The Effect of Network Hierarchy Structure on Performance of ATM PNNI Hierarchical Routing

Baruch Awerbuch *Yi Du*[*]
Department of Computer Science
Johns Hopkins University
Baltimore, MD 21218
{baruch, yidu}@cs.jhu.edu

Yuval Shavitt
Bell Laboratories, Lucent Technologies
101 Crawfords Corner Rd., room 4G-627
Holmdel, NJ 07733-3030
shavitt@lucent.com

Abstract

Networks deploying hierarchical routing are recursively partitioned into sub-networks that do not reveal full details of their internal structure outside their domains. Instead, an aggregated view of certain parameters that are associated with traversal within such sub-networks between some border nodes is advertised. The ATM PNNI standard and the Internet Nimrod architecture both adopt this approach for routing.

This paper studies the effectiveness for ATM hierarchical routing protocols on networks with different hierarchical structures by simulation. Our study shows that, in general, hierarchical source routing performs well compared to global routing strategy which imposes no hierarchy, while utilizing less storage and communication overhead. For certain networks and topologies, hierarchical routing performs better than global routing. Different hierarchies imposed on the same topologies have significant different performance on performance and routing delay. This suggests the necessity of studying the hierarchy design for communication networks using hierarchical routing.

1 Introduction

Routing algorithms can be divided into two major classes: distance vector algorithms [10, 9] and link state algorithms [11, 12]. In distance vector algorithms routers maintain tables with an entry per destination network where the route cost and the next hop to the destination are kept. These tables are exchanged between immediate neighbors, thus the propagation of information about changes in route costs might be slow. It has been shown [7] that distance vector algorithms suffer from slow recovery from link failures, and during the slow recovery time routing loops may occur. In link state algorithms, network routers flood their emanating link costs to the network. Next hop routes are calculated based on a full topology database maintained in each router. Link state algorithms do not exhibit the slow recovery time occurring during link failures.

Both algorithm classes suffer from scalability problems: in distance vector algorithms table size grow linearly with the number of nodes; in link state algorithms communication cost grows with the number of links, and so does the space requirement for topology maintenance. To cope with scalability, hierarchical routing was suggested [1, 8]. In this approach a network is grouped into hierarchy of nodes at various levels. Each node at a high level reports information for underneath topology to other interested nodes. Each node also receives reported routing information from nodes in other domains and exchange them information to other lower level nodes. As a result, each node needs only to maintain partial information about the entire network. Requirement on routing information storage at each node is thus substantially reduced. ATM PNNI [1] routing standards and Internet Nimrod [8] routing protocols both adopts such strategy.

Domains typically correspond to a management organization or a geographical region. We need to either reduce the size of the representation of the structure of these domains, or to hide the details of internal structure for these domains for security reasons [1, 8]. Instead of the full information for the internal structure, a summary, or *aggregation* of routing information for the domain is maintained and reported by the domain. To

[*]Corresponding author.

achieve scalability for routing protocols, aggregation is needed. This is because the complexity of routing protocols grows at least linearly with the number of links in the network for both source routing and link-by-link routing. Many ways to aggregate routing information, called *aggregation schemes*, exist with various tradeoffs on representation size and representation accuracy for routing information.

In this paper we study performance of hierarchical routing for various link cost metrics and aggregation schemes. We are also interested in understanding the performance implication of different hierarchy embedding upon a same network graph to study the effectiveness for hierarchical routing. We choose ATM PNNI standard [1] as representative for the study, since PNNI routing and signaling protocols are mature and its performance has significant implication on the future deployment of ATM networks. We perform simulation studies to compare the performance of both the global routing and hierarchical routing strategy, in terms of their network throughput and control overhead such as signaling delay.

We show that, hierarchical routing could be as effective as global routing when exponential link cost metric is used, though it could incur slightly higher signaling delay. For some topologies, imposing hierarchies would lead to improved routing performance than global routing. We also observe that an increase on the levels of hierarchy does not necessarily lead to better performance. This implies that different hierarchies for same topology could have much different performances. Therefore it is important to study hierarchy designs for hierarchical routing. With saving in storage for routing information and communication overhead, and its capability for security and scalability enhancement, hierarchical routing provides an effective and scalable routing solution for future global teranode networks.

The rest of the paper is organized as follows. Hierarchical routing is introduced in section 2. Section 3 discuss the simulation environment we develop for studying ATM PNNI routing. Simulation parameters under study are detailed in section 4. Section 5 presents the result for the performance study. Section 6 gives the conclusion.

2 Hierarchical source routing

In this paper, we consider *source routing* in general, which is of the broad link-state routing category. To realize a connection request, source node receiving the request would use local routing information maintained by itself to make decision as to which path the connection would take. Routing information for a link is usually presented to routing protocols in the form of a cost metric value. An routing algorithm is executed to use all these link cost metric values to find a feasible path for the connection request. In fact, shortest path algorithm, such as Dijkstra's Algorithm, is widely used for this purpose.

Note that if information about all physical links in the routing domain is maintained at each node, a physical path can always be found for a connection request by the source node. This style of source routing is called *global source routing*. The found path is not necessarily feasible (in terms of resources for all path links for the request), though. A signaling process is invoked to traverse all links on the path according to their order on the path, while reserving bandwidth on the link which has enough bandwidth to support the request. When a link can not support a request, a backtrack (or *crank-back*) is needed to go back to previous node and try another link. This routing is also called *link state* routing and is the approach adopted by Internet routing protocols.

However, often it is not feasible that a source node maintain full information for all physical links. A node can maintain processed, or *aggregated*, information about the network. Using these information, source can find a sequence of partial paths at different abstraction levels, while they together consist of a path for a connection request. The successful establishment for each partial path in this sequence thus realize a route of physical links for the request. These paths are usually in different levels in a topology hierarchy, as this kind of routing is called *hierarchical source routing*. This is the approach deployed by ATM PNNI standard [1]. After the partial paths are determined, a *signaling* process is invoked to traverse links on these path to reserve bandwidth on those links that have enough bandwidth to support the request. When a link can not support a request, a backtrack (or *crank-back*) action makes the traverse back to the previous node and try another link from there.

These two styles of source routing both require each node maintains a view on the entire routing domain, though they can differ in the degree of detail of the maintained information. Besides routing information storage, there are also network overhead when information about state change for each link is transmitted to nodes which maintain information related to these links. Communication overhead is thus incurred to update nodes with up-to-date information. Source routing requires each node store its own routing link state information, and the network must support communication among nodes for link state update. The routing information storage, together with the communication cost, are the two major system expenses for such routing protocols.

Routing information advertised by a node to others is aggregated by *topology aggregation scheme* to reduce routing information storage in nodes and topology information exchange, the main reason why hierarchical routing is scalable. Topology aggregation essentially derives a simpler connectivity graph out of a much more complicated real connectivity graph. Various topology aggregation schemes exist, among them are tree scheme [6] and spanner scheme [2, 13].

In this paper, we focus on the effect of hierarchies of structure on the hierarchical routing performance. That is, we consider how various hierarchies imposed on the same topology would affect the performance of hierarchical routing. ATM PNNI routing protocol is used in the study.

3 STARS: ATM PNNI simulation

We have implemented **STARS** [5], a simulator for ATM routing to study how aggregation schemes affect performance for hierarchical routing [4]. **STARS** is an object-oriented discrete event simulator written in C++ that emulates PNNI routing protocols over hierarchical structured networks. It has a modular architecture that enables easy integration of different modules to achieve various functionality. For example, an generic interface of routing link cost function is provided to permit insertion of various cost functions for study. Similar interface for aggregation schemes for domain routing information summary and topology graphs are also provided. Below we briefly discuss the design of STARS. More details on the design and implementation of **STARS** can be found in [5].

STARS consists of the following components:

- Simulation Engine;
- Topology Aggregation Server;
- Network Topology Generator;
- Network Traffic Load Generator;
- Statistics Collector;
- Graphical Display;

Simulation engine adopts shortest path algorithm using some link cost function to select feasible path. This is a powerful formulation since many existing schemes fit well with it. For example, the so-called min-hop scheme used in Internet routing is can be realized by a shortest path algorithm using constant link cost. On the other hand, link cost functions that consider other link characteristic such as link utilization and link capacity capture more information and promise better performance. Theory suggests that, there exists on-line routing algorithms based on exponential cost link which achieve optimal performance in terms of throughput against off-line algorithm. To facilitate such study, our simulator allows easy incorporation of different link metric functions.

Aggregation information for domains has to be recomputed whenever there is a need to do so (such as a "significant" change occurs in the topology parameters to justify an aggregation for the new topology information). *Aggregation server* accepts a topology and an aggregation method and returns the resulting aggregation metric after the aggregation calculation. Since there could be far too many calls to aggregations, for increased performance besides providing with a capability for direct calls to aggregation functions (which will be executed sequentially on the same machine), the simulator has an option to run several aggregation servers on other machines to alleviate loads on one computer. One aggregation server can serve multiple requests simultaneously.

Network topology generator produces various topologies to be studied, and network traffic load generator generates dynamic random connection requests to the simulator to drive the simulation. A request is represented as by its staring time, source and destination node, and traffic bandwidth requirement. We use a simple topology description format and allows input to specify connection request streams with various arrival/departure distribution and bandwidth requirement. Statistic collector is used to collect data of interest, and a graphical display shows routing signaling progress for the current connection request under establishment.

In the next section, we discuss the simulation parameters and performance measures we use for the simulation experiment.

4 Simulation parameters

Cost link functions and aggregation schemes are clearly two important parameters for routing performance. The network performance is measured by network throughput which is defined as the total flow successfully transmitted by the network, and control overhead such as signaling delay which measures the efficiency of routing establishment. These parameters are detailed in the following.

4.1 Link cost metric

As discussed in above, constant link cost metric is currently used in Internet by min-hop routing, while exponential link metric is promising with theoretical guarantee in performance for on-line routing algorithm. The performance of these two metric is studied in this paper. These two metric are detailed as follows.

constant — The cost for each link is a constant regardless of the currently available bandwidth on the link. This corresponds naturally to minimum hop routing in Internet.

exponential — Link cost is an exponential function of the remaining bandwidth on the link. This cost function is justified by both queueing theory suggestion and opportunity cost functions [4].

4.2 Aggregation methods

Aggregation methods play an important role in network performance. Different aggregation methods incur different combination of computation burden and representation size. This would affect the overall overhead for managing network routing information.

Aggregation schemes studied include Complete (full information is provided), Stars (Average and Diameter), Spanning Tree (Minimum spanning tree or MST, and Random spanning trees or RST), and graph spanner.

Complete — No aggregation is done. The full cost matrix between border nodes is used.

DIA — A star aggregation where the cost of the star radius is half the cost of network diameter.

AVE — A star aggregation where the cost of the star radius is half the average cost of the distances between border nodes.

MST — A minimum spanning tree representation.

RST — A random spanning tree.

Spanner — a t-spanner.

Scheme Complete is used as a basis for comparing performance loss due to aggregation. Due to the online nature of the problem, in some cases, especially those with low link delay, Complete exhibits slightly worse performance than other methods. This is because previously made "optimal" decision could turn out to be bad for future connection requests.

A separate study is conducted on the performance of different aggregation schemes [4]. It is shown that Complete and Minimum Spanning Tree provide good performance almost always among these schemes, while spanner can achieve good performance though with high aggregation computation expenses.

4.3 Network dynamics

Of the many other factors affecting performances besides the link cost functions and aggregation schemes mentioned in the above sections, we are most interested in the following:

- Hierarchical network structure: various hierarchies of network structure, even for the same connectivity graph of lowest-level nodes, could lead to much different network performance. Network hierarchical structure is the main focus in this paper;
- Link delay: link delay is normalized to the connection average holding time. Link reservation process becomes asynchronous due to link delay. Some link bandwidth could be held by some connection request which would eventually fail to use it, yet other requests are prevented from using the bandwidth as it is held by this connection request;
- Network load: traffic load would lead to different network performances such as throughput;
- Link capacity: link capacity affects the chance whether a request can be honored.

4.4 Performance measures

Two measures are used to compare the performance of the algorithms:

Throughput measured by the cumulative length of the realized connections, and

Control overhead measured by the average number of backtracks per connection.

Many parameters and environment attributes can affect the performance. Aggregation methods as described in 4.2 could lead to much difference performance, due to the fact that aggregated information maintained at various levels by hierarchical routing could differ much for various aggregation schemes. However, other parameters might enhance or suppress the differences among the aggregation schemes. The link metric used by the routing algorithm in calculating shortest path is studied, as the minimum hop metric and the exponential cost function (described in section 4.1) is compared. Other parameters under which the hierarchical routing and global routing were examined are link delay, network load, and link capacities.

An observation is that, when backtracks are used in both link cost metrics, the entire network bandwidth is up to consumption regardless of the aggregation method in use. The result is in many cases, throughput is expected to be very similar for various aggregation methods with differences only due to interaction between concurrent reservations. In some cases, larger difference may occur due to bad route selection and incomplete aggregated information.

To gain insight on how different parameters (especially aggregation schemes) affect network performance, we examine topologies which emphasize the differences between parameters under examination. Usually this is carried out on graphs of regular pattern, though the "degree" of regularity differs much among these graphs. The study on these graphs would provide a guide for further investigation on other graphs. In this study, graphs under investigation have the same lowest level connectivity. Different hierarchical structures are imposed to induce different hierarchical levels.

5 Simulation results

In this study, we compare the performance of global routing strategy and hierarchical routing strategy. We consider same physical topology, with induced hierarchy added on for hierarchical routing study.

Theory suggests that exponential cost metric is an optimal policy to ensure that an on-line algorithm is competitive with its optimal off-line counterpart [3]. As a results we selected this metric for our simulations. In section 5.2 we report simulation results that confirm this theoretical result. Our study on other topologies also verifies the superiority of the exponential cost metric for hierarchical routing [4].

We first introduce the network structures under study.

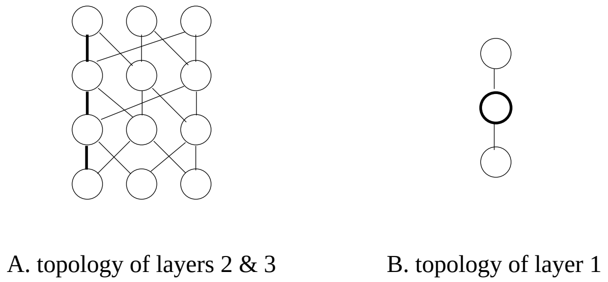

Figure 1. Self-similar Hierarchical structure. The middle node in layer 1 (right hand side) is structured as depicted on the left. In layer 2 each of the 12 nodes is structured recursively the same as a shuffle-exchange. Link capacities: 5 for internal links in layers 2 and 3, 10 for the bold internal links in layers 2 and 3, 10 for links between border nodes, 15 for links between border nodes that both have bold links.

5.1 Topologies under study

We study topologies with certain patterns. That is, we concern about both connection pattern at the lowest level, and hierarchical structures imposed on such lowest level connectivity. These rather "regular" topologies allow us to study on the effect of some individual network parameters.

Graph properties of particular interest in our study are number of nodes, number of edges, average degree of nodes, border nodes (for hierarchy), inter-connection pattern between domains (for hierarchy). Network structures are generated according to the specifications for these values.

In the following sections, a particular topology called *self-similar hierarchical* (SSH) is studied. This topology contains several levels, with each level as a switching network (shuffle-exchange). Figure 1 illustrates this topology.

5.2 Effect of link cost metrics

Consider the performance of exponential cost metric against that of constant metric. Figure 2 shows the ratio of throughput between exponential and constant metric, and Figure 3 shows the ratio of crankbacks between the two metrics. We observe that exponential metric achieves more than 50% (sometimes more than 100%) gain in network throughput against constant metric, while suffers less than 20% of crankbacks (and less than 10% for the promising aggregation schemes, i.e., everything but stars).

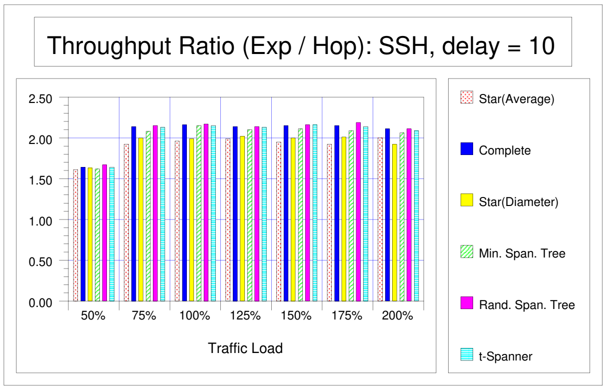

Figure 2. Ratio of throughput between exponential and constant cost metric for topology in figure 1

This shows that exponential cost metric performs much better than constant cost metric. As a result, we use exponential link cost metric for the rest studies.

In the following, we study the effect of imposing hierarchies on the routing performance. We focus on the hierarchical routing performance for different hierarchies that are imposed on same topologies. That is, we study how different hierarchies would affect the routing performance for the same physical network.

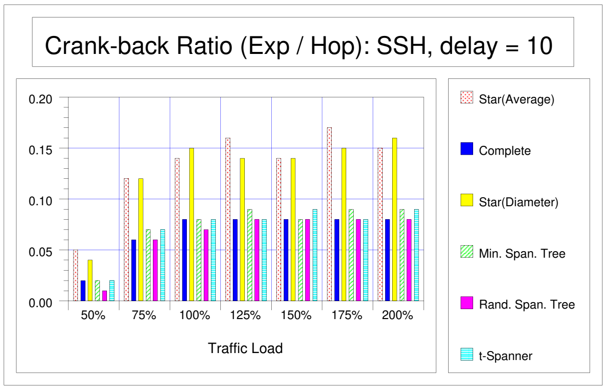

Figure 3. Ratio of crankbacks between exponential and constant cost metric for topology in figure 1

5.3 Effect of hierarchies

SSH topology contains 2 levels of imposed nodes. We study the effect of SSH topology with 1 level of imposed nodes and no imposed levels (global routing).

Consider SSH hierarchy with 2 levels and SSH hierarchy with no levels (physical connectivity). Figure 4 shows the ratio of throughput between 2-level hierarchy and 0-level hierarchy, and Figure 5 shows the ratio of crankbacks between the two hierarchies.

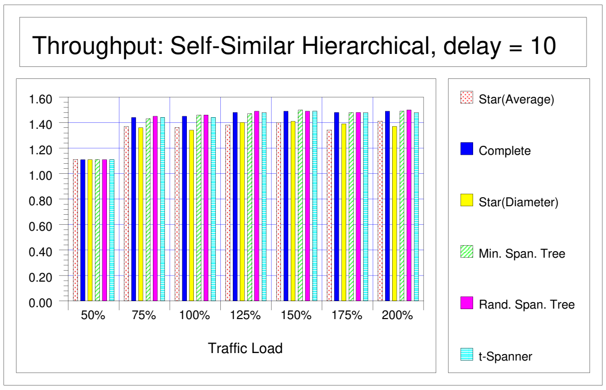

Figure 4. Ratio of throughputs between SSH with 2-level hierarchy and SSH with no hierarchy (global routing). SSH topology is in figure 1

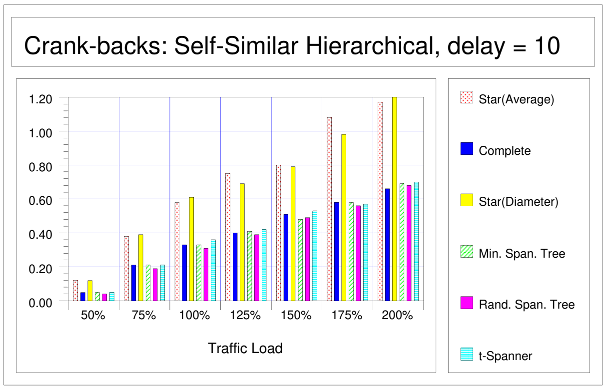

Figure 5. Ratio of crankbacks between SSH with 2-level hierarchy and SSH with no hierarchy (global routing). SSH topology is in figure 1

From the figures, we can see that 2-level hierarchy performs much better than 0-level hierarchy in terms of network throughput. For most of the situations studies, 2-level hierarchy would produce more than 40% throughput. Only for very low loads, where most of the requests for connection establishment are granted the difference is smaller, but is still favorable to the 2-level hierarchy by about 10%.

For crankbacks, not considering the star aggregation, 2-level hierarchy performs better than 0-level hierarchy. For most of the situations, 2-level hierarchy incurs less than 70% of crankbacks compared to 0-level hierarchy. The difference between the two increases as the load decrease, and for very low loads (50%) 2-level hierarchy incurs less than 10% of the crankbacks incurs by the 0-level hierarchy. For the star aggregation schemes, that ere shown to be unsuccessful in general [4], and for high loads 2-level hierarchy is shown to perform worse than 0-level hierarchy.

Therefore, for some networks such as SSH, imposing hierarchies would have better performance in both throughput and signaling delay. A good hierarchy design could guide hierarchical routing process with useful information, leading to improved performances.

Now consider imposing different hierarchies on the same SSH topology. The 2-level hierarchy we used above is compared against a 1-level hierarchy were the high level aggregation is not done, i.e., the complex node of Figure 1-B is replaced with its internal structure (that appears in Figure 1-A. Figure 6 shows the ratio of throughput between 2-level hierarchy and 1-level hierarchy, and Figure 7 shows the ratio of crankbacks between these two hierarchies.

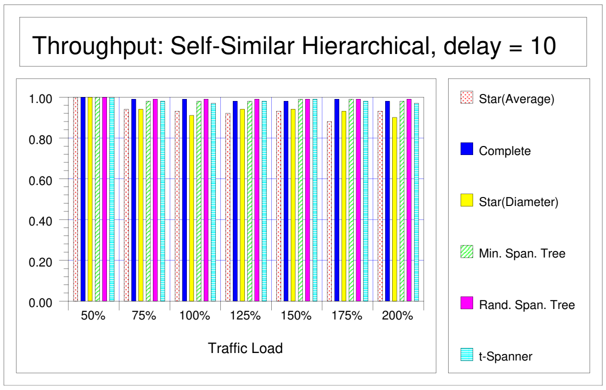

Figure 6. Ratio of throughputs between SSH with 2-level hierarchy and SSH with 1-level hierarchy. SSH topology is in figure 1

Figure 6 shows that, 2-level hierarchy has almost the same throughput as 1-level hierarchy. Actually the throughput for 2-level hierarchy is slightly less than that of 1-level hierarchy.

On the other hand, Figure 7 shows that, 2-level hierarchy performs worse than 1-level hierarchy in terms of crankbacks. Most of situations 2-level hierarchy incurs 50% more crankbacks than 1-level hierarchy, though

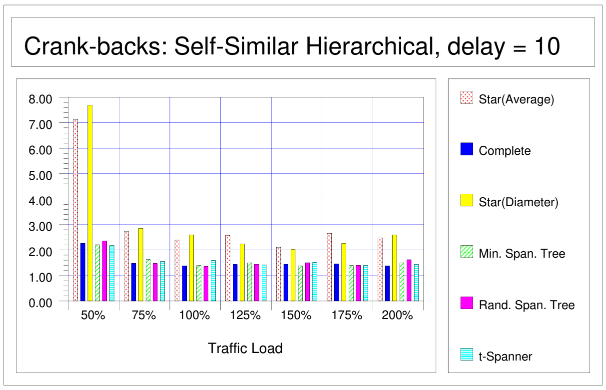

Figure 7. Ratio of crankbacks between SSH with 2-level hierarchy and SSH with 1-level hierarchy. SSH topology is in figure 1

there exists situations where 2-level hierarchy would incur 100% more crankbacks.

This study shows that, different hierarchies would lead to drastically different routing performances, even though these hierarchies are imposed on the same physical topologies. This justifies the necessity to study designing hierarchies for networks to achieve good performances.

6 Conclusion

In this paper, we study the effectiveness for ATM hierarchical routing performance. We study the hierarchical routing performance for networks with different hierarchy embedding. In particular, we have compared the performance of global routing strategy and hierarchical routing strategy, in terms of network throughput and signaling delay for realized connections. In this study, we focus on the performance of various link cost metrics and aggregation schemes.

Our study shows that, in general hierarchical routing performs well when compared to global routing strategy while utilizing less storage and communication overhead with some more computation overhead. For certain networks and topologies, hierarchical routing even performs better than global routing. Different hierarchies imposed on same topologies could have very different performance implications. This suggests the necessity on studying the hierarchy design for communication networks. Another finding is that link cost metric plays important role in routing performances. Our study shows that exponential link cost metric used in hierarchical routing provides an effective routing solution for large networks.

References

[1] Private network-network interface specification version 1.0 (PNNI). Technical report, The ATM Forum technical committee, Mar. 1996. af-pnni-0055.000.

[2] I. Althofer, G. Das, D. Dopkin, D. Joseph, and J. Soares. On sparse spanners of weighted graphs. *Discrete and Computational Geometry*, 9:81 – 100, 1993.

[3] B. Awerbuch, Y. Azar, and S. Plotkin. Throughput-competitive on-line routing. In *34th Annual IEEE Symposium on Foundations of Computer Science*, pages 32 – 40, Oct. 1993.

[4] B. Awerbuch, Y. Du, B. Khan, and Y. Shavitt. Routing through networks with hierarchical topology aggregation. *Journal of High-Speed Networks*, 7(1), 1998.

[5] B. Awerbuch, Y. Du, and Y. Shavitt. A simulation environment for aggregation based hierarchical topolgy aggregation. In *SCS/IEEE SPECTS'98*, July 1998.

[6] Y. Bartal. Probabilistic approximation of metric space and its algorithmic applications. In *37th Annual IEEE Symposium on Foundations of Computer Science*, Oct. 1996.

[7] D. Bertsekas and R. Gallager. *Data Networks*. Prentice Hall, second edition, 1992.

[8] I. Castineyra, J. N. Chiappa, and M. Steenstrup. The nimrod routing architecture, Feb. 1996. Internet Draft, Nimrod Working Group.

[9] C. Hedrick. Routing Information Protocol, June 1988. Internet RFC 1058.

[10] G. S. Malkin and M. E. Steenstrup. Distance-vector routing. In M. E. Steenstrup, editor, *Routing in Communications Networks*, pages 83 – 98. Prentice Hall, 1995.

[11] J. Moy. OSPF Version 2, Mar. 1994. Internet RFC 1584.

[12] J. Moy. Link-state routing. In M. E. Steenstrup, editor, *Routing in Communications Networks*, pages 135 – 157. Prentice Hall, 1995.

[13] D. Peleg and A. A. Schäffer. Graph spanners. *Journal of Graph Theory*, 13(1):99 – 116, 1989.