PNNI And The Optimal Design Of High-Speed ATM Networks

Abdella Battou
Center for Computational Science,
U.S. Naval Research Laboratory. Washington D.C.
AND
Bilal Khan, Sean Mountcastle,
ITT Industries Systems Division,
at the Center for Computational Science,
U.S. Naval Research Laboratory. Washington D.C.
E-mail: {battou,bilal,mountcas}@cmf.nrl.navy.mil

Keywords: ATM Network Design, PNNI Simulation, PRouST

Edited by: Mohsen Guizani
Received: January 15, 1999 **Revised:** July 31, 1999 **Accepted:** August 25, 1999

In addition to being well-dimensioned and cost-effective, a high-speed ATM network must pass some performance and robustness tests. We propose an approach to ATM network topology design that is driven by the performance of its routing protocol, PNNI. Towards this end, we define performance indicators based on the time and traffic required for the protocol to first enter and subsequently return to the meta-stable state of global synchrony, *in which switch views are in concordance with physical reality. We argue that the benefits of high call admission rate and low setup latency are guaranteed by our indicators. We use the PNNI Routing and Simulation Toolkit (PRouST), to conduct simulations of PNNI networks, with the aim of discovering how topological characteristics such as the diameter, representation size, and geodesic structure of a network affect its performance.*

1 Introduction

The size of operational ATM clouds continues to grow at an increasing pace. Both in anticipation of this changing scale, and to insure smooth inter-operation of these networks, the ATM Private Network-Network Interface standard (PNNI) was recently adopted (ATM Forum 1996). PNNI defines a set of protocols for *hierarchical* networks of ATM switches, and is designed to provide efficient and effective routing. In the long term, however, the degree to which PNNI succeeds in this regard will depend crucially on two factors:

First, because PNNI does not mandate specific policies for call admission, route selection, or topology aggregation, these aspects of the protocol remain "implementation-specific". Clearly the degree to which PNNI meets the challenges posed by tomorrow's ATM networks will depend significantly on the success of *switch designers* in devising effective algorithms for the admission and routing of connections, and for the aggregation of topology information.

Second, *network designers* must have the tools and information necessary to design ATM network topologies that are (i) capable of meeting anticipated traffic demands, and (ii) *optimized for performance under PNNI.* In this paper, we shall not address the first of these two issues, that of dimensioning networks to satisfy known costs and traffic demands. Our investigations begin at the point where a network designer, having been given projected traffic profiles and switch/fiber specifications, has arrived at a set of candidate ATM network topologies which appear equally adequate. We argue that although two topologies may appear indistinguishable in terms of the mathematics of QoS requirements, the PNNI protocol exhibits significant differentiation in their performance. Understanding how the PNNI protocol affects network performance is a necessary first step to determining how the adoption of PNNI *should* affect ATM network design. In subsequent sections, we shall describe our simulation experiments and begin developing guidelines for ATM network topology design that take into consideration the specific nature of the PNNI protocol.

2 PNNI Performance Indicators

There are many candidate performance criteria for evaluating the relative merits of network topologies. Here we shall assume that topology design is motivated by increasing the ATM network's call admission rate and decreasing the average connection setup latency. Additionally, we desire that the background traffic due to the PNNI protocol itself, should not be excessively high.

Setup Latency. In the absence of crankback, setup latency within a peergroup is seen to be linearly correlated with the number of hops in the selected path (Niehaus et

al. 1997) and thus may be estimated in the worst case by the network diameter. When crankbacks occur, each failed attempt at valid route selection contributes significant additional latency, required for backtracking to the entry border node, computing an alternate route and then re-traversing the peergroup along the new path. Reduction of setup latency thus requires minimizing the crankback frequency.

Crankbacks and Call Admission Rate. Recent results on PNNI aggregation schemes [2,4] indicate that ATM call admission rate and crankback frequency is directly proportional to the "distortion" present in switches' views of the network. In particular, the experimental data presented in (Awerbuch et al. 1998) confirms the intuition that when a switch has inaccurate (e.g. outdated) views of network topology and metric information, this increases the likelihood that calls entering the peergroup at that switch will be assigned sub-optimal routes. A larger discrepancy between a switch's local information and the underlying reality of the network's state, results in a larger fraction of calls originating at the switch being rejected en-route, hence undergoing crankbacks (and possibly even unwarranted rejection at the source).

Thus, beyond the problem of dimensioning, selecting topologies that will yield high ATM call admission rates and low average setup latency requires that one be able to characterize which topologies minimize the divergence in switches' views.

2.1 Our Approach

Local synchrony. We define *local synchrony* of a peergroup to be the state where every switch in the peergroup has knowledge of the same set of PNNI Topology State Elements (PTSEs). It follows from the logic of the PNNI NodePeer finite state machine, that if a peergroup reaches local synchrony, then all member switches agree about the topology metrics describing their peergroup. Within a PNNI peergroup, each member switch is responsible for originating and flooding accurate information about its internal state and the resource availabilities on its incident links. Thus, modulo any loss due to aggregation schemes, local synchrony may be interpreted as a state in which all members of the peergroup are in agreement not only amongst themselves, but also with the underlying reality of the peergroup's state.

Global synchrony. We define *global synchrony* of a connected ATM network to be the state where every peergroup at every level has reached local synchrony, and the PNNI network hierarchy has reached a unique apex. Admittedly, the notion of global synchrony is "artificial" in the sense that it may be rarely achieved in real dynamic networks where connections are constantly arriving and departing. However, in a simulated network this state is attainable, and we shall use it to probe the rate of PNNI information propagation.

When a switched virtual circuit (SVC) is established in an ATM network, bandwidth availability is altered for links that the SVC traverses. Assuming this change is significant, updated information is re-originated and flooded by each switch incident to the affected links. If the network had reached global synchrony prior to the SVC setup request, these re-originations cause the network to fall out of a state of global synchrony for a brief time, until the new information has reached every node. This naturally leads us to consider:

- **Resynchronization time:** Average time required for the network
  to return to global synchrony, after a single, isolated, random SVC setup.
- **Resynchronization traffic:** Average PNNI traffic required for the network
  to return to global synchrony, after a single, isolated, random SVC setup.

The basic "trial" involved in measuring the above **resynchronization parameters** is as follows: allow the network to reach global synchrony, then inject a connection request between randomly chosen source and destination nodes and measure the time required and bytes transmitted before the network returns to a state of global synchrony. By repeating this trial a large number of times, we obtain an average value, which we call the resynchronization time.

To illustrate the importance of resynchronization time, let us consider two extreme scenarios. First, consider a network where the average time between SVC requests (i.e. 1/ SVC arrival rate) is much higher than the network resynchronization time. In this situation, changes in bandwidth availability induced by an SVC setup will, on average, have time to flood to all other switches in the network prior to the arrival of the next SVC setup request. Thus, routing decisions for each SVC will, on average, be made according to completely accurate information at the originating switch. If an SVC setup is rejected or experiences unacceptably high latency, this undesirable behavior is attributable solely to inadequate dimensioning of the network, because there is no legitimate way to fulfill the request.

In contrast, consider a network where the average time between SVC requests is much lower than the network resynchronization time. In this situation, changes in bandwidth availability induced by an SVC setup have not yet propagated to many switches in the network by the time the next SVC setup request arrives. Thus, the routing decision for an SVC is likely to be made according to stale information. The extent to which the information is stale is determined by the extent to which network resynchronization time exceeds average inter-SVC arrival time. If an SVC setup is rejected or experiences unacceptably high latency, this undesirable behavior may be due to inadequate dimensioning of the network or it may be due to suboptimal routes and unwarranted rejections induced by inaccurate information at the switches.

Major changes in network topology, such as network partitioning due to link/node failure, or re-merging of components upon subsequent recovery, will cause the PNNI

hierarchy to undergo severe restructuring. We define the **boot parameters** below to be indicators of the time and traffic required to reinstate consistent routing information after such catastrophic changes.

- **Boot time:** Time at which the network first reaches global synchrony.
- **Boot traffic:** PNNI traffic required for the network to reach
  global synchrony for the first time.

We take the parameters above as a worst-case estimate of the time and traffic resources required to return to global synchrony. By comparison, the resynchronization parameters described earlier aim to measure the same quantities for the average case, i.e. during normal (stable) operation of the network.

We contend that a network designer, given two topologies that are equivalent with regards to meeting anticipated QoS requirements, must take into consideration their resynchronization and boot times. In particular, reducing these two quantities increases the fraction of time that the network spends in a state of global synchrony. At the same time, the designer must keep a watchful eye on the Boot traffic and Resynchronization traffic to make sure that not too much of the network bandwidth is being expended by PNNI itself.

One way in which the designer might determine the values of the four parameters mentioned above is to take physical measurements of them on two live networks, each configured in the appropriate topology. But this would be difficult to do accurately because of the inherent problems in distributed measurement, and moreover it would completely defeat the intention of *design before implementation!* Alternately, one could simulate the candidate PNNI networks to determine both time and traffic for boot and resynchronization; we follow this latter approach here.

3 Experiments

3.1 The Simulation Environment

Our simulation experiments were carried out using the PNNI Routing and Simulation Toolkit (PRouST), which was developed by the Signaling Group at the Naval Research Laboratory's Center for Computational Sciences. PRouST is a faithful and complete implementation of version 1.0 of the PNNI standard and can be used both to simulate large networks of ATM switches as well as to emulate live ATM switch stacks. In particular, PRouST includes the Hello, NodePeer, and Election finite state machines, all relevant packet encoding and decoding libraries, routing database management, and full support for hierarchy. In addition, a "plug-in" interface for call admission, path selection and aggregation policies is provided. For inter-switch signaling PRouST makes use of the NRL Signaling Entity for ATM Networks (SEAN), which is a complete simulation/emulation library implementing version 4.0 of the ATM User-Network-Interface (UNI) standard. The fidelity of PRouST's PNNI implementation has been demonstrated extensively in live interoperability tests with commercial switches. Both PRouST and SEAN are written in C++ over the Component Architecture for Simulation of Network Objects (CASiNO) described in (Mountcastle et al. 1999), and both are available in the public domain.

In the simulations that follow, all network links operate at the OC3 rate. Link jitter varies uniformly between $+10$ μs and -10 μs for each transmitted message. PNNI messages that enter the switch control port experience a latency of 10 ms. The backplane of the switch routes data traffic on existing virtual circuits at *OC48* rates. These figures, while artificial, are projections of current switch vendor specifications. We found that jitter did not noticeably alter the outcome of our simulations from one trial to the next. The variations were typically less than 1% and for boot and resynchronization time, and less than 5% for boot and resynchronization traffic. The values we have listed in our tables are mean values.

An outline of results presented: We seek to understand what factors influence resynchronization and boot parameters. To this end, we start by simulating single-peergroup networks with grid, chain, ring and star topologies; these results are presented in sections 3.2 and 3.3. We compare these very particular families of topologies with similar experiments using all possible topologies on 7 nodes–these results are described in sections 3.4-3.5. In addition, we simulate 100 randomly generated topologies on 20 nodes, the results of which are described in section 3.6. Finally, in sections 3.7 and 3.8 we address the impact of peergroup size and hierarchy, by simulating several different hierarchic configurations of linear networks.

3.2 Boot and Resynchronization Time

We start by investigating the characteristics of network topology that influence boot and resynchronization times. The NodePeer protocol floods PNNI Topology State Elements over a link whenever the switches incident to the link have discrepant databases. Thus, information will flow from each switch radially until it has reached every other switch in the peergroup. Given this, network boot time and worst-case resynchronization times should be linearly dependent on the diameter of the peergroup.

Simulation Results We simulated PNNI networks of increasing size, specifically, chains, grids rings, and star networks. The tables (1,2,3,4) show the results of the simulations, and are depicted in figures 1 and 2. The figures indicate that PNNI boot time grows linearly in network diameter; for each unit increase in diameter PNNI boot time increases by approximately 21.4 ms, while resynchronization times increase by 10.7 ms.

Chain Length	Network Diameter	Boot Time seconds	Resynch. Time seconds
2	1	30.0864	0.0108
4	3	30.1290	0.0323
6	5	30.1719	0.0537
8	7	30.2146	0.0755
10	9	30.2588	0.0962
20	19	30.4562	0.2035
30	29	30.6859	0.3104

Table 1: Chains—Boot/resynch. times

Grid Size	Network Diameter	Boot Time seconds	Resynch. Time seconds
2x2	2	30.1073	0.0212
3x3	4	30.1498	0.0428
4x4	6	30.1926	0.0636
6x6	10	30.2785	0.1067

Table 2: Grids—Boot/resynch. times

3.3 Boot and Resynchronization Traffic

To study PNNI boot and resynchronization traffic, we will introduce the notion of the representation size of a network: the minimum number of PNNI topology state elements required to fully describe its topology. In our subsequent discussion, we take the representation size of a network to be twice the number of links plus the number of switches.

Because the NodePeer protocol is responsible for transmission of the peergroup's current representation to each component switch, traffic required to reach *initial* global synchrony should be bounded below by a function proportional to the product of the representation size and the number of switches. For chains, grids and rings the number of nodes and the representation size are linearly related, so this product is quadratic in the representation size.

Resynchronization involves flooding updated information about links affected by the SVC, to all members of the peergroup. In the best-case, flooding takes place over a spanning tree and the total traffic required to resynchronize is proportional to the number of switches in the peergroup. In the worst case, when flooding is occurs over every edge in the network, traffic due to resynchronization is proportional to the number of links in the network. For this reason, we consider resynchronization traffic in PNNI networks as a function of representation size.

Ring Length	Network Diameter	Boot Time seconds	Resynch. Time seconds
10	5	30.1654	0.0518
20	10	30.2694	0.1037
30	15	30.3732	0.1555
40	20	30.4932	0.2172
50	25	30.6002	0.2688

Table 3: Rings—Boot/resynch. times

Star Size	Network Diameter	Boot Time seconds	Resynch. Time seconds
5	2	30.0872	0.02172
10	2	30.0886	0.02179
15	2	30.0894	0.02188
20	2	30.0896	0.02191
25	2	30.0909	0.02198

Table 4: Stars—Boot/resynch. times

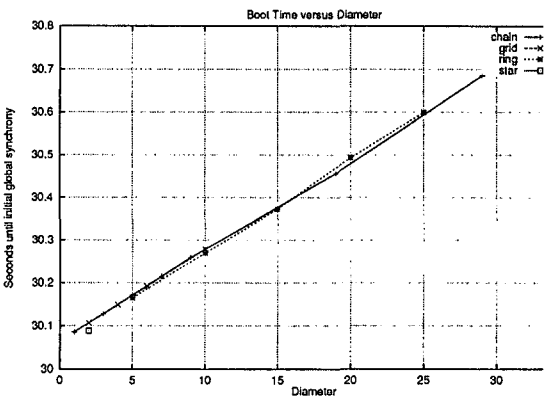

Figure 1: Boot time vs. network diameter

Simulation Results We interpret the traffic data collected from the simulations of the previous section. This data is shown in tables (5,6,7,8). The traffic required to reach initial global synchrony in each of these cases, grows super-linearly with the representation size, as can be seen in figures 3 and 4). Resynchronization traffic also manifests

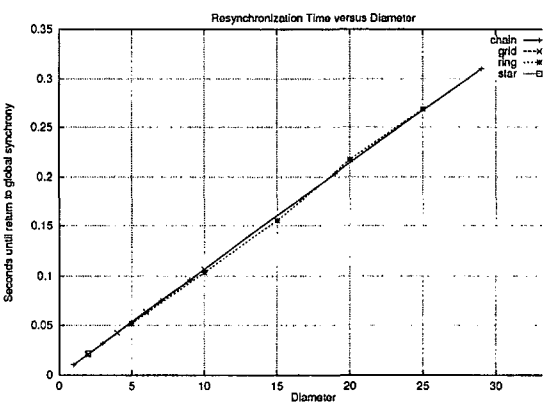

Figure 2: Resynch. times vs. network diameter

super-linear growth as a function of representation size.

Chain Length	Repres. Size	Boot Traffic bytes	Resynch. Traffic bytes
2	4	10292	2967
4	10	56028	13399
6	16	133976	30127
8	22	248124	53247
10	28	394476	82575
20	58	1629364	324487
30	88	3702444	719548

Table 5: Chains—Boot/resynch. traffic

Grid Size	Repres. Size	Boot Traffic bytes	Resynch. Traffic bytes
2x2	12	73284	14412
3x3	33	182232	43240
4x4	64	1674880	260640
6x6	156	3521512	816003

Table 6: Grids—Boot/resynch. traffic

Ring Length	Repres. Size	Boot Traffic bytes	Resynch. Traffic bytes
10	30	201812	2720
20	60	765388	5440
30	90	1379620	108160
40	120	2167876	816680
50	150	3353088	1286000

Table 7: Rings—Boot/resynch. traffic

3.4 General Tests I: All 7 Node Networks

In order confirm the validity of the above conclusions, we conducted the same experiments on the class of all (853 topologically distinct) 7 node connected graphs[1].

Simulation Results Because the 7 node networks all have relatively small diameter, there was little differentiation in their boot and resynchronization times. As the graphs in figures 5 and 6 indicate, boot and resynchronization times were clustered at discrete values spaced 10ms apart. We note that 10ms is the time required for processing of a single PNNI message at the control port of a switch.

[1]The software used to generate all non-isomorphic 7 node graphs was the NAUTY program developed by Brendan McKay.

Star Size	Repres. Size	Boot Traffic bytes	Resynch. Traffic bytes
5	16	30864	9144
10	31	129024	43976
15	46	292724	104776
20	61	526464	191624
25	76	820152	302496

Table 8: Stars—Boot/resynch. traffic

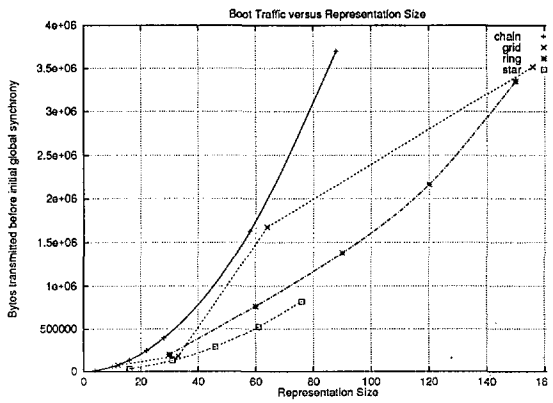

Figure 3: Boot traffic vs. representation size

While the plot does not immediately illustrate this, the distribution of the data points was not uniform over the cluster points; we have plotted the average as a function of diameter to emphasize this. Somewhat surprisingly, on average, resynchronization time seems to grows super-linearly with network diameter. More simulations need to be conducted over a larger class of graphs to determine the cause of this phenomenon. The PNNI boot traffic for these simulations is shown in figure 7, and is linear with an approximate growth rate of 6000 bytes per unit of representation.

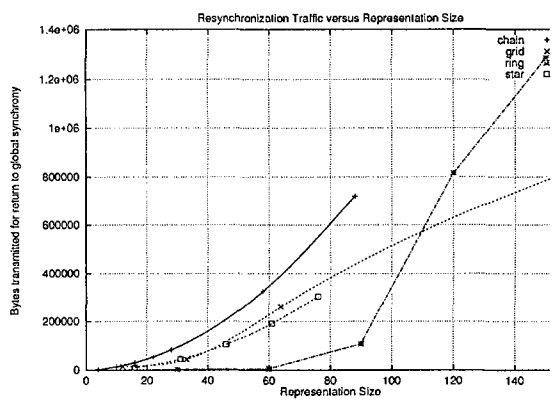

Figure 4: Resynch. traffic vs. representation size

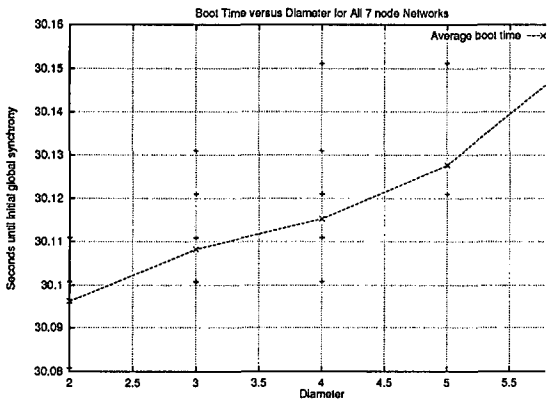

Figure 5: Boot times for 7 node networks

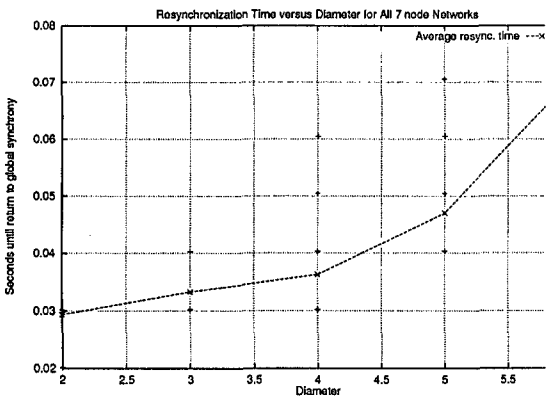

Figure 6: Resynch. times for 7 node networks

3.5 Network Geodesic Structure

The plot shown in figure 8 indicates that that over 7 node graphs of any fixed representation size, there is considerable variation in PNNI resynchronization traffic. We attempted to understand the cause for this differentiation by determining, for each representation size, which topologies achieved the highest and lowest resynchronization traffic. Figure 9 is a partial list of the best and worst performers (only those with representation sizes between 19 and 33 are shown).

We noted that the worst performers had a large number of redundant geodesics between pairs of switches. That is to say, topologies with the worst performance contained a multiplicity of shortest paths between nodes. Scrutiny of the simulations revealed that this multiplicity results in a redundant flooding in the PNNI NodePeer protocol. Figure 10 illustrates the redundant flooding of a PTSE along one edge when two nodes are connected by more than one shortest path. The fact that the worst performers in figure 9 contain many unchorded 4-cycles, whereas the best performers contained many triangles, supports this conclusion.

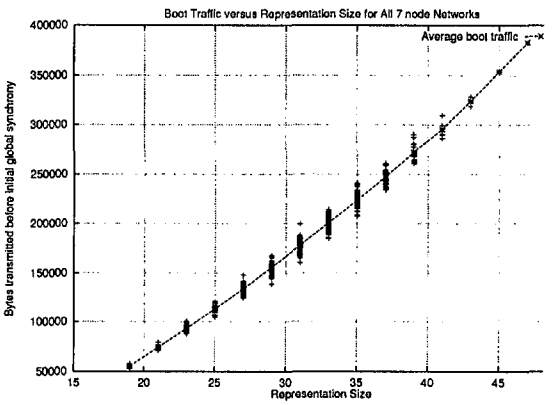

Figure 7: Boot traffic for 7 node networks

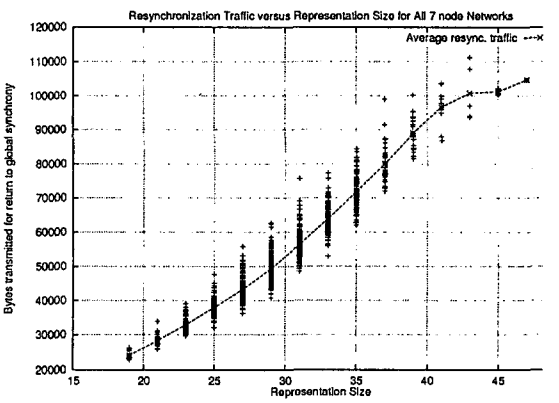

Figure 8: Resynch. traffic for 7 node networks

3.6 General Tests II: Randomly Generated 20 Node Networks

We also conducted measurements of boot and resynchronization parameters for randomly generated networks.

A random topology is generated as follows: 20 nodes are assigned random locations on a grid. Links are added via a random process that repeatedly generates a random node-pair and adds a link between them with probability that decays exponentially with the Euclidean distance between the two nodes (Waxman 1998). Links that will cause the degree of a node to exceed eight are rejected by the random process in order to keep the graph reasonably sparse. The process of adding links terminates when the graph is connected and every node has degree at least 2. In this manner, we generated 8000 random topologies on 20 nodes. These were then sorted into classes, based on the diameter of the generated network, and 10 networks were chosen randomly from each class.

Simulation Results The results of simulations of 100 random 20-node networks are shown in figures 11 and 12. The data indicates that for any given value of diameter, there is a significant variation in the resynchronization and boot times. We were not able to determine the topological structure property that is responsible for this variance, but

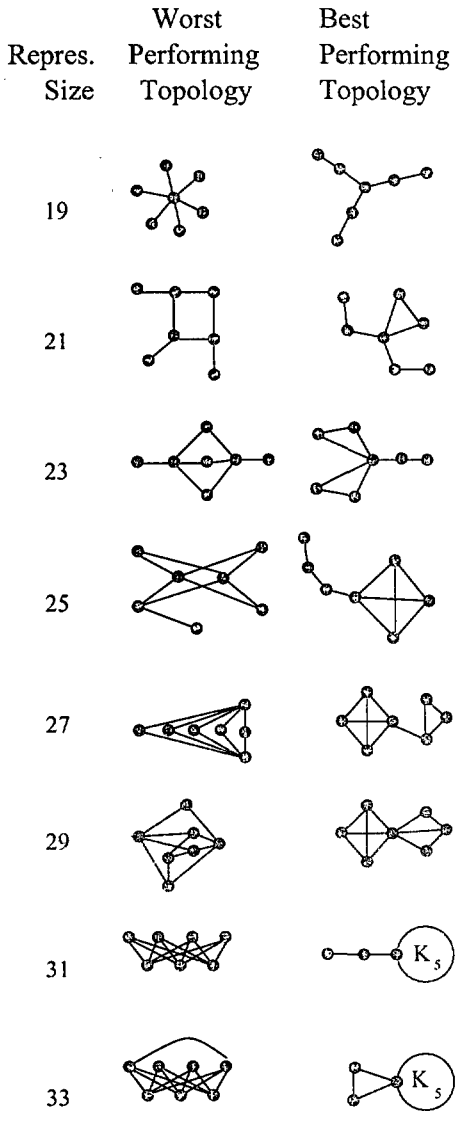

Figure 9: Best and worst 7 nodes topologies (in terms of boot traffic

hope to address the question in our future work. The results of previous sections indicate that if we are given two networks from the same restricted family (such as grids, for example) then diameter alone can serve to predict the approximate value of the resynchronization time. In the set of experiments presented in this section, we realize that, unfortunately, this simple estimation criterion does not hold as well over large mixed families of graphs (e.g. our random set). On the other hand, we remark that the average value of resynchronization and boot times (over the randomly chosen topologies) does increase linearly with diameter.

3.7 Peergroup Size

Another aspect of network design we consider is choice of peergroup size.

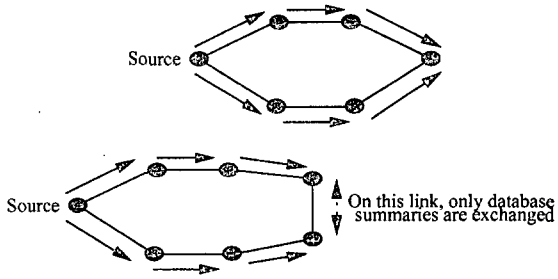

Figure 10: Effects of geodesic structure on traffic

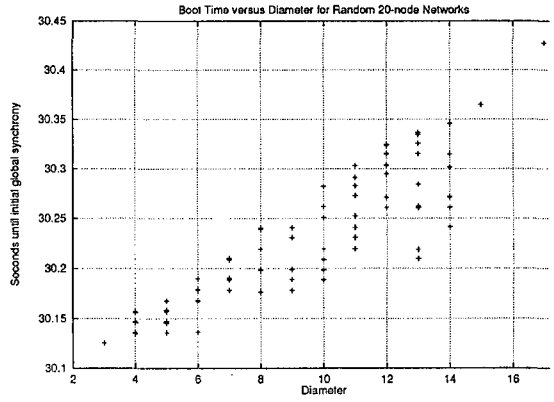

Figure 11: Boot times for 100 random 20-node networks

First, we note that partitioning a network into peergroups will result in more SVCs needing to be established between leaders at the next higher level. It also necessitates logical Hello finite state machines to stabilize over these SVCs and for the election process to conclude at the higher level. These two factors together are responsible for a discrete jump in the PNNI boot time when one moves from single peergroup to multiple peergroup configurations. As we begin to decrease the size of the peergroups, each requires less information to reach local synchrony, since detailed information about distant peergroups is being represented by a single logical node at the higher level. This causes boot traffic and resynchronization traffic to decrease. As we

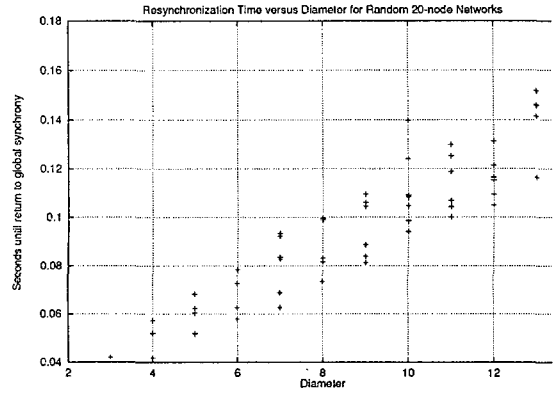

Figure 12: Resynch. times for 100 random 20-node networks

make peergroup size smaller still, an SVC setup on average traverses a larger number of peergroups, which in turn triggers re-aggregation of links at the higher level. Thus beyond a certain point, decreasing peergroup size causes an increase in resynchronization traffic and time.

Simulation Results We simulated chains of 16 and 32 switches, configured with peergroups of sizes 2,4 and 8 (the 16 node examples are illustrated in figures 13). The data from the simulations is presented in tables 9 and 10. In the 32 node chain, going from a single peergroup of 32 nodes to 4 peergroups of 8 nodes produces a jump in the PNNI boot time of over 59 seconds. On the other hand, boot traffic decreases by a factor of 3, and resynchronization traffic by a factor of 6.4. Reduction of the peergroup size from 4 to 2, causes an increase in resynchronization traffic by 30%. This happens because higher level links are re-aggregated and flooded downward in response to the change in resources at the lower level.

Topology name	16-16	16-4	16-2
# of PGs	1	4	8
PG size	16	4	2
Boot time	30.38s	90.26s	90.27s
Boot traffic	1034K	639K	587K
NNI boot traffic	0	1.5K	1.8K
Resync. traffic	0.30s	0.23s	0.24s
Resync. traffic	494K	156K	233K

Table 9: Varying peergroup size for a 16 node chain

Topology name	32-32	32-8	32-4	32-2
# of PGs	1	4	8	16
PG size	32	8	4	2
Boot time	30.73s	75.43s	75.39s	75.49s
Boot traffic	4217K	1401K	1121K	1420K
NNI boot traffic	0	3.1K	3.7K	3.9K
Resync. time	0.64s	0.46s	0.46s	0.51s
Resync. traffic	1981K	306K	689K	898K

Table 10: Varying peergroup size for a 32 node chain

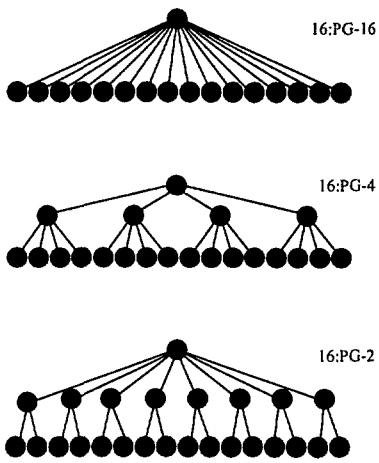

Figure 13: Examples of peergroups simulated.

3.8 Hierarchy

Finally, we consider how the presence of hierarchy affects our performance indicators.

First, we note that a network with many levels of hierarchy requires SVCs to be established between leaders in the same higher level peergroup. It also necessitates logical Hello finite state machines to stabilize over these SVCs and for the election process to conclude in each peergroup at each level. These two factors are the principal cause of the discrete jump seen in PNNI boot time and boot traffic as one considers networks with a greater number of levels. On the other hand, hierarchy localizes the side effects of network changes. In particular, it reduces the number of switches to which updated information must be flooded. Thus hierarchical addressing lowers both resynchronization time and traffic.

Simulation Results We simulated chains of 16 and 32 switches, configured with various hierarchical structure (The 16 node scenarios are depicted in figure 14). The data collected is presented in table 11. A 3-level configuration of the 32 switch chain boots in 75.4897 seconds, while the 5 level one requires 150.3220 seconds to reach initial global synchrony. The 5 level hierarchy has the advantage of improving both resynchronization time and traffic by a factor of 1.3 and 1.8 respectively.

4 Conclusion and Future Work

The simulations conducted using the PNNI Routing and Simulation Toolkit (PRouST) confirmed that topological characteristics such as the diameter, representation size, and geodesic structure do affect the boot and resynchronization times and traffic. These four indicators determine the discrepancy between switches' views of the network and its underlying physical state, and are thus critical to call admission rates and crankback frequency in ATM networks. Partitioning a network into peergroups reduces the

Topology name	16/3	16/5	32/3	32/5
Chain length	8	8	32	32
Hierarchy structure	16,8,1	16,8,4, 2,1	32,16,1	32,16, 8,4,1
Boot time	90.27s	120.32s	75.49s	150.32s
Boot traffic	587K	611K	1420K	1717K
NNI boot traffic	1.8K	3.1K	3.9K	6.8K
Resync. time	0.24s	0.16s	0.51s	0.38s
Resync. traffic	233K	215K	1191K	638K

Table 11: Simulation results for several hierarchy configurations.

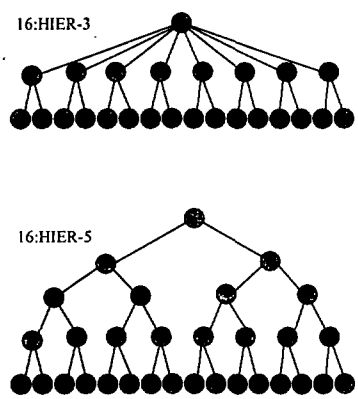

Figure 14: Two examples of hierarchic structures simulated.

boot and resynchronization traffic and introducing hierarchy results in improved resynchronization time and traffic, although it has a minor side-affect of increasing the boot time.

Peergroup size and hierarchy structure are two of the most importance choices a network designer must make. In our future research efforts will focus further on these two parameters. As we have shown in [section 3.7] there is an optimal value beyond which reduction of peergroup size results in increased resynchronization traffic, due to the re-aggregation and downward flooding of higher level links in response to changes at lower levels. We intend to precisely quantify both this optimal value, and the relative tradeoffs between peergroup size, hierarchy structure and resynchronization traffic and time.

5 Acknowledgements

We would like to thank David Talmage and Jack Marsh at the NRL Signaling Group for their contributions to PRouST, SEAN and CASiNO. Also we wish to acknowledge Sandeep Bhat for his assistance, particularly with the internals of the NodePeer finite state machine.

References

[1] ATM Forum. (1996) Private Network-Network Interface Specification, Version 1.0.

[2] Baruch Awerbuch, Yi Du, Bilal Khan, and Yuval Shavitt. (1998) Routing Through Networks with Hierarchical Topology Aggregation. *Journal of High Speed Networks*, vol. 7(1) p.57–73.

[3] W. C. Lee, Topology aggregation for hierarchical routing in ATM networks. *Computer Communications Review*, p. 82-92.

[4] S. Mountcastle, et al. (1999) CASiNO: A Component Architecture for Simulating Network Objects. *Proceedings of the 1999 Symposium on Performance Evaluation of Computer and Telecommunications Systems, July 11-15, 1999*. p. 261-272.

[5] D. Niehaus, et al. (1997) Performance Benchmarking of Signaling in ATM Networks. *IEEE Communications Magazine*, vol. 35 number 8, p. 134-142.

[6] Bernard M. Waxman. (1988) Routing of multipoint connections. *Journal on Selected Areas of Communications*, vol. 6 p. 1617-1622.