Hierarchical Routing With QoS

Wai Sum Lai
AT&T Labs
200 Laurel Avenue
Middletown, New Jersey, USA
Tel: +1-732-420-3712
wlai@att.com

ABSTRACT
We present our simulation-based design of hierarchical routing systems using the following state parameters to support quality of service (QoS): delay, hop count, and available bandwidth. Without revealing its internal structure, a cluster in a hierarchical system can advertise to other clusters its transit QoS characteristics in terms of the state parameters of either minimum-weight paths or widest paths between each of its border-node pairs. Neither of these paths reflects accurately what is actually available inside a cluster to meet user QoS requirements, potentially resulting in sub-optimal routing. Moreover, based on operational experience, we show that advertising widest paths may interfere with normal operational procedures in the addition or deletion of links. After analyzing the pros and cons of these paths, we propose the use of constrained minimum-weight paths for advertisement, to balance the need for smaller weights and sufficient available bandwidth.

Categories and Subject Descriptors
C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design – *Network communications, Packet-switching networks;* C.2.2 [**Computer-Communication Networks**]: Network Protocols – *Routing protocols;* C.2.3 [**Computer-Communication Networks**]: Network Operations – *Network monitoring;* I.6.3 [**Simulation and Modeling**]: applications

General Terms
Algorithms, Design, Performance

Keywords
Hierarchical routing, QoS routing, available bandwidth, widest path, minimum-weight path, path selection, topology aggregation, routing simulation

SimulationWorks 2010 March 15–19, Torremolinos, Malaga, Spain.

1. INTRODUCTION
Hierarchical routing is used in most large networks for routing to scale in size or for network administration to conceal proprietary topology information. In this paper, our focus is on routing scalability. In particular, we investigate the support for quality of service (QoS) in connection-oriented packet-switching networks with hierarchical routing. Examples of protocols running on such networks include Private Network-Network Interface (PNNI) [1] and Multiprotocol Label Switching (MPLS) [2].

In a non-hierarchical or *flat* network, the network topology database in all nodes is synchronized to provide for efficient and optimized routing. However, a flat network scales poorly. By using *topology aggregation*, a multi-level hierarchical network can scale to very large sizes. The trade-off for scalability is route optimization.

We report our operational experience in the design of hierarchical routing with QoS for the AT&T Global Network (AGN). Our study relies on using the simulation-based Capacity Management Tool (CMT). This tool incorporates the entire AGN topology, together with the details of hierarchical clustering structure and routing protocols with various algorithms and features.

Previous studies on hierarchical routing with support for QoS (e.g., [3]) have mainly been based on theoretical models. Our paper is pragmatic and application-oriented, with a discussion of the rationale behind the various design decisions for implementation in an operational, large network.

1.1 Contributions and Summary
In Section 2, we define state parameters and describe QoS routing in terms of weight and available bandwidth. Minimum-weight paths and widest paths optimize these parameters individually. As network state changes, state parameter updates are needed as presented in Section 3. Dampening and timer-based mechanisms are employed to control the update frequencies.

In Section 4 we discuss hierarchical routing and the use of topology aggregation in clustering. We show that advertising either minimum-weight paths or widest paths can cause some routing problems. In particular, from our experience in AGN, advertising widest paths may interfere with normal operational procedures in the addition or deletion of links.

In Section 5, as a solution for advertisement by a cluster, we propose the use of constrained minimum-weight paths with pruning based on carried load characteristics. This method of

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2010.8668
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2010.8668

path selection helps to balance the need for smaller weights and sufficient available bandwidth.

In the appendix, we provide an overview of CMT, our simulation tool. We have been using this tool extensively in our investigations of various routing issues in the design of AGN. As an example application, we describe the use of CMT in a case study of a particular network problem with routing anomalies under failure. The results of this simulation analysis shed us some light on how advertising widest paths by a cluster may interfere with operational procedures and cause undesirable traffic shifts. This impact is explained in Section 4.3.

2. QOS ROUTING

To support QoS routing, *state parameters* describing the characteristics of links and nodes are advertised. These state parameters can be classified as either metrics or attributes.

A *metric*, such as delay, is a state parameter whose effect is cumulative along a path. That is, the values of the metrics of all links and nodes along a given path are combined to determine whether the path meets user QoS requirements.

An *attribute*, such as bandwidth, is considered individually to determine whether a given link or node meets user requirements. Thus, if the attribute value associated with a topological element (either a link or a node) violates the QoS requirements, that element is eliminated from the path selection.

We study routing with three state parameters: delay, hop count, and available bandwidth. In the rest of this section, we consider link state parameters only; nodes are assumed to have zero delays and infinite bandwidth. Node state parameters are discussed later in the context of hierarchical routing (Section 4).

2.1 Minimum-Weight Path Routing

Both delay and hop count are additive metrics. We combine them into a single mixed metric as follows. Each link is assigned a *weight* that is the sum of two components:

- One-way light-load delay over the link, d
  This is essentially the propagation delay, assuming the use of high-speed links and fast nodal processors so that insertion, processing, and queueing delays are negligible.
- Penalty for the link, p
  This is a fixed value for all links.

The weight W of a link L is then $W(L) = d + p$. Our objective of optimal routing is to minimize cost subject to this weight function for link traversal. The relative magnitudes of d and p determine the importance attached to either delay or hop count in finding a path. Thus, for delay-based routing, a small value should be chosen for p.

The weight (or *cost*) of a path P, defined as the sum of the weights of all links along the path, is computed as $W(P) = D + h\,p$, where D is the path delay, i.e., the sum of the link delays, and h is the hop count of path P.

Let $\{P_k \mid k = 1, \ldots, n\}$ be the set of paths for a given source-destination pair with respective weights $W(P_k)$. A path in this set has *minimum weight* if the weights of all other paths in the set are at least as much.

As described in [4], minimum-weight path routing enables the selection of paths with low delay and few hops.

2.2 Widest Path Routing

To support different traffic classes for different classes of service, certain amount of bandwidth on a link is allocated to a specific traffic class. This is the *maximum bandwidth* for the class.

For a given class, the *available bandwidth* of a link is the residual bandwidth (out of the maximum for the class) on the link that is available for new traffic from the class.

For a given class, the available bandwidth of a path is the minimum of the available bandwidth of all links on the path for the class. In other words, it is the *bottleneck bandwidth*.

Let $B(.)$ denotes the available bandwidth for a class. If the path P_k between a source-destination pair have links $\{L_j \mid j = 1, \ldots, m_k\}$, then $B(P_k) = min\ B(L_j)$. The path in the set $\{P_k\}$ that has $max\ B(P_k)$ is referred to as a *widest path* for the given source-destination pair.

Widest path routing finds a path with the most available bandwidth, regardless of the number of hops traversed. When there are multiple widest paths, tie breaking is done by picking the one with the smallest weight.

2.3 Path State Parameters

From the above discussion, the QoS characteristics of a path for each class can be captured by a pair of state parameters: (path weight, available bandwidth of the path for the class).

When there are multiple QoS characteristics, it is generally not possible to pick a single path that simultaneously optimizes all of the state parameters. Given a source-destination pair, a minimum-weight path has the smallest weight among all the paths but not necessarily the most available bandwidth. A widest path has the most available bandwidth, but is usually more *circuitous* (i.e., has more delay and/or hops) than a minimum-weight path. This is because paths with smaller weights get filled up first by virtue of the tie-breaking rule, thereby leaving them with lower available bandwidth.

3. STATE PARAMETER UPDATES

As network state changes, changes in the state parameters need to be advertised so that the old values maintained in various nodes can be updated accordingly.

3.1 Available Bandwidth Updates

Available bandwidth is a dynamic attribute that varies according to the level of traffic traversing a link and the resulting residual link capacity available for additional traffic. In connection-oriented networks, available bandwidth is required to arbitrate whether a given link is suitable to carry a new connection. This arbitration is performed by the connection admission control mechanism.

When parameters frequently change, there is the potential for a network to be overwhelmed by the advertisements of updates. To reduce communication overhead, a *dampening* mechanism is employed to reduce update frequency by limiting advertisements below a set threshold. For example, changes in available bandwidth can be measured in terms of a proportional difference

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2010.8668
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2010.8668

from the last value advertised and are advertised only if they are significant. To avoid the possibility of long periods with only small changes without triggering updates, a timer-based mechanism is additionally used.

3.2 Weight Updates

While bandwidth information is rather dynamic, link weights are relatively static. Link weight changes are usually driven by link status changes.

For example, maintenance or failure of a link generally causes its weight to be set to a very large value. When installing a new link, during the *soak-in period* for line quality monitoring, link weight is also set to a high value to discourage traffic from using the link. Such weight assignment is referred to as *cost-out*. The cost-out weight is typically chosen to be several times larger than the weight of the expected longest path in the entire network. When the link is ready for service, it is then assigned the normal delay-based weight. This is *cost-in*.

Similarly, cost-out is used to prepare for the removal of a link from service to discourage new connections from getting on the link.

As a result of such operational practice, dampening is not applied to link weights; any change is considered significant.

3.3 Path Computation

Paths can be computed on demand based on the QoS requirements specified by an application requesting a connection. Each request initiates an instance of the process for path computation. When connection requests are frequent, it is possible for a network to be overloaded with such activities.

Alternatively, paths can be computed in the background based on the state parameters received prior to connection establishment time. Recomputation of these pre-computed paths is usually triggered when advertisements of significant changes are received from other nodes.

Due to time delays in updates, the actual network state tends to drift away from the last advertised values maintained by different nodes. So, it can happen that a connection arrives at a cluster only to find out that its available bandwidth has been depleted to the extent that the bandwidth requirement of the connection can no longer be met. The blocked connection has to be rolled back to try an alternative path toward the final destination, resulting in a longer setup time. This may also lower overall connection throughput.

Under normal operation, the need for crankbacks such as above can be minimized by using suitably designed updating mechanisms so that changes in state parameters can be distributed timely to different nodes for good path selection. A properly engineered network is also required to reduce blocking.

Under failure, it is possible that a large number of connections affected by the failure may need to be rerouted simultaneously. It would be difficult for the updating mechanisms to keep up with the big and rapid changes in the actual network state. In this situation, crankbacks are highly likely.

4. HIERARCHICAL ROUTING

In hierarchical routing, the nodes of a network are classified into different *clusters*. These clusters are grouped into higher-level clusters, and this process iterates recursively. Finally, at the highest level, there is only a single top-level cluster for the entire network.

It is also possible to group links instead of nodes to form link-based clusters [5]. In this paper, to simplify description, we focus only on node-based clustering as described above.

At the lowest level, a node corresponds to a switching system or a router in the network. Each cluster is represented by a single logical node at the next higher level. At each level, within a cluster, each node maintains the topology and detailed state information about the links and other nodes in the same cluster. A cluster advertises outside only a summary, or an aggregated view, of its internal structure. This ability to use a single logical node to represent a cluster of connected nodes limits the number of topological elements generating updates on their states, and so significantly improves scalability.

After aggregation, the network is represented by a simpler topology. Information about the state parameters of individual links and nodes outside a cluster is often lost. Use of this imprecise information may cause routing to suffer from distortion, that is, the cost of going through the aggregated network may not be the minimum.

4.1 Node State Parameters

To aid routing into or transit across a cluster, the aggregated topology description being advertised needs to convey the QoS characteristics from an entry border node to some interior point (the destination) in the cluster, or in transit to an exit border node, respectively. (For node-based clustering, a *border node* is a node that has a direct link to a node outside of the cluster.)

Generally, this requires a cluster to advertise, for each traffic class, the transit QoS characteristics across the cluster for every entry-exit pair, i.e., from one border node to another. Given a cluster, the QoS characteristics between a border-node pair can be derived from the state parameters of the internal paths connecting the corresponding border-node pair. The resulting state parameters associated with this fully connected mesh of border nodes then become the *node state parameters* of the cluster.

Depending on the routing algorithm used, the state parameters of either minimum-weight paths or widest paths between border-node pairs can be used to set the values of the node state parameters of a cluster.

At each node in a cluster, the sets of node state parameters received from other clusters enable the node to decide which cluster to use for routing, as well as to select appropriate entry border node to each cluster and the appropriate associated exit border node.

4.2 Advertising Minimum-Weight Paths

Suppose that a node in a cluster computes minimum-weight paths to different destinations. Receipt of minimum transit weights advertised by different clusters enables the node to select a concatenated path of transit clusters to a destination cluster with low latency and few hops.

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2010.8668
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2010.8668

Note that links or nodes with small weights tend to attract traffic. In particular, a cluster that advertises small transit weights relative to others will encourage traffic to go through it, consuming its available bandwidth. The available bandwidth advertised by a cluster under this scheme therefore may be very low. A cluster would not be chosen if the bandwidth required by a connection exceeds what is advertised, even though there may actually be other paths with larger weights but sufficient available bandwidth through the cluster.

For example, Figure 1 shows two paths P_1 and P_2 through a cluster with state parameters (W_1, B_1) and (W_2, B_2), respectively. If $W_1 < W_2$, then P_1 is the minimum-weight path and so (W_1, B_1) is advertised as the transit QoS characteristics of the cluster. If the bandwidth requirement B_c of a connection is such that $B_1 < B_c \leq B_2$, then this connection will not transit the cluster because it is deemed to have insufficient bandwidth, even though sufficient bandwidth does exist on P_2.

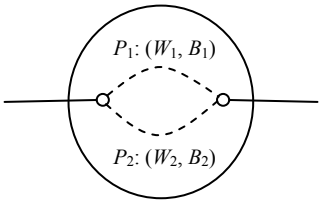

Figure 1. Advertising minimum-weight paths can prevent transit.

4.3 Advertising Widest Paths

In using widest paths, a cluster selects those links with more bandwidth available for subsequent connections, thereby minimizing the probability of blocking.

Widest paths tend to have more hops and larger weights. The advertisement of large transit weights by a cluster tends to discourage transit through it. This may result in non-optimal routing with the selection of longer paths through other clusters, as illustrated by the two examples to follow.

4.3.1 Example 1

Figure 2 shows a network with source cluster A, destination cluster Z, and three transit clusters T, T_1, and T_2. If P_1 is the widest path in T with $B_1 > B_2$, then (W_1, B_1) is advertised as the transit QoS characteristics of T. There are two inter-cluster paths from A to Z: the upper path via T, and the lower path via T_1 and T_2. If either of these paths meets a connection's bandwidth requirement, and if the upper path has a smaller weight than the lower path, then the upper path will be selected for routing.

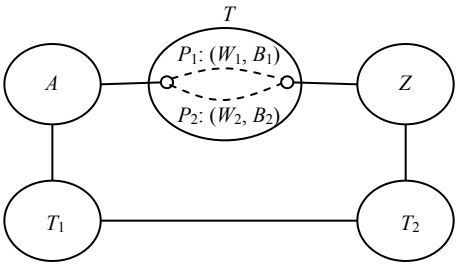

Figure 2. Advertising widest paths with large weights can discourage transit.

Similarly inside T, if either P_1 or P_2 has sufficient available bandwidth, the ingress node in T will take the one with a smaller weight through T. Thus, the actual path selected for a connection to transit a cluster may be different from the one advertised. We will discuss more of this aspect later in Section 5.3.

Now suppose that a new link is installed inside T somewhere along path P_2. (See Figure A.2 in Appendix A for an example.) During the soak-in period prior to service, this new link is assigned a very large weight (as discussed in Section 3.2) so that it does not attract any traffic. As a result, all of its bandwidth is available. By including this new link as part of P_2, the new path P_2 has a much larger weight than P_1, i.e., $W_2 >> W_1$. If P_2 now also has more available bandwidth than P_1, i.e., $B_2 > B_1$, then the transit QoS characteristics of T is advertised as (W_2, B_2). The very large weight of W_2 will force A to switch to the lower path through T_1 and T_2 to Z, even though the unadvertised path via P_1 had adequate available bandwidth and a smaller weight.

Note that if T is not a cluster as in a flat network, the large weight assigned to a new link during soak-in will achieve its intended effect of discouraging traffic. That is, since the topology of T is visible to A when T is not a cluster, A will continue to take P_1 to Z. In this case, the new link has no impacts on routing during its soak-in.

As a result of advertising widest paths, the large weight assigned to a new link inside cluster T is apparently visible to A, even though the topology of T is invisible outside. The side effect of this "visibility of the invisible" is unintended and undesirable as the resulting shifts in regular traffic flows may cause disruptions in network operations.

After the soak-in period, the new link will be assigned its normal in-service weight. The A-Z traffic will then revert back to the upper path via T.

It appears that an expedient to avoid the above problem of selecting a new link during its soak-in period as part of the widest path is to artificially lower its bandwidth. While this is theoretically possible, it is not done in a live network. When installing a link, the parameters for the physical characteristics of the link need to be configured properly according to their in-service values so that testing during the soak-in period can be done to assess the link quality. Once the soak-in period terminates with all the quality tests being passed, the link can be placed in service. Link bandwidth is a physical characteristic that we would like to test to ensure that the link could support the speed at the specified error rates. So, in practice, we would not lower the bandwidth of a new link during its soak-in to avoid the problem.

4.3.2 Example 2

Figure 3 shows another example with a similar problem in routing. Both paths P_1 and P_2 are widest paths for their respective border-node pairs in cluster T. Each has sufficient available bandwidth to meet user bandwidth requirement.

Suppose that the upper A-Z path via P_1 through T has a smaller weight than that of the lower path via P_2 and T_1. Then the upper path is used for A-Z routing. Installing a new link along P_1 will have the same effect of flipping paths for A-Z traffic as that in

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2010.8668
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2010.8668

Figure 2 for Example 1. That is, during the soak-in period, A is forced to switch to the lower path via P_2 and T_1 to Z.

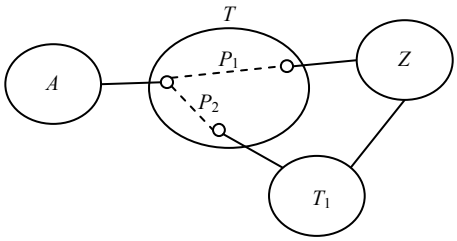

Figure 3. Another problematic scenario for advertising widest paths.

5. BALANCING WEIGHT AND AVAILABLE BANDWIDTH

In view of the foregoing problems, it is tempting to let a cluster use both minimum-weight paths and widest paths so that more accurate information can be captured in the node state parameters. Generation and maintenance of both sets of paths consume more processing cycles and require more storage space at all nodes. Advertisement of the state parameters for both sets of paths consumes more bandwidth on all the links over which they are sent. In the interest of minimizing overheads, this approach would not be practical.

Our proposal is for a cluster to advertise something in between widest paths and minimum-weight paths, with a view to balancing the need for paths with sufficient available bandwidth and smaller weights. In other words, our objective is to use "good" paths: paths that have small weights and most likely will have the necessary available bandwidth to accommodate new requests, even though they may not accurately reflect the actual bandwidth availability inside a cluster.

5.1 Capacity Management

The first line of defense is that a network should be properly engineered to meet service level objectives. For example, a history of bandwidth utilization measurements should be kept to facilitate the analysis of traffic patterns and growth. Individual link loads need to be monitored at several thresholds. When the highest threshold is reached or exceeded, a trigger is issued to order extra capacity, either to install additional links or upgrade existing ones.

By the same token, high-cost underutilized links should be removed from service if it is determined that they are neither required for trended growth nor to meet survivability and/or latency objectives.

5.2 Constrained Minimum-Weight Paths

The problem with minimum-weight paths is that they may have low available bandwidth. This would prevent a cluster from being used by any connection requiring more than the advertised value, even though there are paths within the cluster that have sufficient available bandwidth.

To overcome this problem, *constrained* minimum-weight paths are constructed by pruning those links with little bandwidth available. A sufficiently high pruning value is used to avoid a cluster not being selected because of a perceived lack of available bandwidth.

How should pruning values be set? In hierarchical routing, a cluster does not reveal its topology outside and it does not know in advance the requirements of incoming connection requests. Setting pruning values too low can lead to paths that may not meet user bandwidth requirements. Setting pruning values too high can lead to paths that may be circuitous and so may not meet user delay requirements.

We need to find a way to estimate the proper pruning values for different clusters to use so that links with available bandwidth below some thresholds can be pruned. (Note that since delay is cumulative over a path, links are generally not pruned based on the delays of individual links.) This will enable a cluster to use constrained routing to select paths with sufficient available bandwidth and low latency, and then use the QoS characteristics of these paths for advertisement. For any given border-node pair of a cluster, there is a possibility that constrained routing may only yield paths with available bandwidth below the pruning value. In this case, the widest path for the border node-pair is advertised instead.

Our solution to the estimation problem is based on the usual practice for a network to maintain the load profiles of all the connections that it carries. From these collected data, probability distributions about the bandwidth requirements of the connections in different classes of service visiting different clusters can be generated. Suitable pruning values for each class for different clusters can be derived therefrom to minimize blocking, i.e., to yield a high probability for the admission tests for new connections to be passed. For example, the 95th percentile of bandwidth requests may be used to set pruning values.

As connection characteristics may change over time, these pruning values may need to be reviewed periodically. Also, since different clusters may have different link speeds, the pruning values may need to adjusted accordingly on a per-cluster basis.

5.3 Transiting a Cluster

A cluster advertises the state parameters of constrained minimum-weight paths. When a connection arrives at the cluster, the actual path that is selected for transit may not necessarily be the constrained minimum-weight path that was used to advertise the state parameters. So long as a path within the cluster can be found that meets the bandwidth required, that path can be used for transit. We have mentioned this possibility in Section 4.3 on the advertisement of widest paths.

This alternative path may actually have a smaller weight and so may improve the overall routing. Also, depending on the position of the bottleneck link in the constrained minimum-weight path, in using the alternative path the connection may not necessarily reduce the advertised available bandwidth of the constrained minimum-weight path.

6. CONCLUSIONS

We have been using the simulation-based Capacity Management Tool to study various issues in the design of hierarchical routing with QoS for the AT&T Global Network. These studies show that advertisement of aggregated information in hierarchical routing may lead to non-optimal routing. Also, use of either minimum-

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2010.8668
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2010.8668

weight paths or widest paths for aggregation can cause problematic routing scenarios.

When the details inside a cluster are not revealed outside, there is the problem of selecting appropriate and relevant QoS information to be advertised, while minimizing overheads. For any solution to be effective, what matters is that the advertised information of a cluster allows a connection to get through the cluster with the requested bandwidth with a high probability, should the cluster be selected on a path toward the destination. This is the proof of the pudding.

In this spirit, to minimize the impacts of information loss as a result of aggregation, we propose the use of constrained minimum-weight paths with pruning based on carried load characteristics. This enables a cluster to select those paths that balance the need for low delay and sufficient available bandwidth. By advertising the state parameters of these paths, there is a high likelihood that the cluster will have the necessary available bandwidth to meet new demands. This helps to minimize the probability of blocked connections, thereby reducing the need for crankbacks.

7. ACKNOWLEDGMENTS

The proposal in this paper was based on our operational experience in AGN. Ed Sierecki provided very helpful input and discussion in the implementation. Herb Shulman also provided guidance in this work.

The simulation-based Capacity Management Tool (CMT) developed by Bob Kelley, Doug Jones, and Clem McCalla was instrumental in the study to validate the different design concepts. Ron Levine of Cisco Systems provided some of the details of switch operations.

Manuel Villén-Altamirano of the Universidad Politécnica de Madrid in Spain, co-chair of the Industry Track (*SimulationWorks 2010*) of the 3rd International ICST Conference on Simulation Tools and Techniques (*SIMUTools 2010*), kindly invited us to present this paper. He also went over the paper with a fine-toothed comb, and made numerous helpful suggestions for improvement.

8. REFERENCES

[1] The ATM Forum Technical Committee, "Private Network-Network Interface Specification Version 1.1 (PNNI 1.1)," *af-pnni-0055.002*, Apr. 2002.

[2] E. Rosen, A. Viswanathan, and R. Callon, "Multiprotocol Label Switching Architecture," Internet Engineering Task Force (IETF) RFC 3031, Jan. 2001.

[3] W.Y. Tam, K.S. Lui, S. Uludag, and K. Nahrstedt, "Quality-of-Service Routing with Path Information Aggregation," *Computer Networks*, Vol. 51, No. 12, Aug. 2007, pp. 3574-3594.

[4] W.S. Lai, E. Rosenberg, L. Amiri, M. Ball, Y. Levy, H. Shulman, H. Tong, and M. Ungar, "Analysis and design of AT&T's Global PNNI Network," *Proc. 2005 IEEE Pacific Rim Conference on Communications, Computers, and Signal Processing (PACRIM 2005)*, Victoria, B.C., Canada, 24-26 Aug. 2005, pp. 129-132.

[5] Y. Lai and W.S. Lai, "A Graph-Theoretic Model of Routing Hierarchies," *Proc. IEEE International Conference on Advanced Information Networking and Applications Workshops (WAINA 2009),* Bradford, UK, 26-29 May 2009, pp.1118-1123.

APPENDIX A: APPLICATION OF SIMULATION

In this appendix, we give an overview of the simulation-based Capacity Management Tool (CMT) used in our study. As an example of its application, we describe a case study of its use to determine the root cause of a routing anomaly detected under a link failure in the AT&T Global Network (AGN).

A.1 Simulation Tool

CMT is an in-house tool that simulates the different routing protocols used in the network, together with their various algorithms and features.

CMT uploads network data daily from AGN. These include configuration and utilization data of different network elements such as switches, routers, trunks, and fiber spans. Also included are all the established connections, along with their respective service classes, allocated bandwidth and restoration requirements, average and peak loads, and sundry other characteristics.

From these uploaded data, CMT maintains the current view and some history of the entire AGN topology, together with the details of hierarchical clustering structure. By processing the collected data, CMT produces various visual map displays and generates reports on a number of network metrics, such as utilization of trunks and ports, delay and hop count distributions of connections, etc. These aids are used by operations personnel to manage AGN.

CMT offers a user interface whereby its view of the network can be modified by adding and/or deleting network elements, or by changing the hierarchical clustering structure, per user specification. Any such changes in network topology may affect the paths used by the established connections. A user has the option of either rerouting these connections or keeping them on their old paths. An established connection will be released if it is not rerouted and its old path no longer exists, e.g., due to the deletion of a network element that previously carried the connection.

Similarly, the routing of any mix of connections can be analyzed, e.g., by adding new connections with various characteristics as described above, or by deleting selected established connections. Newly added connections are routed according to the routing protocols used, assuming the currently available bandwidth in the network.

Through these capabilities, a user can investigate the behaviors of different network models of AGN. For example, CMT can be used as a modeling tool to perform network capacity analysis for traffic growth forecast, or to study various design alternatives and trade-offs.

A.2 A Detective Story

In retrospect, the routing scenarios illustrated in Examples 1 and 2 in Section 4.3 seem "obvious" enough. However, we were not aware of the unintended interaction described therein until it

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2010.8668
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2010.8668

actually happened in the live network. We now describe a case study involving CMT that helped uncover the problems we encountered in Section 4.3.

Figure A.1 depicts a highly simplified diagram of a small segment of the network in our case study. The links are labeled with their respective weights. *T* is a cluster, with only border nodes shown colored in black. Some of the other nodes in the diagram are actually also clusters, but they are not germane to our exposition of the problem and hence are not shown as such.

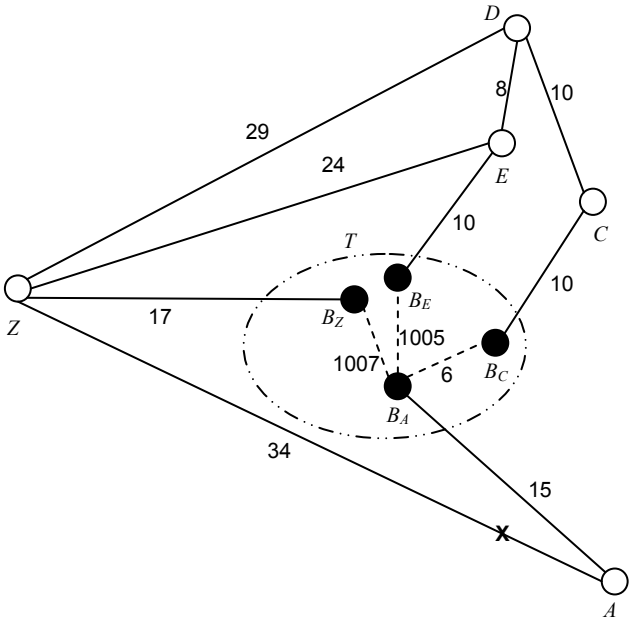

Figure A.1. Network under failure.

Normally, the traffic from *A* to *Z* takes the direct link joining the two nodes. This link failed due to a cable cut incident. Operations personnel noticed that, after failure, the connections from *A* were rerouted to go via *T*, *C*, and *D* to *Z*, with significant increases in latency. It was puzzling why that more direct paths to *Z*, e.g., via *T* only, or via *T* and *E*, were available (i.e., with sufficient available bandwidth) but not used.

Using CMT, a detailed simulation analysis of what's going on inside *T* reveals the reason for the much longer path being taken. After the failure of the *A*-*Z* direct link, *A* will look for a minimum-weight path (with sufficient available bandwidth) to *Z* using the link weights it sees plus the advertised weights for any cluster that the *A*-*Z* connections must transit. From the advertised weights of *T* displayed in Figure A.1 (to be explained below), the very large difference in the advertised B_A-B_C weight relative to the B_A-B_E and B_A-B_Z weights overrides all other weight considerations and causes the path through *C* and *D* to selected by *A*.

Figure A.2 shows a simplified diagram of cluster *T*, displaying only on those aspects that are relevant to the routing problem at hand. As shown, there are three offices inside *T*. Each inter-office link is labeled with a pair of numbers: its weight and the percentage load carried. The percentage available bandwidth on a link can be computed as (100 – percentage load carried).

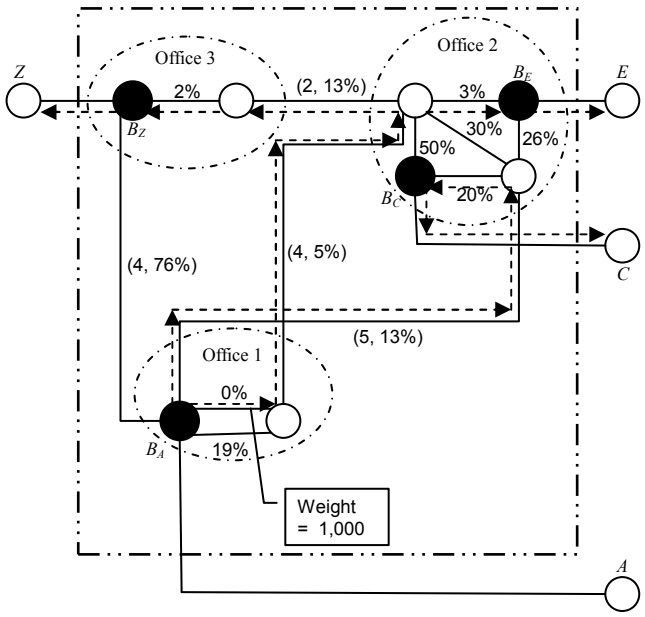

Figure A.2. Cluster *T* and some widest paths.

Each intra-office link is assumed to have a weight of 1 (not shown in the diagram) and is labeled only with its percentage load.

Just prior to the cable cut event, a new parallel link with a weight of 1000 was installed in Office 1.

T advertises its node state parameters based on the widest paths between its border-node pairs. Figure A.2 shows the three widest paths from the border node B_A to the other three border nodes. Their respective weights are:

B_A-B_C: 5 + 1 = 6
B_A-B_E: 1000 + 4 + 1 = 1005
B_A-B_Z: 1000 + 4 + 2 + 1 = 1007

Thus, the high available bandwidth of the new link in Office 1, coupled with the large weight assigned to it, has unduly influenced the computation of the node state parameters of *T* when widest paths are used. This results in the sub-optimal routing of *A*-*Z* traffic.

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2010.8668
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2010.8668