Laboratoire Bordelais de Recherche en Informatique
UMR 5800 - Université Bordeaux I, 351, cours de la Libération,
33405 Talence CEDEX, France

Research Report RR-1262-01

A Space Lower Bound for Routing in Trees

by Pierre Fraigniaud and Cyril Gavoille

September, 2001

A Space Lower Bound for Routing in Trees

Pierre Fraigniaud* Cyril Gavoille†

Abstract

The design of compact routing schemes in trees form the kernel of sophisticated strategies for compact routing in arbitrary graphs. This paper focuses on the space complexity for routing messages along shortest paths in trees. It was recently shown that the family of n-node trees supports routing schemes using addresses and routing tables of size $O(\log^2 n/\log\log n)$ bits per node, if the output port numbers of each node are chosen by an adversary. This paper shows that this result is tight, that is the sum of the sizes of the address and of the local routing table is at least $\Omega(\log^2 n/\log\log n)$ bits for some node of some tree.

Keywords: algorithms and data structures, computational and structural complexity, compact routing.

*CNRS, Laboratoire de Recherche en Informatique, Université Paris-Sud, 91405 Orsay cedex, France. *pierre@lri.fr*.

†Laboratoire Bordelais de Recherche en Informatique, Université Bordeaux I, 33405 Talence Cedex, France. *gavoille@labri.fr*

1 Introduction

As it is well known, a *routing scheme* consists of two parts:

1. a pre-processing algorithm in charge of, giving a network G, setting data structures (e.g., routing tables, node addresses, port labeling, etc.) on which the routing decisions in G will be based;

2. a routing protocol, i.e., a distributed algorithm, running at each node of G, which mainly determines, for every message entering the node, the output port on which the message has to be forwarded. Note that the routing protocol may also be in charge of other tasks such as updating the headers, etc.

For instance, a classical routing scheme is the so-called *routing table* which stores a look-up table at each node. This table returns, for each possible destination address of a message, the output port on which the message has to be forwarded. To be more precise, upon reception of a message M, a node x considers the *header* attached to M. The header contains the *address* of the destination, say node y. This address is denoted by address(y). Node x then extracts from its look-up table the *output port number* p corresponding to the entry address(y). Finally M is forwarded on the incident link of x labeled by p. We note that the header is not modified here, and the entire routing of M is determined by the information address(y) originally stored in the header of M by the source node. The pre-processing algorithm is in charge of constructing the look-up tables. Giving different addresses between 1 and n to the nodes of an n-node network, and labeling the incident links of each node x by different integers between 1 and $\deg(x)$, yields look-up tables of size $O(n \log n)$. The question is: can we do better? I.e., can we design routing schemes requiring less space at each node of the network? The answer is "yes", and the main tool to achieve that improvement is a combination of clustering and tree covers. In particular, we refer to the hierarchical routing schemes presented in [1, 2, 13], and to the more recent schemes presented in [3, 4, 14, 15]). Although the stretch factors (i.e., the maximum, over all pairs of nodes, of the ratio between the length of the route over the length of a shortest path between two nodes) resulting from these schemes might be not optimal, the improvement in term of memory space is often significant. All these schemes use as subroutine a shortest path routing schemes in some (partial) spanning trees of the network. This motivates the design of shortest path routing schemes for trees.

There are several complexity issues related to the design of routing schemes. Here we list three of them. (1) The size of the addresses and of the headers. For instance, routing tables use addresses and headers of $\lceil \log n \rceil$ bits[1], but other schemes may use longer addresses – e.g., the scheme in [13] uses addresses on $O(\log^2 n)$ bits, though headers on $O(\log n)$ bits only. (2) The size of the local data structure stored at each node. This size plus the size of the addresses is informally called *memory requirement*. And (3) The length of the routes, i.e., the stretch factor. In [14] it is proved that the previous results in the seminal

[1] All logarithms are in base two.

paper [13]. Namely, they show that any network support a routing scheme with stretch $4k-5$ using local data structures of size $\tilde{O}(kn^{1/k})$, and using addresses and headers of size $O(k\log^2 n/\log\log n)$, for every integer parameter $k \geqslant 2$. Using an *handshaking* technique (an agreement of $O(\log n)$ bits between the source and the destination) they even reduced the stretch to $2k-1$. Results and well established Erdös conjectures [5] from structural graph theory allow to believe that the space-time tradeoff is optimal. However, it is let open whether one could decrease the size of the addresses and headers.

As shown in [6, 15], these complexity issues strongly depend on whether or not we have the ability to set the output port numbers during the pre-processing phase. The *fixed-port* model specifies that the port numbers are fixed (say by an adversary), as opposed to the *designer-port* model which specifies that the port numbers can be fixed by the designer of the scheme. The routing schemes in [13, 15] are based on sub-routing schemes on trees in the fixed-port model. Indeed, every node uses several routing protocols and hence its incident links (i.e., ports) must be labeled once for all the protocols. In [6, 14], it is shown that, under the fixed-port model, there is a routing scheme using addresses and local data structures of size $O(\log^2 n/\log\log n)$ bits.

Our result

In this paper, we prove a tight $\Omega(\log^2 n/\log\log n)$ lower bound for the memory requirement of shortest path routing in trees under the fixed-port model. An important consequence of this result is that the routing schemes in [6, 14] are optimal as far as memory requirement is concerned. Another important consequence is that the sizes of the headers and addresses used in [13, 15] cannot be improved based on routing on spanning trees. Therefore, other techniques must be derived in order to decrease these sizes, say to $O(\log n)$, if possible. Our lower bound is obtained via sophisticated counting arguments presented in the forthcoming sections.

Previous works related to routing in trees

Several previous works addressed the routing problem in trees (results on compact routing in arbitrary networks have been recently surveyed in [8]). Under the fixed-port model, in [6] it is shown that every routing scheme that selects addresses in the range $[1,n]$, has an $n-o(n)$ bit local routing table for some n-node tree, whereas a scheme using $n+o(n)$ bits and $[1,n]$ range addresses exists [6]. Better performances are achieved under the designer-port model. In [7] it is constructed a scheme using $O(\sqrt{n})$ bit local routing tables and $[1,n]$ range addresses, and this is a tight bound according [4]. On the other hand, slightly augmenting the address size is worth since, for instance, under the fixed-port model, [3] describes a routing scheme that uses $3\log n$ bits for the addresses and $O(\sqrt{n}\log n)$ bits for each local data structure. This is also worth for the designer-port model, since n-node trees have a routing scheme with only $c\log n$ bits of memory requirement [6], for a small constant $c>1$. It is even proved in [15] that c can be reduced to $c=1+O(1/\log\log n)$. As said before, the significance of the fixed-port model has been shown in [13, 14], and a routing scheme for trees using addresses and local data structures of size $O(\log^2 n/\log\log n)$ bits has been independently presented in [6] and [14].

2 Forwarding Labeling Scheme

We are given arbitrary trees in which the edges incident to every node x are labeled by distinct integers in $\{1,\ldots,\deg(x)\}$. Note that an edge (x,y) receives two labels, one at x and one at y, and these two labels may be distinct. We often refer to these labels by the use of *port*, port i of node x referring to the incident edge of x labeled i (at x). In this section, unless specified otherwise, all trees and graphs are assumed to be port-labeled as described above.

Consider an arbitrary family $\mathcal{F}$ of graphs. In the fixed-port model, every routing scheme on $\mathcal{F}$ assigns to every graph $G \in \mathcal{F}$, and to every node x of G, two entities: (1) an address for x, denoted by $\text{address}(x)$; and (2) a data structure used for routing at x, denoted by $\text{data}(x)$. Moreover the scheme defines a routing protocol, i.e., an algorithm *send* (which usually depends on the family $\mathcal{F}$) that is in charge of routing. For every pair of distinct nodes x and y of G, $\mathit{send}(\text{data}(x), \text{address}(y))$ returns the label of an edge incident to x on a shortest path from x to y.

Hereafter, we denote by $\text{length}(\text{address}(x))$ and by $\text{length}(\text{data}(x))$ the size (in bits) of $\text{address}(x)$ and of $\text{data}(x)$ respectively.

Theorem 1 *In the fixed-port model, every shortest path routing scheme on n-node trees satisfies that there is a tree, and a node x of that tree, for which*

$$\text{length}(\text{address}(x)) + \text{length}(\text{data}(x)) \;=\; \Omega(\log^2 n/\log\log n).$$

The remaining of the paper is dedicated to proving Theorem 1. The proof is based on the following definition which belongs to the informative labeling theory (cf. [10, 11, 12] and [9] for an overview).

Definition 1 *Given a family $\mathcal{F}$ of graphs, a* forwarding labeling *on $\mathcal{F}$ is a pair (L,R) where R is called the routing function, and L is called the labeling function. For every $G \in \mathcal{F}$ and every node x of G, L assigns a non-empty binary string $L(x)$ to x, and, for any two distinct nodes x, y of G, $R(L(x), L(y))$ returns the label of an edge incident to x on a shortest path from x to y.*

If there exists a routing scheme with $\text{length}(\text{address}(x)) + \text{length}(\text{data}(x)) \leqslant \alpha$, then, we can derive a forwarding labeling with labels of length $\alpha + O(\log\alpha)$ as follows. Each label $L(x)$ is obtained by the concatenation of $\text{address}(x)$ and $\text{data}(x)$, preceded by a sequence $s(x)$ of $O(\log\alpha)$ 1's followed by one 0 describing on how many bits is coded $\text{address}(x)$. The routing function R can be computed from the routing protocol *send* by extracting the fields $\text{data}(x)$ and $\text{address}(y)$ from the labels $L(x)$ and $L(y)$, by the use of $s(x)$ and $s(y)$. Hence a lower bound on the size of the labels of any forwarding labeling yields a lower

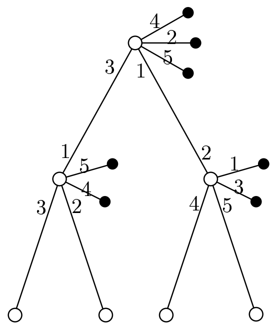

Figure 1: A port-labeled tree $T \in \mathcal{T}$ for $h = 2$, $d = 5$, and $t = 2$.

bound on length(address(x)) + length(data(x)) of any routing scheme, up to an additive factor bounded by the logarithm of this bound.

To compute a lower bound on the size of the labels in a forwarding labeling on trees, we proceed as follow. For any family $\mathcal{T}$ of trees, and for any forwarding labeling (L, R) on $\mathcal{T}$, we construct the set $\mathcal{L}$ consisting of all the labels assigned by L on all the trees of $\mathcal{T}$, i.e., $\mathcal{L} = \{L(x) \mid x \in T, T \in \mathcal{T}\}$. There exists some label $\ell \in \mathcal{L}$ with length$(\ell) \geqslant \log(|\mathcal{L}| + 2) - 1$. Indeed, there are at most $\sum_{i=1}^{k} 2^k = 2^{k+1} - 2$ non-empty binary strings of length at most k, so that every set $\mathcal{L}$ of non-empty binary strings requires some strings of length at least $\lceil \log(|\mathcal{L}| + 2) \rceil - 1$. Hence our aim is to derive a lower bound on $|\mathcal{L}|$, independent of (L, R). For that purpose, let h, d, t be three positive integers (depending on n, and to be specified later) such that (1) $t < d$, (2) $d \cdot t^h = O(n)$, and (3) $t = \Theta(h) = \Theta(\log n / \log\log n)$. Let T_0 be the rooted tree obtained from a complete t-ary tree $\overline{T}_0$ of depth h in which we add $d - t - 1$ leaves at every internal node, and $d - t$ leaves at the root. We consider the family $\mathcal{T}$, depending on h, d, and t, composed of all the possible port-labeled trees obtained from T_0 (see Fig. 1). For every tree $T \in \mathcal{T}$, we denote by $\overline{T}$ the underlying labeled t-ary tree of depth h. $\overline{T}$ has $|\overline{T}| = \sum_{i=0}^{h} t^i = (t^{h+1} - 1)/(t - 1)$ nodes, and T has $(|\overline{T}| - t^h)(d - t - 1) + 1$ nodes.

The following definition is a specific variant of forwarding labeling, valid for trees in $\mathcal{T}$ only:

Definition 2 *A* relaxed forwarding labeling *on $\mathcal{T}$ is a forwarding labeling where the condition on the routing function is relaxed and is required to apply only for pairs (current node, destination node) where the current node and the destination node belong to $\overline{T}_0$, and the current node is an ancestor of the destination node.*

Let (L, R) be a relaxed forwarding labeling on $\mathcal{T}$, and let $\mathcal{L} = \{L(x) \mid x \in T, T \in \mathcal{T}\}$. We assume that (L, R) is chosen such that $|\mathcal{L}|$ is minimal. By definition, every forwarding labeling is a relaxed forwarding labeling, therefore, for every forwarding labeling (L', R') on $\mathcal{T}$, we have $|\mathcal{L}'| \geqslant |\mathcal{L}|$ where $\mathcal{L}' = \{L'(x) \mid x \in T, T \in \mathcal{T}\}$. Hence our aim is to derive a lower bound on $|\mathcal{L}|$. This is the main key point of the proof developed in the next section.

3 Proof of Theorem 1

The whole section consists, given the family $\mathcal{T}$ described in the previous section, and a relaxed forwarding labeling (L, R) on $\mathcal{T}$, to derive an $\Omega(\log^2 n/\log\log n)$ lower bound for $\log|\mathcal{L}|$.

For every non-leaf node x of the (non labeled) tree $\overline{T}_0$, let $x_1, \ldots, x_t$ be its t children. For every non-leaf node x of a port-labeled tree $\overline{T}$, let ϕ_x denotes the ordered sequence of port-labels such that $\phi_x[i]$ is the label of the edge leading to x_i. Let $\mathcal{S}$ be the set of all possible t-sequences of distinct integers in $1, \ldots, d$, i.e.,

$$\mathcal{S} = \{(s_1, \ldots, s_t) \mid 1 \leqslant s_i \neq s_j \leqslant d \text{ for all } i \neq j\}.$$

We have $|\mathcal{S}| = d!/(d-t)!$, and, for every non-leaf $x \in \overline{T}$, $\phi_x \in \mathcal{S}$. For every $s \in \mathcal{S}$, we define $\mathcal{T}(s)$ as the set of port-labeled trees T of $\mathcal{T}$ such that $\phi_r = s$, where r is the root of T. Observe that $\mathcal{T} = \bigcup_{s\in\mathcal{S}} \mathcal{T}(s)$. Now, for every port-labeled tree $\overline{T}$, and every $i \in \{1, \ldots, t\}$, we denote by $\overline{T}^{(i)}$ the labeled subtree of $\overline{T}$ of root the ith child of the root of $\overline{T}_0$. Let us also define, for every $s \in \mathcal{S}$,

$$\mathcal{W}(s) \;=\; \bigcup_{T\in\mathcal{T}(s)} \left\{ \Big(L(r), L(x_1), \ldots, L(x_t)\Big) \text{ where } r \text{ root of } T \text{ and } x_i \in \overline{T}^{(i)} \text{ for all } i \right\}.$$

Clearly,

$$\bigcup_{s\in\mathcal{S}} \mathcal{W}(s) \;\subset\; \mathcal{L} \times \ldots \times \mathcal{L}, \tag{1}$$

where $\mathcal{L}$ appears $t+1$ times. On the other hand, we have:

Lemma 1 *For all $s \neq s' \in \mathcal{S}$, $\mathcal{W}(s) \cap \mathcal{W}(s') = \varnothing$.*

Proof. Assume that there exists $(\ell_0, \ell_1, \ldots, \ell_t) \in \mathcal{W}(s) \cap \mathcal{W}(s')$. Therefore, there exist two port-labeled trees $T \in \mathcal{T}(s)$ and $T' \in \mathcal{T}(s')$, of roots r and r' respectively, such that: (1) $\ell_0 = L(r) = L(r')$; and (2) for every $i \in \{1, \ldots, t\}$, $\ell_i = L(x_i) = L(x'_i)$ for some x_i's in $\overline{T}^{(i)}$ and some x'_i's in $\overline{T}'^{(i)}$. Since $s \neq s'$, let i be such that $s[i] \neq s'[i]$. The routing function R is such that $R(L(r), L(x_i)) = \phi_r[i] = s[i]$ because $T \in \mathcal{T}(s)$. Thus, $R(\ell_0, \ell_i) = s[i]$. On the other hand, considering now T', we obtain that $R(L(r'), L(x'_i)) = \phi_{r'}[i] = s'[i]$, that is $R(\ell_0, \ell_i) = s'[i]$, a contradiction. ∎

A direct consequence of Lemma 1 is that $|\bigcup_s \mathcal{W}(s)| = \sum_s |\mathcal{W}(s)|$. Hence, from Equation 1,

$$\sum_{s\in\mathcal{S}} |\mathcal{W}(s)| \;\leqslant\; |\mathcal{L}|^{t+1} \tag{2}$$

and therefore $|\mathcal{S}| \leqslant |\mathcal{L}|^{t+1}$, i.e.,

$$|\mathcal{L}| \;\geqslant\; |\mathcal{S}|^{1/(t+1)}. \tag{3}$$

Expressing $\mathcal{S}$ as a function of the degree d of the port-labeled trees in $\mathcal{T}$, we have:

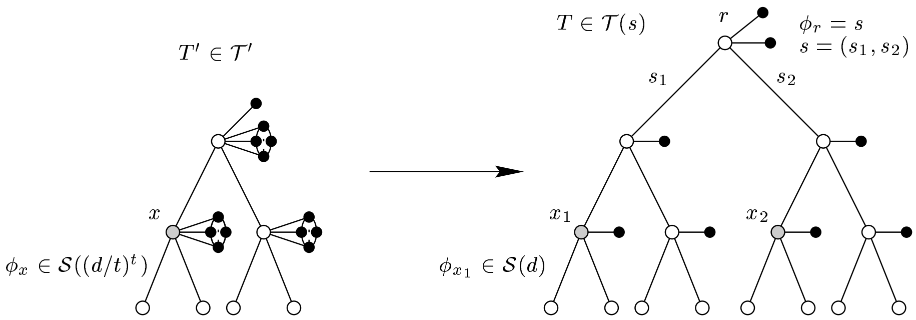

Figure 2: Transformation of $T' \in \mathcal{T}'$ into a tree $T \in \mathcal{T}(s)$.

Lemma 2 *For all $d \geqslant t$, $|\mathcal{S}(d)| \geqslant (d/t)^t$ and $\left|\mathcal{S}\left((d/t)^t\right)\right| \leqslant |\mathcal{S}(d)|^t$.*

Proof. We have $|\mathcal{S}(d)| = t!\binom{d}{t}$, and $\binom{d}{t} \geqslant (d/t)^t$ for all $d \geqslant t$. So, $|\mathcal{S}(d)| \geqslant (d/t)^t$ and $|\mathcal{S}(d)|^t \geqslant (d/t)^{t^2}$. On the other hand, $|\mathcal{S}(d)| \leqslant d^t$, thus $|\mathcal{S}((d/t)^t)| \leqslant (d/t)^{t^2}$. It follows that $\left|\mathcal{S}\left((d/t)^t\right)\right| \leqslant |\mathcal{S}(d)|^t$. ∎

The next lemma gives an inductive lower bound on $|\mathcal{L}|$ as a function of the height h and the degree d of the trees.

Lemma 3 *Let $f(h,d) = |\mathcal{L}|$. For all $d \geqslant t$,*

$$f(h,d)^{t+1} \ \geqslant\ |\mathcal{S}(d)| \cdot f\left(h-1, (d/t)^t\right).$$

Proof. From Equation 2, $|\mathcal{L}|^{t+1} = f(h,d)^{t+1} \geqslant \sum_{s \in \mathcal{S}(d)} |\mathcal{W}(s)|$. We will show that $|\mathcal{W}(s)| \geqslant f\left(h-1, (d/t)^t\right)$ for every $s \in \mathcal{S}(d)$. For that purpose, we construct a relaxed forwarding labeling (L', R') on the family $\mathcal{T}'$ of parameter $h' = h-1$ and $d' = (d/t)^t$ that uses labels in $\mathcal{W}(s)$ only. Let $s \in \mathcal{S}(d)$, and let $\sigma : \mathcal{S}\left((d/t)^t\right) \to \mathcal{S}(d)^t$ be a one-to-one function to be specified later (such a function exists from Lemma 2). Thanks to σ, we associate a port-labeled tree $T \in \mathcal{T}(s)$ to every port-labeled tree $T' \in \mathcal{T}'$, as follows (cf. Fig. 2). To every node x of $\overline{T}'$ is associated a sequence $(x_1, \ldots, x_t)$ of nodes of $\overline{T}$ where x_i is the node corresponding to x in the subtree $\overline{T}^{(i)}$. We set the port-labels of the x_i's such that $\phi_{x_i} = \sigma(\phi_x)[i]$. The port-labels of the root r of T is set such that $\phi_r = s$. Finally, we fix arbitrarily all remaining port-labels of T.

Let us consider a tree $T' \in \mathcal{T}'$, and a node x of $\overline{T}'$. By the previous transformation, let T be the associated tree in $\mathcal{T}(s)$, with root r. Let $(x_1, \ldots, x_t)$ be the associated sequence of nodes in the subtrees $\overline{T}^{(1)}, \ldots, \overline{T}^{(t)}$. We set

$$L'(x) \ = \ (L(r), L(x_1), \ldots, L(x_t)).$$

We can easily check that $L'(x) \in \mathcal{W}(s)$, thus the labeling function L' uses at most $|\mathcal{W}(s)|$ labels for the family $\mathcal{T}'$. To construct R' we need to specify the one-to-one mapping σ. Let

$$\mu_1 : \{1, \ldots, d/t\}^t \to \left\{1, \ldots, (d/t)^t\right\}.$$

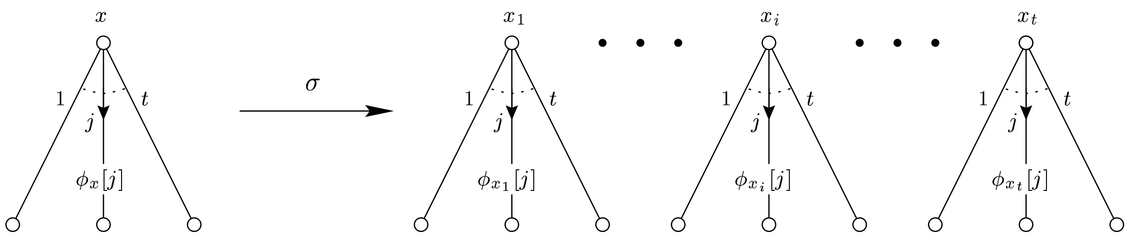

Figure 3: The port-label transformation.

and

$$\mu_2 : \{1,\dots,t\} \times \{1,\dots,d/t\} \to \{1,\dots,d\}$$

be two arbitrary bijections. For instance $\mu_1(a_1,\dots,a_t) = 1 + \sum_{i=1}^{t}(a_i - 1)(d/t)^{i-1}$ and $\mu_2(a,b) = (a-1)d/t + b$, but any bijections would fit as well, and the exact values of μ_1 and μ_2 do not really matter. For every pair $(i,j) \in \{1,\dots,t\}^2$, let

$$p(i,j) \;=\; \mu_1^{-1}\left(\phi_x[j]\right)[i],$$

i.e., the ith element of the sequence $\mu_1^{-1}\left(\phi_x[j]\right) \in \{1,\dots,d/t\}^t$ where $\phi_x[j]$ is the label of the port of x leading to its jth child. The function σ is then defined by

$$\sigma\left(\phi_x\right)[i] \;=\; \left(\phi_{x_i}[1],\dots,\phi_{x_i}[t]\right) \;=\; \left(\mu_2(1,p(i,1)), \mu_2(2,p(i,2)),\dots,\mu_2(t,p(i,t))\right). \quad (4)$$

In other words, by μ_1, the port-label $\phi_x[j] \in \left\{1,\dots,(d/t)^t\right\}$ is decomposed into a sequence of t integers in $\{1,\dots,d/t\}$, namely $p(1,j),\dots,p(t,j)$. By μ_2, the pair $(j,p(i,j))$ is assigned to $\phi_{x_i}[j]$ (see also Fig. 3).

Observation 1. For any j, $\phi_x[j]$ can be obtained from the sequences of port-labels $\phi_{x_1}[j],\dots,\phi_{x_t}[j]$ by the formula:

$$\phi_x[j] \;=\; \mu_1\Big(\mu_2^{-1}\left(\phi_{x_1}[j]\right)[2],\dots,\mu_2^{-1}\left(\phi_{x_t}[j]\right)[2]\Big). \quad (5)$$

As for any sequence, the "[2]" stands here to designate the second element of the pair $\mu_2^{-1}\left(\phi_{x_i}[j]\right) \in \{1,\dots,t\} \times \{1,\dots,d/t\}$.

Observation 2. For any i, $\phi_{x_i}[j] \neq \phi_{x_i}[j']$ for all $j \neq j'$. Indeed, from Equation 4, $\phi_{x_i}[j] = \phi_{x_i}[j']$ implies $\mu_2(j,p(i,j)) = \mu_2(j',p(i,j'))$, that is $j = j'$.

The second observation shows that, for every i, $\phi_{x_i} \in \mathcal{S}(d)$. Therefore $\sigma\left(\phi_x\right) \in \mathcal{S}(d)^t$ as required. The first observation allows us now to define R'. Given $T' \in \mathcal{T}'$, consider two labels $L'(x) = (\ell_0,\ell_1,\dots,\ell_t)$ and $L'(y) = (m_0,m_1,\dots,m_t)$ where x is an ancestor of y, and y belongs to $\overline{T}'$. We define R' such that

$$R'(L'(x),L'(y)) \;=\; \mu_1\left(\mu_2^{-1}\left(R(\ell_1,m_1)\right)[2],\dots,\mu_2^{-1}\left(R(\ell_t,m_t)\right)[2]\right),$$

that is Equation 5 in which each term $R(\ell_i, m_i)$ replaces $\phi_{x_i}[j]$. Let T be the tree associated to T', and let $x_1, \ldots, x_t$ and $y_1, \ldots, y_t$ be the nodes of T associated to nodes x and y of T', respectively. Since x is an ancestor of y in $\overline{T}'$, x_i is an ancestor of y_i in $\overline{T}$. Moreover, if the shortest path from x to y goes through, say, the jth child of x in $\overline{T}'$, then, for every i, the shortest path from x_i to y_i goes through the jth child of x_i in $\overline{T}$. In other words, in the tree $\overline{T}$, $R(L(x_i), L(y_i))$ returns $\phi_{x_i}[j]$ for all i. Since, for every i, ℓ_i and m_i are respectively the labels $L(x_i)$ and $L(y_i)$, it follows that

$$R'(L'(x), L'(y)) \;=\; \mu_1\left(\mu_2^{-1}(\phi_{x_1}[j])[2], \ldots, \mu_2^{-1}(\phi_{x_t}[j])[2]\right).$$

Therefore $R'(L'(x), L'(y)) = \phi_x[j]$, as desired. This completes the construction of the relaxed forwarding labeling in $\mathcal{T}'$, and hence the proof of the lemma. ∎

Let $\alpha = 1/(t+1)$ and $\gamma(d) = (d/t)^t$. One can check that for all $d \geqslant 2t$, $\gamma(d) \geqslant d$. For every $i \geqslant 0$, let $\gamma^{(i)}$ be the ith iterate of the function γ, that is $\gamma^{(0)} = \mathrm{Id}$, and $\gamma^{(i)} = \gamma \circ \gamma^{(i-1)}$ for every $i \geqslant 1$.

Lemma 4 *For all $d \geqslant 2t$, $|\mathcal{L}| \geqslant \prod_{i=1}^{h} \gamma^{(i)}(d)^{\alpha^i}$.*

Proof. From Lemma 3, $|\mathcal{L}| = f(h, d) \geqslant (|\mathcal{S}(d)| \cdot f(h-1, \gamma(d)))^\alpha$. By Lemma 2, $|\mathcal{S}(d)| \geqslant \gamma(d)$ for all $d \geqslant t$, thus $f(h, d) \geqslant (\gamma(d) \cdot f(h-1, \gamma(d)))^\alpha$. Since for all $d \geqslant 2t$, $\gamma(d) \geqslant d$ and $d \geqslant t$, then $\gamma^{(i)}(d) \geqslant t$ for every $i > 0$. So, one can reapply Lemma 3, and we obtain,

$$\begin{aligned} f(h,d) &\geqslant (\gamma(d) \cdot f(h-1, \gamma(d)))^\alpha \;\geqslant\; \gamma(d)^\alpha \cdot \left(\gamma^{(2)}(d) \cdot f(h-2, \gamma^{(2)}(d))\right)^{\alpha^2} \\ &\geqslant \gamma(d)^\alpha \cdot \gamma^{(2)}(d)^{\alpha^2} \cdots \gamma^{(h-1)}(d)^{\alpha^{h-1}} \cdot f(1, \gamma^{(h-1)}(d))^{\alpha^{h-1}}. \end{aligned}$$

Again by Lemma 2, $\left|\mathcal{S}\left(\gamma^{(h-1)}(d)\right)\right| \geqslant \gamma\left(\gamma^{(h-1)}(d)\right)$. By Equation 3, $f(1, \gamma^{(h-1)}(d)) \geqslant \left|\mathcal{S}\left(\gamma^{(h-1)}(d)\right)\right|^\alpha \geqslant \gamma^{(h)}(d)^\alpha$. It follows that $f(h, d) \geqslant \prod_{i=1}^{h} \gamma^{(i)}(d)^{\alpha^i}$. ∎

Lemma 5 *For all $d \geqslant t^2$ and $t \geqslant 2$, $\log|\mathcal{L}| \geqslant (1 - e^{-h/t}) \cdot t \cdot \log(d/t^2)$.*

Proof. One can check that, for all $i > 0$, $d \geqslant t^2$, and $t \geqslant 2$,

$$\gamma^{(i)}(d) \;=\; \frac{d^{t^i}}{t^{t+t^2+\ldots+t^i}} \;=\; \frac{d^{t^i}}{t^{t(t^i-1)/(t-1)}} \;\geqslant\; \frac{d^{t^i}}{t^{2t^i}} \;=\; \left(\frac{d}{t^2}\right)^{t^i}.$$

From Lemma 4, we get $\log|\mathcal{L}| \geqslant \sum_{i=1}^{h} \alpha^i \log\left(\gamma^{(i)}(d)\right) \geqslant \sum_{i=1}^{h} (\alpha t)^i \log\left(\frac{d}{t^2}\right)$. Now, $\sum_{i=1}^{h} (\alpha t)^i = t\left(1 - \left(\frac{t}{t+1}\right)^h\right)$. Since $\left(\frac{t}{t+1}\right)^h \to e^{-h/t}$ as $t \to +\infty$, the result follows. ∎

To complete the proof of Theorem 1, it just remains to set suitable values for d, h, and t. This is the goal of this last lemma:

Lemma 6 *For n large enough, $d = \sqrt{n}$, and $t = h = \log d / \log\log d$, every tree in $\mathcal{T}$ has at most n nodes, and* $\log|\mathcal{L}| > (\log n)^2/(7\log\log n)$.

Proof. The number of nodes of any tree of $\mathcal{T}$ is

$$N \;=\; \big((t^{h+1}-1)/(t-1) - t^h\big)\,(d-t-1)+1 \;\leqslant\; d\cdot t^h.$$

For $h = t = \log d/\log\log d$, we get :

$$\log(t^h) \;=\; t\log t \;=\; \frac{\log d}{\log\log d}\cdot\log\left(\frac{\log d}{\log\log d}\right) \;\leqslant\; \log d. \tag{6}$$

Therefore $d\cdot t^h \leqslant d^2 \leqslant n$, and $N \leqslant n$. From Lemma 5,

$$\log|\mathcal{L}| \;\geqslant\; c\cdot t\cdot\log\big(d/t^2\big) \;\geqslant\; c\cdot t\cdot\log d - O(\log(t^t)),$$

where $c = 1 - e^{-1} > 4/7$. Therefore, from Equation 6,

$$\log|\mathcal{L}| \;\geqslant\; c\cdot\frac{\log^2 d}{\log\log d} - O(\log d) \;\geqslant\; \frac{c}{4}\cdot\frac{\log^2 n}{\log\log n} - O(\log n) \;>\; \frac{\log^2 n}{7\log\log n}\;.$$

This completes the proof of Theorem 1. ∎

4 Conclusion

In this paper, we have shown that, in the fixed-port model, shortest path routing in trees has a memory requirement of at least $\Omega(\log^2 n/\log\log n)$, and this bound is tight. Beside the significance of the result in regard with previous upper bounds for routing in trees and in arbitrary networks, this result must also be compared to the best upper bound on the memory requirement for shortest path routing in graphs of bounded treewidth. Up to our knowledge, the best result is due to [12] where it is shown that there exists a shortest-path routing scheme for graphs on treewidth k using addresses and local data structures of size $O(k\log^2 n)$. For tree (treewidth 1), this bound is not tight, by a factor $\log\log$. We hence let as an open problem the question of whether one can design (under the fixed-port model) routing schemes with memory requirement $O(k\log^2 n/\log\log n)$ for graphs of treewidth k. For instance, can we route on shortest paths in outer-planar graphs (treewidth 2) using addresses and local data structures of $O(\log^2 n/\log\log n)$ bits only?

References

[1] Baruch Awerbuch and David Peleg. Sparse partitions. In 31^{th} *Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 503–513. IEEE Computer Society Press, October 1990.

[2] Baruch Awerbuch and David Peleg. Routing with polynomial communication-space trade-off. *SIAM Journal on Discrete Mathematics*, 5(2):151–162, May 1992.

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

[4] Tamar Eilam, Cyril Gavoille, and David Peleg. Compact routing schemes with low stretch factor. In 17^{th} *Annual ACM Symposium on Principles of Distributed Computing (PODC)*, pages 11–20. ACM PRESS, August 1998.

[5] Paul Erdös. Extremal problems in graph theory. In *Publ. House Cszechoslovak Acad. Sci., Prague*, pages 29–36, 1964.

[6] Pierre Fraigniaud and Cyril Gavoille. Routing in trees. In 28^{th} *International Colloquium on Automata, Languages and Programming (ICALP)*, volume 2076 of Lecture Notes in Computer Science, pages 757–772. Springer, July 2001.

[7] Cyril Gavoille. A survey on interval routing. *Theoretical Computer Science*, 245(2):217–253, 2000.

[8] Cyril Gavoille. Routing in distributed networks: Overview and open problems. *ACM SIGACT News - Distributed Computing Column*, 32(1):36–52, March 2001.

[9] Cyril Gavoille and David Peleg. Compact and localized distributed data structures. Research Report RR-1261-01, LaBRI, University of Bordeaux, 351, cours de la Libération, 33405 Talence Cedex, France, August 2001.

[10] Cyril Gavoille, David Peleg, Stéphane Pérennes, and Ran Raz. Distance labeling in graphs. In 12^{th} *Symposium on Discrete Algorithms (SODA)*, pages 210–219. ACM-SIAM, January 2001.

[11] David Peleg. Informative labeling schemes for graphs. In 25^{th} *International Symposium on Mathematical Foundations of Computer Science (MFCS)*, volume 1893 of Lecture Notes in Computer Science, pages 579–588. Springer, August 2000.

[12] David Peleg. Proximity-preserving labeling schemes. *Journal of Graph Theory*, 33:167–176, 2000.

[13] David Peleg and Eli Upfal. A trade-off between space and efficiency for routing tables. *Journal of the ACM*, 36(3):510–530, July 1989.

[14] Mikkel Thorup and Uri Zwick. Approximate distance oracles. In 33^{rd} *Annual ACM Symposium on Theory of Computing (STOC)*, pages 183–192, Hersonissos, Crete, Greece, July 2001.

[15] Mikkel Thorup and Uri Zwick. Compact routing schemes. In 13^{th} *Annual ACM Symposium on Parallel Algorithms and Architectures (SPAA)*, pages 1–10, Hersonissos, Crete, Greece, July 2001. ACM PRESS.