In Proc. First ARL/ATIRP Annual Conference,
College Park, MD (Jan. 1997), pp. 257-263.

VIRTUAL TREE PATH MANAGEMENT FOR BATTLEFIELD NETWORKS

Adarshpal S. Sethi and Prashant Ramarao
Department of Computer and Information Sciences
University of Delaware, Newark, DE 19716
{*sethi, prashant*}*@cis.udel.edu*

ABSTRACT

Virtual Trees (VTs) are source-rooted trees in ATM networks in which VCs originating at the source but going to different destinations can share some of the bandwidth pre-allocated to the VT, thus providing an additional multiplexing advantage over Virtual Paths (VPs). We propose two fast bandwidth allocation algorithms for both VTs and VPs that are approximate in the sense that they do not always yield optimal configurations. We describe the results of a simulation study which shows that Virtual Tree configurations have almost 40% lower cost than corresponding Virtual Path configurations. Applications of this work to dynamic routing schemes for battlefield wireless networks are outlined.

Keywords: Virtual Paths, Virtual Trees, Dynamic Routing, Bandwidth Allocation.

INTRODUCTION

The technique of Virtual Paths is used in ATM networks to perform bandwidth allocation for virtual circuits and to simplify setting up of virtual circuits in response to connection requests. The Virtual Path concept provides several advantages such as allowing simpler network architectures, eliminating the need for call-by-call routing resulting in very short connection setup times, and allowing easier implementation of dynamic bandwidth allocation schemes [1], [2]. In our previous work, we have extended this concept of Virtual Paths to propose a new technique called Virtual Trees that can be used for bandwidth allocation in ATM networks [3], [4]. A Virtual Tree (VT) corresponds to pre-allocated bandwidth along a set of links in the network that form a tree rooted at a source node and leading to various destinations. Virtual Trees retain all the advantages of Virtual Paths but have a better performance potential because of the increased multiplexing between traffic flowing from the same source to different destinations over common links. The effect is a reduced probability of blocking (rejecting connection requests), better utilization of allocated capacity, and flexibility in determining routes when the VT is set up.

The Virtual Paths used in a network must be periodically modified to match the dynamically changing traffic patterns in the network. A Virtual Path configuration refers to the set of Virtual Paths that are to be used during a period of time. There are several proposed algorithms that deal with the problem of finding such configurations [5], [6] given predicted traffic patterns between source-destination pairs and capacity constraints for the network links. We have previously proposed a bandwidth allocation model for Virtual Trees [3] that yields optimal VT configurations in a similar manner to VP configurations.

One of the main difficulties in using these optimization models is that determining optimal configurations is an NP-Hard problem [7]. We have developed approximate optimization models that, while not yielding an optimal solution, provide good solutions in a reasonable amount of time. Two such models named Graph Reduction and Underloaded Neighbors respectively, both start with shortest path configurations and modify them in different ways to try and satisfy link capacity constraints. Experiments conducted by us on random graphs of various sizes have shown that these models typically provide reductions in allocated bandwidths for VTs of around 40% over VPs for networks with a few hundred nodes.

Battlefield wireless networks can exploit the advantages of Virtual Trees because of source concatenation at the data link and physical layers which has an effect very similar to that of VC multiplexing in ATM

networks. For this reason, our work on algorithms for constructing Virtual Tree configurations holds great promise for application to path and bandwidth management in battlefield networks.

VIRTUAL PATHS AND VIRTUAL TREES

The Asynchronous Transfer Mode (ATM) is a switching and multiplexing technique developed for Broadband ISDN (B-ISDN) networks that makes it possible to achieve the multiplexing of different kinds of traffic while providing individual quality of service guarantees for each traffic type. The CCITT standard for the ATM Layer defines two types of connections: *Virtual Channel (VC) connections* and *Virtual Path (VP) connections*. The concept of a Virtual Channel (VC) is similar to that of a virtual circuit in traditional networks, namely, a logical unidirectional association that defines a connection between a source and its destination. To establish a VC, the source node sends a request that propagates through intermediate nodes to the destination requesting allocation of the required bandwidth. Because of the propagation delay (which assumes increasing importance as speeds increase), and the processing delay at each node of the network, VC establishment by this method can be slow and unacceptable for real-time needs, particularly when connections are established on a per-burst basis.

In ATM networks, Virtual Paths (VPs) solve this problem by providing a pre-defined route with pre-defined bandwidth between a source-destination pair. A Virtual Path is commonly defined as a bundle of virtual channels delimited by two Virtual Path terminators. Multiple VCs may exist simultaneously within a Virtual Path between a given source-destination pair. Virtual Channels are allowed to share the bandwidth pre-allocated to the Virtual Path they belong to, yet the sharing of the Virtual Path's bandwidth depends on the bandwidth allocation technique being used. In some cases, a Virtual Channel may need to use more than one Virtual Path to associate a source with its destination (i.e., to establish a connection); the VC may then be carried by a concatenation of Virtual Paths through a sequence of intermediate VP terminators.

Virtual Channels are allowed to share the bandwidth pre-allocated to the VP they belong to. But the technique of Virtual Paths does not involve the policing of the source's traffic to enforce the amounts of bandwidth pre-allocated to each VP; therefore, each source must keep its traffic within the parameters negotiated at the call admission to ensure that the quality of service offered by the network will meet the requirements of all the sources using the network. Once a VP has been set up with a specified bandwidth, the source node must decide on each call request whether or not that call can be carried by the VP. In doing so, it is useful to refer to the term *equivalent bandwidth* which is the amount of bandwidth necessary to accommodate the aggregate traffic of a set of K bursty sources while complying with a specific quality of service [8]. The quality of service may be measured in terms of the buffer overflow probability or the cell loss probability. Other desirable quality of service measures are delay and delay jitter, but these are harder to use in computing equivalent bandwidth.

We have proposed a new technique called Virtual Trees (VTs) to be used in a manner similar to the way Virtual Paths are currently used [4], [3]. In VTs, there is pre-defined bandwidth available for use by a given source node, just as in VPs. Whereas in VPs, the path and the corresponding bandwidth is defined separately for each destination, in VTs some of the links and their bandwidths could be shared by VCs going to different destinations. This takes the form of a tree rooted at the source node whence the name Virtual Tree.

In a Virtual Path, different Virtual Channels may share the bandwidth of the VP resulting in better multiplexing of the traffic than if the network allocated bandwidth individually for each VC [2]. Because of this multiplexing effect, a smaller amount of bandwidth may be allocated to the VP than the sum of the bandwidths of all VCs traversing it. Put another way, the Equivalent Bandwidth required by each VC decreases as more traffic is multiplexed on the same VP. However, multiplexing in VPs is limited to traffic flowing between the same source-destination pair.

In Virtual Trees, the multiplexing effect of VPs is enhanced by permitting sharing of bandwidth among traffic flowing to different destinations. It is still limited to traffic originating at a single source, as that

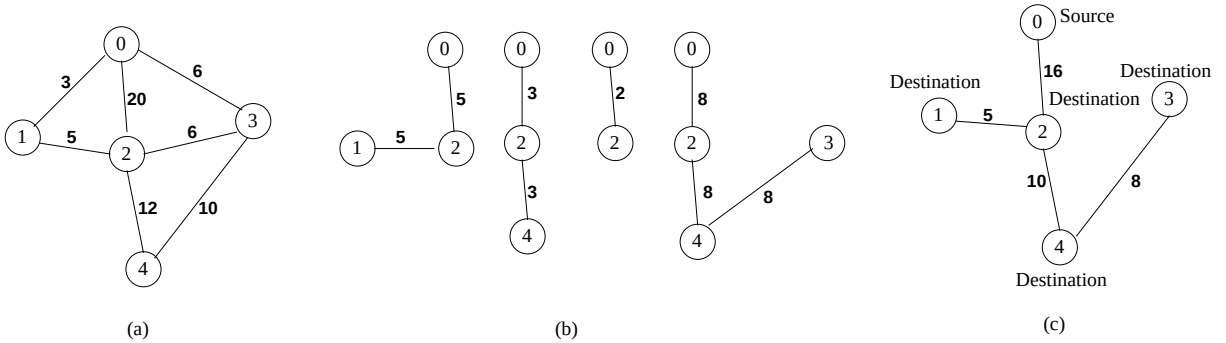

Fig. 1. (a) A computer network G. Numbers along lines show link capacity. (b) A set of Virtual Paths in G originating at source 0 with numbers indicating bandwidth allocated to VP. (c) A Virtual Tree VT_0 from G rooted at the node 0. In a Virtual Tree, the amount of bandwidth allocated to a given link does not have to match the sum of equivalent bandwidths from the paths that use that link.

is what allows the source node to decide on call admission without sending out a call setup request to the destination. But a Virtual Tree allows the bandwidth allocated to a particular source node to be used flexibly for various destinations. The effect is a reduced probability of blocking (rejecting connection requests), better utilization of allocated capacity, and flexibility in determining routes when the VT is set up. These effects are obtained by preserving all the major advantages of VPs with only the slight additional cost of a somewhat more complex call admission algorithm at the source node.

The concept of a Virtual Tree and its relationship with the paths for source-destination pairs is depicted in Figure 1. Part (a) of this Figure shows a network with five nodes connected by links with specified bandwidths. The traffic from each source-destination pair is transmitted along the route selected for each one of them. Part (b) shows a set of Virtual Paths for the source-destination pairs (0-1), (0-2), (0-3), and (0-4) with a bandwidth allocated to each VP based on predicted traffic patterns. Part (c) shows a Virtual Tree rooted at the source node 0 for the same set of source-destination pairs. Bandwidth is now allocated to each link of the VT as shown. The total bandwidth allocated to a Virtual Tree's link is the equivalent bandwidth of all the traffic multiplexed in it; therefore, it might happen that the sum of the individual equivalent bandwidths used by the VPs at a given link will not match the equivalent bandwidth allocated to the same link in the VT.

One of the important problems that needs to be addressed is: Given an ATM network topology with various physical link capacities, and given a predicted traffic pattern between various source-destination pairs, how do we determine a Virtual Tree configuration that will carry the given traffic and minimize the total network cost? The total network cost here is defined as a weighted sum (over all links) of the total bandwidth allocated on each link to all the Virtual Paths or Trees weighted by the link cost. We have presented in [3] an optimization model that yields such a VT configuration.

It has been shown that the problem of determining optimal configurations for Virtual Paths is NP-Hard [7]. Clearly, the algorithm to solve the optimization model for Virtual Trees is at least as hard. We have conducted simulation studies of our optimization model and found that it was difficult to use it for networks of more than about 10 nodes because of the excessive time required to find the solution. We also observed that, for networks of less than 10 nodes, Virtual Trees resulted in an Average Call Blocking Probability that was approximately 11-18% lower than the corresponding figure for Virtual Paths. While encouraging, this is not a dramatic improvement. We would expect to see a more significant improvement in performance for larger networks because there is larger flexibility in constructing trees when more alternate paths are available and also because there are more opportunities to gain from multiplexing of traffic.

APPROXIMATE ALGORITHMS

We have developed two approximate optimization algorithms for finding VT and VP configurations that,

while not yielding an optimal solution (in terms of minimizing total network cost), provide good solutions in a reasonable amount of time. The two algorithms named *Graph Reduction* and *Underloaded Neighbors* respectively, both start with shortest path configurations and modify them in different ways to try and satisfy link capacity constraints. Both algorithms start by constructing *Unconstrained* paths in which link capacities are not taken into account during the process of bandwidth allocation. We present here only a brief outline of the two algorithms and then describe the simulation results comparing the performance of the algorithms on VTs and VPs.

The basic algorithm for determining Virtual Path configurations is based on the All Pair Shortest Path algorithm. This algorithm minimizes the total system cost without taking into account the link capacities and hence it is possible that some of the links become overloaded. In VPs the mutliplexing is done over all Virtual Circuits for the same source-destination pair. This algorithm is easily modified to construct Virtual Tree configurations, again assuming no capacity constraints. In Virtual Trees the idea is to form trees with a particular source node as root and its various destinations as leaves. If it so happens that a particular link is shared by multiple paths originating from the same source then they can be multiplexed so that the effective bandwidth requirement is reduced and hence the cost of the network is also reduced. The Virtual Trees use the paths created by the All Pair Shortest Path algorithm, and all the paths starting from a particular node are grouped together to form a tree.

When capacity constraints are added, some of the links in the VP and VT configurations obtained by the above method could become overloaded. The two constraining algorithms developed by us identify these overloaded links and try to reroute some of the traffic through those links so that the capacity constraints are satisfied. In the first method called *Graph Reduction*, we identify all the overloaded links and find shortest paths and costs considering the "reduced graph" - a graph with only those links that can withstand at least one more source-destination flow through it. The alternate paths for the traffic that has to be rerouted is calculated using this reduced graph. This algorithm is used for both Virtual Trees and Virtual Paths. In the second constraining method called *Underloaded Neighbors*, for every overloaded link we try to find all the source-destination pairs which contribute to the traffic through it, then we attempt to reroute the traffic using a neighbor of the destination which does not use an overloaded link and results in the minimum increase of the overall cost among all such neighbors.

We simulated the unconstrained and the two constrained algorithms for both Virtual Paths and Virtual Trees on a set of randomly generated network topologies satisfying certain specified constraints. Specifically, the number of nodes, N, in the network was varied from 10 to 200. For each size, the links were generated randomly satisfying minimum and maximum degree constraints, such that the number of links was of the order $NlogN$ and the network was connected. Each link was assigned the same cost and the same bandwidth. Also, equal traffic was assumed between all source-destination pairs. The total cost for both Virtual Path and Virtual Tree configurations was computed with each of the three algorithms (the unconstrained and the two constrained). These costs were then averaged over multiple runs with independently generated networks for each network size to factor out statistical variations. We also computed the average normalized cost for both VPs and VTs, where the normalized cost for each network is the cost of the VP or VT configuration for that network obtained with the appropriate constraining algorithm divided by the Unconstrained Virtual Path cost for that network.

Figure 2 shows the results of this experiment. This figure shows the variation of the average cost with number of nodes in the network for both VP and VT configurations in the unconstrained case as well as in the two constrained cases. As expected, the average cost increases as the network size increases. This is because, although the traffic generated between each source-destination pair is constant, the number of such pairs increases at a rate that is order N^2 where N is the number of nodes. Also, the constrained costs are higher than the corresponding unconstrained cost for each of the VP and VT cases. However, VTs give a lower cost than VPs in all cases. The two constraining algorithms seem to behave almost identically for VTs, while for VPs, the reduced

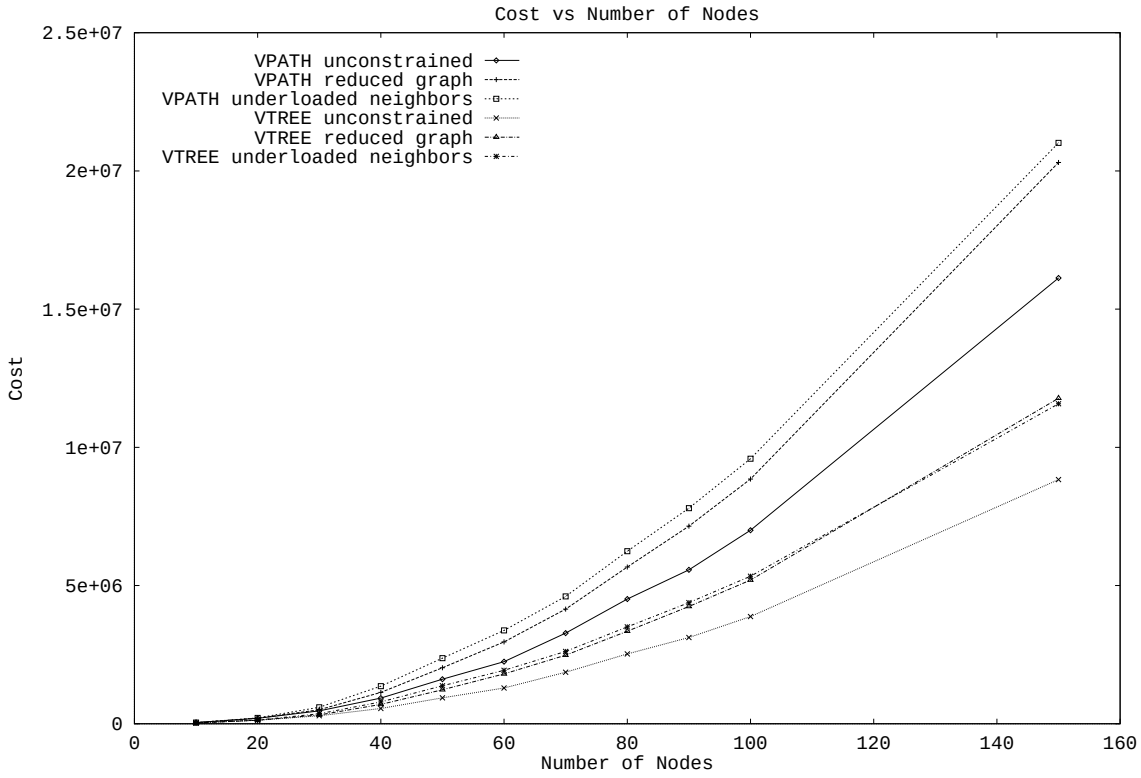

Fig. 2. Simulation results for Approximate Algorithms for VTs and VPs.

graph algorithm works slightly better than the underloaded neighbors algorithm.

Figure 3 shows the variation of normalized cost with number of nodes. Since all costs are normalized by the unconstrained VP cost, the cost for the Unconstrained VP configurations is the constant 1.0 for all numbers of nodes. This figure allows us to quantify the improvement obtained by each of the algorithms over the Unconstrained VP configuration. The normalized cost for the Unconstrained Virtual Tree configurations initially decreases as the number of nodes is increased because small network sizes provide less opportunity to multiplex multiple paths over common links. But with more than about 70 nodes, this factor does not play an important role and the normalized cost of VTs settles down to slightly less than 0.6. This represents a more than 40% improvement over Virtual Paths.

When capacity constraints are introduced, the cost of the resulting configurations is obviously more than the cost of the corresponding unconstrained configurations for both VPs and VTs. It appears that in the long run the normalized constrained costs stabilize at around 0.75 for VTs and around 1.25 for VPs. This again represents a 40% improvement for Virtual Trees over Virtual Paths.

APPLICATION TO WIRELESS BATTLEFIELD NETWORKS

Recently, Virtual Trees have been proposed to be used in two different contexts in wireless networks for managing handoff between mobile terminals and base stations. Acampora and Naghshineh [9] use a tree called a Virtual Connection Tree to modify existing VCs when a mobile node is handed off to a new base station. However, their tree is not rooted at the source like our tree is and also uses distinct paths from the root to the base stations, thus not taking advantage of multiplexing over overlapping links. Veeraraghavan et al. [10] use a Virtual Sink Tree to find paths to a given destination. A Sink Tree is rooted at the destination instead of the source, and takes advantage of multiplexing over common links,

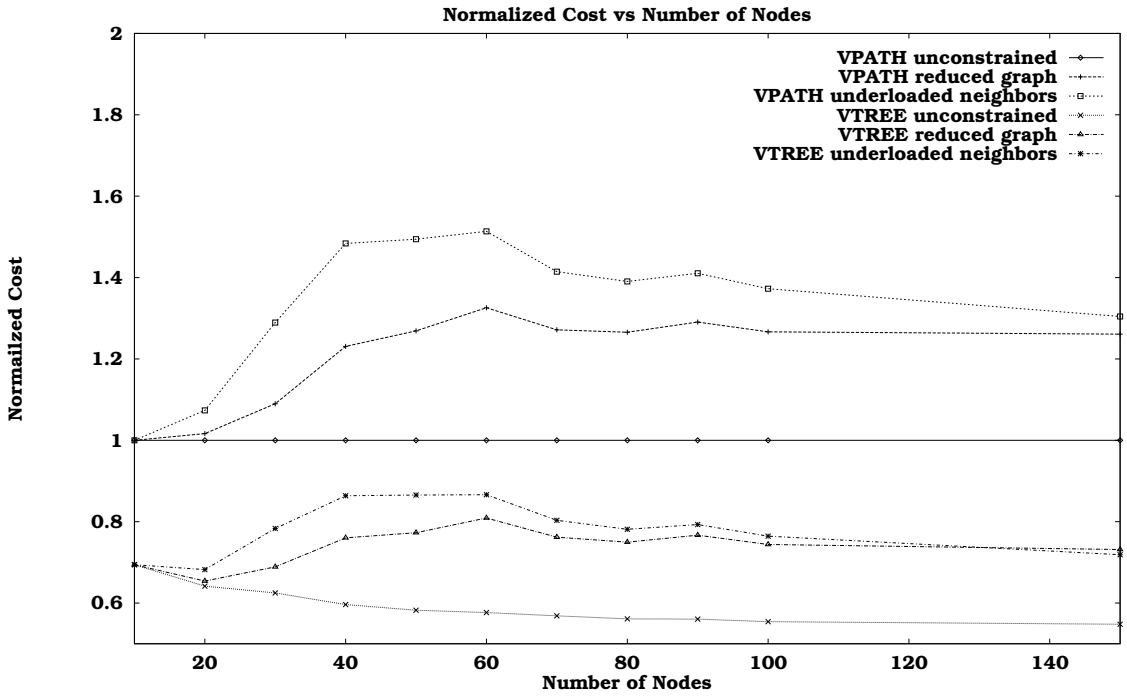

Fig. 3. Normalized Costs vs Number of Nodes for VTs and VPs.

but VCI setup is slower.

Since a tactical battlefield network will not have fixed base stations to manage handoff, but is instead likely to have peer-to-peer communications, the main use of Virtual Trees in battlefield networks will be to manage the setup of end-to-end paths between nodes that wish to communicate with each other. These networks are not likely to use ATM technology, so the multiplexing of cells between VCs cannot be exploited in the way that VPs and VTs have done in ATM networks. However, there are similar advantages in wireless networks of using source concatenation of packets flowing to different destinations. The MIL-STD-188-220A suite of protocols [11] being developed for Combat Net Radio (CNR) for digitized battlefields uses source concatenation at both the data link and physical layers. Because of the large net access delays, low bandwidths, and random access protocols used for media access, it makes sense to transmit more than one data link PDUs within a single transmission when a node gains access to the channel. For this reason, it is advantageous to set up the network layer paths in the form of trees to derive maximum benefit from shared links in the paths leading from the same source to different destinations. The algorithms developed by us to set up Virtual Tree configurations can be used for this purpose.

Another aspect of tactical battlefield networks is the dynamically changing configuration due to both mobility of the nodes and enemy action. While an initial VT configuration can be used by the nodes at start-up, this will rapidly become sub-optimal or even unusable as the underlying network topology changes. We propose to use a partially decentralized approach to handle this problem. Whenever a link in the network goes down because it is no longer available, or when a new link is detected, an indication is provided by the data link layer to the network layer that results in a topology update. With our approach, each node keeps track of the network topology in its immediate vicinity and makes local adjustments to its own VTs as the topology changes. Each subnet or a small number of subnets collectively have a designated path manager node that also keeps track of subnet topology. Periodically, the path manager runs an incremental algorithm to generate new VT configurations and updates the configurations being

used by all the nodes within its domain. The use of a path manager will prevent congestion hot spots from arising, a possibility if the nodes act in a purely distributed manner without any coordination. On the other hand, the partially decentralized nature of the algorithm that allows the nodes to make changes to their own VTs will ensure that a failure of the path manager does not result in a total breakdown of communication.

CONCLUSIONS

In summary, we have shown that the use of Virtual Trees can give significant cost and performance advantages over the use of Virtual Paths for bandwidth allocation and management. We have designed an optimization model for Virtual Tree configurations and have proposed two approximate algorithms to generate such configurations. The use of Virtual Trees is advantageous in ATM networks because of the increased multiplexing that results from sharing common links between VCs that flow from the same source to different destinations. In non-ATM wireless networks, a similar advantage can be obtained by using source concatenation of data link and physical layer frames. In FY97, we will be working on applying our results and algorithms for Virtual Trees to battlefield wireless networks in order to design algorithms for dynamic route allocation and path management using tree structures in networks with changing topologies.

The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government.

REFERENCES

[1] Y. Sato and K. Sato. Virtual path and link capacity design for ATM networks. *IEEE Journal on Selected Areas in Communications*, 9(1):104–111, January 1991.

[2] J.A.S. Monteiro, M. Gerla, and L. Fratta. Statistical multiplexing in ATM networks. *Performance Evaluation*, 12(3):157–167, June 1991.

[3] A.S. Sethi. A model for virtual tree bandwidth allocation in ATM networks. In *Proc. Infocom '95, 14th Annual Joint Conference of the IEEE Computer and Communications Societies*, pages 1222–1229, Boston, MA, 1995.

[4] A.S. Sethi and A. Mock. Virtual trees - a new technique for bandwidth allocation in ATM networks. In *Proc. Second Intl. Conf. on Telecommunication Systems, Modeling and Analysis*, pages 113–119, Nashville,TN, March 1994.

[5] M. Gerla, J.A.S. Monteiro, and R. Pazos. Topology design and bandwidth allocation in ATM nets. *IEEE Journal on Selected Areas in Communications*, 7(8):1253–1262, October 1989.

[6] J.Y. Hui, M.B. Gursoy, N. Moayeri, and R.D. Yates. A layered broadband switching architecture with physical or virtual path configurations. *IEEE Journal on Selected Areas in Communications*, 9(9):1416–1426, December 1991.

[7] I. Chlamtac, A. Farago, and T. Zhang. Optimizing the system of virtual paths. *IEEE/ACM Transactions on Networking*, 2(6):581–587, December 1994.

[8] M. Decina, T. Toniatti, P. Vaccari, and L. Verri. Bandwidth assignment and virtual call blocking in ATM networks. In *Proc. Infocom '90, Ninth Annual Joint Conference of the IEEE Computer and Communications Societies*, pages 881–888, San Francisco, CA, 1990.

[9] A.S. Acampora and M. Naghshineh. An architecture and methodology for mobile-executed handoff in cellular ATM networks. *IEEE Journal on Selected Areas in Communications*, 12(8):1365–1375, October 1994.

[10] M. Veeraraghavan, M. Karol, and K.Y. Eng. Implementation and analysis of connection-management procedures in a wireless ATM LAN. In *Proc. IEEE International Conference on Universal Personal Communications*, 1996.

[11] Military Standard - Interoperability Standard for Digital Message Device Subsystems (MIL-STD 188-220A). July 1996.