Upper and Lower Bounds for Routing Schemes in Dynamic Networks (Abstract)

Yehuda Afek
AT&T Bell Labs &
Computer Science Department
Tel-Aviv University

Eli Gafni *
Computer Science Department
U.C.L.A. and Tel-Aviv University

Moty Ricklin
Computer Science Department
Tel-Aviv University

Abstract

We present an algorithm and two lower bounds for the problem of constructing and maintaining routing schemes in dynamic networks. The algorithm distributively assignes addresses to nodes and constructs routing tables in a dynamically growing tree. The resulting routing scheme routes data messages over the shortest path between any source and destination, assigns addresses of $O(\log^2 n)$ bits to each node, and uses in its routing tables $O(\log^3 n)$ bits of memory per incident link, where n is the final number of nodes in the tree. The amortized communication cost of the algorithm is $O(\log n)$ messages per node.
We also give two lower bounds on the tradeoff between the quality of routing schemes (i.e., their stretch factor) and their amortized communication cost in general dynamic networks.

1 Introduction

Routing data messages is a fundamental task in any data communication network. A *routing scheme* is a distributed protocol that directs the data traffic. To control the data traffic in the network, most routing schemes use two basic information structures; First, an *address* is assigned to each node. Each data message carries the address of its destination. Second, a *routing function*, R_v, is associated with each node v. When a data message arrives at a node the routing function of that node is applied to the destination address of the data message to yield the next link (or links) on which the message will be forwarded (i.e., `next-link` $= R_v(Address(destination))$). [1]

In this paper we investigate the design and analysis of algorithms that construct and maintain routing schemes in dynamic networks. Previous work [KK77, KK80, PU88, ABNLP88a] was mainly concerned with the tradeoff between the quality of the routes (stretch factor) and the memory requirements, in static networks. Network changes require expensive pre-processing to reconstruct the routing scheme over the whole network. The newly constructed structure is used until the next change. As the size of existing networks grow, presumably topological changes will occur more often and the reconstruction of the routing schemes might not keep up with the changes, overloading the network with control messages. Hence, dynamic routing schemes are an essential mechanism in the operation of such networks. These schemes should adapt to dynamically changing networks with low adaptation cost, while maintaining fixed addresses. (I.e., addresses should not change despite topological changes.)

Following [ABNLP88a], we consider the following measures and characteristics of routing schemes:

- **Stretch factor:** The stretch factor of a routing scheme is the maximum over all node pairs (s,t) of the ratio between the number of messages sent when routing a single message from s to t and the shortest distance from s to t, assuming the scheme has reached quiescence following the last topological change. [2] (We assume 1 unit of cost for transmitting a message over any link.)

- **Space requirement:** The maximum memory required to store the routing function at any node. [3]

*Supported by NSF Presidential Young Investigators Award under grant DCR84-51396 and Matching fund from XEROX Co. under grant W881111.

[1] Routing schemes may use more sophisticated data structures and techniques, such as changing the header of the message (the destination address) as the message traverses its path to the destination; For a more accurate definition of static routing schemes see [PU88].

[2] Clearly, at times of topological changes an adversary can force the routing schemes to behave poorly, e.g. with an $\Omega(n)$ stretch factor.

[3] The space requirement of a routing scheme is an important parameter because it effects the over-all performance of the routing scheme. For example, if the space requirement is high then

- **Address size:** The maximum number of bits required to represent the address of any node.

- **Fixed addressing:** Once the routing scheme assigns a node an address, that address will never change. (This definition differs slightly from the *Name independence* of [ABNLP88a].)

- **Adaptation cost:** The communication and time complexities of the algorithms that adapt the routing functions (and addresses if addressing is not fixed) following topological changes. These costs are amortized over the number of changes.

In this paper we provide both positive and negative results on the complexity of dynamic routing schemes. On the positive side we present an efficient routing scheme for dynamically growing trees with fixed addressing, constant stretch factor and logarithmic cost of adaptation per new node. The negative results are two lower-bounds on the tradeoff between the adaptation costs and the stretch-factors of dynamic routing schemes.

Routing schemes for static tree topologies have played a major role in the design of static routing schemes in the past [SK85, vLT87, PU88, ABNLP88b, ABNLP88c, ABNLP88a]. Similarly, the tree routing scheme can be used as an efficient routing scheme in arbitrary dynamically growing networks by considering a spanning tree of the network. In addition, the design of distributed protocols for other problems in dynamically growing trees has been an important theoretical question in distributed computing, and this work is an important step towards a more general solution to these questions [AAPS87, AP87, AS87, PU88, BYK86].

Routing schemes for static trees have been intensively studied [SK85, vLT87, FJ86]. Most of these are Interval Routing (ITR) schemes. In an ITR scheme the address of each node is an integer. The routing function at each node associates with each incident link a set of intervals, such that the collection of sets at any single node are disjoint. To send a message to a node whose address is a, the interval containing a is found and the message is routed over the link associated with that interval. Note that at each node one link can be considered the default link: if the address of a node is not in any of the explicit intervals then the messages for that node are routed over the default link.

A simple ITR scheme which is based on Depth First Search (DFS) is given in [SK85]. In this scheme, the nodes of a rooted tree are numbered with depth-first-search numbers, and the interval of each link is the range of dfs-numbers of its descendants. This scheme has stretch-factor 1, address size $\log n$ bits, and $\log n$ bits of memory per incident link to store the intervals. Van Leeuwen and Tan [vLT87] built on this idea to present a scheme for dynamically growing trees that does not preserve fixed addressing and has an $O(n)$ adaptation cost per topological change.

An ITR scheme for dynamically growing trees that does preserve fixed addressing and has logarithmic adaptation cost is presented in [AAPS87], where the problem of name assignment is considered. The resulting ITR scheme has stretch factor 1, address size at most $O(\log n)$ bits, but its space complexity is $O(n \cdot \log n)$ bits.

In this paper we present a new generalization of the DFS-based ITR scheme for dynamically growing trees. The new scheme uses fixed addressing and has address size $O(\log^2 n)$, uses $O(\log^3 n)$ bits of memory per incident link, and has a logarithmic adaptation cost. An interesting aspect of our scheme is that it maintains a global structure on a dynamically growing network with only local (logarithmic) cost.

Following the upper bounds for dynamically growing trees we present lower bounds on the tradeoff between the stretch factor and adaptation cost of routing schemes for dynamic networks. In the first lower bound, we prove that any routing scheme in an arbitrary dynamic network (in which links may come up and go down [AAG87]), whose stretch factor is less than k must send $\Omega(n/k)$ messages per topological change to update the routing functions at the various nodes. Second, we prove the same lower bound on routing schemes for dynamically growing trees in which the address of a node is required to be its *id* (name independence [ABNLP88a]). The lower bounds allow unbounded memory and message size.

The design and analysis of sub-linear amortized cost, fixed addressing routing schemes for dynamically growing, arbitrary topology networks remains as an open question.

Network Model: Throughout the paper the standard model of asynchronous networks is assumed with the following variations: In Sections 2 the model of a dynamically growing tree [AAPS87, BYK86] is assumed. In this model, links never fail, but new links and nodes may join the tree network at any time. In the lower bounds, Section 3, we use the model of [AAG87] in which links may go down and come up an arbitrary number of times.

each time a data message is routed we might need to retrieve information from a slow memory [CGK88] thus dramatically effecting the delay of data messages in the network.

2 The algorithm

2.1 overview

The algorithm to be presented is a generalization to the dynamic case of the static ITR scheme. As in the static case the scheme is based on Depth-First-Search numbering, but unlike the static case the algorithm employs an increasing number of DFS's as the dynamic tree grows. Each of these DFS's will have an index, and each node that was searched by a DFS will maintain an ITR table for it, with that index. The first DFS that passes through node v will assign it the address (x, m), where x is the index of that DFS and m is v's DFS number by which the DFS numbered v (to be denoted by $dfs(x, v)$).

To describe an index of a DFS, suppose that the root of the tree has conducted its ith DFS. When a node v obtains a DFS token for that DFS for the first time, it marks all the links incident to it. The DFS will be forwarded by v only over these nodes. In the subtrees that will join it later through new unmarked incident links, v recursively will play the role of a root. Each DFS is recursively given an index (i, m, l) by the node, v, which initiates it. Where i is the index of the last DFS initiated by an ancestor of v, and in which v has participated. The entry m is $dfs(i, v)$, the numbering of v in DFS i. Finally, l is a count of the number of DFS's initiated at v since participating in DFS i. Thus an index of a DFS can be viewed as an alternating sequence of DFS counts and DFS numbers that terminates with a DFS count $(i_0, m_0, ..., i_k)$. A name of a node is a similar sequence but terminates with a DFS number.

A key invariant in the algorithm is that when a node is ready to initiate a DFS with an index whose last digit is i, the number of nodes in its subtree in the dynamic tree is greater than 2^i. To maintain this, a node v can only initiate a DFS with index $(i_0, m_0, ..., i_{k-1}, m_{k-1}, i_k)$ if $i_{k-1} > i_k$. (Thus, the i's are monotonically decreasing in any DFS index.) Recursively, when v gets a signal requesting the initiation of the $i_k + 1$st DFS and $i_k + 1 = i_{k-1}$, it refrains from doing this and instead signals the root requesting its $(i_{k-1} + 1)$st DFS. The signal to the root to start the $(i_{k-1} + 1)$st DFS is passed by on a request message from node v to the root that travels up the tree. Per each DFS we want only one such message to travel on a link. Consequently, when a request message has to travel on a link that has been traversed already by the same request the message is safely stopped. Either the root has already started the $(i_{k-1}+1)$st DFS, or there is another request message already closer to the DFS initiator.

There may be several DFS's initiated by different nodes searching overlapping parts of the network concurrently. We notice that the indices of DFS's can be totally ordered using lexicographic order. We employ the rule that a node participates in the highest DFS it observes. All messages and operations associated with lower DFS's are purged.

Routing is performed as follows: When node v is to forward a message to address x, it looks up the sequence of DFS indices in x, and compares them with the DFS indices in which it has participated. If it does not find one index of its own that agrees with one of the indices of x, it then forwards the message to its father. Otherwise it uses the ITR table of that index to forward the message to the dfs numbered node that appears in x after the index.

In the full version of the paper we show numerous variations of this algorithm. All these algorithms though share the same worst case complexity:

Theorem 1 *The algorithm outlined above has the following worst case complexities given that the dynamic tree grew to a final size of n nodes:*
Address size - $O(\log^2 n)$ bits.
Routing tables - $O(\log^3 n)$ bits per node + $O(\log^2 n)$ bit per link.
Amortized adaptation cost - $O(\log n)$ messages.
Stretch Factor - 1.

Finally we notice that a node may join and wait indefinitely for a DFS to traverse it, because the DFS traverses an unbounded number of nodes that joined later than it. To prevent this situation, we amend the DFS traversal. First we maintain the stopped DFS request messages at a node in a queue. When the DFS arrives its search complies with the order of the queue. Furthermore, a DFS whose last digit is i does not travel to depth greater than 2^i from its initiator, and similarly backtracks from a node after it has given addresses to more than 2^i of its children.

3 Lower Bounds

In this section we prove that any routing scheme in an arbitrary dynamic network whose stretch factor is less than some constant k has an amortized adaptation cost of at least $\frac{n}{16k}$ in both messages and time.

Model and assumptions:

1. We assume the model of a dynamically changing network as in [AAG87]. The set of nodes is fixed and links may come up and go down an arbitrary number of times.

2. Each node has unlimited memory.

3. Message size is unbounded.

Before proving the lower bounds we refine the complexity measures. We consider two types of messages: *data messages* and *control messages*. The control messages are the messages exchanged by the routing scheme to compute the routing functions and addresses of the nodes following a topological change. The data messages are the messages sent when a data message is routed from its source to its destination. (A message may be of both types.)

A routing scheme in a dynamic network can be in one of two global state, *quiescent*, or *updating*. An updating state is one which, in the absence of further topological changes, can lead to the transmission or reception of control messages. Clearly, a quiescent period (between two updating periods) ends with the occurrence of a topological change.

If the stretch factor is measured during arbitrary updating intervals, the stretch factor of any scheme can be shown to be $\Omega(n)$. Hence, we consider the stretch factor of a routing scheme in a dynamic network as the maximum over all u, w $u \neq w$ of the ratio between the cost (number of data messages) of sending a data item from node u to w and the length of the shortest path between u and w during any one quiescent interval. That is, assuming that the system is quiescent from the time the data is sent from u until all the resulting data messages reach their destination.

Dynamic routing schemes may be designed in such a way that control messages are sent only if the delivery of a data message is required, i.e., following a topological change no control messages are sent until a data message is sent. Moreover, the data messages may also serve as control messages, updating the routing scheme data structures as they go. This and similar considerations indicate that the definitions of stretch factor and adaptation cost do not entirely capture the behavior of efficient dynamic routing schemes. We leave such a formulation as an open problem. However, the simplifying notions introduced in this section do not effect or weaken our lower bounds.

Theorem 2 *The amortized adaptation cost of any routing scheme for arbitrary dynamic networks of size n with a constant stretch factor $< k$ is at least $\frac{n}{16k}$.*

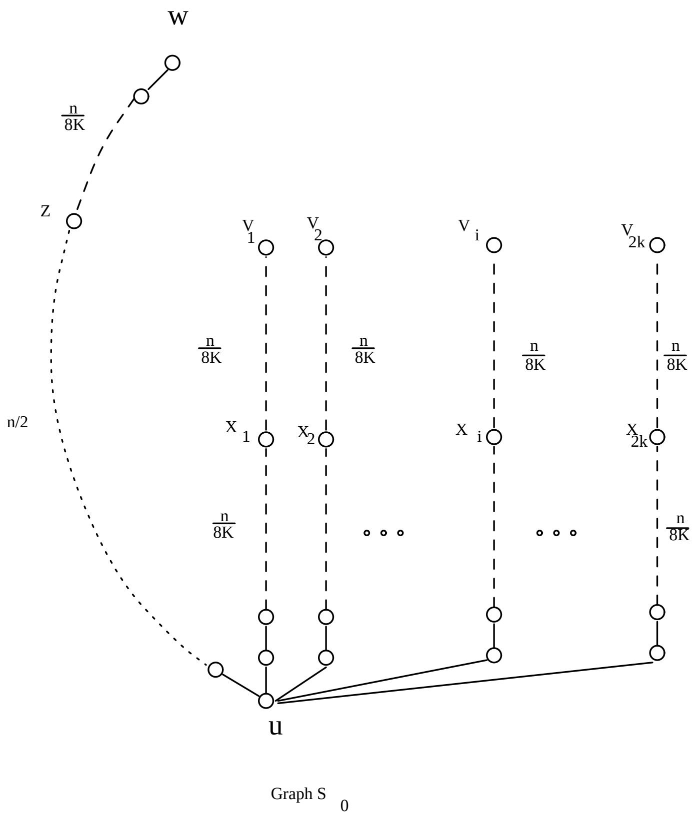

Figure 1: Network for the lower bound proof.

The theorem is proved by considering the network of Figure 1.

Lemma 1 *There is a scenario of m topological changes in the network of Figure 1 such that any routing scheme with a constant stretch factor smaller than $k \le n/16$ will send at least $\frac{mn}{16k}$ control messages.*

Proof: For the proof we consider $2k+1$ variants on the topology of the network in Figure 1: $T_0, T_1, ..., T_{2k}$. T_0 is the topology of the network in Figure 1. In this topology there are $2k+1$ disjoint paths emanating from node u. One path is $(n/2) - 1$ long. The node at the other end of this path is called w. The node on this path which is at distance $n/(8k)$ from w, is called z. All the other paths are $(n/4k)$ long and are numbered from 1 to $2k$. For every $1 \le i \le k$, x_i is the node at the middle of the $i-th$ path, and v_i is the node at the end of path i. T_i is the same as T_0 plus an edge from w to v_i.

We call a state of the system S_i, if in this state the routing scheme is quiescent and the topology is T_i. Note that the local states of processors are not necessarily the same each time the system enters a state in S_i. For the lower bound proof we define a scenario as a sequence of $\{S_{i_0}, S_{i_1}, ..., S_{i_j}, S_{i_{j+1}}, ...\}$ such that, the next quiescent state after the system leaves a quiescent state in S_{i_x} is in $S_{i_{x+1}}$. We are interested in scenarios in which the system alternates between states in S_i for $0 \leq i \leq 2k$. Note that in between each two such states the system goes through an updating state. Such scenarios are denoted $\{S_{i_0}, S_{i_1}, ..., S_{i_x}\}$ for an arbitrary x.

Claim 2 *There is a scenario $\{S_0, S_{i_1}, S_0, \cdots, S_0, S_{i_m}\}$ such that either*

1. *at least $\frac{mn}{16k}$ control messages were sent, or*
2. *there is a prefix of the scenario, $\{S_0, S_{i_1}, S_0, \cdots, S_0, S_{i_j}\}$, such that in state S_{i_j} it costs more than $n/4$ to send a data message from u to w (which is at least k times the shortest path between u and w in T_{i_j}).*

Proof of Claim: By induction on m. Clearly the lemma holds for $m = 0$. Induction step: let SC_m be such a scenario. If case (2) holds on SC_m then obviously it holds on $\{SC_m, S_0, S_1\}$ and we are done. Otherwise, we define $2k+1$ scenarios and run them all in parallel. Scenario $SC_{m,i}$, for $i = 1, ..., 2k$, is defined to be $\{SC_m, S_0, S_i\}$ $i \neq 0$, and $SC_{m,0}$ to be $\{SC_m, S_0\}$, such that: 1) In all the scenarios u is required to send a data message to w, after the system enters the last quiescent state and 2) assume that the system behaves as a synchronous system with the same delay on the links in all the scenarios. Now assume that for every scenario $SC_{m,i}$ $1 \leq i \leq 2k$, the number of control messages after the last topological change $< \frac{n}{16k}$. Let u send a message to w in $SC_{m,0}$. Define j in $SC_{m,0}$ as follows:

1. If data messages (resulting from the delivery of a data message from u to w), have reached all the x_i nodes before the first data message has reached z. Then define j so that x_j is the last such node to receive a data message.
2. Otherwise, let j be the index of some x_j which did not receive the first data message before a data message reached z.

We claim that in $SC_{m,j}$ the delivery of the messages from u to w costs at least $n/4$. Since the control messages sent in the final stable state of $SC_{m,j}$ have passed less than $\frac{n}{16k}$ links, the states of all the processors and edges, aside from those on the path, $(z, \cdots w, \cdots, x_j)$ is the same final state in both $SC_{m,0}$ and $SC_{m,j}$, as long as the first data message from u has not reached z or x_j. Thus the data messages sent in $SC_{m,j}$, have either reached z, or have reached all the x nodes, because the message must eventually reach w. Therefore, in both cases the messages traversed at least $n/4$ links. ∎

Finally, we can prove a similar lower bound for routing schemes in dynamically growing trees which possess the following property:

Definition 1 Name independence: *A routing scheme is name independent if the addresses of the nodes are dictated to the scheme* a priori*, e.g. are their id's. Note this requirement is stronger than the fixed addressing requirement, as proven by our upper bound and the following:*

Theorem 3 *The amortized adaptation cost of any name independent routing scheme for dynamically growing trees of final size n, and with a constant stretch factor $< k$ is $\Omega(n/k)$.*

The proof of Theorem 3 is an extension to the proof of Theorem 2 and is omitted from this summary.

4 Discussion

We have introduced the question of the design and analysis of routing schemes in dynamic networks, and have given an algorithm for dynamically growing trees and lower bounds for general dynamic networks. Many problems remain open:

- Give a more precise definition of routing schemes in dynamic networks and define more sophisticated complexity measures.
- Our algorithm assigns to each node of a dynamically growing tree an address of size $O(\log^2 n)$ bits and has space complexity $O(\log^3 n)$ bits. It remains open whether there exists an algorithm that assigns addresses of size $o(\log^2 n)$ bits while keeping the space complexity poly-logarithmic per link.
- Design an efficient routing scheme for arbitrary topology dynamically growing networks (i.e., without failures).

Acknowledgment: We thank Michael Merritt for helpful discussions and criticism of earlier versions of this paper.

References

[AAG87] Y. Afek, B. Awerbuch, and E. Gafni. Applying static network protocols to dynamic networks. In *Proc. of the 28th IEEE Annual Symp. on Foundation of Computer Science*, pages 358–370, October 1987.

[AAPS87] Y. Afek, B. Awerbuch, S. Plotkin, and M. Saks. Local management of a global resource in a communication network. In *Proc. of the 28th IEEE Annual Symp. on Foundation of Computer Science*, October 1987.

[ABNLP88a] B. Awerbuch, A. Bar-Noy, N. Linial, and D. Peleg. Compact distributed data structures for adaptive routing. In *Proc. of the 21st Ann. ACM Symp. on Theory of Computing*, page To appear, May 1988.

[ABNLP88b] B. Awerbuch, A. Bar-Noy, N. Linial, and D. Peleg. Improved routing strategies with succinct tables. Technical Report MIT/LCS/TM-354, M.I.T., 1988.

[ABNLP88c] B. Awerbuch, A. Bar-Noy, N. Linial, and D. Peleg. Memory-balanced routing strategies. Technical Report MIT/LCS/TM-369, M.I.T., 1988.

[AP87] B. Awerbuch and S. Plotkin. Approximating the size of a dynamically growing distributed network. Technical Report MIT/LCS/TM-328, M.I.T., April 1987.

[AS87] Y. Afek and M. Saks. Detecting global termination condition in the face of uncertainty. In *Proceedings of the ACM Symp. on Principles of Distributed Computing*, August 1987. Also AT&T Technical Memorandum, August 1986.

[BYK86] R. Bar-Yehuda and S. Kutten. Fault toletant leader election with termination detection, in general undirected networks. Technical report, Duke University, Durham, January 1986.

[CGK88] I. Cidon, I. Gopal, and S. Kutten. New models and algorithms for future networks. Submitted, February 1988.

[FJ86] Greg N. Frederickson and Ravi Janardan. Seperator-based strategies for efficient message routing. In *Proc. of the 27th IEEE Annual Symp. on Foundation of Computer Science*, pages 428–437, October 1986.

[KK77] L. Kleinrock and F. Kamoun. Hierarchical routing for large networks; performance evaluation and optimization. *Computer Networks*, 1:155–174, 1977.

[KK80] L. Kleinrock and F. Kamoun. Optimal clustering structures for hierarchical topological design of large computer networks. *Computer Networks*, 10:221–248, 1980.

[PU88] David Peleg and Eli Upfal. A tradeoff between size and efficiency for routing tables. In *Proc. of the 20th Ann. ACM Symp. on Theory of Computing*, pages 43–52, May 1988.

[SK85] N. Santoro and R. Khatib. Labelling and implicit routing in networks. *The Computer Journal*, 28:5–8, 1985.

[vLT87] J. van Leeuwen and R. B. Tan. Interval routing. *The Computer Journal*, 30(4):298–307, 1987.