seeds using the algorithm in [11]. For any two points u, v in P , we define the distance between them to be the minimum shortest path in the two trees. That is $d_H(u, v) = \min\{d_{H_1}(u, v), d_{H_2}(u, v)\}$.

Theorem 1. *For any metric (P, d) with expansion rate k and two HSTs H_1, H_2 , there is a constant c such that for any two nodes $u, v \in P$,*

$$E[d_H(u, v)] = E[\min\{d_{H_1}(u, v), d_{H_2}(u, v)\}] \leq c \cdot k^4 \cdot d(u, v).$$

For two nodes $u, v \in P$, denote their lowest common ancestor (LCA) in H_i by $\text{LCA}_i(u, v)$, for $i = 1, 2$. And denote $\text{LCA}(u, v) = \min\{\text{LCA}_i(u, v), i = 1, 2\}$. Thus $d_H(u, v) = 2^{i+2}$ if $\text{LCA}(u, v)$ is at $i+1$. Now we have $E[d_H(u, v)] = \sum_{i=0}^{\delta-1} \text{Prob}\{\text{LCA}(u, v) \text{ is at level } i+1\} \cdot 2^{i+2}$. With the following Lemma that bounds the probability that $\text{LCA}(u, v)$ is at $i+1$ (the proof is in the Appendix), we can prove the Theorem.

Lemma 1.

$$\text{Prob}\{\text{LCA}(u, v) \text{ is at level } i+1\} \leq \begin{cases} 0, & \text{if } 2^{i+2} < d(u, v); \\ 3k^4 \cdot d^2(u, v)/2^{2i-4}, & \text{if } 2^{i-2} \geq d(u, v). \end{cases}$$

Proof (Theorem 1). With the above lemma, we can prove Theorem 1 easily. Suppose j^* is the smallest i such that $2^{i+2} \geq d(u, v)$,

$$\begin{aligned} E[d_H(u, v)] &= \sum_{i=0}^{\delta} \text{Prob}\{\text{LCA}(u, v) \text{ is at level } i+1\} \cdot 2^{i+2} \\ &\leq \sum_{i=j^*+4}^{\delta} [3k^4 \cdot \frac{d^2(u, v)}{2^{2i-4}}] \cdot 2^{i+2} + \sum_{i=j^*}^{j^*+3} 2^{i+2} \\ &\leq 2^7 \cdot 3k^4 \cdot d^2(u, v)/2^{i^*} + 14d(u, v) \leq (96k^4 + 14) \cdot d(u, v). \end{aligned}$$

Routing with Two HSTs. To route a message from a source to a destination node, we check each set of labels to see which tree gives a lower LCA (lowest common ancestor). That tree will provide a path with only constant stretch. We remark that the storage requirement for each node is very low, in the order of $O(\log n)$.

4 Resilience to Node Failures Using Two HSTs

In this section we show that using two trees, instead of one, can improve the routing robustness substantially. For a pair of node u, v , if the path connecting them is disconnected on the first tree, it is still possible that there is a path between them on the second tree. Thus one can switch to the second tree for a backup route and recover from sudden, unforeseen failures instantaneously, akin to the path splicing idea [18].

Robustness of a single random HST. We first examine the properties of a single HST in terms of node failure. When a node u fails, any path on the HST that uses a cluster with u as the center is disconnected. We examine how many such pairs there are. The

worst case is that u is a center of a cluster near the root of the HST – this will leave big components and $\Omega(n^2)$ number of pairs disconnected. For example, if the node $\pi(1)$ fails. However, since the construction of the HST uses random permutations (assuming the adversary has no control over the choice of this random permutation, as in standard settings of randomized algorithms), a single node failure is unlikely to be near the root. The following theorem works for any metric (P, d) with constant doubling dimension.

Theorem 2. *Given a node u and an HST, the expected number of nodes within clusters with u as center is $O(\log \Delta)$, where Δ is the aspect ratio of the metric (P, d) with constant doubling dimension.*

Proof. Suppose a node x is within a cluster with u as the center, say this cluster is at level i . Then we know that $d(u, x) \leq \beta 2^i$ and u is the highest rank node in $B(x, \beta 2^i)$. Now, take $\ell_u(x)$ as the lowest level j such that $d(u, x) \leq \beta 2^j$. Clearly, $\ell_u(x) \leq i$. Thus $B(x, \beta 2^{\ell_u(x)}) \subseteq B(x, \beta 2^i)$. That is, u is the lowest rank node at level $\ell_u(x)$ as well. The probability for that to happen is $1/|B(x, \beta 2^{\ell_u(x)})|$. Thus the probability that x is inside a cluster with u as center is no greater than $1/|B(x, \beta 2^{\ell_u(x)})|$.

Now, the expected number of nodes within clusters with u as center, W , is,

$$\begin{aligned} W &= \sum_x \text{Prob}\{x \text{ is in a cluster centered at } u\} \leq \sum_x 1/|B(x, \beta 2^{\ell_u(x)})| \\ &= \sum_j \sum_{x \in B(u, \beta 2^j) \setminus B(u, \beta 2^{j-1})} 1/|B(x, \beta 2^j)| \leq \sum_j \sum_{x \in B(u, \beta 2^j)} 1/|B(x, \beta 2^j)|. \end{aligned}$$

Now, recall that the metric (P, d) has constant doubling dimension γ . Thus we can cover the point set $B(u, \beta 2^j)$ by balls of radius $\beta 2^{j-1}$, denoted as sets B_1, B_2, \dots, B_m , $m \leq 2^\gamma$. Since the points in B_j are within a ball with radius $\beta 2^{j-1}$, all the points within B_j are within distance $\beta 2^j$ of each other. That is, for a node $y \in B_i$, $B_i \subseteq B(y, \beta 2^j)$. Thus $|B_i| \leq |B(y, \beta 2^j)|$, where $y \in B_i$. Now we group the points of $B(u, \beta 2^j)$ first by the balls they belong to, and then take the summation over the balls.

$$\begin{aligned} W &\leq \sum_j \sum_{x \in B(u, \beta 2^j)} 1/|B(x, \beta 2^j)| = \sum_j \sum_i^m \sum_{x \in B_i} 1/|B(x, \beta 2^j)| \\ &\leq \sum_j \sum_i^m \sum_{x \in B_i} 1/|B_i| = \sum_j \sum_i^m |B_i| \cdot 1/|B_i| = \sum_j m \leq 2^\gamma \delta = O(\log \Delta). \end{aligned}$$

The above lemma shows that the total number of pairs disconnected if one random node is removed is bounded by $O(n \log \Delta)$.

Robustness of Two Random HSTs. We now examine the robustness property of using two random HSTs and bound the number of pairs ‘disconnected’ in both trees, i.e., their routes by using both HSTs go through u . For this case we assume that (P, d) has both constant upper and lower bounded growth ratio. By using two trees we reduce the expected number of disconnected pairs from $O(n \log \Delta)$ to $O(n)$.

Theorem 3. *The number of pairs of nodes disconnected in two HSTs, constructed using independent random permutations, is a fraction of $O(1/n)$ of all pairs, for a metric (P, d) with both constant upper and lower bounded growth ratio.*

Proof. Take a pair of nodes x, y , the paths connecting the two in both trees are disconnected if and only if in each of the tree, exactly one node is in a cluster with u as center and another one is not in any cluster with u as center. Denote by $P_u(x)$ the probability

that x is in a cluster with u as the center. $P_u(x) \leq 1/|B(x, \beta 2^{\ell_u(x)})|$. Thus the expected number of pairs of nodes disconnected after node u is removed is,

$$\begin{aligned} W_2 &= \sum_y \sum_x 4[P_u(x)]^2 [1 - P_u(y)]^2 \leq \sum_y \sum_x 4[1/|B(x, \beta 2^{\ell_u(x)})|]^2 \\ &= 4n \sum_j \sum_{x \in B(u, \beta 2^j) \setminus B(u, \beta 2^{j-1})} 1/|B(x, \beta 2^j)|^2 \\ &= 4n \sum_j (|B(u, \beta 2^j)| - |B(u, \beta 2^{j-1})|) / |B(x, \beta 2^j)|^2. \end{aligned}$$

If (P, d) has constant bounded growth ratio k , we know that $\rho^- \beta^k 2^{jk} \leq |B(x, \beta 2^j)| \leq \rho^+ \beta^k 2^{jk}$ for constants $\rho^- \leq \rho^+$. Thus

$$W_2 \leq 4n \sum_j [\rho^+ \beta^k 2^{jk}] / [\rho^- \beta^k 2^{jk}]^2 = 4n \sum_j \rho^+ / (\rho^-)^2 \cdot 1 / (\beta^k 2^{jk}) = O(n).$$

Robustness of Two HSTs With Reversed Rank. An alternative method to use two trees for robust routing is to construct the second tree to be as different as possible from the first tree. One idea is to build the second HST H_2 by using permutation π_2 , as the reverse of the permutation π_1 used in H_1 . As an immediate consequence of that, suppose x is in a cluster with u as the center in H_1 , then x can not be inside any cluster with u as center in H_2 . This is because the rank of x is greater than u in π_1 , and the rank of x must be smaller than the rank of u in π_2 . Thus x can never nominate u in H_2 . This says that the set of nodes ‘chopped off’ by the failure of u in H_1 will not be chopped off in H_2 , ensuring certain robustness of routing. We also evaluate this method by simulations and it performs no worse than the two random HSTs.

5 Simulations

This section evaluates our two HSTs mechanism in terms of path stretch and reliability against node or link failures. We run our simulation on two data sets. The first data set is a unit disk graph on a network of nodes deployed using perturbed grid model, a widely used model for wireless sensor networks. To be specific, the networks are generated by perturbing n nodes of the $\sqrt{n} \times \sqrt{n}$ grid in the $[0, 1]^2$ unit square, by 2D Gaussian noise of standard deviation $\frac{0.3}{\sqrt{n}}$, and connecting two resulting nodes if they are at most $\frac{2}{\sqrt{n}}$ apart. The average degree of the network generated in this way is about 5. The second data set is the Sprint backbone network topology inferred from Rocketfuel [26], which has 314 nodes and 972 edges.

We first study the routing stretch by using 1 HST and 2 HSTs respectively assuming no node or link failures. We also examine the number of pairs disconnected when using one HST and two HSTs respectively. For the two HSTs, we carry out simulations for both random HSTs and a pair of HSTs whose node rankings are in reverse of each other. Next, we compare our scheme with the path splicing approach [18] when link or node failures exist. We conducted two sets of experiments using two randomly constructed HSTs, and a pair of HSTs with reversed rank. For both methods, the next hop has a probability p to fail at each step. In case of a failure, we route a message using one HST or one splicing, and switch to another HST or splicing instance when we encounter a link or node failure on the next hop. The path stretch is computed only on the messages that reach the destination before TTL runs down to zero.

Summary of simulation results. Our observations from these experiments are:

- *Small path stretch.* Without failure, the path stretch from two HSTs improves significantly over a single HST (Figure 1 [left]). In case of failures, using two HSTs gives worse stretch compared with path splicing, but reducing the routing table size significantly (Figure 1 [right]).
- *Extremely good resilience.* The maximum number of disconnected pairs using one HST can be bad, roughly 85% but using two HSTs the number drops to below 10% (Figure 2). Combining 2 HSTs with path splicing, our routing performance (i.e., the delivery rate), is nearly as good as using $2n$ spanning trees in the path splicing method (Figure 3).

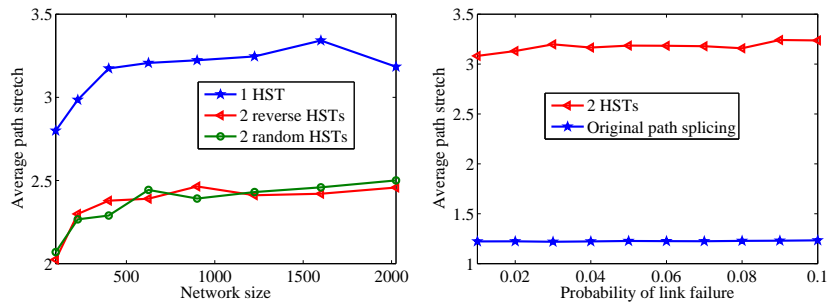


Fig. 1. [Left] Average path stretch using 2 HSTs v.s. 1 HST for the unit disk network (for each network size, we sample 20 different networks and take the average value); [Right] Path stretch using 2 HSTs v.s. path splicing on the Sprint topology, where each underlying link fails with probability p from 0.01 to 0.1.

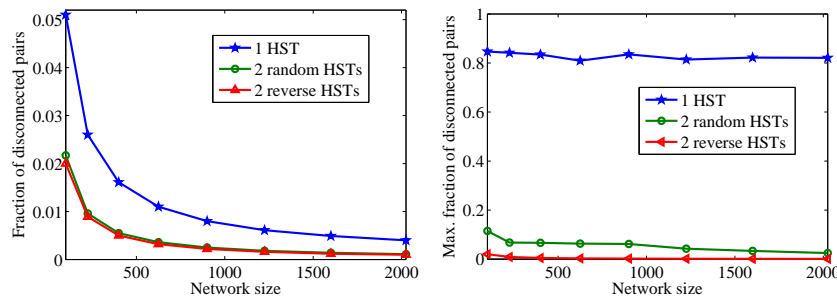


Fig. 2. By setting each node fail and removing all its adjacent edges from the network, we compute the fraction of disconnected pairs using 1 HST v.s. 2 HSTs. [Left]: average value. [Right]: maximum value. Results are the average for 20 unit disk networks for each network size.

References

1. I. Abraham, C. Gavoille, A. V. Goldberg, and D. Malkhi. Routing in networks with low doubling dimension. In *Proc. of the 26th International Conference on Distributed Computing Systems (ICDCS)*, July 2006.
2. I. Abraham and D. Malkhi. Name independent routing for growth bounded networks. In *SPAA '05: Proceedings of the seventeenth annual ACM symposium on Parallelism in algorithms and architectures*, pages 49–55, 2005.
3. S. Alstrup, C. Gavoille, H. Kaplan, and T. Rauhe. Nearest common ancestors: a survey and a new distributed algorithm. In *SPAA '02: Proceedings of the fourteenth annual ACM symposium on Parallel algorithms and architectures*, pages 258–264, 2002.

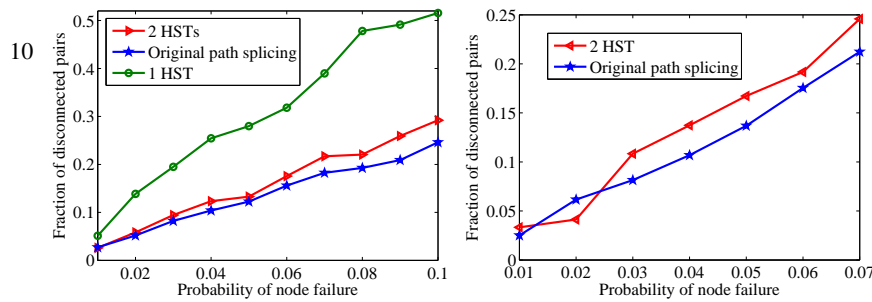


Fig. 3. The fraction of messages that are not delivered to the destination when next hop node fails with probability p . [left]: results on Sprint network. [Right]: average results on 50 unit disk graphs with 400 nodes. **What is the TTL used here?**

4. A. Atlas and A. Zinin. Basic specification for ip fast reroute: Loop-free alternates. September 2008.
5. Y. Bartal. Probabilistic approximation of metric spaces and its algorithmic applications. In *FOCS '96: Proceedings of the 37th Annual Symposium on Foundations of Computer Science*, page 184, Washington, DC, USA, 1996. IEEE Computer Society.
6. Y. Bartal. On approximating arbitrary metrics by tree metrics. In *STOC '98: Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 161–168, New York, NY, USA, 1998. ACM.
7. Y. G. C. Reichert and T. Magedanz. Two routing algorithms for failure protection in ip networks. In *Proc. ISCC*, 2005.
8. H. T.-H. Chan, A. Gupta, B. M. Maggs, and S. Zhou. On hierarchical routing in doubling metrics. In *SODA '05: Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 762–771, 2005.
9. L. J. Cowen. Compact routing with minimum stretch. In *SODA '99: Proceedings of the tenth annual ACM-SIAM symposium on Discrete algorithms*, pages 255–260, 1999.
10. T. Eilam, C. Gavoille, and D. Peleg. Compact routing schemes with low stretch factor. *J. Algorithms*, 46(2):97–114, 2003.
11. J. Fakcharoenphol, S. Rao, and K. Talwar. A tight bound on approximating arbitrary metrics by tree metrics. In *STOC '03: Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, pages 448–455, New York, NY, USA, 2003. ACM.
12. J. Gao, L. J. Guibas, N. Milosavljevic, and D. Zhou. Distributed resource management and matching in sensor networks. In *Proc. of the 8th International Symposium on Information Processing in Sensor Networks (IPSN'09)*, pages 97–108, April 2009.
13. J. Gao and L. Zhang. Tradeoffs between stretch factor and load balancing ratio in routing on growth restricted graphs. *IEEE Transactions on Parallel and Distributed Computing*, 20(2):171–179, February 2009.
14. A. Gupta, R. Krauthgamer, and J. R. Lee. Bounded geometries, fractals, and low-distortion embeddings. In *FOCS '03: Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science*, pages 534–543, 2003.
15. D. Karger and M. Ruhl. Find nearest neighbors in growth-restricted metrics. In *Proc. ACM Symposium on Theory of Computing*, pages 741–750, 2002.
16. G. Konjevod, A. W. Richa, and D. Xia. Optimal-stretch name-independent compact routing in doubling metrics. In *PODC '06: Proceedings of the twenty-fifth annual ACM symposium on Principles of distributed computing*, pages 198–207, 2006.
17. N. Linial, E. London, and Y. Rabinovich. The geometry of graphs and some of its algorithmic applications. *Combinatorica*, 15:215–245, 1995.
18. M. Motiwala, M. Elmore, N. Feamster, and S. Vempala. Path splicing. *SIGCOMM Comput. Commun. Rev.*, 38(4):27–38, 2008.
19. E. Ng and H. Zhang. Predicting Internet network distance with coordinates-based approaches. In *Proc. IEEE INFOCOM*, pages 170–179, 2002.

20. D. Peleg. *Distributed Computing: A Locality-Sensitive Approach*. SIAM Monographs on Discrete Mathematics and Applications, 2000.
21. C. G. Plaxton, R. Rajaraman, and A. W. Richa. Accessing nearby copies of replicated objects in a distributed environment. In *Proc. ACM Symposium on Parallel Algorithms and Architectures*, pages 311–320, 1997.
22. P. Raghavan and C. D. Thompson. Provably good routing in graphs: regular arrays. In *Proceedings of the 17th annual ACM Symposium on Theory of Computing*, pages 79–87, 1985.
23. K.-W. K. S. Ray, R. Guerin and R. Sofia. Always acyclic distributed path computation. *To appear in IEEE/ACM Transactions on Networking*, 2009.
24. R. Sarkar, X. Zhu, and J. Gao. Spatial distribution in routing table design for sensor networks. In *Proc. of the 28th Annual IEEE Conference on Computer Communications (INFOCOM'09), mini-conference*, April 2009.
25. M. Shand and S. Bryant. Ip fast reroute framework. June 2009.
26. N. Spring, R. Mahajan, D. Wetherall, and T. Anderson. Measuring isp topologies with rocketfuel. *IEEE/ACM Trans. Netw.*, 12(1):2–16, 2004.
27. M. Thorup and U. Zwick. Compact routing schemes. In *SPAA '01: Proceedings of the thirteenth annual ACM symposium on Parallel algorithms and architectures*, pages 1–10, 2001.
28. S. I. Y. Ohara and R. V. Meter. Mara: Maximum alternative routing algorithm. In *Proc. IEEE INFOCOM*, 2009.
29. D. Zhou and J. Gao. Maintaining approximate minimum steiner tree and k-center for mobile agents in a sensor network. In *Proc. of the 29th Annual IEEE Conference on Computer Communications (INFOCOM'10)*, March 2010.

6 Appendix

To prove Lemma 1, we first evaluate the probability that in one tree, say, H_1 , the probability that u, v have a lowest common ancestor at level j , $1 \leq j \leq \delta$.

Lemma 2.

$$\begin{aligned} & \text{Prob}\{\text{LCA}_1(u, v) \text{ is at level } i + 1\} \\ & \leq \begin{cases} 0, & \text{if } 2^{i+1} < d(u, v); \\ k^2 \cdot d(u, v)/2^{i-2}, & \text{if } 2^{i-2} \geq d(u, v). \end{cases} \end{aligned}$$

Proof. First, if $w = \text{LCA}_1(u, v)$ is at level $i + 1$, then $d(w, u) \leq \beta_{i-1} \leq 2^i$, $d(w, v) \leq \beta_{i-1} \leq 2^i$. By triangle inequality $d(u, v) \leq d(u, w) + d(w, v) \leq 2^{i+1}$. Thus in the first case of the lemma, the probability is 0. Suppose j^* is the smallest i such that $2^{i+2} \geq d(u, v)$. In the following we focus on the second case, i.e., $i \geq j^* + 4$.

If u, v belong to different clusters at level i , we say that the decomposition D_i separates u, v at level i . Thus $\text{LCA}_1(u, v)$ is at level $i + 1$ if and only if D_i separates u, v and $D_j(j > i)$ does not. Thus,

$$\text{Prob}\{\text{LCA}_1(u, v) \text{ is at level } i + 1\} \leq \text{Prob}\{D_i \text{ separates } (u, v)\}.$$

Take this level i such that D_i separates u, v . There is a node w such that one of u, v is first assigned to w and the other is not. We say that w settles the pair u, v at level i . Such a node w is unique, as once the pair u, v is settled it won't be settled again. Thus we will consider the union of the probability for each node w of P to possibly settle u, v . If w

settles u, v and u is assigned to w , we say w cuts u out. Summarizing the above, we have $\text{Prob}\{D_i \text{ separates } (u, v)\} = \sum_w \text{Prob}\{w \text{ settles } u, v\} = \sum_w \text{Prob}\{w \text{ cuts } u \text{ out}\} + \sum_w \text{Prob}\{w \text{ cuts } v \text{ out}\}$.

Let K_i^u be the set of nodes in P within distance 2^i to node u , and let $k_i^u = |K_i^u|$. We rank the node in K_i^u with increasing order of distance from u : $w_1, w_2, \dots, w_{k_i^u}$. For a node w_s to cut u out of the pair u, v at level i , it must satisfy the following conditions: (i) $d(u, w_s) \leq \beta_i$. (ii) $d(v, w_s) > \beta_i$. (iii) w_s settles u, v . Thus β_i must lie in $[d(u, w_s), d(v, w_s)]$. But we have $d(v, w_s) \leq d(v, u) + d(u, w_s)$ by triangle inequality. so the length of interval $[d(u, w_s), d(v, w_s)]$ is at most $d(u, v)$. Since we choose β_i uniformly from the range $[2^{i-1}, 2^i]$, the probability for β_i to fall into this interval is at most $d(u, v)/2^{i-1}$.

We also need to bound the probability that it is w_s that cut u out of the pair u, v , not others in K_i^u . First we note that the points that are very close to both u, v cannot possibly settle u, v . In fact, w_s must lie outside K_{i-2}^u for $i \geq j^* + 4$. Suppose otherwise, w_s is in K_{i-2}^u , and u is assigned to w_s , then v must be assigned to w_s too, by triangle inequality, $d(v, w_s) \leq d(v, u) + d(u, w_s) \leq 2^{i-2} + 2^{i-2} \leq 2^{i-1} \leq \beta_i$ (note that $i \geq j^* + 4$). Thus only those in $w_{k_{i-2}^u+1}, w_{k_{i-2}^u+2}, \dots, w_{k_i^u}$ can separate u, v in level i . Since we have a random permutation on the node rank, the probability for w_s to be the first center assigned to u is at most $1/s$. Then the probability that u is cut out of the pair (u, v) at level i is bounded by

$$\sum_{s=k_{i-2}^u+1}^{k_i^u} \frac{1}{s} \cdot \frac{d(u, v)}{2^{i-1}} = \frac{d(u, v)}{2^{i-1}} \cdot (H_{k_i^u} - H_{k_{i-2}^u}),$$

where $H(m)$ is the harmonic function.

For a metric with expansion ratio k , we have $k_i^u \leq k \cdot k_{i-1}^u \leq k^2 \cdot k_{i-2}^u$. Then

$$H_{k_i^u} - H_{k_{i-2}^u} = \sum_{s=k_{i-2}^u+1}^{k_i^u} \frac{1}{s} < \sum_{s=k_{i-2}^u+1}^{k_i^u} \frac{1}{k_{i-2}^u} = \frac{k_i^u}{k_{i-2}^u} - 1 \leq k^2.$$

Thus, we have $\text{Prob}\{D_i \text{ separates } (u, v)\} = d(u, v) \cdot \frac{k^2}{2^{i-2}}$, as required in the theorem.

Now we are ready to prove Lemma 1.

First, if $\text{LCA}(u, v)$ is at level $i + 1$, then at least in one tree the lowest common ancestor is at level $i + 1$, the probability of which is 0 if $d(u, v) < 2^{i+2}$, as shown in Lemma 2. In the following we focus on the second case when $2^{i-2} \geq d(u, v)$.

If $\text{LCA}(u, v)$ is at level $i + 1$, the first time (smallest level) that u, v belong to different clusters is i in one tree and is $j \geq i$ in another tree. Denote by $P_1(i)$ and $P_2(i)$ the probability that $\text{LCA}_1(u, v)$ and $\text{LCA}_2(u, v)$ are at level $i + 1$ respectively.

$$\begin{aligned} & \text{Prob}\{\text{LCA}(u, v) \text{ is at level } i + 1\} \\ &= P_1(i) \sum_{j=i+1}^{\delta} P_2(j) + P_2(i) \sum_{j=i+1}^{\delta} P_1(j) + P_1(i)P_2(i) \end{aligned}$$

By using Lemma 2. Now we have

$$\begin{aligned} & \text{Prob}\{\text{LCA}(u, v) \text{ is at level } i + 1\} \\ &\leq 2k^2 \frac{d(u, v)}{2^{i-2}} \sum_{j=i+1}^{\delta} [k^2 \frac{d(u, v)}{2^{j-2}}] + [k^2 \frac{d(u, v)}{2^{i-2}}] [k^2 \frac{d(u, v)}{2^{i-2}}] \\ &= 3k^4 \cdot d^2(u, v) / 2^{2i-4}. \end{aligned}$$