Name-Independent Compact Routing in Trees

Kofi A. Laing*

Abstract

Given an undirected graph with positive edge weights, define $N_q(v)$ for each node v to be the set of q nodes closest to v (including v itself and breaking ties by node ID). It is shown that the nodes of any tree can be colored with one color per node drawn from a set of q colors, so that, for each node v, each color appears exactly once in $N_q(v)$ (but for arbitrary graphs, verifying whether they can be similarly colored for constant q is NP-Complete). As an application of the first result, we present a generalized tradeoff scheme that, for any fixed constant k and any $2 \leq b \leq O(n^{1/k})$, uses space $O(n^{1/k} \log_b n \log n)$, headers of length $O(\log_b n \log n)$, makes intermediate routing decisions in $O(\log b)$ time, and achieves stretch $2^k - 1$. For fixed k, this realizes $O(\log n)$ sized headers when we choose $b = n^{1/k}$. A single-source compact routing tradeoff scheme achieves the a stretch of $2k - 1$ with the same space and header constraints. Our schemes use underlying optimal name-dependent tree routing algorithms from from Thorup and Zwick, and from Fraigniaud and Gavoille.

1 Introduction

We begin by introducing the compact routing problem after which we formulate a related graph multicoloring problem which may be used in solving the compact routing problem. An instance of the compact routing problem is an undirected graph $G = (V, E)$ with edge weights. The problem is to specify routing tables $Table_v[\,]$ at each node v of the graph, and an algorithm for routing between pairs of nodes with the help of these tables. Typically one attaches either a constant or writable header to a packet, and the routing algorithm determines the next edge on the route path, and possibly the next value of the header when headers are writable, as a function of the incoming header and the routing table in the current node.

One measure of the quality of the routing tables and algorithm is the *stretch* obtained, which is defined as the maximum over all pairs of source and destination, of the ratio of the route path length to the length of the shortest path. We seek to get a good tradeoff between the stretch obtained and the amount of space available in each node for storing the routing tables, as a function of the network size.

The name-independent model assumes that each node comes labeled in the input graph, and that the node names may not be reassigned by the routing algorithm designer to assist the algorithm. A packet arrives in the network with only the original given name for its destination. Any topology-dependent information required for routing to this destination must be looked up within the network itself. In contrast, the name-dependent model allows the routing algorithm designer to reassign names (let us call these routing *labels*) to the nodes to encode a limited amount of topological information about the graph, and thus improve the routing algorithm. In this case a packet arrives in the network with the routing label for its destination node, and how the source node knows this label is not specified.

The fixed-port model assumes that each node v in the graph has distinct names $\{1, \ldots, deg(v)\}$ for the edges incident on v, and that there is no relationship between the two names assigned to an edge by the two nodes at its ends. In contrast the designer-port model [6] assumes that the routing algorithm designer may relabel the ports based on the topology of the graph, and specify the routing algorithm in terms of the new port names.

*Computer Science Department, Tufts University, Medford, MA 02155. laing@eecs.tufts.edu

In the name-independent fixed-port model, Awerbuch et al.[2] obtained a compact routing scheme with stretch $O(k^2 3^k)$ for arbitrary graphs using space $\tilde{O}(n^{2/k})$ and $O(\log n)$ headers. This was subsequently improved to $O(2^k)$ for arbitrary graphs using $\tilde{O}(n^{2/k} \log D)$ space where D is the diameter of the network, and headers of size $O(\log n)$ by Peleg[9]. Another scheme by Awerbuch and Peleg [3] obtained a stretch of $O(k^2)$ in networks using $\tilde{O}(n^{2/k} \log D)$ space, and headers of size $O(\log n)$.

Recent work has mostly been in the name-dependent model, which does not address the problem of how a source node finds out the name-dependent label of a destination node. This includes a stretch 3 scheme by Cowen which requires $\tilde{O}(n^{2/3})$ space and uses $O(\log n)$ headers[4], and a stretch 5 scheme by Eilam et al.[5] which uses $\tilde{O}(n^{1/2})$ space. Tradeoff schemes for name-dependent routing were established by Thorup and Zwick[10] with stretch $2k-1$ and space $\tilde{O}(kn^{1/k})$ provided we perform handshaking – which refers to the exchange of information between two nodes prior to computing a bit string to be included in a packet header. Without handshaking, the stretch they obtain is $4k-5$.

A recent paper by Arias, Cowen, Laing, Rajaraman and Taka addressed the name-independent version of the compact routing problem in arbitrary graphs with significant improvements [1] – a stretch 3 scheme is obtained for single-source routing with $\tilde{O}(n^{1/2})$. A stretch of 5 is obtained for routing from any source in arbitrary graphs with $\tilde{O}(n^{1/2})$ space and $O(\log^2 n)$ headers. They show how to improve the header size to $O(\log n)$ at the expense of either increasing the stretch to 7 or increasing the space to $\tilde{O}(n^{2/3})$. They also present a tradeoff scheme that achieves a stretch of $1 + (k-1)(2^{k/2} - 2)$ for space $\tilde{O}(n^{2/k})$, and another with stretch $16k^2 + 4k$ which improves on the result of Awerbuch and Peleg [3] with space $\tilde{O}(n^{2/k} \log D)$, with the former being of practical interest for small values of k ($k \leq 8$) [1]. Both of these tradeoff schemes use $o(\log^2 n)$ sized headers.

Central to these compact routing algorithms of [1] are variants of the following coloring problem: Given an undirected n-vertex graph $G = (V, E)$ with positive edge weights, and with a multicoloring of the nodes from the color set $\{1, \ldots, q\}$, let $C_v \subset [q]$ denote a subset of colors assigned to node v. Denote by $N_q(u)$ the set of q nodes closest to u (including u and breaking ties by node id). We say $N_q(u)$ is *fully-colored* if for every color $\tau \in [q]$, there exists a node $v \in N_q(u)$ such that $\tau \in C_v$. Define $\Gamma(G, q)$ to be the minimum c such that there exists a multicoloring satisfying for all $v \in G$: (1) $N_q(v)$ is fully-colored and (2) $|C_v| < c$. A result for the general version of this coloring problem (see Lemma 3.1 in [1]) implies that, for general graphs G, $\Gamma(G, \sqrt{n}) = O(\log n)$. In this paper we show the perhaps surprising result that for any tree, $\Gamma(G, q) = 1$. The proof is purely graph-theoretic, and constructive. We also show a result that is interesting in its own right – that for any constant q, deciding whether or not $\Gamma(G, q) = 1$ for an arbitrary graph is NP-Complete.

As an application of the first result, we obtain improved compact routing tradeoff schemes for trees under the name-independent fixed-port model. Section 2 presents the result that $\Gamma(G, q) = 1$, which entails a new coloring algorithm for the implementation of distributed dictionaries of lookup information for translating name-independent (given) names to topology-dependent (routing) labels in trees. In Section 3 we show that that the problem of deciding whether $\Gamma(G, q) = 1$ for any *constant* q is NP-Complete (even when the edge weights are $O(1)$). Section 4 presents two compact routing schemes: a tree-based all-pairs scheme which uses $\tilde{O}(n^{1/k})$ space and $O(\log n)$ headers for fixed constant k, and obtains a stretch bound of $2^k - 1$, as well as another single-source scheme which achives a stretch of $2k - 1$. Compared with naively applying the *more general* exponential-stretch algorithm of Arias et al.[1] to trees, the first scheme yields an $O(\log n)$ factor improvement in the space requirement by avoiding the probabilistic randomized distribution of $O(\log n)$ layers of colors (and this could be significant for small memory and/or embedded devices), an $O(\log n)$ factor improvement in header size, and a $O(k)$ factor improvement in stretch.

2 Tree Neighborhood Coloring Result

We now prove that for every weighted tree T on n nodes and any $q \leq n$, $\Gamma(T, q) = 1$. That is,

Theorem 2.1 *Let $q \geq 1$ be a constant integer, and let $T = (V, E)$ be a tree with $n \geq q$ nodes and edge weights $w : E \longrightarrow \mathcal{R}^+$. There exists a mapping $c : V \longrightarrow [q]$ such that for every node $v \in V$, $N_q(v)$ is distinctly colored.*

First, the following informal discussion puts this result in context: it is trivial to see that linear arrays can be colored in such a way that *all connected subgraphs* of size q are distinctly colored (not just the neighborhoods based on distance and breaking ties lexicographically by node ids). This is by applying the colors cyclically from one end to the other, as shown in figure 1.

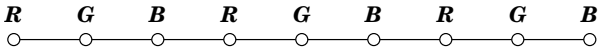

Figure 1: In this 9 node path three colors can be applied cyclically in the order $R, G, B, R, \ldots$ along the path, so that all connected subgraphs of size 3 are distinctly colored.

On the other hand, there is no way to color an n node tree such that *all connected subsets* of size q are distinctly colored (not unless $q = n$ when $n \geq 3$). This is easily seen by considering the star graph with $n - 1$ leaves.

It is therefore pleasantly surprising to find that by only considering neighborhoods defined in terms of distances and breaking ties based on node id, we *can* color a tree with one layer (one coloring function) using q colors such that every neighborhood of size q is distinctly colored – this is illustrated in figure 2.

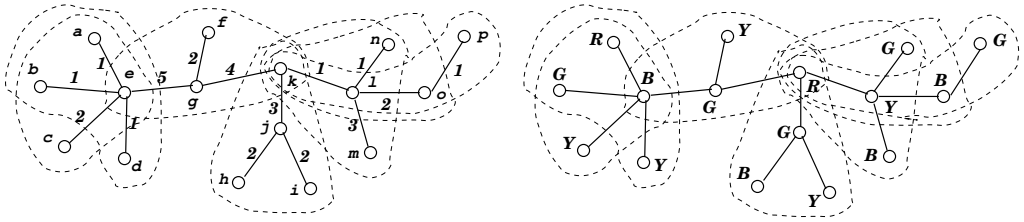

Figure 2: In the 16 node tree on the left, the node names (shown alphabetically) and edge weights are shown, with the resulting neighborhoods. The same tree is shown on the right, with the four colors $\{R, G, B, Y\}$ assigned so that every neighborhood contains the four distinct colors.

A final observation is that cycles whose length is a multiple of q can be colored (in the stronger sense of coloring all connected subgraphs of size q). However, as would be expected, not all planar graphs can be colored using a single color layer, as can be seen by considering odd-length cycles whose length is not a multiple of q, where q is odd, and where all edges have unit weights. Our proof of Theorem 2.1 uses the following lemma:

Lemma 2.2 (Awerbuch et al.[2]) *Let $G = (V, E)$ be a graph with positive edge weights, and let $q \geq 1$ be a constant integer. If $t \in N_q(u)$ and v is a node on a shortest path from u to t, then $t \in N_q(v)$.*

Proof of Theorem 2.1: By construction: Algorithm 2.3 below correctly computes a coloring such that each node v has a neighborhood $N_q(v)$ that is fully colored. The proof consists of three subclaims.

Algorithm 2.3 **ColorTree***(WeightedTree T, Num q):*
1: $B \leftarrow \{r\}$, *where r is an arbitrary node in T*
2: *color the nodes of $N_q(r)$ distinctly.*
3: **while**$(B \neq V)$
4: *choose an edge $\{u, v\}$ where $(u, v) \in B \times \overline{B}$*

5: *color* $N_q(v) \setminus N_q(u)$ *with colors in* $N_q(u) \setminus N_q(v)$
6: $B \leftarrow B \bigcup \{v\}$

Claim A: *Every time the test in line 3 is evaluated, the induced graph* $T[B]$ *is connected.*

This is clear from lines 1, 4 and 6.

Claim B: *Just before line 5, none of the nodes in* $N_q(v) \setminus N_q(u)$ *is colored.*

A sufficient condition is for all the colored nodes in $N_q(v)$ to be in $N_q(u)$. To show this, consider cases on the position of a *colored* node $s \in N_q(v)$. When u is on the shortest path from v to s, and Lemma 2.2 easily implies $s \in N_q(u)$. Otherwise, u is not on the shortest path from v to s, and there exists a node $u' \in B$ such that $s \in N_q(u')$, because the only way a node can be colored is by being in the neighborhood of a node in B. Now observe that if we deleted the edge $\{u, v\}$, u' would be in the same component as u, since $T[B]$ is connected (by claim A), and $v \notin B$. Also s would be in the other component because a shortest path from v to s does not go through u. It follows that the path from u' to s goes through the edge $\{u, v\}$. Therefore u is on the *shortest* path from u' to s and again, Lemma 2.2 implies that $s \in N_q(u)$.

Claim C: *Every time the test in line 3 is evaluated,* $N_q(v)$ *is fully-colored for every* v *in* B.

By induction. This is clearly true the first time line 3 is evaluated, because $B = \{r\}$ and $N_q(r)$ is fully-colored. Now suppose the claim holds after the first t evaluations (when $|B| = t$). Let $\{u, v\}$ be the t^{th} edge chosen in line 4. By the inductive hypothesis, $N_q(u)$ is fully-colored. Therefore none of the colors in $N_q(u) \setminus N_q(v)$ is in $N_q(u) \bigcap N_q(v)$. Moreover $|N_q(u) \setminus N_q(v)| = |N_q(v) \setminus N_q(u)|$, so after executing line 5, $N_q(v)$ will be fully-colored, since none of the nodes in $N_q(v) \setminus N_q(u)$ were colored (by Claim B).

Theorem 2.1 follows immediately from Claim C when Algorithm 2.3 terminates with $B = V$. □

3 Neighborhood Coloring is NP-Complete for Constant q

In this section we prove the following theorem:

Theorem 3.1 *Let* $q \geq 3$ *be a constant integer. and let* $G = (V, E)$ *be an* n*-node graph with positive edge weights. The problem of deciding whether* $\Gamma(G, q) = 1$ *is NP-Complete (even when edge weights are* $O(1)$*).*

Proof: The problem is in NP because given a coloring of the nodes with one color per node, we can compute all the neighborhoods and then check whether every neighborhood of size q contains an instance of every color, all in polynomial time.

We say an edge-coloring of a connected graph G' using colors drawn from a total order is *unique-minimum* if at every node, there is a unique edge with minimum weight. An algorithm to find a unique-minimum coloring is to compute a maximal matching, and assign all the edges in the matching a weight of w_1. Now every edge is incident on a node that is matched, or else the matching was not maximal. Consider any singleton nodes that are not yet matched, and arbitrarily associate each one to an incident edge, and label that edge $w_2 > w_1$. Finally label all remaining edges $w_3 > w_2$. This algorithm creates a unique-minimum edge-coloring using three colors in an arbitrary graph, and runs in polynomial time.

We show NP-hardness as follows. Let $G' = (V', E')$ be a connected instance of the problem q-COLORABILITY OF GRAPHS, which is well-known to be NP-Complete [7]. Determine a unique-minimum edge-coloring of G' using the preceding algorithm, using the weights $\{w_1, w_2, w_3\} = \{q^1, q^2, q^3\}$.

Construct G from G' by independently replacing every edge $\{u, v\}$ in G' of color q^i by the new extra nodes $x_1^{uv}, \ldots, x_{q-2}^{uv}$ and edges $\{u, x_1^{uv}\}$, $\{x_{q-2}^{uv}, v\}$, both of weight q^i, and edges $\{\{x_j^{uv}, x_{j+1}^{uv}\} \mid 1 \leq j < q-2\}$, all of weight 1, forming a path $(u, x_1^{uv}, x_2^{uv}, \ldots, x_{q-2}^{uv}, v)$ in G. No other edges contain the nodes x_i^{uv}.

We claim that for any $1 \leq j \leq q-2$, $N_q(x_j^{uv}) = \{u, x_1^{uv}, \ldots, x_{q-2}^{uv}, v\}$, because $d(x_j^{uv}, u) = q^i + (j-1) < q^i + q$ and $d(x_j^{uv}, v) = q^i + (q-j-2) < q^i + q$. The distance from x_j^{uv}) to any other node on the

chain is also less than $q^i + q$ because they are closer to x_j^{uv}) than u and v, and the distance from x_j^{uv} to any other node not on the chain is at least $q^i + q$. The chain $(u, x_1^{uv}, x_2^{uv}, \ldots, x_{q-2}^{uv}, v)$ is of size q, and accounts exactly for the desired neighborhood size, thus proving the claim.

Secondly we claim that for every node u, $N_q(u)$ is of the form $(u, x_1^{uv}, x_2^{uv}, \ldots, x_{q-2}^{uv}, v)$, where $\{u, v\} \in G'$ has the minimum edge weight among the edges incident on u. This is because for every x_i^{uv} on the chain, $d(u, x_i^{uv}) < q^i + q$ and $d(u, v) < 2q^i + q$. Moreover, by choice of v, the distance from u to any adjacent node other than x_1^{uv} is at least $q^{i+1} > 2q^i + q$ by the unique minimum property.

Thus, all distinct neighborhoods of size q in G are of the form $\{u, x_1^{uv}, x_2^{uv}, \ldots, x_{q-2}^{uv}, v\}$, where $\{u, v\} \in G'$ has the minimum edge weight among the edges incident on u. The distinct neighborhoods therefore correspond one-to-one to the edges in G'. Note that this reduction is planarity-preserving, can be performed in polynomial time and results in edge weights that are $O(1)$.

Finally we claim that a q-coloring of G' exists if and only if $\Gamma(G, q) = 1$. First, given a q-coloring $c' : V' \longrightarrow [q]$ of G', we color each node u in G with its color $c'(u)$ in G'. For each edge $\{u, v\}$ in G' we also color the nodes x_i^{uv} in G arbitrarily with the colors in $[q] \setminus \{c'(u), c'(v)\}$. Clearly the neighbhorhood $N_q(u)$ is fully-colored. Since this is independently true for each edge, every neighborhood in G is fully-colored and $\Gamma(G, q) = 1$. Conversely given a q-coloring $c : V \longrightarrow [q]$ of G such that every neighborhood in G is fully-colored, we color each node u in G' with its color $c(u)$ in G. Now consider any edge $\{u, v\}$ in G'. Since $\{u, v\} \subset N_q(x_1^{uv})$ and $N_q(x_1^{uv})$ is fully-colored, we know that $c(u) \neq c(v)$, so the resulting coloring is a good q-coloring on G'. □

Corollary 3.2 *Given a planar graph $G = (V, E)$ with positive edge weights, the problem of deciding whether $\Gamma(G, 3) = 1$ is NP-Complete.*

Proof: In this case we apply the same reduction from an instance of 3-COLORABILITY OF PLANAR GRAPHS WITH MAXIMUM DEGREE 6, (which is implicitly shown to be NP-complete — pages 85–89; [7]).

4 A Compact Routing Tradeoff Scheme for Trees

In this section we present two compact routing schemes for trees – an all-pairs scheme and also a single-source scheme. These schemes combine some ideas from [1] with our new coloring schemes. First we establish a common basis of notation and then develop the two schemes. We use the following result of Thorup and Zwick[10]. A similar result was obtained independently by Fraigniaud and Gavoille[6]:

Lemma 4.1 ([10, 6]) *Given a tree T with edge weights there exists a shortest-path name-dependent compact routing algorithm using $O(b \log n)$ space per node and $O(\log_b n \log n)$ headers.*

Let $Label(v)$ be the routing label assigned by the Algorithm of Lemma 4.1 to the node v in a tree T, and let $Table_v.ND[\,]$ denote the name-dependent routing table it stores at node v. We will define a distributed dictionary $Table_v.DD[\,]$ and the complete routing table $Table_v[\,]$ will consist of both these tables $Table_v.ND[\,]$ and $Table_v.DD[\,]$.

The routing algorithm is similar to that in [1] but is based on a much smaller distributed dictionary data structure, so there are differences in the details. For simplicity we assume that n is a k^{th} power and define the alphabet $\Sigma = \{0, \ldots, n^{1/k} - 1\}$. We also define the i^{th} neighborhood $N^i(v)$ of a node v as the set of $n^{i/k}$ nodes closest to v, breaking ties lexicographically by node name. Finally we define $\sigma_i(u)$ to mean the length i prefix of (u written in number base $n^{1/k}$, and padded on the left with zeros till it is of length k).

4.1 All Pairs Tradeoff Scheme

For each $i \in \{0, \ldots, k-1\}$ we color T with a coloring function $c_i : V(T) \longrightarrow \Sigma^i$ so that for every node v, every neighborhood $N^i(v)$ contains every word in Σ^i. We also define $c_k : V[T] \longrightarrow \Sigma^k$ by $c_k(u) = \langle u \rangle_{n^{1/k}}$, where $\langle u \rangle_{n^{1/k}}$ denotes writing u in number base $n^{1/k}$ and padding it on the left with zeros so that it is of length k. Note that for all nodes u, $c_0(u) = \varepsilon$.

Algorithm 4.2 **SetUpRoutingTableAt***(node u):*

1: *Store $Table_u.ND[\,]$ for name-dependent routing.*
2: **for** *every* $0 \leq i < k$
3: $\alpha \leftarrow c_i(u) \in \Sigma^i$; $\beta \leftarrow \sigma_i(u) \in \Sigma^i$
4: **for** *every pair* $(\lambda, \tau) \in \{\alpha, \beta\} \times \Sigma$
5: $v' \leftarrow$ *closest node to u in* $\{v \mid c_{i+1}(v) = \lambda\tau \text{ or } \sigma_{i+1}(v) = \lambda\tau\}$
6: $Table_u.DD[\lambda\tau] \leftarrow (v', Label(v'))$

Storage Analysis: $Table_u.ND[\,]$ requires $O(b \log n)$ space[10, 6]. In $Table_u.DD[\,]$, there are k values of i, and for each we provide extensions of two strings α and β. Each of these is extended by $n^{1/k}$ values of τ, and for each such extension we store a label $Label(v')$ of length $O(\log_b n \log n)$. This comes to a total of $O(b + kn^{1/k} \log_b n)$ words per node. The complete routing table $Table_u[\,]$ therefore requires only $\tilde{O}(n^{1/k})$ space when $2 \leq b \leq n^{1/k}$, for constant k. □

Algorithm 4.3 **Route***(source s, destination t):*

1: $v_0 \leftarrow s$
2: **for** *($i = 0$; $i < k$; $i++$):*
3: $(v_{i+1}, Label(v_{i+1})) \leftarrow Table_{v_i}.DD[\sigma_{i+1}(t)]$
4: **if** *($v_i \neq v_{i+1}$):*
5: *route optimally to v_{i+1} using $Label(v_{i+1})$ and $Table_q.ND[\,]$ for intermediate nodes q*

Lemma 4.4 *For each $0 \leq i < k$, $d(v_i, v_{i+1}) \leq 2^i d(s,t)$, and delivery to v_{i+1} is guaranteed.*

Proof: Proof is by induction. For basis case $i = 0$, $d(s, v_1) \leq 2^0 d(s,t)$ holds, because v_1 is chosen to be the closest node to s from the set

$$\{u \in N^1(s) \mid c_1(u) = \sigma_1(t)\} \bigcup \{u \in V[T] \mid \sigma_1(u) = \sigma_1(t)\}$$

The (unique) member of the first subset is in $N^1(s)$, and t itself satisfies the second condition. So whether $t \in N^1(s)$ or not, we have $d(s, v_1) \leq d(s,t)$, by definition of $N^1(s)$. We have stored $Label(v_1)$ at s, and the underlying optimal tree routing algorithm guarantees delivery to v_1 using the tables $Table_q.ND[\,]$ for intermediate nodes q.

Let us assume that the claim is true for $0 \leq i \leq r-1 < k-1$, and bound $d(v_r, v_{r+1})$ as follows: let v^*_{r+1} be the closest node to s, such that $c_{r+1}(u) = \sigma_{r+1}(t)$ or $\sigma_{r+1}(u) = \sigma_{r+1}(t)$

Since v_{r+1} and v^*_{r+1} both satisfy the condition $c_{r+1}(u) = \sigma_{r+1}(t)$ or $\sigma_{r+1}(u) = \sigma_{r+1}(t)$, we have $d(v_r, v_{r+1}) \leq d(v_r, v^*_{r+1}) \leq d(v_r, s) + d(s, v^*_{r+1})$. Now note that $d(s, v^*_{r+1}) \leq d(s,t)$, and we also have $d(v_r, s) = d(s, v_r) \leq \sum_{i=0}^{r-1} d(v_i, v_{i+1})$ since $d(s, v_r)$ is a shortest distance, and $\sum_{i=0}^{r-1} d(v_i, v_{i+1})$ sums up the distance of a walk between the endpoints s and v_r. So $d(v_r, v_{r+1}) \leq d(s,t) + \sum_{i=0}^{r-1} 2^i d(s,t)$ by the inductive hypothesis, and therefore $d(v_r, v_{r+1}) \leq 2^r d(s,t)$.

We know that delivery to v_{r+1} is assured because delivery to v_r is guaranteed by the inductive hypothesis, and the underlying name-dependent algorithm delivers a packet reliably from v_r to v_{r+1}. □

Theorem 4.5 *Algorithm 4.3 uses space $\tilde{O}(n^{1/k})$, uses $O(\log_b n \log n)$ headers, makes intermediate routing decisions in time $O(\log b)$ and has stretch $2^k - 1$.*

Proof: This follows immediately from Lemma 4.4, since the length $p(s,t)$ of the route obtained from s to t satisfies $p(s,t) \leq d(s,t) \sum_{i=0}^{k-1} 2^i = (2^k - 1)d(s,t)$, and delivery is guaranteed to t. The stretch of the algorithm is therefore $(2^k - 1)$. □

As recommended by [10], we could set $b = 2$ in Theorem 4.3 to optimize the speed of routing decisions. Alternatively we could exploit the $\tilde{O}(n^{1/k})$ space available and set $b = n^{1/k}$, to obtain $O(k \log n)$ headers in the fixed port model. This would worsen the space requirements of previous name-dependent tree routing algorithms which use asymptotically less space than $\tilde{O}(n^{1/k})$.

4.2 Single Source Scheme

In this section we present a compact routing scheme which obtains stretch $2k - 1$ for single source routing, given $\tilde{O}(n^{1/k})$ space and $O(\log n)$ headers for fixed constant k.

For each $i \in \{0, \ldots, k-1\}$ we color the nodes $N^i(r)$ with an arbitrary bijective coloring function $c_i : N^i(r) \longrightarrow \Sigma^i$. This ensures that the neighborhood $N^i(r)$ contains every word in Σ^i, with one word per node. We also define $c_k : N^k(r) \longrightarrow \Sigma^k$ by $c_k(u) = \langle u \rangle_{n^{1/k}}$, where $\langle u \rangle_{n^{1/k}}$ denotes writing u in number base $n^{1/k}$ and padding it on the left with zeros so that it is of length k. Recall that $N^k(r) = V[T]$.

Note that $c_0(r) = \varepsilon$ (here ε denotes the empty string). In the following algorithm the condition involving c_{i+1} should be considered to be false when $i = k - 1$, since c_k is undefined.

Algorithm 4.6 **SetUpRoutingTableAt**(*node u*):
1: *Store $Table_u.ND[\,]$ for name-dependent routing.*
2: *Let t be the the smallest i such that $u \in N^i(r)$.*
3: **for** *every $t \leq i < k$*
4: $\alpha \leftarrow c_i(r) \in \Sigma^i$; $\beta \leftarrow \sigma_i(r) \in \Sigma^i$
5: **for** *every pair $(\lambda, \tau) \in \{\alpha, \beta\} \times \Sigma$*
6: $v' \leftarrow$ *closest node to r in* $\{v \mid c_{i+1}(v) = \lambda\tau \text{ or } \sigma_{i+1}(v) = \lambda\tau\}$
7: $Table_u.DD[\lambda\tau] \leftarrow (v', Label(v'))$

Storage Analysis: $Table_u.ND[\,]$ requires $O(b \log n)$ space[10, 6]. In $Table_u.DD[\,]$, there are at most k values of i, and for each we provide extensions of two strings α and β. Each of these is extended by $n^{1/k}$ values of τ, and for each such extension we store a label $(v', Label(v'))$ of length $O(\log_b n \log n)$. This comes to a total of $O(b + kn^{1/k} \log_b n)$ words per node. The complete routing table $Table_u[\,]$ therefore requires only $\tilde{O}(n^{1/k})$ space when $2 \leq b \leq n^{1/k}$, for constant k. □

Algorithm 4.7 **RouteFromRootTo**(*destination t*):
1: $v_0 \leftarrow r$
2: **for** *($i = 0$; $i < k$; $i++$):*
3: $(v_{i+1}, Label(v_{i+1})) \leftarrow Table_{v_i}.DD[\sigma_{i+1}(t)]$
4: **if** *($v_i \neq v_{i+1}$):*
5: *route optimally to v_{i+1} using $Label(v_{i+1})$ and $Table_q.ND[\,]$ for intermediate nodes q*

Lemma 4.8 *The distance $d(r, v_1) \leq d(r,t)$, and for each $1 \leq i < k$, $d(v_i, v_{i+1}) \leq 2d(r,t)$, and for each $0 \leq i < k$, delivery to v_{i+1} is guaranteed.*

Proof: First we show that for all $0 \leq i \leq k$, $d(r, v_i) \leq d(r, t)$. This is because v_i is the closest node to r in

$$\{u \in N^i(r) \mid c_i(u) = \sigma_i(t)\} \bigcup \{u \in V(T) \mid \sigma_i(u) = \sigma_i(t)\},$$

and the (unique) member of the first subset is in $N^i(r)$, and t itself satisfies the second condition. So whether $t \in N^i(r)$ or not, we have $d(r, v_i) \leq d(r, t)$, by definition of $N^i(r)$.

For every $1 \leq i < k$, $d(v_i, v_{i+1}) \leq 2d(r, t)$ clearly holds, because $d(v_i, v_{i+1}) \leq d(v_i, r) + d(r, v_{i+1})$ by the triangle inequality and $d(v_i, r) = d(r, v_i)$ in undirected trees.

The algorithm has the property that each v_i (except when $i = k$) stores $Label(v_{i+1})$, and the underlying optimal tree routing algorithm guarantees delivery to v_{i+1} using the tables $Table_q.ND[\,]$ for intermediate nodes q. □

Theorem 4.9 *Algorithm 4.7 uses space $\tilde{O}(n^{1/k})$, uses $O(\log_b n \log n)$ headers, makes intermediate routing decisions in time $O(\log b)$ and has stretch $2k - 1$.*

Proof: This follows immediately from Lemma 4.8, since the length $p(r, t)$ of the route obtained from the root r to t satisfies $p(r, t) \leq (1 + \sum_{i=1}^{k-1} 2)d(r, t) = (2k - 1)d(r, t)$, and delivery is guaranteed to t. The stretch of the algorithm is therefore $(2k - 1)$. □

Again as recommended by [10], we have the option of setting $b = 2$ in Theorem 4.7 to optimize the speed of routing decisions, or using all of the $\tilde{O}(n^{1/k})$ space available by setting $b = n^{1/k}$, to obtain $O(k \log n)$ headers in the fixed port model.

5 Conclusions and Open Problems

Gavoille and Gengler have shown a lower bound of 3 for general graphs in the name-dependent model [8], which also applies to the name-independent model. Considering the n node star shows that this lower bound also applies to trees in the name-independent fixed-port model. In order to route optimally the hub (or center) node of the star needs to be able to encode all $n!$ permutations on n integers in its routing table. This requires at least $\log(n!)$ bits, which is easily shown to be $\omega(\sqrt{n} \log n)$, by Stirling's formula. Since suboptimal routing from the hub is necessary, any example of a suboptimal route results in a stretch of at least 3.

Since for any n we can choose k large enough so that $n^{1/k}$ becomes as small as a constant q, the result of section 3 indicates the hardness of applying the ideas of the compact routing scheme in subsection 4.1 in an arbitrary graph G using tight space requirements (assuming this general coloring approach, and using exactly $\Gamma(G, n^{1/k})$ blocks per node). Approximation algorithms are therefore required for finding the number of "layers of color" needed for a given graph and neighborhood size. A first solution would be the randomized coloring algorithm of [1] which easily gives a $O(\log n)$ approximation.

It would be interesting to know the best possible stretch attainable with $\tilde{O}(n^{1/k})$ space on trees (and also in general graphs) under the name-independent fixed-port model. Finally it would be great to characterize families of graphs with different values of $\Gamma(G, q)$.

References

[1] M. Arias, L. Cowen, K. Laing, R. Rajaraman, and O. Taka. Compact routing with name independence. In *Proc. 15th ACM Symposium on Parallel Algorithms and Architectures*, pages 184–192, 2003.

[2] B. Awerbuch, A. Bar-Noy, N. Linial, and D. Peleg. Compact distributed data structures for adaptive network routing. In *Proc. 21st ACM Symp. on Theory of Computing*, pages 479–489, May 1989.

[3] B. Awerbuch and D. Peleg. Routing with polynomial communication - space trade-off. *SIAM J. Disc. Math*, 5(2):151–162, 1992.

[4] L. Cowen. Compact routing with minimum stretch. *J. of Algorithms*, 38:170–183, 2001.

[5] T. Eilam, C. Gavoille, and D. Peleg. Compact routing schemes with low stretch factor. Technical Memo RR-1195-98, Laboratoire Bordelais de Recherche en Informatique, Jan. 1998. To appear in PODC98.

[6] P. Fraigniaud and C. Gavoille. Routing in trees. In 28^{th} *Int'l. Colloquium on Automata, Languages and Programming (ICALP)*, volume 2076 of LNCS, pages 757–772. Springer, 2001.

[7] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, New York, NY, 1979.

[8] C. Gavoille and M. Gengler. Space-efficiency of routing schemes of stretch factor three. *Journal of Parallel and Distributed Computing*, 61:679–687, 2001.

[9] D. Peleg. Distance-dependent distributed directories. *Information and Computation*, 103(2):270–298, 1993.

[10] M. Thorup and U. Zwick. Compact routing schemes. In *Proc. 13th ACM Symposium on Parallel Algorithms and Architectures*, pages 1–10. ACM, July 2001.