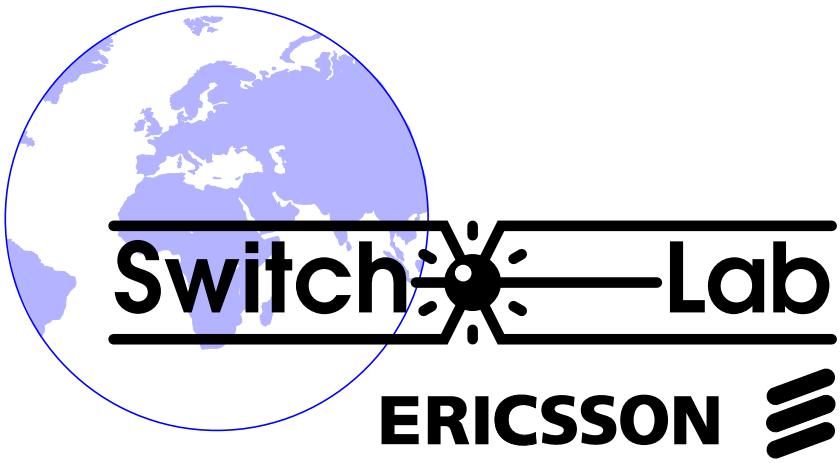

Multicast Routing in ATM and IP over ATM Networks

Bartosz Mielczarek

Chalmers University of Technology

Master's Thesis in
Digital Telecommunication Systems and Technology

Supervisor: Harald Brandt
SwitchLab
Ericsson Telecom AB

Examiner: Bengt Wedelin
Department of Information Theory
Chalmers University of Technology

Stockholm, 1997-12-16

Abstract

The main objective of multicast is to find a tree connecting many sources and destinations, and which consists of paths that fulfil a set of given constraints such as cost, delay and delay variation. The established tree should not disturb other traffic and should provide fast and efficient transport of data packets.

ATM networks impose an additional constraint on multicast routing: as for now, there is no support for bidirectional one-to-many or many-to-many connections - this is due to the connection-oriented character of ATM layer and its inability to distinguish cells arriving from different sources on a same path. Currently, there are two most important approaches to the problem: one approach depends on building many one-to-many source routed connections, whereas the second approach uses dedicated multicast servers for many-to-many connections.

A routing algorithm can concentrate on the cost and/or delay constraints. The simulations show that the delay-oriented algorithms perform significantly better than the cost-oriented ones in one-to-many connections. In many-to-many connections, however, the situation is not that clear - when multicasting servers are used, none of the algorithms is universally good. This may imply that completely different algorithms should be employed for different applications.

Dynamic construction of a multicast tree is another important subject. When new nodes want to join the existing multicast session, they may connect to the tree in different ways. According to the conducted simulations, a new node should attach to the tree via the shortest path to the source.

Another important issue is the interaction between the currently existing multicast protocols and ATM routing. Due to the worldwide spread of the Internet, which is based on the TCP/IP protocol, it is necessary to discuss the possibilities of using current existing techniques over fast ATM networks. Different routing layer approaches are discussed and possible interaction schemes suggested.

Table of Contents

Appendices

List of Figures

List of Tables

1 Introduction

This report is a result of research work for a Master of Science degree of Chalmers University of Technology. The whole project was carried out at the SwitchLab research facilities of Ericsson Telecom AB in Stockholm.

1.1 What is multicasting?

With the advance of modern digital telecommunication technologies, it is becoming more and more necessary to employ the most effective methods of managing the traffic in networks. The traditional simple algorithms are no longer sufficient in more demanding tasks like sending real-time video data over the networks.

The traditional approach to routing assumed a simple one-to-one connection with only one type of factor influencing the choice of a path - link cost. The link cost function can depend on the length of a link, its load or the administrative cost of establishing the connection. However, new types of connections, namely one-to-many and many-to-many with multiple routing factors are required for many applications like:

- video-conferencing
- video-on-demand, music-on-demand and similar services
- software-upgrades
- group-ware (i.e. a group of people working on the same document)

Indeed, any application sending large amounts of data to many users demands multicasting techniques to avoid blocking of available network resources. The June 97 edition of the Byte magazine includes the article “Multicast to the Masses” in which a real life example of Boeing is presented. The upgrade of software on engineers’ workstations started on Saturday morning and involved sending approx. 20MB of data to every workstation. Due to the excess blocking, the procedure lasted until Tuesday, which caused considerable delay in the company activities. The problem was brought about by the old multicast technique, based on establishing many point-to-point connections for the same data, which causes excessive bandwidth occupation and large delays.

Multicast routing is significantly more complex than unicast routing - choosing the best path usually involves taking account of three factors: cost, delay and delay variation. There are no simple algorithms providing best possible tree - the problem is known to be NP-complete [7] which means that finding the best solution involves a search time that grows exponentially with the number of nodes in the graph.

The problem of dynamic multicasting is another important issue. What happens if a node wants to join the multicast tree - should the tree be reorganized or can the node just attach to the closest tree branch?

1.2 Scope of the thesis

The objective of the thesis was to investigate the performance of different algorithms used for routing of a multicast tree. Such comparison gives an idea about the complexity of algorithms compared to their revenue: increased throughput, better distribution of link load, maintaining the crucial parameters of connections etc.

The main tasks were:

- Compare different techniques of multicasting - both static and dynamic.
- Implement and test different multicast algorithms.
- How should multicasting be performed in ATM networks?
- How should TCP/IP and ATM multicasting interact?

In the following chapters, it is assumed that the reader is familiar with basic networking concepts such as routing, flooding, topology table etc. Detailed information about these concepts can be found for example in [5].

Basics of the Asynchronous Transfer Mode are not discussed in the report either - only the specific features influencing multicast techniques.

1.3 Disposition

The report consists of eight chapters in which the above problems are addressed.

A general introduction to multicast techniques is presented in Chapter 2.

Chapter 3 explains the peculiarities of ATM and ways of circumventing the particular problems arising from the employment of this type of network.

Chapter 4 and 5 consist of presentations and results of simulations of some of the existing algorithms. These chapters are the main result of the thesis work.

Chapter 6 addresses the problems of interaction between the most widely used Internet protocol TCP/IP and ATM networks. Since there already exist routing protocols for IP networks, it should be discussed how they can work with ATM protocols.

Chapter 7 concludes with suggestions for possible solutions and future studies.

In the last part of the thesis, there is a list of references, presentation of a simulator environment, employed networks and a glossary.

2 General concepts

This chapter presents basic description of problems involved in multicast routing. This is done without regard to any particular type of network or service.

Different routing approaches are briefly presented as well as their possible outcomes and applications.

2.1 Unicast, multicast and broadcast.

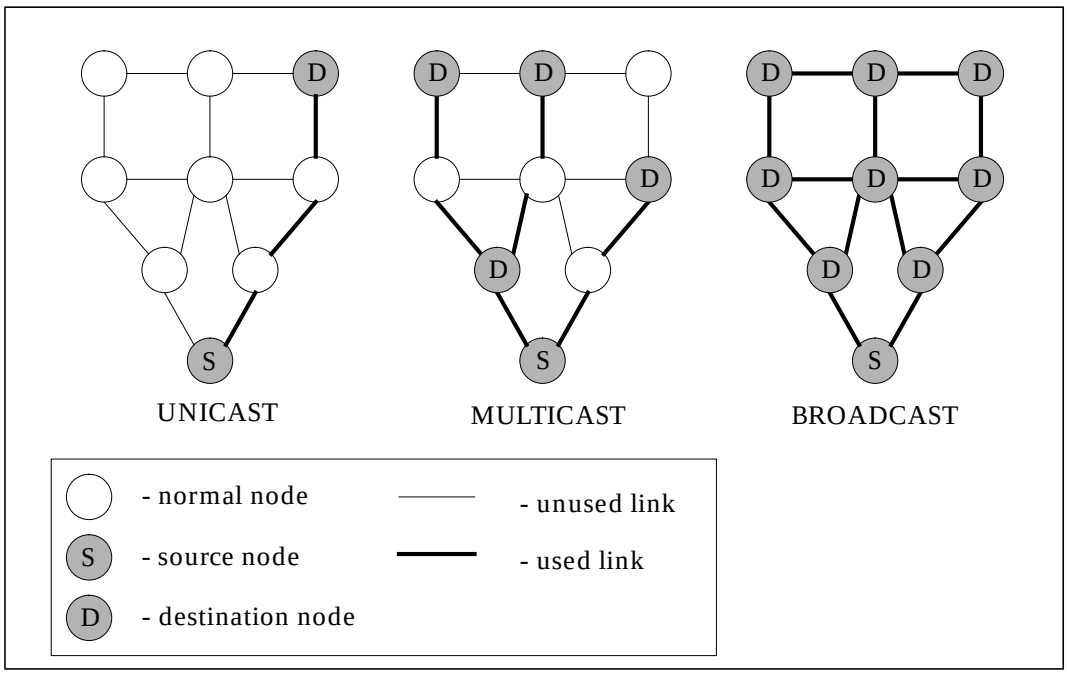

Figure 2.1: Different types of connections. *Unicast involves establishing a direct one-to-one connection. Multicast sources send the same data to many destinations - broadcasting is the form of multicasting where all the nodes in the network are destinations.*

In modern networks, there are three types of connections depending on the number of receivers. The most common connection type is unicast where there is only one transmitter and only one receiver. These connections are used for example in phone calls, telnet applications, WWW browsers and mail applications.

Multicast involves one or more sources and more than one destination. One-to-many connections are most important in practical applications and can be used for transmission of video streams to many users (for example documenting a scientific experiment), sending identical data (for example software upgrades) to many users or manipulating remote processes. Many-to-many connection is mostly used in video-conferencing where all participants see each other, and ‘group-ware’ conferencing where a group of people work on the same document simultaneously (’white board’ style).

Broadcast is a well known technique from LAN networks and involves sending the same data to all destinations within the same level of hierarchy of a network. This technique is particularly applicable in local networks due to their physical architecture (token ring, shared bus etc.). Broadcast is quite easy to achieve also in large networks with simple techniques such as flooding. This type of connection is usually used to transmit bulletins or software upgrades within a small company network

2.2 Multicast constraints

Finding the optimal connection tree in multicast is much more complex than in unicast or broadcast. The optimality of a tree depends on at least three factors, and in the case of services with specified Quality of Service (QoS), additional parameters such as delays and loss ratios must be taken in account [22]. The most important factors will be discussed below.

2.2.1 Cost

Every link in a network has an associated cost. This can be expressed as the load of the link, its delay, administrative cost etc.

It is most common (see [4]) to express the cost of the link in terms of its load and a fixed administrative weight as in the following equation

$$c = c_{load}(1 - 0.5c_{fix}) + 0.5c_{fix} \quad (1)$$

In [6], it is also proposed to base the cost function on the maximum number of receivers that can share that link. This leads to a very complicated formula with nonlinear terms.

In the previous projects, it was usually assumed that the fixed cost is equal to zero and the load of the link was based purely on its load. This simplifies the analysis even though it gives less realistic conditions of a simulation. In the following simulations, two forms of cost function will be used:

The square cost function:

$$c = l^2 \quad (2)$$

The ‘Delay cost’ function based on delay distribution on loaded links[1].

$$c = \frac{1}{10}\left[\frac{1}{1 - min(load, \frac{10}{11})} - 1\right] \quad (3)$$

The load of the link is computed according to the formula:

$$l = \frac{(ECR_{MAX} - ECR_{FREE}) + ECR_{REQ}}{ECR_{MAX}} \quad (4)$$

A few words about the ECR (Effective Cell Rate) parameters. When calculating the cost, only the maximum capacity of the link and the requested ECR of the connection is known precisely. The available bandwidth is obtained from the network state table which contains the data about all the existing links. This table is created by a mechanism defined in PNNI standards as flooding [1]. This method sends the encapsulated information about the loads, QoS parameters etc. to all the nodes in the network. Since it is always delayed, the real load of the link can differ from the one stored in the state table. The detailed description of the flooding techniques and means of alleviating the problems derived from imperfect knowledge of the network state (for example oscillations) is presented in [5]. This thesis does not cover these problems, it is assumed that the data in the network state table is always valid.

1. The difference between ‘Delay cost’ function and cell delay should be kept in mind - the first is a cost function depending on the load of the link, while the other is an actual delay of cells travelling on this link.

2.2.2 Delay

All paths involve some delay in the transfer of information. Depending on the nature of the call, it can be caused by physical distance, processing time or queuing.

This parameter is extremely important in real-time communication. While it may be acceptable to have a second or two of delay when using a WWW browser or an ftp client, such long delays make video or voice connections difficult or impossible. One of the main objectives of the multicast algorithms presented in this report is to keep the delay below the requested value.

It is worth noting that the delay factor differs significantly from the cost factor. It is sufficient to keep the delay under some specified value - human senses cannot distinguish whether their conversation is delayed by 40 ms or 80ms. In case of cost, we are interested in minimizing it as much as possible - it does make a difference if we assign the connection to the link with 40% or 80% utilization.

The following figure presents the delay concept in a more detailed way:

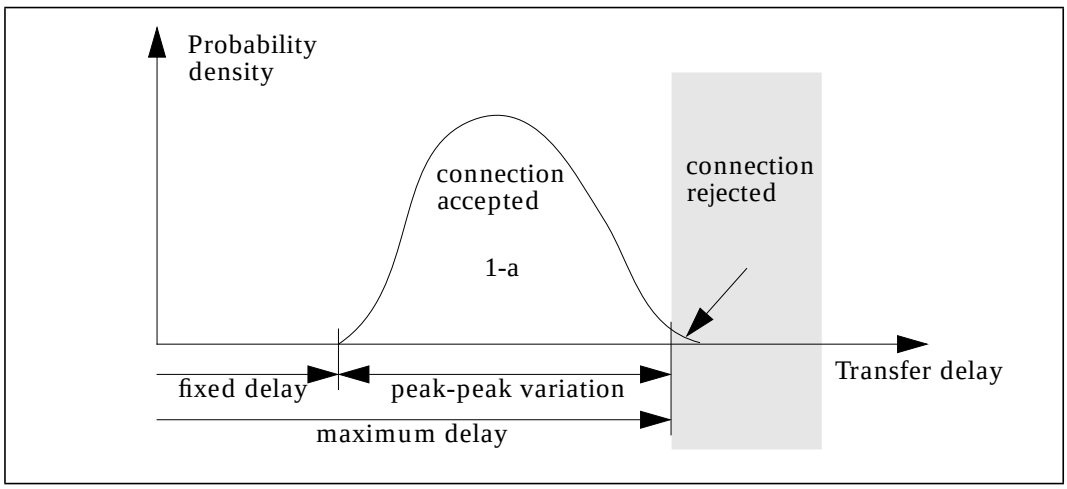

Figure 2.2: Transfer delay probability density model. *The delay may be increased if the network state changes. The maximum increase of the delay is called the Cell Delay Variation (CDV). If the increase is higher than the CDV, the connection is rejected.*

As we can see, the delay can vary - it is impossible to predict the situation of the network, so the fixed delay is only an approximation of the bottom limit of delay. The concept of Cell Delay Variation is defined in [2] - it is a maximum increase of delay (in addition to the fixed delay) for which the cell is still accepted[1].

2.2.3 Delay variation

The third important factor is the delay variation - a concept uniquely associated with multicast techniques.

Assume that we have four destinations and one source. The information travels on different paths and consequently experiences different delays. This might influence some specific applications such as process control, where it is desirable to synchronize the moment of arrival of the same data stream to different destinations. In case of ‘classical’ connections, such as file distribution or bulletin sending, this difference has no significance what

1. It is important to understand the difference between cell delay variation (CDV) and delay variation (DV). The first refers to a single link while the latter to a group of links. See Figure 2.3 on the next page

so ever. In case of real-time processing, where delay is significant, it seems to be sufficient to provide cell arrival under a fixed delay threshold - as it was mentioned before, delay differences are undetectable by human senses, so even if different video streams arrive with different delays but under the desired delay threshold, it will cause no disturbance. When it is absolutely necessary to synchronize the streams, it is possible to use buffering at a destination (see [11]), which is perhaps the most obvious approach. Unfortunately, it requires detailed cooperation between multicast group members and demands memory buffers which may be a limiting factor. Nevertheless, delay variation seems to be the least important of the factors involved in choosing optimal multicast tree. Moreover, it is important to realize that minimization of the delay variation may involve increasing delays of links and / or cost of the tree, so usually, it may not be worth the effort.

Below a graphical illustration of the DV concept is presented:

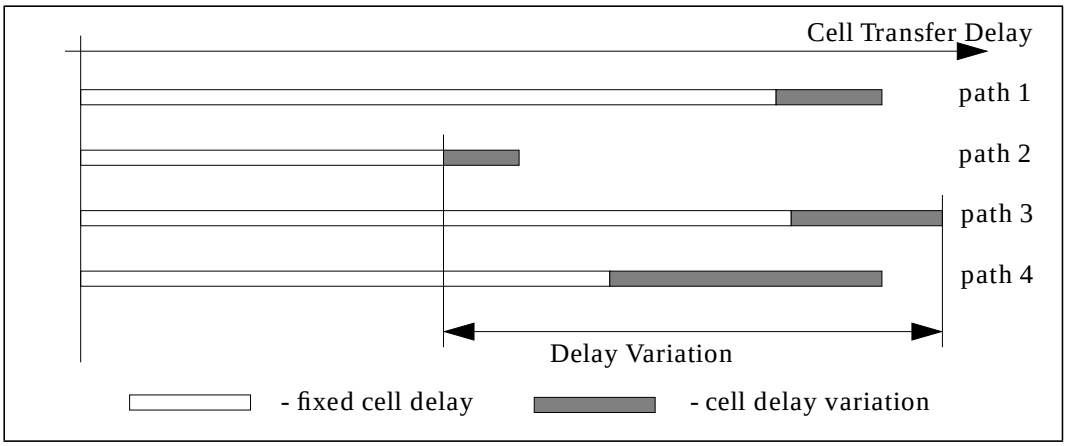

Figure 2.3: Delay variation concept. *The delay difference between paths in the multicast tree may influence the connection - in certain applications it should be kept as small as possible.*

2.3 Routing methods

This section presents an overview of the most important problems and particular methods used in multicast.

2.3.1 Where to route?

In existing networks, there are two basic types of routing:
- Hop-by-hop routing
- Source routing

The first type lets the routers between the source and the destination sequentially choose the next hop. This approach simplifies the routing process and takes into account the actual load of the links, since a router knows perfectly the distribution of traffic in its links. Moreover, such a technique requires no flooding (less bandwidth occupation) and no topology tables.

Source routing means construction of the whole routing path at the source router. This technique requires flooding and therefore has difficulties in coping with rapid changes of link loads - the topology table quickly becomes obsolete.

Multicast routing is almost always source routing. The reasons for this are the following:
- the complexity of algorithms would cause intermediate routers to slow the traffic and possibly increase their costs.
- it is very difficult to provide connection parameters like delay using the hop-by-hop technique - routers may not know the whole path and therefore cannot choose the opti-

mal link in terms of delay.
- with hop-by-hop, the possibility of loop creation increases significantly with many paths transferring the same data. This requires additional loop-detection algorithms which complicates matters even more.
- it is more difficult for hop-by-hop routers to respond to changes in network architecture caused by link failures and repairs.

In this thesis, only source routing will be investigated.

2.3.2 Multicast algorithm categories

Multicast routing algorithms generally fall into two categories: shortest path and Steiner tree algorithms [7].

Shortest path algorithms minimize the distance from source to each destination. The distance can be expressed in terms of delay, cost or other factors. This class of algorithms is derived from classical unicast algorithms (like Dijkstra or Bellman-Ford) and the problem is always solvable in polynomial time.

The Steiner tree algorithms minimize the total cost of the multicast tree. These problems are NP- complete [7] - the computational complexity is significantly higher than shortest path algorithms. Moreover, there are no simple and efficient algorithms solving the problem for an arbitrary network - there are only heuristics which, in the worst case, give a tree with twice the cost of the optimal tree [7][8][9].

The difference between the outcomes of both algorithms is presented below:

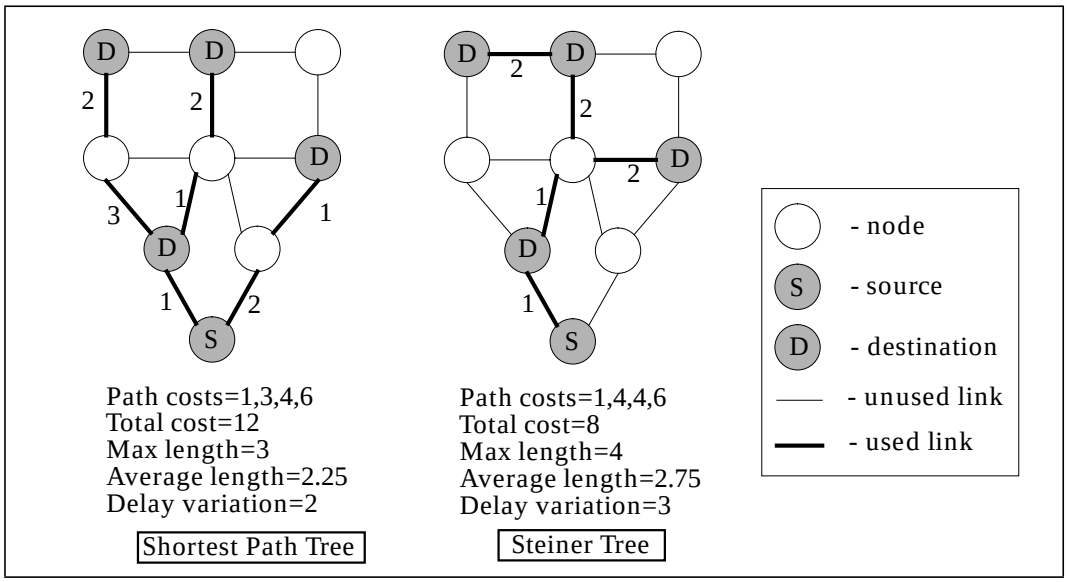

Figure 2.4: **Comparison of shortest path and Steiner trees.** *In the Shortest Path Tree, the individual costs of the paths are minimized. The Steiner Tree algorithm tries to minimize the overall cost of the tree, which results in longer paths and increased Delay variation.*

Comparing the above simple examples, one can easily see that the Steiner Tree is only optimal in terms of cost - its delay and delay variation performance are worse than the Shortest Path Tree, which is only optimal in terms of delay. It is, therefore, obvious that neither of these algorithms is general, and the only solution is a combination of both techniques providing sub-optimal parameters for cost, delay and delay variation.

It is, in fact, extremely complicated to derive an algorithm finding an optimal tree with respect to all three constraints discussed in the previous sections. The problem is also proved to be NP-complete and is often referred to as constrained Steiner tree problem.

2.3.3 Shared tree vs source specific trees for many-to-many problem

The above considerations were valid mostly for one-to-many connections. In many cases, however, it is necessary to establish many-to-many connections, where a set of transmitters and a set of receivers communicate with each other using the same connection.

There are two approaches to this problem:

a) Source specific multicast trees.

This method is a straightforward application of one-to-many technique to all sources of the group. Each source constructs this type of tree to all other sources and keeps track only of its own connections.

b) Shared tree.

This method assumes the creation of one tree connecting all members of the group - it is a more effective technique in the sense of resource utilization.

A graphical example of both techniques is presented below:

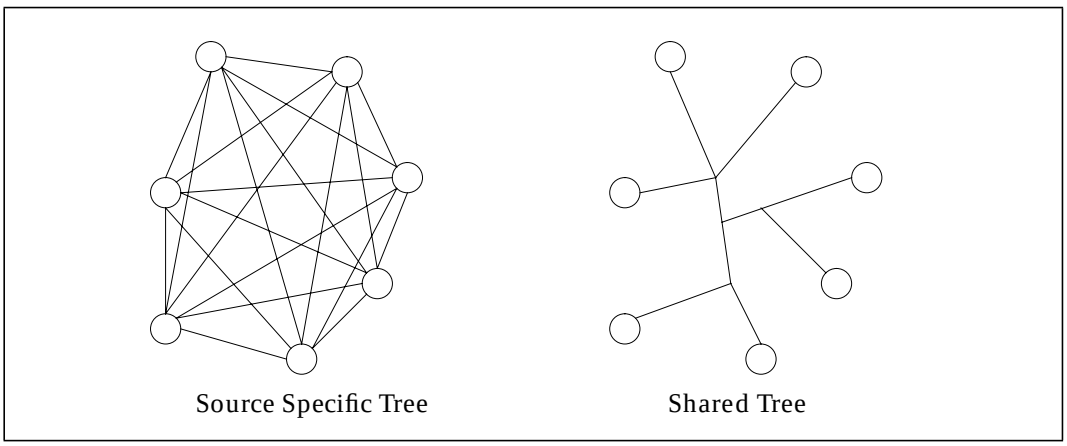

Figure 2.5: Examples of Source Specific and Shared Trees. *The Source Specific Tree is the mesh of many one-to-many multicast trees. The Shared Tree is a single tree connecting all the multicast group members.*

Both techniques have their strong and weak points:

Source Specific Tree (SST)	Shared Tree (ST)
construction of the tree is straightforward	construction of the tree is complicated
traffic is spread over many links	traffic is concentrated in few links
it is easy to keep delays in range	delays are usually longer than in SST
each source has to keep track of all connections - good synchronization protocols are needed	the management of the tree requires less processing and signalling - much better scalability than SST
dynamic reorganization of a tree is difficult - all sources have to recalculate their trees	dynamic reorganization of the tree is much easier - the new nodes need just to attach to the closest node of the existing tree

Table 2.1: Comparison of Source Specific and Shared Trees

The main limiting factor involved with ST method is the calculation of the optimal tree. To facilitate that calculation, it is a standard procedure to choose a node which is a centre of a shared tree - that allows for much simpler algorithms to be used in path calculations. Such a centre can be one of the following types:

a) **administrative centre**
- maintains the ST (i.e. joining and leaving nodes)
- keeps ST members informed about the current status of the connection
- does not have to be a destination node

b) **distribution centre**
- first two tasks are identical to the administrative centre
- is an intermediate destination for all the sources
- additionally processes the incoming data

The data flow controlled in the administrative manner travels over the shared tree without having to pass through the centre. In the case of a distribution centre, first a unicast link between source and the centre is established, and then a one-to-many connection between the centre and all the receivers is set up. It may, therefore, happen that the same data will traverse the same link in two opposite directions. Also, delays can be significantly higher in case of distribution centres.

However, as suggested in [6], a distribution centre can adjust data flow when different destinations demand different quality of service. Video distribution is a good example - some receivers may have insufficient bandwidth or processing power to accommodate the full resolution video flow. There is, however, a possibility of sending packets of data containing only basic data and other packets with supplementary information (Fourier transform for example separates low frequency components from high frequency data, which is responsible for better resolution). The distribution centre can create two trees - one for basic and one for supplementary data, and send information in a desired way. Unfortunately, there has been little research done in that field.

Using the ST also involves a non-trivial task of choosing a centre. It is a crucial issue, since the centre will be responsible, not only in the sense of traffic management, but also for its dependability - any failure in the centre will cause a total blockage of all multicast connections. To alleviate this problem, some protocols use multiple ST with more than one centre - the interested reader can find more information about this in [7], [18] and [20].

Results of Wei and Estrin's research (quoted in [7]) show that the SST achieves on average up to 20% smaller delay and 30% less traffic concentration than ST serving the same multicast group. The cost of ST is, however, on average 10% less than SST.

2.3.4 Dynamic multicast tree construction

Another important issue in multicasting is the dynamic reorganization of a tree. It may happen that a node wants to leave or join a tree during the established session. While leaving the tree involves simply pruning of the appropriate leaf, joining the session is a considerably more complicated problem.

The simplest algorithm was proposed by Doar and Leslie ([7]) - the joining node simply connects to the existing tree over a shortest path to the source. Their results show that on average, the cost of such a tree is 50% higher than the classically created tree. While it might seem reasonable for only one additional node (taking into consideration the simplicity of the algorithm), the increase of traffic after several joins can be many times higher than the optimal one. These results will be verified in Chapter 5.

Another approach assumes joining to the closest node already belonging to the tree - this algorithm performs better in terms of cost but there is no guarantee of keeping the delay of the new path under the requested limit.

Kadirire ([7]) suggested to generate the original tree so that the average distance to all the other nodes is minimal (GSDM algorithm). Such a tree can be more easily accessed by new nodes, but its performance can be far from satisfactory. Such approach is rather questionable since building a specific tree just because some nodes may wish to join it in future is a clear waste of resources. Moreover, multicast connections involving human interaction are most likely to be quasi-static (it is hard to imagine a video conference in which the participants change every minute) so it should be enough to employ some of the simple techniques.

Generally, the ST algorithms are much easier to adapt to dynamic reorganization:

- they involve fewer links than SST
- it is usually sufficient to create a link to the closest node in a ST
- the centre can take care of all 'hand-shakes' and perform so-called root-initiated join [1] which facilitates the procedure

The SST algorithms involve exchanging messages between all the members of the group following leaf-initiated join. In light of the research already conducted in the field, it is also necessary to at least partially reconstruct each tree. This features make this type of connections only suitable for very small groups with acceptable 'drop-outs' of connections.

3 ATM multicasting

In the following chapter, particular features of Asynchronous Transfer Mode which can influence the multicast routing are presented. A short discussion of advantages and disadvantages of ATM is included, as well as examples of simple hardware multicasting switches.

3.1 Why multicast in ATM?

The structure of today's networks is highly heterogeneous - there exist technologies based on simple token ring LAN technologies and advanced fibre optic based networks.

One of the most promising technologies which can be used over different media is ATM. Its technological simplicity allows for fast and reliable transfers and good scalability. It is also the first widely used technology in which the issues of guaranteed quality of transfer were addressed.

The previous chapter describes the constraints needed to successfully transmit multicast quality-sensitive connections over the networks. Two of the parameters were the delay and the delay variation. The ATM standard uses the so called Quality of Service (QoS) concept [2], [22]. It is a set of parameters characterizing the performance of an ATM connection:

Negotiable parameters	**CDV**	**Peak-to-peak Cell Delay Variation**
	MCTD	**Maximum Cell Transfer Delay**
	CLR	Cell Loss Ratio
Not negotiable parameters	CER	Cell Error Ratio
	SECBR	Severely Errored Cell Block Ratio
	CMR	Cell Misinsertion Rate

Table 3.1: Quality of Service parameters for ATM. *In multicast routing, only the two first parameters are used - the links not fulfilling the other ones are simply pruned off.*

The first two parameters are the most important ones for routing. The rest of them are taken care of by a Connection Admission Control (CAC) [1] and signalling protocols [3] - they do not influence routing (except for the fact that the links with parameters violating the given set of QoS constraints are excluded from the routing). As one can see, this concept facilitates the process of delay conscious routing (no additional protocols are needed). This fact, together with high capacity connections offered by ATM, makes this technology particularly suited for real-time applications.

3.2 Drawbacks of ATM multicasting

As any technology, also ATM suffers from certain limitations. The simplicity of the low level layers (fixed cell size, organization of switches etc.) makes it fairly easy to design the physical part of the network. The network management, however, is extremely complicated since it has to take into account many more factors than the traditional network protocols do.

A short presentation of ATM principles follows.

3.2.1 ATM basics

In ATM, all data transport is organized in fixed size cells. The cells are forwarded between nodes via so called Virtual Channel Connections (VCC), which in turn are grouped in Virtual Path Connections (VPC). A VPC can be established between nodes over many physical links - from the network point of view these nodes become logically adjacent. This technique helps reduce the control cost by grouping connections with common paths into single units. The network deals then with groups of connections instead of a large number of individual connections. There exists also a possibility of using point-to-multipoint VCCs which directly facilitate multicasting.

3.2.2 ATM Adaptation Layer

ATM Adaptation Layer (AAL) provides the mapping services between the lower ATM layer and higher layers, thus providing semantic independence (the type of payload is not important for ATM layer). According to ITU-T I.362, it is to perform the following services:

- handling of errors
- segmentation and reassembly of larger data blocks
- handling of lost and misinserted cells
- flow and timing control

Up till now, four service types of AAL were specified:

	Class A	**Class B**	**Class C**	**Class D**
Timing relation between source and destination	required		not required	
Bit rate	constant	variable		
Connection mode	connection-oriented			connectionless
AAL Protocol	1	2	3/4, 5	3/4

Table 3.2: AAL service classification. *In present networks, connection-oriented classes are most widely used.*

Only Class C and Class D will be considered in further sections. The main difference between them lies in the connection mode. AAL 5 providing connection-oriented service is currently the most important class for data applications. The following diagram shows the cell structure in AAL 3/4 and AAL 5:

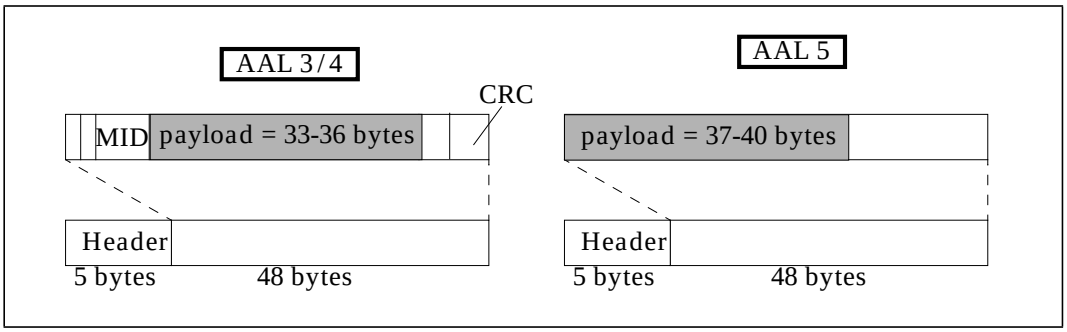

Figure 3.1: AAL 3/4 and AAL 5 cell structure. *AAL5 class does not use cell identifiers (MID) which allow for connection-less mode. Its payload is larger and processing faster.*

As it can easily be seen, the AAL 3/4 cell contains more overhead than AAL 5. Two fields are especially important: CRC (Cyclic Redundancy Code) providing error correction and MID (Multiplexing Identifier) providing cell flow control. The overhead helps to keep the error rate lower than AAL 5 (although it is usually low enough) and allows connectionless mode, while reducing the payload. The other drawback with the 3/4 mode is the increased processing time. These factors attribute to relatively lower popularity of this mode. Moreover, using the 10 bit long MID field for distinguishing between multicast destinations would seriously reduce the number of possible receivers.

The AAL 5, however, presents another problem. While being perfect for unicast bidirectional connections, it is much more difficult, if not impossible, to use multicast bidirectional connections. The reason is explained in the following graph:

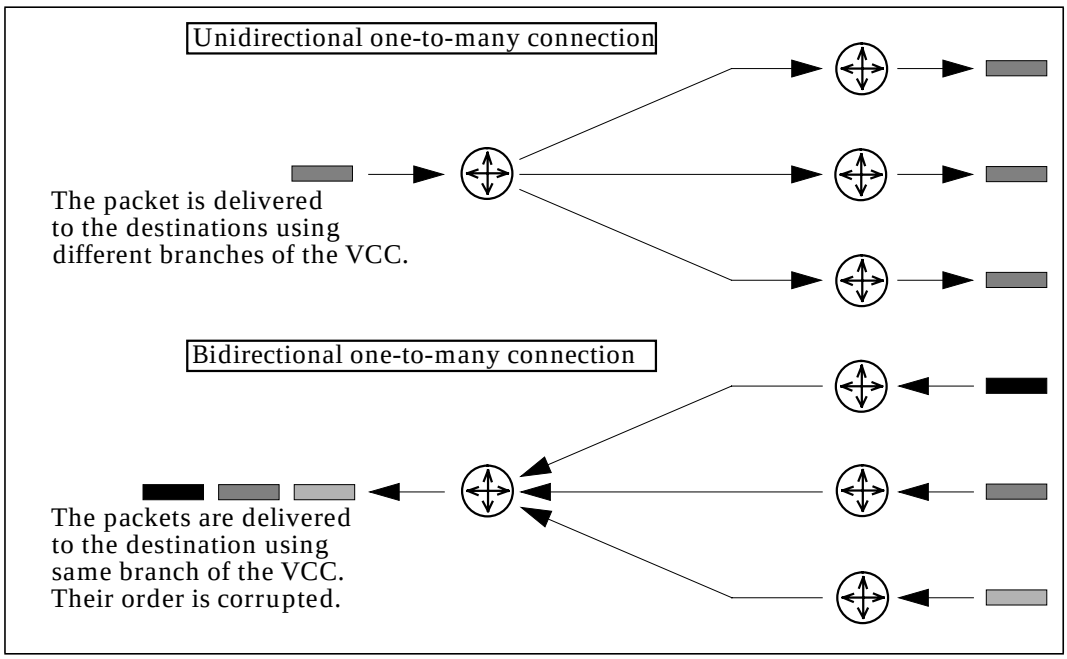

Figure 3.2: Difference between unidirectional and bidirectional multicasting in ATM. *AAL5 assumes connection-oriented mode. In a many-to-one connection, cells may arrive in a random order making it impossible to distinguish between them at the ATM level.*

The cells can arrive from many destinations in a random order and there is no easy way of rearranging them at the ATM level. This is the main reason why true (bidirectional) multipoint-multipoint (or even point-multipoint) connections are not possible in ATM at the current version of the standard.

3.3 Methods of multicasting in ATM networks

There is no simple way of circumventing the above problem. There are at least 4-5 methods ([4], [16], [17]) which aim at making multicasting in ATM possible.

3.3.1 Overlaid point-to-multipoint connections (VC Mesh)

This straightforward technique is similar to the SST routing. Each source establishes unidirectional point-to-multipoint connections with all the other members of the group. It suffers from similar drawbacks as SST: complicated procedure of joining and reduced scalability. On the other hand it can be applied without any additional changes in the network infrastructure i.e. establishing servers or high-capacity links and its performance is usually good. This method is supported by the Multicast Address Resolution Server (see 6.3).

3.3.2 Multicast Server

This concept is based on the ST algorithms with a distribution centre. It assumes an existence of a dedicated multicast server (MCS), whose task is to resequence and redirect data flows to the appropriate nodes. Each source establishes an intermediate VC connection with the server, which in turn retransmits the received packets via a point-to-multipoint connection to all the receivers, ensuring that one packet is fully transmitted prior to the next being sent. It is a more universal solution than the previous one, but demands additional investments in hardware and creates a potential bottleneck - with many sources it can take time to resequence huge amounts of data. The potential failure of the system must also be taken into consideration. A side effect of this method is that the nodes which are both senders and transmitters will receive their own packets back from the MCS. Similar to the VC mesh, this method is also supported by the MARS.

3.3.3 Multiple resequencing

A version of the multicast server is proposed in [16]. Instead of using only one server, this method proposes using many distribution centres buffering and resequencing data streams at the packet level. It alleviates problems connected with a single bottleneck, availability of VCC numbers and reduces the hardware requirements of the distribution centres. It also makes a problem of selecting a multicast tree centre less critical, since the ‘responsibility’ is spread over many such centres. This technique could be especially useful in geographically spread groups, where delays on the path source-server-destination could be too long. Using local servers connected by backbone links would reduce this problem.

3.3.4 VP-Multicasting

This technique is based on VPC routing and requires one VPC connection to be established among all members of the multicast group. Each member has its own VCC within same VPC, which provides proper sequencing of data cells. Moreover, the separate VPCs can be used to group VCCs with different QoS requirements. This technique demands, however, complex signalling protocols and means to assign unique VCC identifiers to nodes - such protocols do not exist yet. Nevertheless, this method looks very promising - upon completion of extended signalling protocols, this technique could become the most important of all multicasting methods.

3.3.5 Cell buffering

In [17] an architecture called SEAM (Scalable and Efficient ATM Multicast), was introduced. This technique assumes the existence of a so called cut-through method based on AAL5 specification. Each packet of AAL5 ends with a special EOP bit - upon receiving this bit the destination assumes that all previous cells belonged to this packet. If the switch can recognize EOP bit of different incoming VCC, it can reorganize the outcoming stream on one VCC without corruption of order of the cells. This is done with the help of buffering at the switch - the first incoming stream gets priority while the other ones are being buffered. Scalability of this technique, as well as its complexity, are, however, questionable. Two important issues arise:

- what happens if one stream is so long that it blocks the buffer?
- what happens if slow streams gets priority over a fast one?

To sum up - it is rather doubtful that this techniques will be of great importance.

3.3.6 Collision detection

Some techniques assume treating VCC as a common medium and use techniques similar to ALOHA. These methods usually use the CRC field in a trailer of AAL5 payload. This complicates the system and is only applicable to low bitrate, non-real-time applications.

3.3.7 Methods summary

It is clear that, until VP multicasting is developed, only the two first methods are widely applicable. A multiple resequencing modification of the multicast server technique may also be useful, but the detailed research in this field is beyond the scope of this thesis. Therefore, in the following chapters, only the VC mesh and the multicast server will be simulated.

3.4 Examples of multicast ATM switches

Having discussed the problem of joining VCCs, it is important to investigate construction of a switch capable of cell duplication. There are currently two approaches to the problem:

3.4.1 Integrated Switch

The integrated switch architecture is designed specially to provide the multicasting capabilities. The duplication of ATM cells can be done in so-called Shared Medium ATM Switching Elements (ASE), Shared Memory ASE or Space Division ASE. The elements are in turn grouped into larger ATM Switching Fabrics (ASF) using almost always a form of space division.The simple diagrams presenting the idea of the integrated switch is shown below:

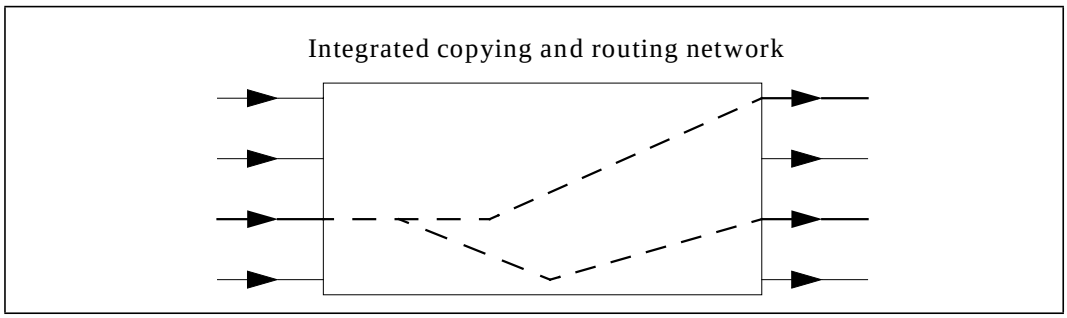

Figure 3.3: **Integrated Switch architecture.** *The switch is designed specially for multicasting - any modernization of the existent network involves replacement of old switches.*

The most important problem connected with this type of switch is that it usually requires exchanging current unicast switches with the new, multicast capable ones. This may preclude fast and cheap updating of networks.

The other general problem is usually a limited number of cell copies possible to create simultaneously. To circumvent this problem, it has been proposed to introduce so called recursive copy generation, which introduces a reverse loop enabling the cells to re-enter the router and leave on a different output port. After recirculation, the cells can be treated as normal unicast cells or be duplicated again. In [15], a 16x16 binary multicast switch with recursive multicast tree creation is evaluated, and its performance seems to be satisfactory. This type of switch suffers, however, from the necessity of having an internal clock multiplier, since the cell duplicating process must be done within one cycle of the outside network. This can create problems with using this kind of a switch in very fast networks.

3.4.2 Cascaded Switch

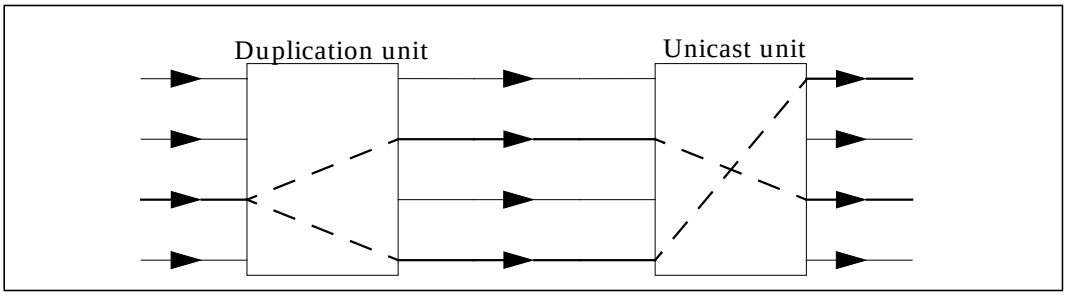

Figure 3.4: Cascaded Switch architecture. *The Duplication unit can be added to the existent unicast switch making it easy to upgrade the network to support multicast.*

This architecture consists of two separate devices. The first one performs only duplication of cells, whereas the second one is a classical unicast switch. The most important advantage of this architecture is the fact, that currently existing infrastructure of ATM switches can be easily adapted to support multicast just by adding a duplication unit to the existing unicast devices. The cost of such an update can, however, be higher than a dedicated switch with the integrated architecture.

Summing up:

both architectures are equally good and their application depends only on the desired structure of network: for homogenous multicasting networks, it is better to use dedicated switches and for networks with only few nodes supporting multicast, a better solution is to upgrade these nodes with duplication units. Referring to the previous considerations: multicasting with many overlaid connections requires almost every node to support the extended features, whereas the server based multicasting can work with fewer number of such nodes (especially with more than one resequencing centre). The hardware structure of the network therefore plays an additional role in choosing the type of multicasting strategy.

4 Algorithms

4.1 Existing algorithms and their applications

In the last 10 years there has been a considerable research aimed at obtaining universal multicast-capable algorithms. Apart from the three constraints discussed in the previous parts, there are two more important factors influencing the choice of algorithm:

- complexity of the algorithm
- allowing both symmetric and asymmetric networks (different properties along the links depending on the direction of traffic)

4.1.1 Classical algorithms

The most general classification of classical algorithms divides them into unconstrained shortest path (USP) algorithms and constrained shortest path (CSP) algorithms.

a) **Unconstrained Shortest Path algorithms**

The most popular USP algorithm is the well known Dijkstra algorithm. It performs very well in almost all cases, creates no loops and has moderate complexity. It always finds the optimal tree in only one aspect - cost or delay. When used with delay, it is usually referenced as Least Delay (**LD**) Dijkstra, when used with cost - Least Cost (**LC**) Dijkstra.

The optimality of the algorithm in terms of only one variable is a serious disadvantage in multicast - the LC tree may violate delay constraint (see section 2.2.2). When used with delay in LD algorithm, the cost of the tree may be much higher than necessary, causing an additional increase in blocking. On the other hand, if the Dijkstra algorithm cannot find a tree within requested delay limit, no other algorithm can.

The LC tree is only an approximation to the Steiner Tree problem - usually it performs much worse. There is, however, a class of algorithms designed to find the sub-optimal tree in terms of cost only. The most known ones are:

- **KMB** (Kou, Markowski, Berman) - optimal only for symmetric networks, only 5% worse than ST [8]
- **RS** (Rayward, Smith) - best one, only for symmetric networks [8]
- **TM** (Takahashi,Matsuyama) - simpler than KMB, similar performance [6].

b) **Constrained Shortest Path algorithms**

The above algorithms suffer from their inability to constrain the tree within a given delay limit. While it is possible that their outcome fulfils all the constraints (especially in small networks with short links), it does not have to be this way generally. The following algorithms attempt to find the lowest cost tree within the given delay and/or delay variation constraints:

-**BSMA** (Bounded Shortest Multicast Algorithm) - algorithm starts with constructing the LD tree, then it replaces high cost links with low cost links keeping the required delay limit. It always finds the constrained least cost tree if it exists [9].

-**CCDVMA** (Cost Conscious Delay and Delay Variation Multicast Algorithm) - this algorithm uses the k shortest paths Dijkstra algorithm giving k least cost or least delay paths. It minimizes all three multicast constraints [11].

-**CDK** (Constrained Dijkstra) - this is a name of a class of algorithms trying to merge the properties of the LD and LC trees [9].

-**FB** (Fallback algorithm) - this sequential method uses the Dijkstra algorithm to minimize the weighted linear function of delay and cost. If the delay constraint is violated, the weight for delay is increased and the algorithm starts again.

-**KPP** (Kompella, Pasquale, Polyzos) - a version of the constrained Dijkstra algorithm

with integer delay constraint. It ensures finding the constrained tree if it exists.[9]
-**MSC** (Multicast SemiConstrained algorithm) - simple algorithm based on the LD Dijkstra algorithm. It chooses the best path to a destination node from the paths containing its neighbouring nodes [9],[10]. A simple modification allowing for delay variation control is proposed.

4.1.2 Neural network algorithms

Apart from the classical methods of multicast tree calculation, a class of algorithms using neural networks has been proposed. Their main advantage is that, instead of sequential processing of the topology data, they calculate the whole multicast tree in a parallel way.

a) **The Random Neural Network (RNN)**

The RNN architecture is presented in [14]. The algorithm starts with a traditional algorithm finding the sub-optimal Steiner Tree, for example RS or KMB. After that, a single neuron is assigned to each vertex in the graph, and excitatory and inhibitory weights are adjusted to represent the cost of each edge. In the next step, the procedure enumerates different spanning trees with costs lower than the cost of the solution given by the original heuristic - this way the result can never be worse.

The simulations show that the heuristics with RNN improvement are better in (on average) 10-15% of the cases. This result is not very impressive, especially keeping in mind the following disadvantages:

- the graph has to be symmetrical - there is only one connection between neurons representing the link cost
- the algorithm does not minimize the delay in the multicast tree
- the complexity of the algorithm is higher than already relatively complex algorithms finding sub-optimal Steiner Tree - the improvement in performance is not significant enough to justify increased computational load

Summing up: it is doubtful that this algorithm will be of any great importance in practical multicast routing.

b) **Potts Neuron Approach**

In [13], an interesting algorithm solving the Multiple Shortest Path problem is presented. Instead of assigning a fixed system of neurons to a network, a group of Potts neurons is used for each connection request defined by a start and an end node. The network assigns a loop-free path to each request (treated as a unicast request), minimizing their respective total lengths and keeping the traffic load within link capacities. The problem reduces to one-to-many multicast when all starting nodes are the same - it is impossible to use the algorithm directly for many-to-many connections since the outcome of routing can consist of several unconnected trees (see [13]). The algorithm has the following advantages:

- the graph can be asymmetrical
- the algorithm minimizes the delay while keeping the load of the links under specified capacity
- the simulation of the algorithm shows that the algorithm yields good approximate solutions

The neurons in this method use information only from the neighbouring nodes. While being attractive for distributed implementations, it also creates problems discussed in Section 2.3.1 (hop-by-hop routing). The other problem generally connected with neural networks is that the convergence is not generally guaranteed. It may be interesting, however, to simulate the behaviour of the algorithm in future research.

4.1.3 Complexity of the algorithms

Algorithm	DV	Total cost compared to Steiner Tree	Complexity	Comments
UNCONSTRAINED SHORTEST PATH ALGORITHMS				
Steiner Tree	poor	1	NP-complete	reference
LD	average	< | D |	$O(\|V\|^2)$	
KMB	poor	average:<1.039 worst:<1.14	$O(\|D\|\|V\|^2)$	
RS	poor	average:<1.013 worst:<1.05	$O(\|V\|^3)$	symmetric only
TM	poor	average:<1.039 worst:<1.14	$O(\|D\|\|V\|^2)$	numbers from KMB
CONSTRAINED SHORTEST PATH ALGORITHMS				
BSMA	good	?	$O(k\|V\|^3\log\|V\|)$	arbitrary k
CCDVMA	very good	?	$O(kl\|D\|\|V\|^4)$	arbitrary k,l
CDK	good	<1.07	$O(\|V\|^3)$	
FB - Fallback	average	?	$O(k\|V\|^2)$	arbitrary k
KPP	good	?	$O(\Delta\|V\|^3)$	Δ - integer delay bound
MSC	good	?	$O(\|V\|^2)$	
improved MSC	good	?	$O(\|D\|\|V\|^2)$	
NEURAL NETWORK ALGORITHMS				
RNN	poor	?	$O(\|V\|^3 + \|V\|^2)$	symmetric only
Potts	?	?	$O(\|D\|\|V\|^3)$	depends on links

Table 4.1: Complexity and average performance of algorithms. *Complexity is given for the worst case. Shadowed entries represent algorithms chosen for implementation in PLASMA. Symbols: ? - information unavailable | V | - number of all nodes in the network, | D | - number of multicast destinations.*

It is obvious that some algorithms are rather unrealistic because of their complexity. Below is a comparison of approximate time complexity of some of the algorithms for the Milanet network with 50% of all nodes being members of a multicast group.

LD	TM	BSMA	CCDVMA	CDK	FB	MSC	i-MSC	RNN	Potts
196	1370	9430	2420000	2740	588	196	1370	2940	19208

Table 4.2: Example of time complexity of different algorithms for the Milanet network. *The numbers were calculated using the following parameters: | D | =7, | V | =14, k=3, l=3 (the last two parameters are chosen according to [11]).*

4.2 Implemented algorithms

This section presents the definition of a multicast problem and the selected algorithms designed to solve it. The selection of the algorithms was based on their computational complexity and simplicity of implementation - all of them are based on the basic Dijkstra algorithm. Only one of the algorithms can minimize delay variation: the reasons for neglecting this constraint are discussed in Section 2.2.3.

4.2.1 Definition of the problem

The network is a simple[1], connected, directed graph $G = (V,E)$, where V means the set of nodes and E the set of edges in the network. The existence of an edge $l = (u,v)$ implies the existence of the edge $l' = (v,u)$. Every edge $l \in E$ has two non-negative weights associated with it: $D(l)$ representing the delay experienced by data packets travelling on the link l and $C(l)$ representing the cost of using the link l. The network may be asymmetric, hence it may happen that: $C(l) \neq C(l')$ and / or $D(l) \neq D(l')$.

An application requires a point-to-multipoint connection with a source $s \in V$ transmitting data to a multicast group $M \subseteq V$. The algorithm should construct a multicast tree $T = (V_T,E_T)$, where $V_T \subseteq V, E_T \subseteq E$, spanning all nodes $d \in M$. Nodes which are not members of M may be contained in the multicast tree as relay nodes and simply forward packets along adjacent links.

We define $P_T(s,d) \subseteq E_T$ to be the set of links in the path from s to $d \in M$ in the tree.

Problem definition: Given a network $G = (V,E)$, a multicast group $M \subseteq V$, a source node $s \in V$, a link delay function $D{:}E \rightarrow \Re^+$, a link cost function $C{:}E \rightarrow \Re^+$, a delay tolerance Δ and a delay variation tolerance δ, does there exist a tree $T = (V_T,E_T)$ spanning s and all the nodes in M, such that:

$$\forall d \in M \qquad \sum_{l \in P_T(s,d)} D(l) \leq \Delta \tag{5}$$

$$\forall d_1,d_2 \in M \qquad \left| \sum_{l \in P_T(s,d_1)} D(l) - \sum_{l \in P_T(s,d_2)} D(l) \right| \leq \delta \tag{6}$$

$$\sum_{l \in E_T} C(l) = min(\forall E_T) \tag{7}$$

The constraint (5) is the delay constraint, the constraint (6) is the delay variation constraint and (7) is responsible for minimizing the tree cost. All implemented algorithms fulfil constraints (5) and (7). Only the improved MSC algorithm attempts to fulfil constraint (6).

In the following algorithm descriptions, Dijkstra algorithm is assumed to be known. For detailed description of this algorithm see [5]. The common procedure Add allowing link sharing is presented below:

```
Algorithm A.Procedure Add
1.      for all  l ∈ P_T(s,d)
2.          if   l ∉ E_T     E_T = E_T ∪ l
3.      endfor
4.      for all  v ∈ P_T(s,d)
5.          if   v ∉ V_T     V_T = V_T ∪ v
6.      endfor
```

1. There is only one edge between an ordered pair of nodes

4.2.2 Least Delay Dijkstra (LD)

This is the simplest possible algorithm which provides finding the delay-limited tree if only it exists. The algorithm finds the least delay paths from the source to each of the destinations and then merges them to form a single tree. The complexity of the algorithm is very low and it always finds the best possible tree in terms of delay constraint. The algorithm neglects, however, the cost factor.

Algorithm 1.LD algorithm

1. $T = (\emptyset,\emptyset)$
2. for all $d \in M$
3. find LD path $P_T(s,d)$ from s to d
4. if $\sum_{l \in P_T(s,d)} D(l) > \Delta$ return FAILURE
5. Add $P_T(s,d)$ to T
6. endfor
7. return SUCCESS

4.2.3 Takahashi,Matsuyama Steiner Tree heuristic (TM)

This is the simplest ST heuristic - it will be implemented as a reference algorithm. The heuristic starts with finding the least cost path of all the LC paths from the source to the destinations. After acquiring the path to the tree, all next destinations join the tree via the least cost path to one of the nodes already incorporated in the multicast tree. The advantage of the heuristic is its relatively low complexity - the delay factor is, however, neglected - this algorithm is a direct opposite of the LD algorithm.

Algorithm 2.TM algorithm

1. $T = (\emptyset,\emptyset)$
2. $minCost = \infty$
3. for all $d \in M$
4. find LC path $P_T(s,d)$ from s to d
5. if $\sum_{l \in P_T(s,d)} C(l) < minCost$ store $P_T(s,d)$
6. endfor
7. if $P_T(s,d) = \emptyset$ return FAILURE
8. Add $P_T(s,d)$ to T
9. for all $d \in M, d \neq$ first acquired destination
10. $minCost = \infty$
11. for all $v \in V_T$
12. find LC path $P_T(v,d)$ from v to d
13. if $\sum_{l \in P_T(v,d)} C(l) < minCost$ store $P_T(v,d)$
14. endfor
15. if $P_T(s,d) = \emptyset$ return FAILURE
16. Add $P_T(v,d)$ to T
17. endfor
18. return SUCCESS

4.2.4 Fallback algorithm (FB)

This algorithm combines the simplicity of the Dijkstra algorithm with possibility of recalculation of the tree when the delay constraint is violated. It employs a cost function being a linear combination of the load and delay of the link:

$$c = \text{CostWeight} \cdot l + \text{DelayWeight} \cdot \frac{D(l)}{\Delta} \quad (8)$$

The algorithm starts with finding the least cost tree obtained by setting the weight for the delay to 0 and using Dijkstra algorithm with the cost parameter (LC). If the obtained tree does not fulfil the delay constraint, the delay weight is doubled (in the first run it is set to 1) and the algorithm is run again. The process continues until the delay constraint is fulfilled or the maximum number of attempts reached. The complexity of the algorithm is moderate and it is easy to adapt the algorithm to different networks just by changing the weights and the number of attempts

Algorithm 3.FB algorithm

1. $T = (\emptyset,\emptyset)$
2. CostWeight=1
3. DelayWeight=0
4. for i=1 to NUMBER_OF_ATTEMPTS
5. find LD-LC tree T using CostWeight,DelayWeight
6. if $\forall d \in M \quad D(P_T(s,d)) \le \Delta$ return SUCCESS
7. DelayWeight=2xDelayWeight
8. if DelayWeight=0 set DelayWeight=1
9. endfor
10. return FAILURE

4.2.5 Multicast SemiConstrained algorithm (MSC)

This simple algorithm has the same low complexity as the LD Dijkstra but its performance, according to the simulations from [9] and [10], is very good. The algorithm constructs two matrices containing delay and cost values for LD paths from the source to the destinations. For each destination, the algorithm calculates paths visiting all neighbouring nodes i.e. if a node has five neighbours, the algorithm will calculate five paths - each of them will pass one of the neighbouring nodes. The paths with delay longer than the given limit will be excluded from the tree by setting their cost to infinity. In the next step, a node with the longest delay to the source is chosen and the path with the lowest cost leading to this node is selected. This process is repeated for all the nodes in the network. In the last step the broadcast tree is pruned to consist only of the multicast destinations.

This method finds different paths to the destinations without resorting to the more complex k shortest paths Dijkstra algorithm. The algorithm is easy to implement and modify (for example changing LD tree from the Step 1 to combined LD-LC tree discussed in FB algorithm).

Its complexity is much lower than comparable constrained tree algorithms like BSMA or KPP allowing for the easy and fast computation of the appropriate tree.

Algorithm 4.MSC algorithm

1. find LD tree T to all $d \in V$
2. construct matrix $DELAY(j,k) = min\left(\sum_{l \in P_T(s,j)} D(l)\right)$, k-neighbour node
3. construct matrix $COST(j,k) = min\left(\sum_{l \in P_T(s,j)} C(l)\right)$, k-neighbour node
4. if $DELAY(j,k) > \Delta$ $DELAY(j,k) = COST(j,k) = \infty$
5. for all $d \in M$
6. if $DELAY(d,k) == \infty$ for all k return FAILURE
7. endfor
8. $T = (\varnothing, \varnothing)$
9. for all $d \in V$
10. choose a node $v \notin V_T$ with maximum delay using $DELAY(j,k)$
11. choose the LC path $P_T(s,v)$ from s to v using $COST(j,k)$
12. Add $P_T(s,v)$ to T
13. endfor
14. for all $d \notin M$
15. delete $P_T(s,d)$ from T
16. endfor
17. return SUCCESS

4.2.6 Improved SemiConstrained algorithm (iMSC)

The MSC algorithm does not minimize the delay variation. The problem can be alleviated by a simple modification of the algorithm, without significant increase of complexity (compared to other algorithms like CCDVMA). The algorithm starts with a MSC tree and checks whether the delay variation constraint is met. If not, then the longest link in the tree is recalculated using LD Dijkstra and the DV constraint is checked once again. The algorithm stops when the DV constraint is fulfilled or when all the paths in the original tree where exchange by LD paths.

Algorithm 5.iMSC algorithm

1. find MSC tree T
2. while $\exists d_1, d_2 \in M$ $(|D(P_T(s,d_1)) - D(P_T(s,d_2))| > \delta)$
3. if $\forall (d \in M)$ were replaced return FAILURE
4. find $d \in M$ with maximum delay $D(P_T(s,d))$
5. find LD path $P'_T(s,d)$
6. replace $P_T(s,d)$ with $P'_T(s,d)$ in the tree T
7. endwhile
8. return SUCCESS

5 Simulations

This chapter covers simulations of the algorithms presented in the previous chapter. The algorithms will be tested in three different networks, with different loads and multicast group organization.

5.1 Configuration

All simulations were conducted using some common settings:

- Each simulation covers 8 hours, with measurements done after each 60 minutes of simulated time. After each hour, the background traffic is increased by 20% starting from 10% of the nominal traffic load. This way the algorithms are tested in networks with a wide spectrum of traffic - from 10% up to 150% of nominal capacity (it should be understood that the notion 'nominal capacity' is only an approximation, the real nominal capacity may be higher - that is the reason for testing traffic with 'overload')
- There are two types of traffic: bidirectional low bitrate background traffic of ordinary phone (voice) connections, and unidirectional high bitrate traffic of video connections. Voice traffic is transported using unicast. It is routed adaptively with respect to QoS parameters (see [22]). All video connections are multicasted. The parameters of both types of traffic are presented below:

-**Voice**:

CBR	CER	CLR	CMR	MCTD	CDV	DV
64kbps	1e-5	0	0	0.15s, (international WAN) 0.015s, (national WAN) 0.0015s,(Milanet)	0.0035s	-

Table 5.1: Phone service class parameters. *Only shadowed parameters are used in tests.*

- **Video**:

CBR	CER	CLR	CMR	MCTD	CDV	DV[a]
1000kbps	1e-5	0	0	0.15s, (international WAN) 0.015s, (national WAN) 0.0015s,(Milanet)	0.0035s	0.075s, 0.0075s, 0.00075s

a. this value is used only in some simulations

Table 5.2: Video service class parameters. *Only shadowed parameters are used in tests.*

where:

- The CBR values are chosen to provide a bitrate ratio of 1:16 - see [22].
- The MCTD is chosen according to CCITT G.114 Delay Recommendations which state a delay under 150ms[1] as "acceptable for most user applications".
- The CDV value is chosen according to [22].
- The DV value (Delay Variation among different paths in multicast tree) is chosen using the rule of thumb as half of the maximum delay.[2]

1. The decrease of MCTD in case of national WAN and Milanet was done to accommodate for smaller delays in the links and to obtain realistic path lengths.
2. This results in different DV for different networks.

- Traffic is measured in Erlangs -in this case, the number of simultaneous connections.
- The Network Blocking Probability is defined as the ratio of the number of rejected connection requests to the number of all connection requests.
- It is assumed that the voice connections can be multiple i.e. many connections can exist between the same pair of transmitter-receiver and the average holding time equals 300s (5 minutes). The arrival model of the video traffic is such that for each network there are on average 60 video connection requests per hour. Moreover, the number of simultaneous video connections is: 1 for Milanet network, 5 for national WAN and 10 for international WAN[1]. These numbers differ from the voice model since this type of connections is more 'serious' i.e. it takes more time to establish them and they significantly increase the load of the links. Therefore it is rather unlikely that there can be a lot of simultaneous video connections between the same sources and sinks. This model serves well also in the other types of multicast traffic- for example large data streams.
- State dependent flooding is used, new loads are flooded when the change was larger than 5%.The data is flooded immediately, without any time delay.
- Blocking is only caused by overflow of links. All Connection Admission Controls (CAC) and sinks will accept the calls if only they fulfil the QoS and load parameters.
- The only delays in the network are caused by VPCs. It is assumed that processing takes negligible time.
- Connection Admission Controls are of PeakBitrate type, which means that they only check the Peak Cell Rate (PCR) parameter of a connection. As for QoS, the only negotiated parameter is (according to ATMF - Traffic Management Specification 4.0 [2]) MCTD.
- The routers always give the whole path from the source to the destination.
- If only one multicast destination is blocked, the whole connection is abandoned (this way the routing algorithm is tested for its ability to find a complete tree).
- The multicast group members are chosen randomly, but in a way so there exist both short and long distances between the members - this way the delay varies between the paths and enables comparison of algorithms with respect to the DV parameter.
- The reader is encouraged to study Appendix A, in which the detailed description of networks is presented.

5.2 Experiment I: one-to-many multicast

In the following experiment, simulations of the most typical one-to-many connections are conducted. This type of connection is the most popular multicast - it can be used in software distribution, data distribution, process control etc.

- For each network, three different sources are chosen, each transmitting data to many receivers.
- The sources are placed in nodes with different features such as number of outcoming links, capacity of the links, density of surrounding links, switching capabilities etc.
- The number of destinations in each multicast group is approximately 50% of all nodes in the network. The receivers are distributed randomly.
- Many-to-one connections are not tested - it is assumed that a receiver wishing to establish a bidirectional connection with the source opens a separate unicast connection to it.
- Five independent simulations are conducted. As mentioned before, each simulated hour covers approximately 60 routing requests and there are three different routers (sources) - this way the total number of multicast trees generated per hour is equal to 5x60x3=900. All the presented parameters are averaged parameters of all 900 trees.

1. The number of Erlangs refer to the average number of connections - it may happen that there are more or fewer simultaneous connections.

5.2.1 Maps of networks

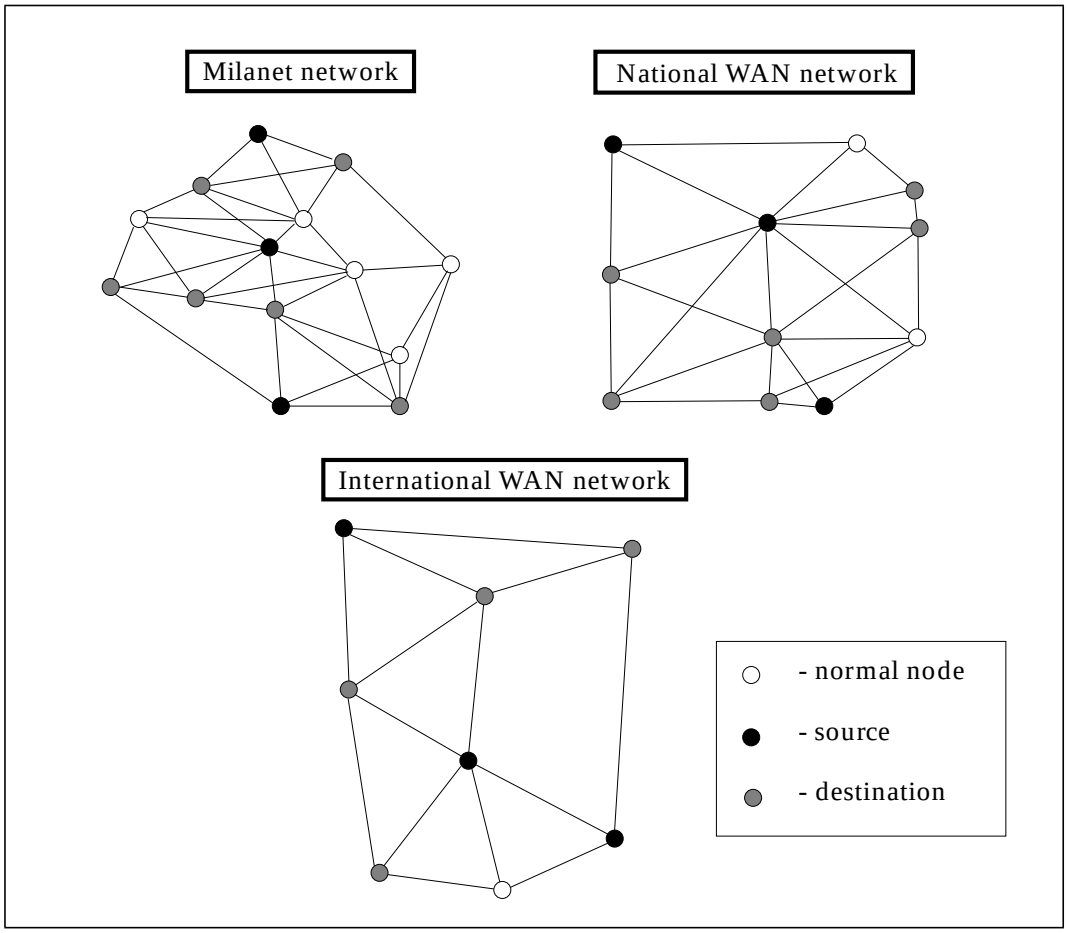

Figure 5.1: Network diagrams for one-to-many multicast. *Each source establishes one-to-many connections to all destinations. Sources are chosen in locations with different characteristics (i.e. number of links, capacity, concentration of nodes etc.). Destinations are chosen in approx. 50% of all the nodes and are placed randomly. The output of simulations is an average of the outcomes of all three multicast groups within each type of network.The detailed description of the networks (i.e. capacities and delays of the links) is presented in Appendix A.*

5.2.2 Efficiency of the algorithms

One of the most important measures allowing to asses suitability of different routing algorithms is their ability to find a proper tree with different background traffic in different networks and in different locations of the source. In Figure 5.2 one can see the comparison between the implemented algorithms (see chapter 4) in Milanet, National and International network in multicast configurations as above in Figure 5.1. The first four algorithms (that is LD, ST, FB and MSC) do not take delay variation into account. The fifth algorithm - iMSC tries to construct the tree with DV requirement fulfilled. It is the only tested algorithm which tries to minimize cost, delay and delay variation of the multicast tree.

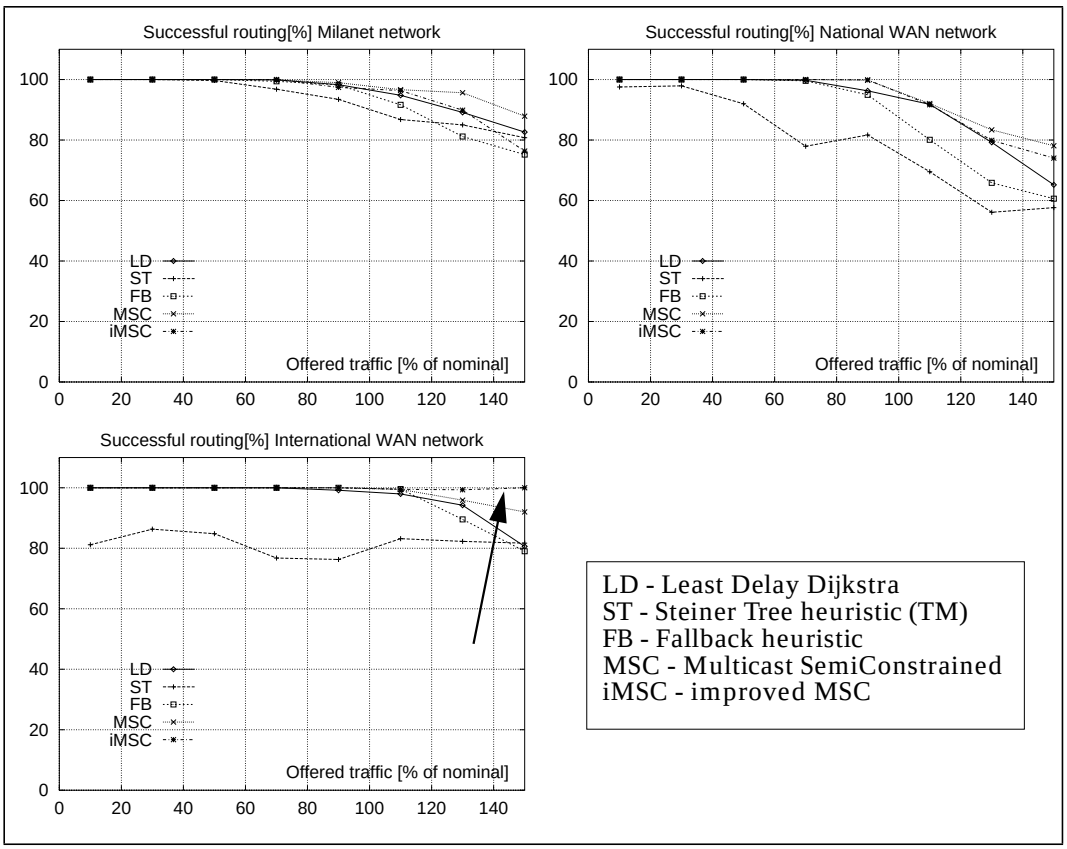

Figure 5.2: Efficiency of algorithms. *Successful multicast connections as a percentage of all multicast connection requests for different algorithms. With the increase of background traffic none of the algorithms can find the suitable tree in all cases. The Steiner Tree heuristics is clearly the worst algorithm with respect to efficiency - MSC and iMSC seem to be the best algorithms.*

In all the networks, cost-based algorithms as the Steiner heuristics and Fallback heuristic (based on Least Cost Dijkstra) do not perform exceptionally well. The clear winners are both MSC algorithms and the LD algorithm places itself in the middle of the rank.

The parameters of the trees generated by respective algorithms will be compared in the following sections but it is already clear to see that the delay is the crucial constraint - its violation discards the whole tree, the cost plays the secondary role and no algorithm concentrating on this constraint is likely to succeed - even in a relatively compact network as Milanet where delays of the links are rather small, the cost-conscious algorithms fail on average 10% more often than delay-conscious ones. In larger networks, the number reaches 20% and more.

An interesting thing can be noticed in the International network where iMSC algorithm performs better than MSC even though it minimizes also the delay variation. Theoretically, it should always be a little worse than MSC but in this case it performs better by around 7-8%. The reason for that could be the fact that this algorithm discards very long links in favour of shorter ones. This could suggest that very long paths are more likely to cause a routing failure due to the unnecessary loading of too many links. Paradoxically, the same can be said about very short paths (the LD algorithm), which congest the same shortest links coming out from the source. The optimal algorithm would therefore try to avoid both direct, shortest paths and longest (and cheapest) paths.

5.2.3 Network blocking probability

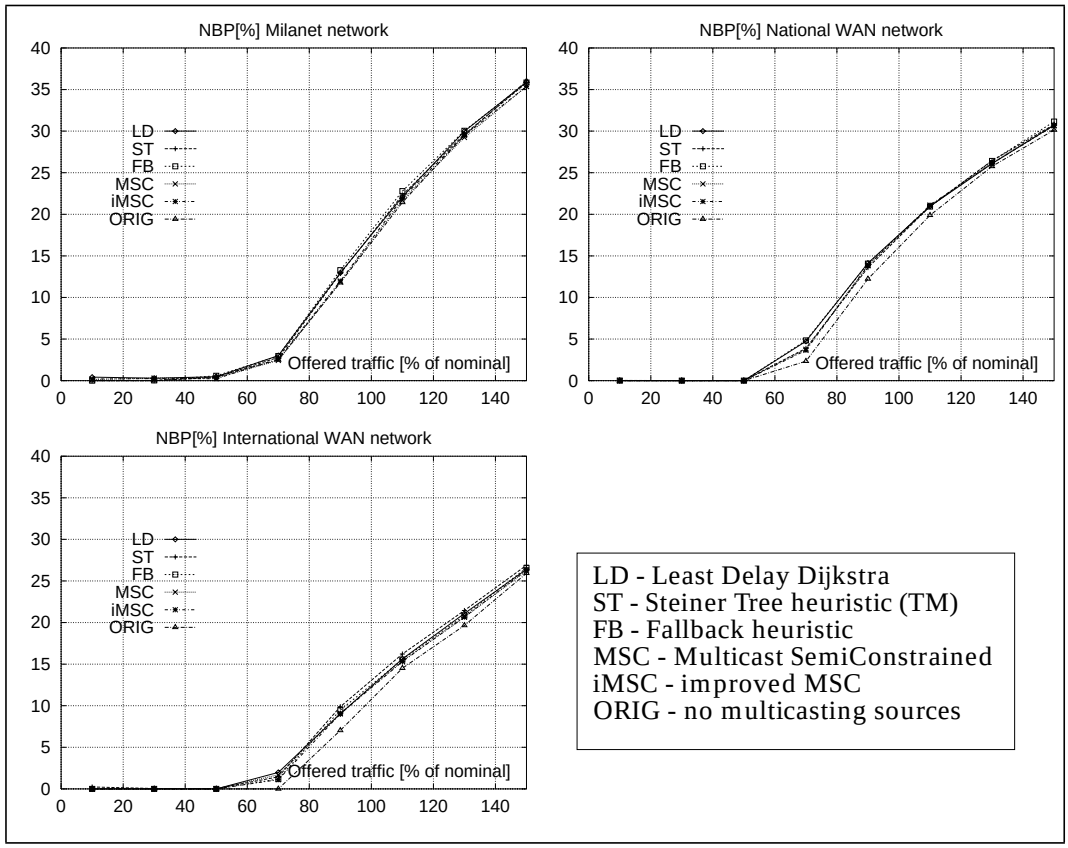

Figure 5.3: Network Blocking Probabilities. *Surprisingly, there is no significant difference in the Network Blocking probabilities of different algorithms. While they all increase the NBP of the original network with only background traffic by at most 3%, the differences between them are less than 1%.*

Another important parameter of an algorithm is the blocking probability it causes in the network. Even if we use the algorithm which manages to find the proper multicast tree in 100% cases, it will be unuseful if it causes an extensive blocking of the rest of the traffic. This is one part of the more general question of fairness (see [22]) which will not be discussed here. It is enough to say that the algorithm that causes much increased blocking probability of the rest of the traffic is usually unacceptable.

In Figure 5.3, one can see that the tested algorithms are almost identical when it comes to the global blocking probability. This is clearly an important factor influencing the choice of the algorithm - some of them perform better without degrading the background traffic.

It is worth considering why cost-conscious algorithms are not significantly better than delay-oriented ones. The most probable cause is that they tend to choose two links with lower loads rather than one link with the higher load. The form of the cost function (the delay cost function- see section 2.2.1) is such that the cost remains relatively constant until 60% of the maximal capacity of the link. That is why a link with 60% utilization will be discarded in favour of two links with 30% utilization. While this technique may work well with the relatively low bandwidth unicast connections, it tends to spread heavy traffic on too many links when high bandwidth multicast streams are transported. This is probably the reason why the FB algorithm is slightly better than the TM - it uses linear cost function of the load of the link so the above situation might be avoided. Anyway, the usability of cost-conscious algorithm seems to be questionable.

5.2.4 Cost of the tree

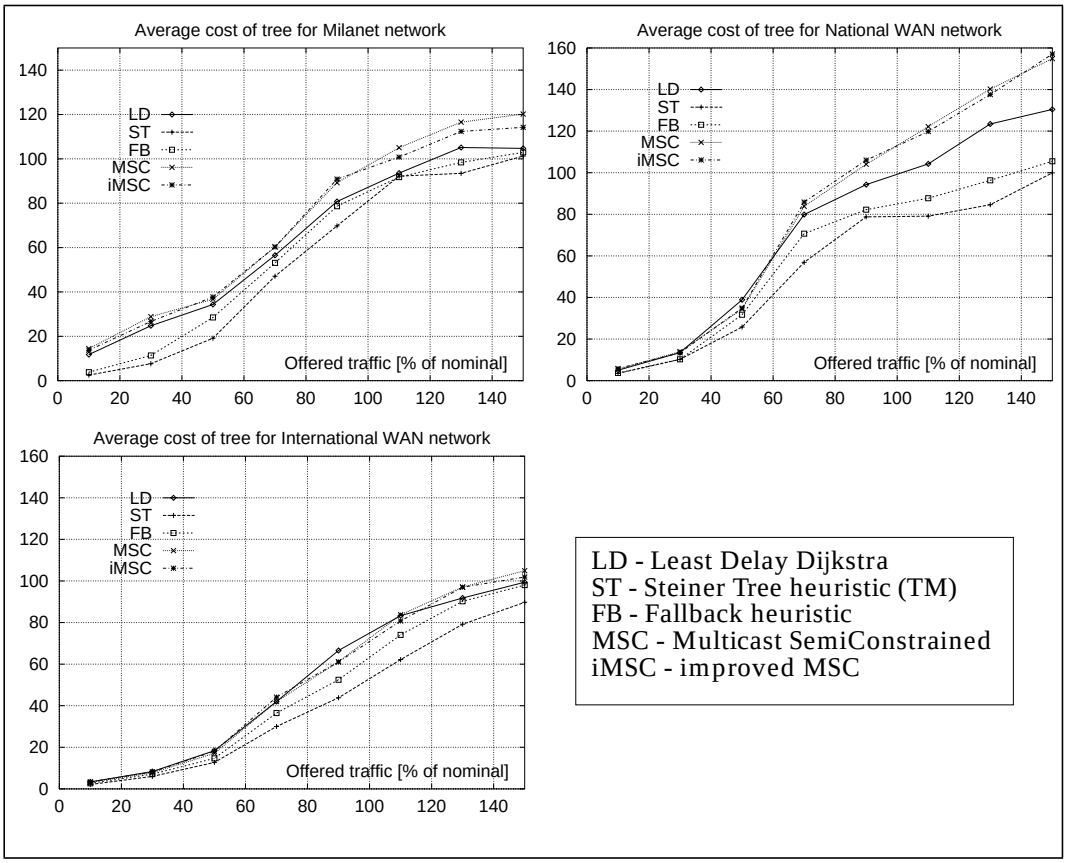

Figure 5.4: Cost of the tree. *The cost of the tree is calculated using the sum of all its branch costs. The function used is the cost delay function (see section 2.2.1).The comparison of the algorithms is rather obvious - TM has the least cost tree and the FB algorithm is slightly worse. The other algorithms are more 'expensive' - on average 20% for Milanet, 60% for the National WAN and 10% for the International WAN. As it could be seen in the previous section, a relatively large difference of tree cost between TM and MSC algorithm is not reflected in the Network Blocking Probability.*

In the above graph, one can clearly see the difference between the cost-oriented algorithms and the delay-oriented ones. The Steiner Tree heuristic (TM) can be even 60% better than the MSC tree in case of the national WAN network. In the other networks, the difference is smaller but still visible. It is, however, worth noting that this difference causes no problems with increased blocking probabilities of the rest of the traffic. One could argue that it is because the multicast connections are much fewer than unicast ones. It is of course possible that cost-oriented algorithms would perform better when the number of multicast connections would be comparable with unicast ones, but this situation is rather unlikely to happen in the real applications. The majority of all traffic would probably be traditional unicast traffic.

Another problem worth investigation, would be to include the administrative costs of the connections in the cost functions. In this case, the cost of the tree would be obviously another important factor in choosing the optimal algorithm. It could be possible, for example, to base the choice of routing algorithm on the demanded quality of connection - if the transfer delay is not crucial (like in software distribution) the Steiner heuristic could be a perfect solution, minimizing the dollar cost of the whole operation.

5.2.5 Maximum delay

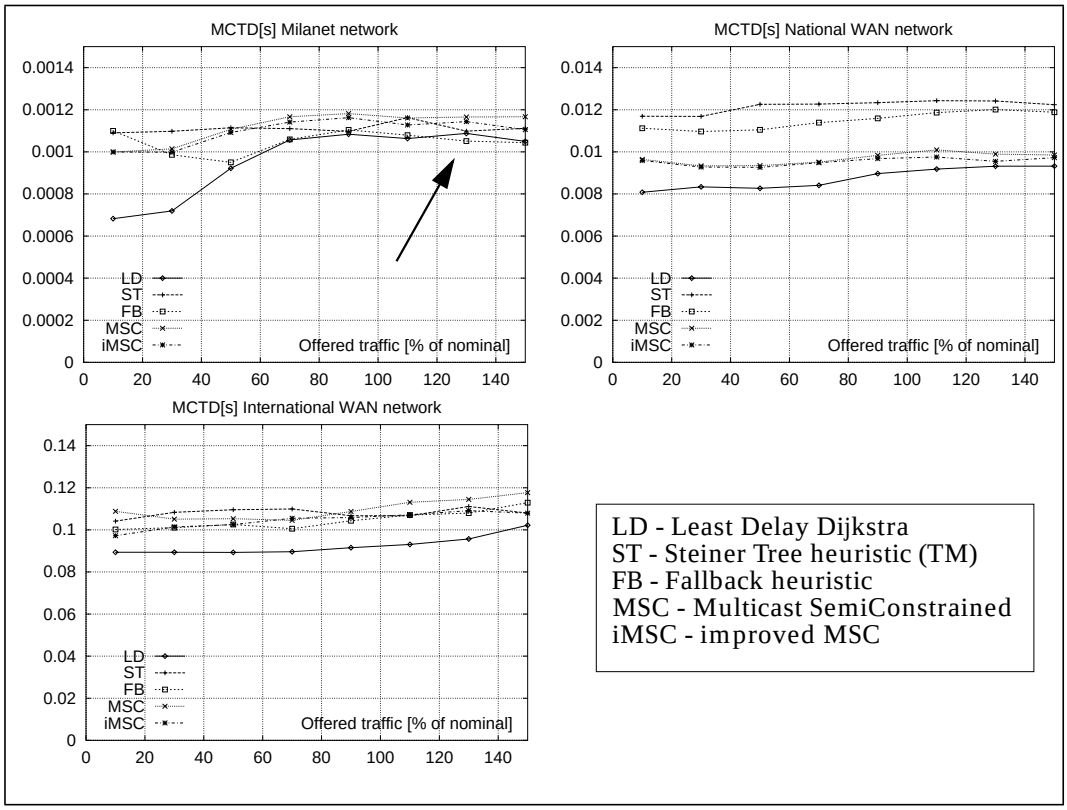

Figure 5.5: Maximum Cell Transfer Delay. *The graphs present the average maximum delay in trees generated by different algorithms in different networks. Only trees fulfilling delay constraints were taken into account. According to the definition, the Least Delay Dijkstra yields the shortest paths. The ST heuristic is the worst with respect to maximum delay.*

As mentioned before, the Maximum Cell Transfer Delay is a crucial factor for construction of a multicast tree. Contrary to parameters like the NBP or the cost of a tree, delay does not have to be minimized - it is sufficient to keep it under the specified threshold since most often the small differences between the delay are not recognizable. All the above graphs present the comparison between the average MCTD for different algorithms and networks. The LD algorithm provides minimum delay in almost all the cases - only for high traffic concentration in the Milanet network is the delay longer than for the FB algorithm. This apparently paradoxical outcome of the simulations can be easily explained - since the LD algorithm always chooses the shortest path, it may happen that it will congest the same links over and over again causing the necessity of using longer paths at the next routing. Other algorithms that respect also the cost of the link will tend to spread the traffic over many different links. This problem is most likely to appear in local networks with relatively low capacity links. As it can be seen from the other two graphs (National and International networks), the LD algorithm does give the shortest paths - these networks, however, have much higher link capacity than Milanet. The longest links are the outcome of the Steiner Tree heuristic - this is also the reason why this algorithm performs so poorly.

The rest of the algorithms keep the delay between the two above extremes. The average delay of the MSC is usually somewhat larger than the iMSC one. This reflects pruning of the longest paths and substituting them by shorter ones when violation of the Delay variation occurs.

5.2.6 Delay variation

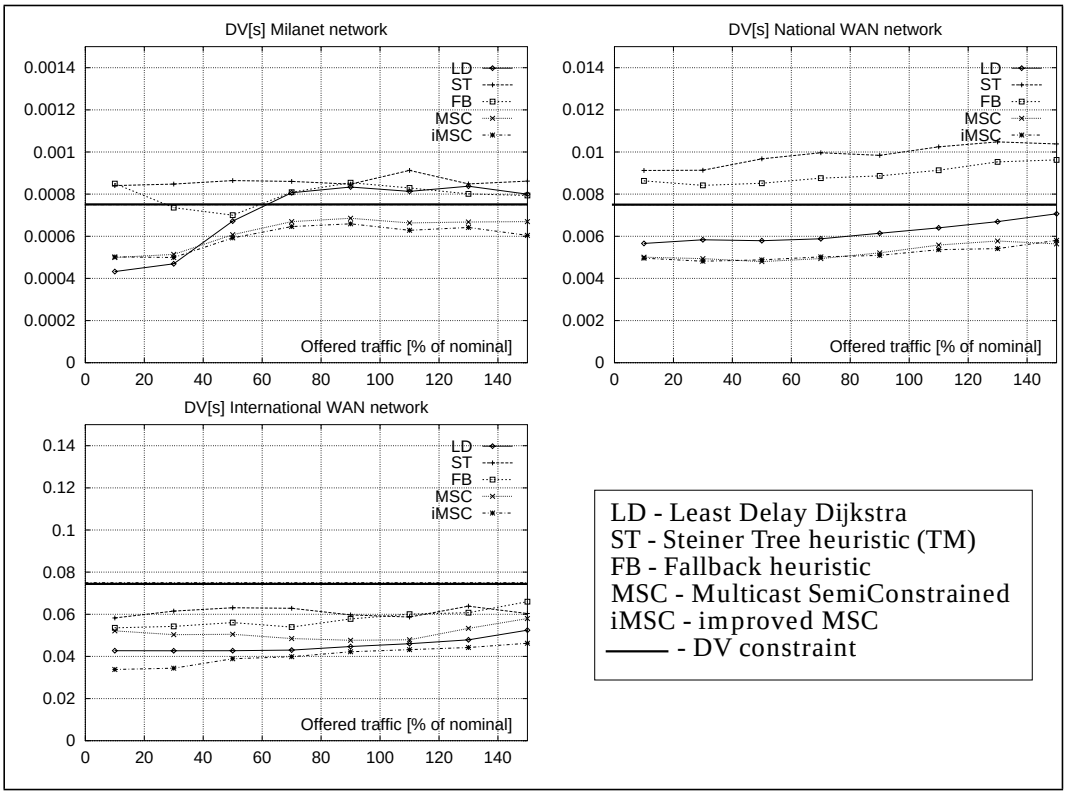

Figure 5.6: Delay Variation. *The average difference between the delays of the shortest and the longest path in the tested trees is compared in the above graphs.Only the iMSC algorithm attempts to minimize the DV constraint. For comparison purpose, the DV threshold is represented as the horizontal black line. Minimizing this parameter seems questionable - the delay-oriented algorithms usually fulfil the limit without any special attempt (LD and MSC). Moreover, in the networks with longer links and/or bigger differences between delays of links, all the algorithms seem to meet this requirement.*

In Section 2.2.3, the third parameter, apart from the cost of the tree and its maximum delay, was discussed. The Delay Variation represents a difference between the shortest and the longest path in terms of transfer delay. Although this is not a crucial factor in choosing or discarding a multicast tree, it may be interesting to compare the algorithms with respect to this parameter too. Having specified the maximum delay variation as half of the maximum transfer delay, it seems that almost all the delay-conscious algorithms fulfil this requirement. There is no significant difference between the MSC and the iMSC algorithm, indicating that it is not absolutely necessary to concentrate additional effort on minimizing the DV - a good general algorithm should produce a tree which is acceptable also with respect to this parameter.

Long links and much different link delays cause the variation of the delay to be naturally minimized - this is due to the fact that a significant change of a path delay would almost always violate the maximum cell delay constraint. In such networks, it is rather unlikely that the algorithm will choose a path crossing more nodes (cities for example) than necessary, especially when their capacity is usually sufficient for demanding connections.

5.2.7 Conclusions

According to the simulations presented in the previous sections, the best algorithm for the one-to-many problem seems to be the Multicast SemiConstrained Algorithm. It outperforms the other algorithms in all the simulated environments, while being only as complex as the basic Dijkstra algorithm. A simple modification allowing for explicit Delay Variation control improves its performance with respect to this parameter. The worst algorithm is without doubt the Takahashi-Matsushita Steiner Tree heuristic. Also the other cost-conscious algorithm (Fallback) is no competition even for the LD algorithm. Despite the fact that the trees generated by the TM and the FB algorithm have significantly lower costs than the ones generated by the other algorithms, the overall network blocking probability remains almost the same. All things considered, the obvious conclusion is that the cost-conscious methods are generally not useful for real-time applications. The delay must be the first minimized parameter - even the FB algorithm which attempts to minimize both cost and delay fails more often than the LD Dijkstra.

It is perhaps interesting to explain the good performance of such a simple algorithm as the MSC - being based on the simple Dijkstra algorithm, it usually outperforms it by at least 10%. Similar to the version of the Dijkstra algorithm that returns k shortest paths, the MSC also returns more than one path. The k shortest path Dijkstra, however, has a complexity proportional to the third power of the number of nodes in the network while the MSC algorithm keeps the complexity proportional to the second power of the number of nodes. It can achieve this by considering more than one shortest path leading to the destination node via different neighbouring nodes. If the destination node is close (in terms of the number of intermediate hops) to the source, the paths crossing its neighbours will most likely differ significantly from each other. If the destination is distant from the source, the most probable outcome of the MSC algorithm will be paths differing only in the direct proximity of the destination node which is the most obvious approach (the algorithm does not have to calculate long and impractical paths - all real paths will be similar to the least delay path). This approach makes the algorithm more suitable for real networks (reduced search time). The k shortest paths Dijkstra will find the same paths as the MSC algorithm at the expense of the exhaustive search for all paths - also the ones which could not be used in practical applications (the delay constraint). The MSC algorithm considers only the most probable paths reducing the search time significantly.

5.3 Experiment II: many-to-many multicast

In the following experiment, many-to-many connections are simulated. The application of this type of multicast is mostly video conferencing where all members transmit and receive video streams at the same time.

- Multicast groups consist of 33%, 66% and 100% of all the nodes (in case of National and International WAN, the node represents a group of close nodes connected to each other - see Appendix A)
- In the overlay multicasting method, all group nodes establish one-to-many unidirectional connection (VC mesh) to all the other members. The nodes transmit and receive data streams at the same time.
- In Core Based Trees (‘core’ is a distribution centre, see 2.3.3), one node is selected to be a distribution centre (‘rendez-vous’ point). Each multicast group member transmits to the CBT node using ordinary unicast connection. The node is then responsible for redistribution of the data to all other members of the group establishing unidirectional one-to-many connections.
- The CBT server node is chosen arbitrarily in the approximate centre of the network and with high-capacity links leading to it.

5.3.1 Maps of networks

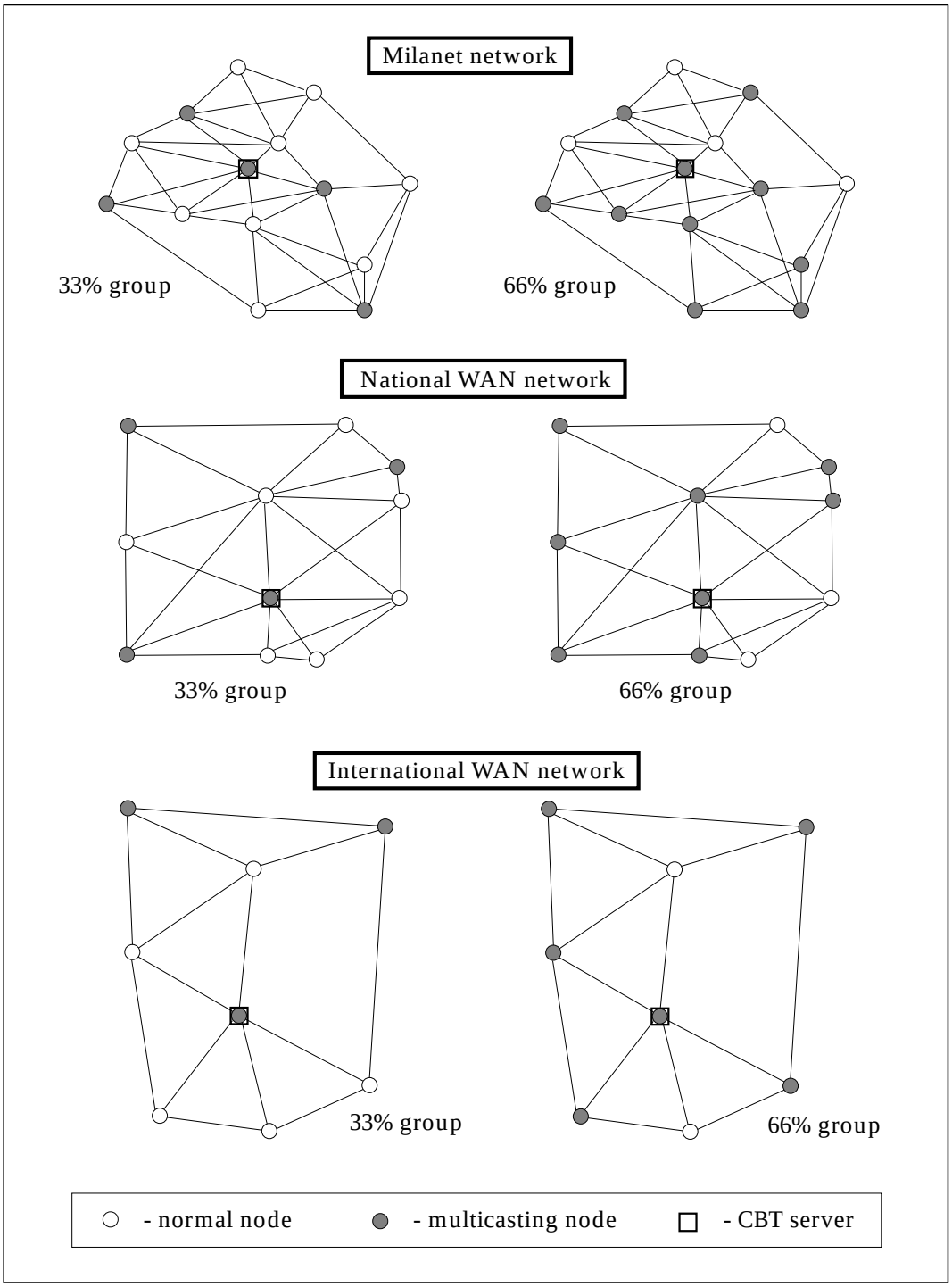

Figure 5.7: Network diagrams for many-to-many multicast. *All multicast group members are chosen randomly. If the overlaid technique is used, all members are equal, i.e. they send and receive data from all the other members of the group. In the case of Core Based Tree (the distribution centre is marked with a square), each node establishes unicast connection with the CBT server, which in turn distributes the data stream to all other members of the multicast group.*

5.3.2 Milanet

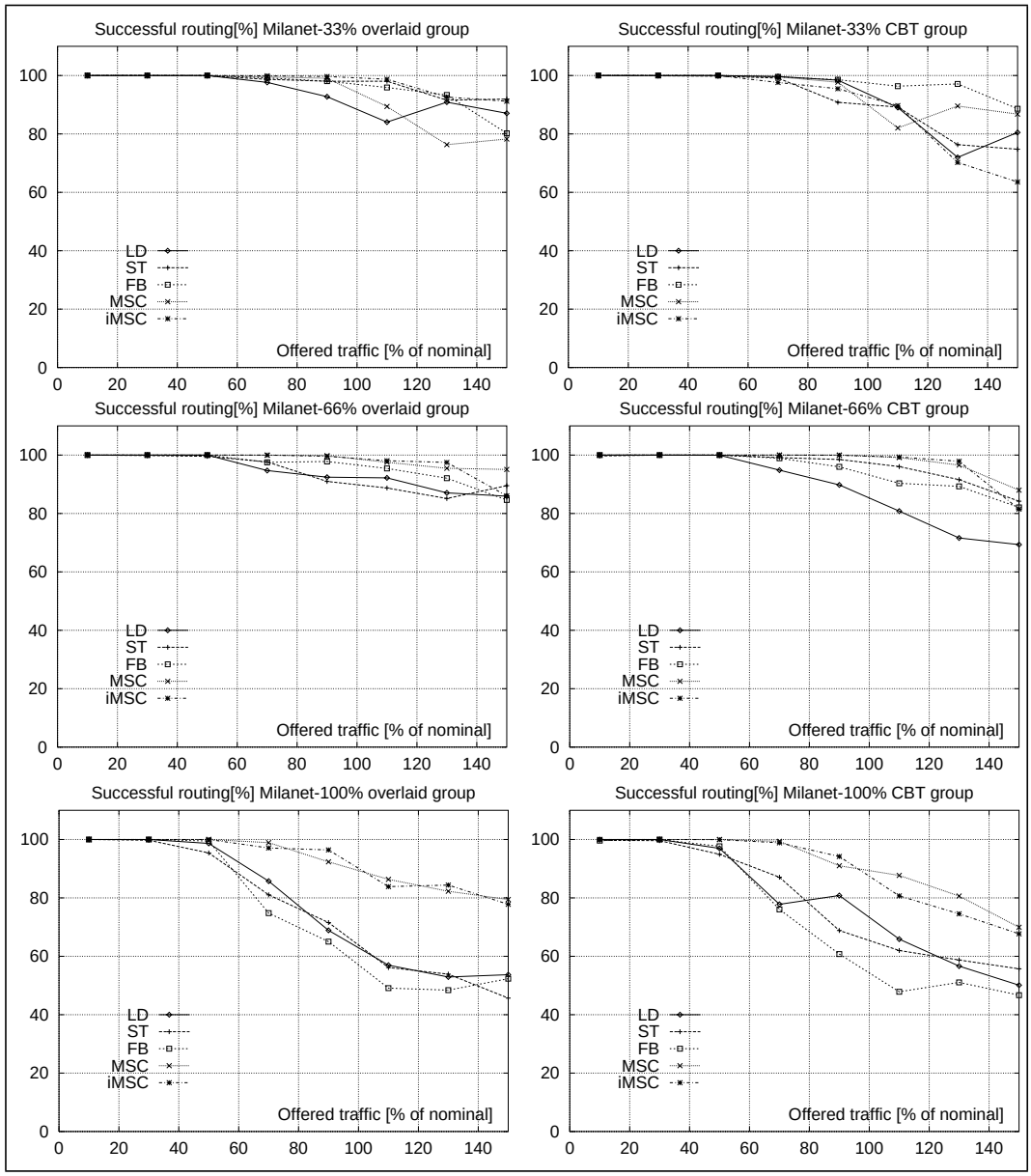

Figure 5.8: Efficiency of algorithms (Milanet). *Successful multicast connections as a percentage of all multicast connection requests for different algorithms.The Core Based Trees are more difficult to compute due to the increased delay and concentration of the traffic in fewer links.*

The above graphs show that construction of the many-to-many multicast trees is a more difficult task than the ordinary one-to-many tree. Moreover, the difference between the tested algorithms is not as conspicuous as in the Experiment I (except for the 100% group, where the difference between the cost-conscious and delay-conscious algorithms is obvious). The reason for this is probably the high traffic caused by many point-to-point connections - for such a situation, it is difficult to find a good tree for any algorithm. Generally for the overlaid trees, the MSC algorithm performs best.

It is, however, more difficult to construct a CBT tree - the concentration of the traffic around the core is usually very high. It may also happen that the same data will traverse the link in both directions, increasing the overall load.

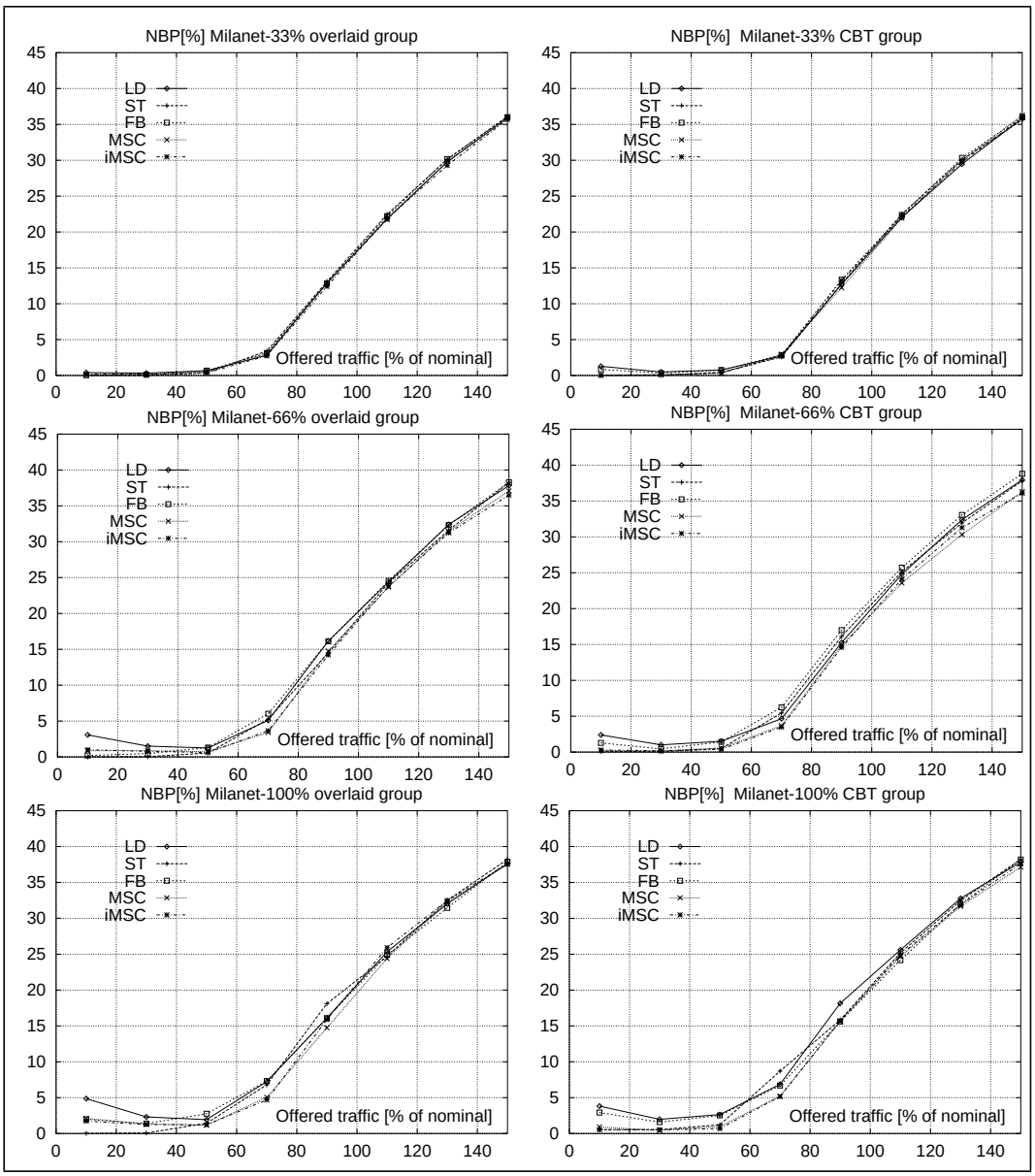

Figure 5.9: Network Blocking Probabilities (Milanet). *There is no significant difference in the network blocking probabilities between different algorithms. The cost-conscious algorithms are, however, a little worse than the delay-conscious ones.*

In the Milanet network, all the algorithms give similar blocking probabilities. Many-to-many connections increase the overall blocking of the traffic by just 5% (compare Figure 5.3 and Figure 5.9), which is surprisingly little, compared to the overall increase of traffic caused by many sources transmitting high bitrate data streams. The CBT trees have a little larger gaps between blocking probabilities of the algorithms (see Milanet 66%) than the overlaid ones. This method is particularly sensitive to even small changes in tree construction since the concentration of traffic around the core node is much higher than around any of the nodes in source-routed trees. Generally, the LD algorithm tends to block more traffic due to its policy of choosing the shortest links with no respect to their load (until they get permanently saturated and do not allow for any additional connections).

5.3.3 National WAN

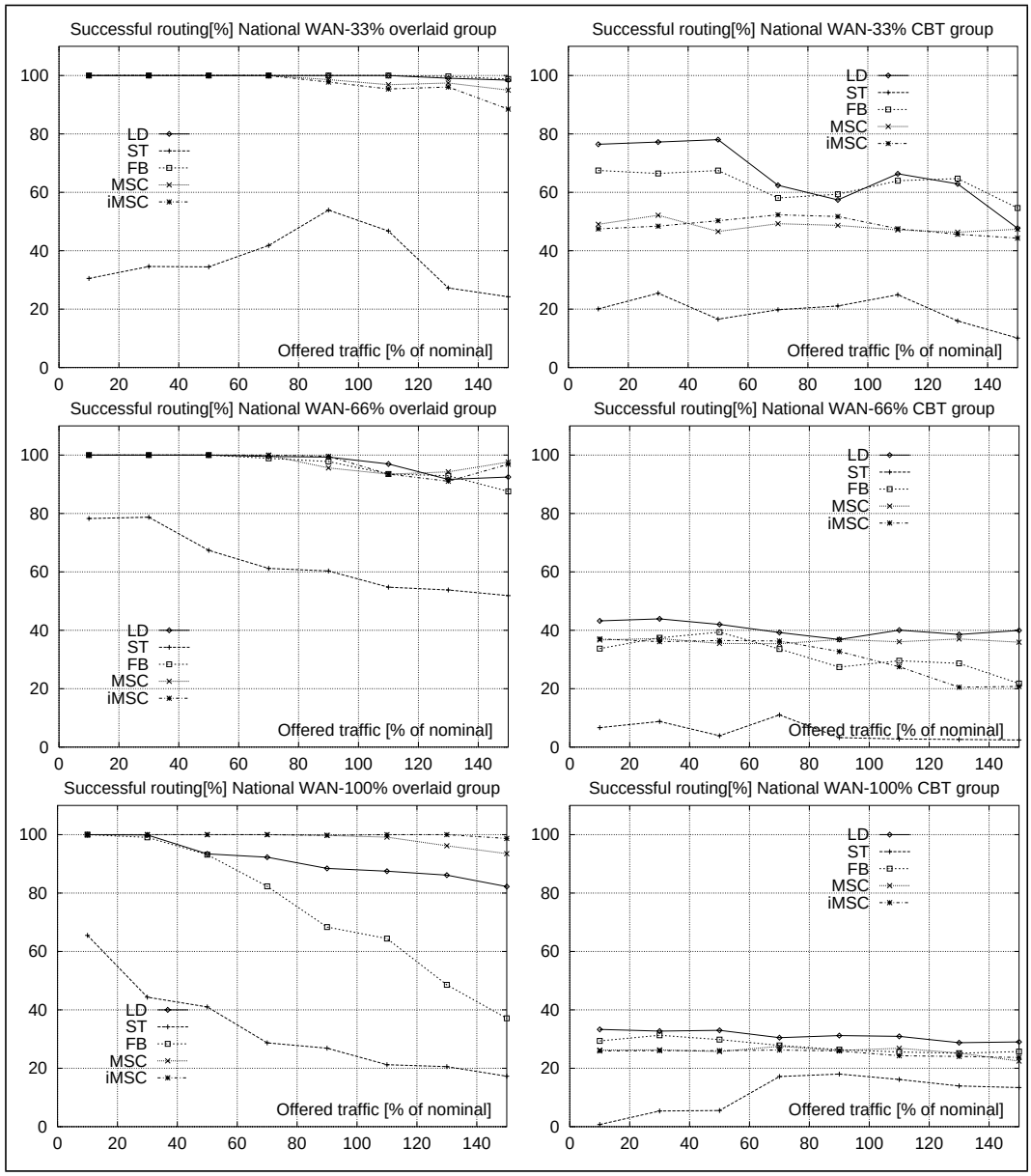

Figure 5.10: Efficiency of algorithms (National WAN). *Successful multicast connections as a percentage of all multicast connection requests for different algorithms.The CBT algorithms perform rather poorly, due to the increased delay caused by the necessity to travel the links twice when the source and the destination nodes lie close to each other.*

In the relatively high-capacity network with longer links, the ST heuristic fails in the majority of the cases. For the overlaid model, the MSC algorithms still perform very well, but for the CBT trees, the basic LD algorithm seems to create the best trees - this is probably the result of the bandwidth margin in the links - there is not so much need for minimizing the load cost of the trees but the delay is a crucial factor in the CBT trees.

The Fallback algorithm performs surprisingly well in some cases - it is linked to the fact that after a few steps it resembles the LD algorithm (the weight for the delay is much larger than the weight for the cost - see Section 4.2.4).

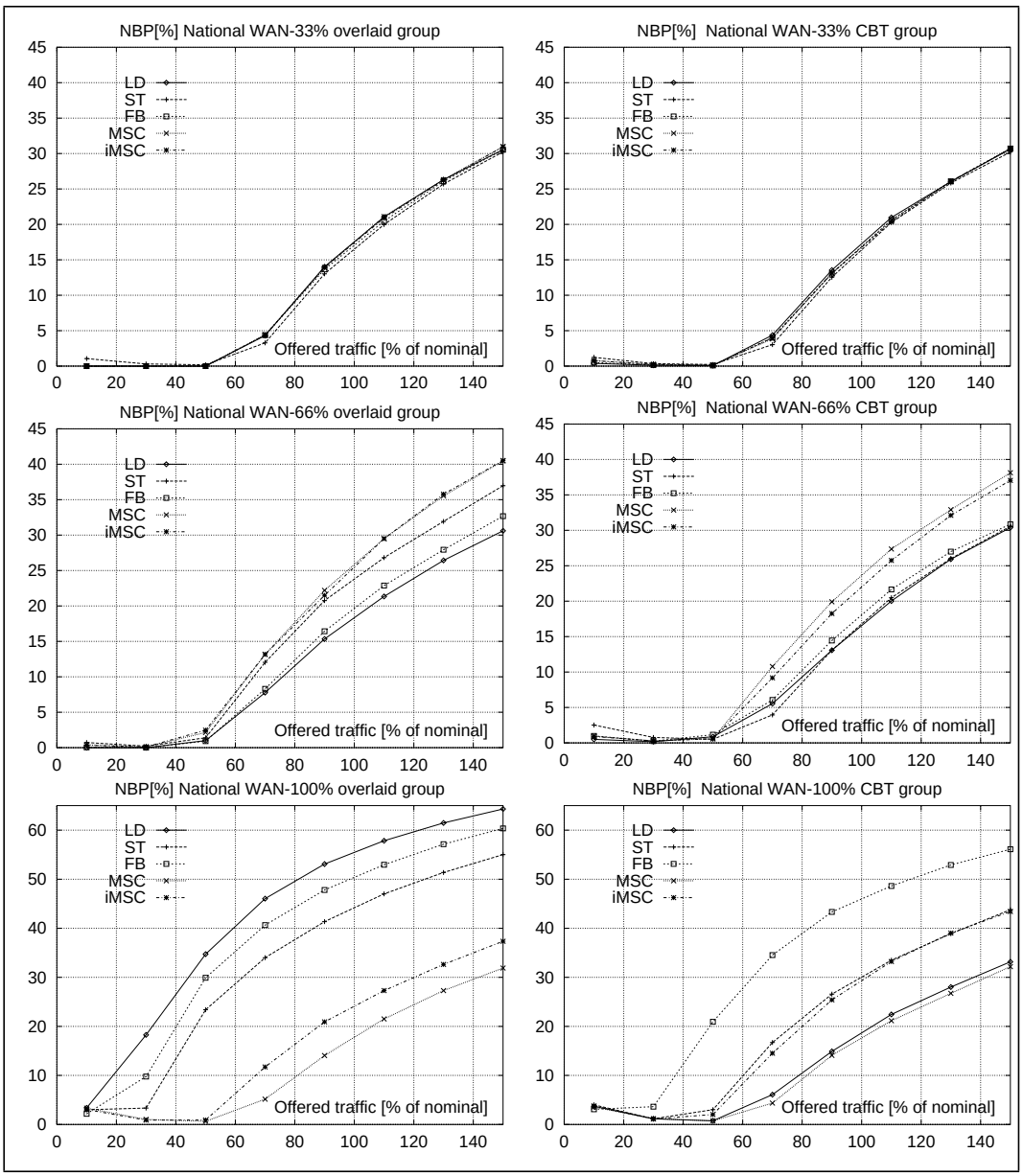

Figure 5.11: Network Blocking Probabilities (National WAN). *The differences between the algorithms for the 66 and 100% group are clearly visible. The best overall performance is offered by the MSC and the LD algorithms - with the exception of the 100% overlaid tree where the LD algorithm fails because the same shortest paths are chosen by too many connections*

In this type of network, one can easily see the huge differences between the algorithms. There is no single algorithm performing best - for the CBT trees, the LD algorithm gives the most ‘friendly’ trees - they hardly increase the blocking compared with the network without any multicasting (see Figure 5.3). For the overlaid trees, it is not that clear which algorithm is best - for the 66% group, the LD works well, but for the 100% group, it is already the worst one, causing double increase of the blocking compared with the original network. It is likely that with many destinations, too many connections operate on the relatively few shortest links to the source, causing excessive overload. In this case (similarly to the one-to-many problem) an algorithm respecting the load (as the MSC) will be better.

5.3.4 International WAN

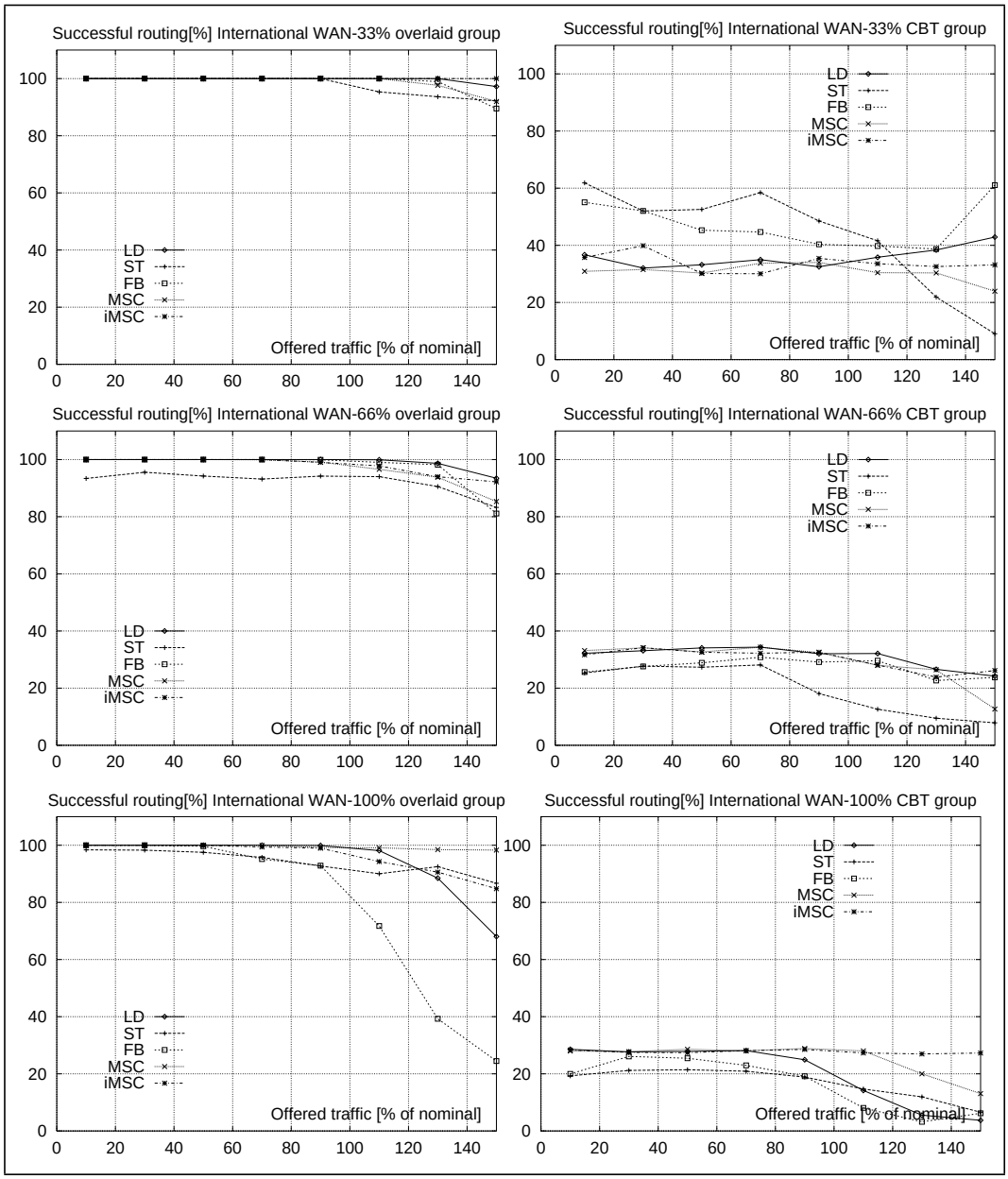

Figure 5.12: Efficiency of algorithms (International WAN). *Successful multicast connections as a percentage of all multicast connection requests for different algorithms.The Core Based Trees are difficult to find for all the requests due to the increased delay caused by double traversing of the links when the source and the destination nodes lie close to each other.*

The observations for the International network are basically similar to the National network. The delay-conscious algorithms perform better (especially the MSC). The only exception is the 33% CBT group where the pseudo-Steiner tree performs best, most probably due to the accidental location of the nodes.

It is worth noticing that with increase of the geographical size of the network (large delays) introducing the CBT algorithms seems questionable - it will create too long delays and use additional bandwidth resources to transfer the same data along different directions of the same links.

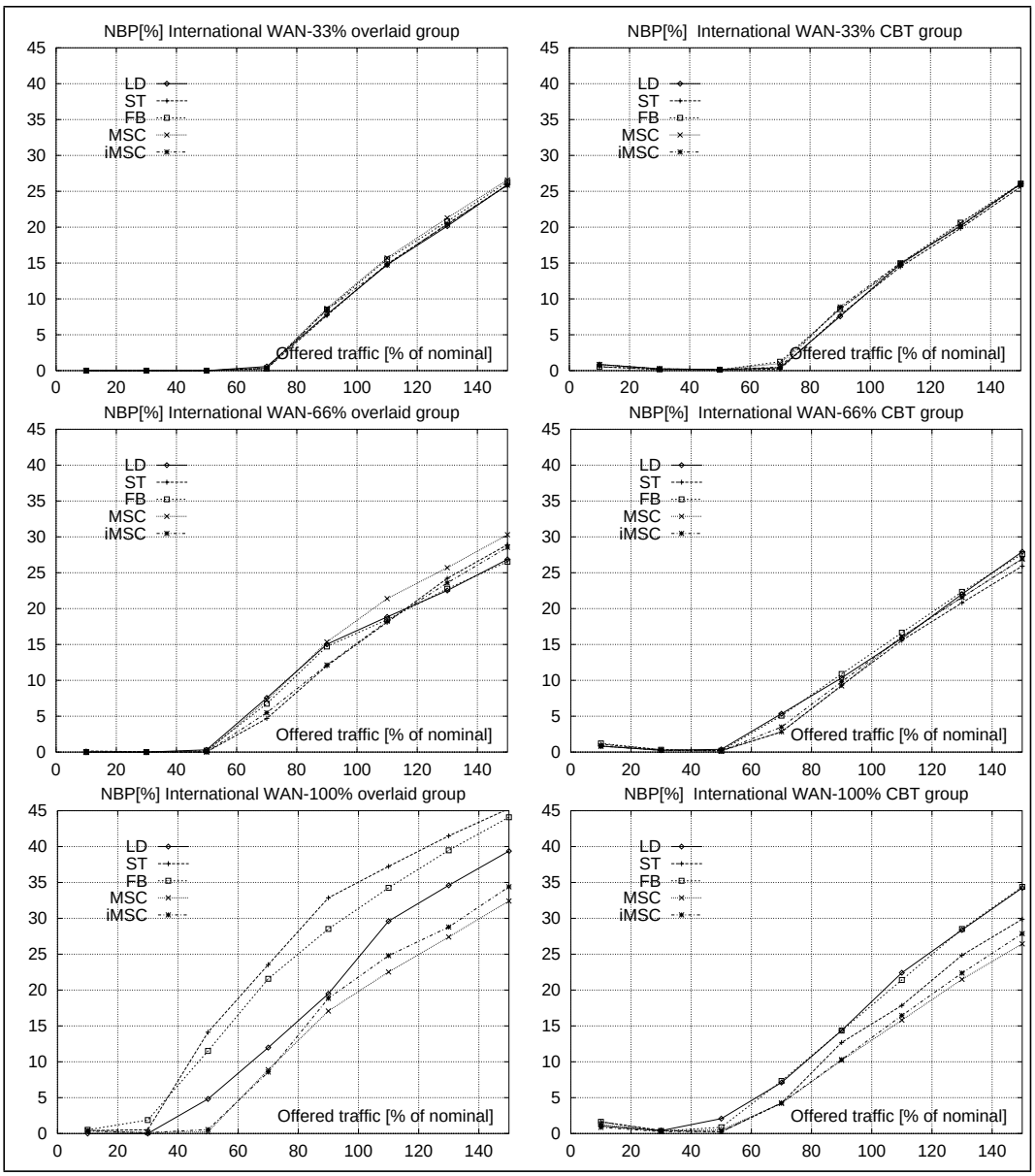

Figure 5.13: Network Blocking Probabilities (International WAN). *Again, the differences between the algorithms for the 100% group are clearly visible. The best performance is offered by the MSC algorithm. The overlaid trees generally cause a higher probability of blocking the background traffic than the CBT trees.*

The third type of the network gives another picture of the tested algorithms. While similar to the National network, the LD algorithm gives higher blocking than the MSC, the absolute worst with respect to the NBP parameter is the pseudo-Steiner tree for the 100% overlaid group - it spreads too much traffic over too many links.

The comparison between the CBT trees and the overlaid trees is not that clear now. The first ones block less traffic and save resources (VC numbers) but they are much harder, sometimes impossible, to compute due to the inherent additional link delays (not to mention the processing delay at the CBT multicast server).

5.3.5 Conclusions

The many-to-many routing is a significantly more difficult task than the one-to-many. The requirement that all the nodes have to communicate with each other introduces new possibilities of a multicast tree construction - it may be constructed as many source-routed overlaid trees or a single tree with a core. Moreover, the size of the multicast group begins to play a much more important role - the difference between 33% and 100% groups proved to be enormous - no simple scaling can be applied. The general problem with constructing the CBT trees is caused by the inherent additional transfer delay caused by traversing the unicast link to the server prior to joining the actual multicast tree. It may be acceptable in case of local networks (like Milanet) but for geographically spread ATM clouds connected only by single backbone links, this strategy will almost always fail. The possible solution is to establish multiple CBT servers so that each ATM cloud has at least one such server, and connect them with the permanent backbone links.

The simulations lead to the conclusion that none of the tested algorithms are possible to use for all cases in both one-to-many and many-to-many routing problems. The MSC seems to be the best algorithm for one-to-many and multiple one-to-many overlaid trees, whereas the LD performs better for the CBT trees. Therefore, it may be necessary to try to design an algorithm capable of handling both problems in an optimal way - there is, however, no guarantee that such an algorithm exists.

A temporary solution is to sequentially try to find a tree using both the MSC and the LD algorithms - since they are both based on the same Dijkstra routing method, their interaction can be realized in a straightforward manner.

It is clear that none of the cost-oriented algorithms can be used in a general way. While occasionally they may find better solutions than the delay-oriented methods, their overall performance leaves a lot to wish for - the most obvious conclusion is to abandon them for the real-time traffic: the difference in blocking probabilities introduced by their lower tree costs are negligible and in most cases not worth considering.

5.4 Experiment III: dynamic reorganization of the tree

In the following experiment, dynamic reorganization of the multicast tree is tested. The previous experiments were conducted assuming static routing, i.e. the router received the list of all the group members and constructed the tree in one step. In real life it may, however, be necessary to add and remove destination nodes to the previously established tree. In the ATMF specification [3], the possibility of so called Leaf Initiated Join is introduced - the new node decides if it wants to participate in the multicast session and joins to the point-to-multipoint VC which was already established. Abandoning the group is usually much simpler - the sender simply prunes the appropriate branch of the tree.

In the experiment, the following settings were used:

- the simulated network is the Milanet with the multicast group as in the Experiment I
- each time, the static tree is constructed for a subset of all destination nodes. The remaining nodes are added to the established tree using either ‘naive’ multicasting (see [7]) connecting via the shortest path to the source (if the path meets a node already belonging to the tree, the connection is established with this node) or trying to connect to the closest (in terms of delay) node already belonging to the tree.
- the abandoning of the tree is not simulated.
- the LD algorithm is not simulated - the dynamic tree is identical to the static tree.
- each simulation is repeated five times, the final result is the average of all five results.
- the CBT trees are not simulated - in this method, adding a new node is similar to the source-routed method, the only difference is that in the CBT tree the new node will try to find the best route to the core, not the source.

5.4.1 20% increase

In the first simulation, a static group is formed by five nodes and the sixth is added dynamically - this corresponds to the 20% increase of the group size.

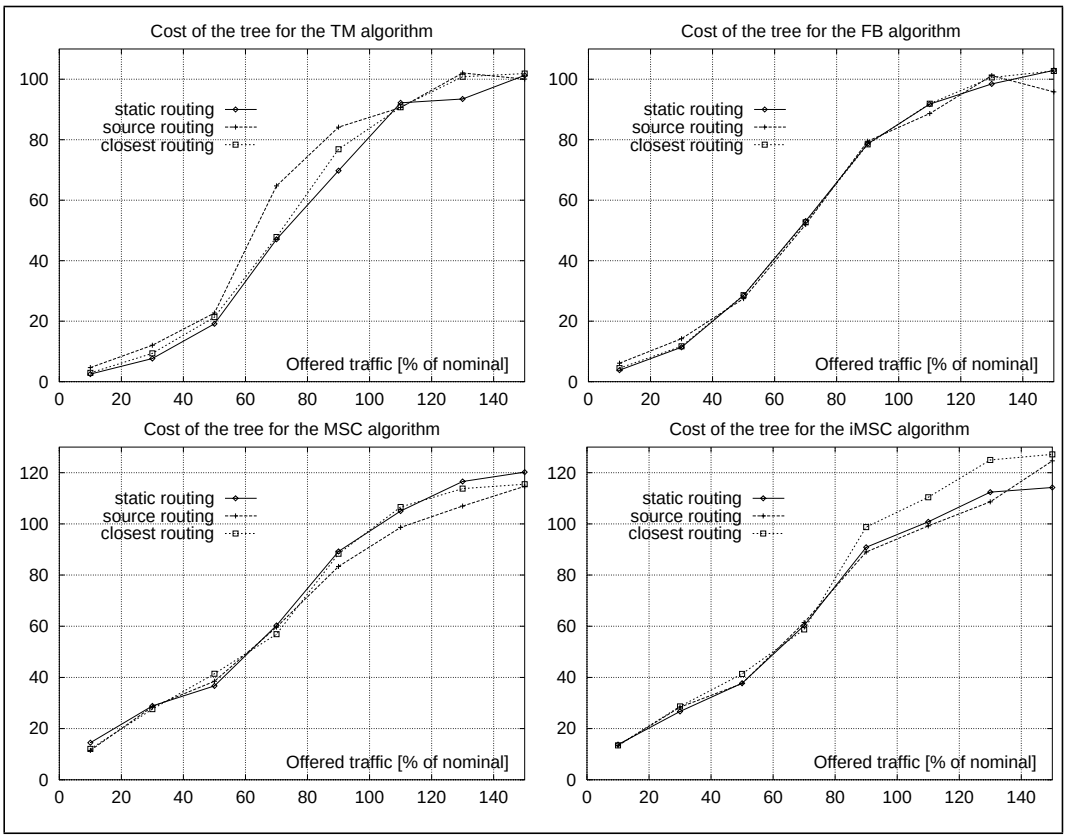

Figure 5.14: Cost of the dynamic tree (20% increase). *In real applications, the dynamically changing number of group members may cause a necessity to partially rebuild the tree. In the above graphs, three techniques are compared. The first one is the static group of 6 members constructed with different algorithms. The second technique first constructs the tree with five members and then adds the sixth one via the shortest path to the source. The third method is similar to the second, but the new member is added to the multicast tree via the shortest path to any of the nodes already included in the tree.*

As it can be seen, there is no significant increase in the cost of the dynamically constructed trees. Only the static TM algorithm performs better than the other types of algorithms. For the FB, MSC and iMSC the 'naive' multicasting (routing directly to the source) yields lower tree cost than the original one, and the method that connects the new node to the closest tree member via the shortest path usually results in higher cost trees - this result does not agree with the result presented in [7] where the 'naive' multicasting was approximately 50% worse than the static one. This difference may, however, be caused by the much smaller number of nodes in the Milanet network (14 nodes compared to 100 nodes quoted in [7]) and / or simulation technique. Even if the cost of the tree is much higher, it is still quite probable that this technique could be widely used - as we could see from the Figures 5.3 and 5.4, an increase of the tree cost does not necessarily mean an increase of the blocking probability. The 'naive' technique is the simplest method - it does not require the complete knowledge of the multicast tree - the new node just needs to know which is the source of the data stream. The other advantage of this method is the delay which is guaranteed to be kept under the specified limit.

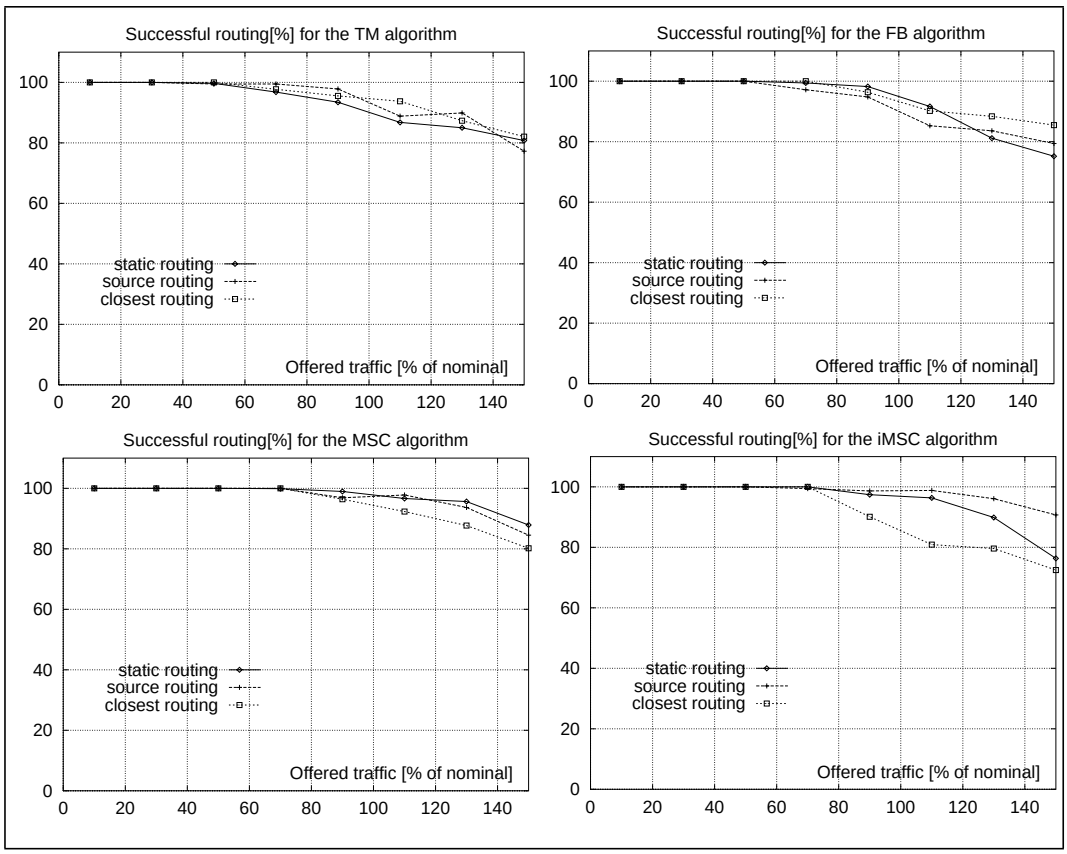

Figure 5.15: Efficiency of the dynamic algorithms (20% increase). *The percentage of the successful routing attempts for two dynamic multicast algorithms compared with the static routing efficiency. Finding the closest tree node is rather ineffective - most probably due to the delay constraint violation. For the MSC algorithms, source routing is evidently better than the basic algorithm.*

The above graphs present the efficiency comparison between the three routing approaches. The batch tree creation and the source routing seems to perform better in case of delay-conscious algorithms (MSC and iMSC) and the closest node routing is apparently better for the cost-oriented trees. The obvious reason for the occasional failure in the closest node routing is the violation of the delay constraint when the closest node happens to be also the node with the longest distance to the source. It is not clear, however, why the source routing performs worse in case of the FB trees, perhaps these trees block to many links around the source so it is more difficult to connect to it directly.

On the other hand, finding the direct path to the source performs better than batch routing in case of the iMSC algorithm. This is probably caused by the similar reasons why the iMSC algorithm (eliminating the longest paths of the tree) sometimes outperforms the MSC method - statistically, the longest links are the most probable cause of blocking. Anyway, having selected MSC as the most effective algorithm for the one-to-many connections, it seems natural to use also the source routing to attach new nodes to a tree created by this algorithm.

5.4.2 100% increase

In the second simulation, the static group consists of three nodes an three new ones are added dynamically - this way the group is doubled

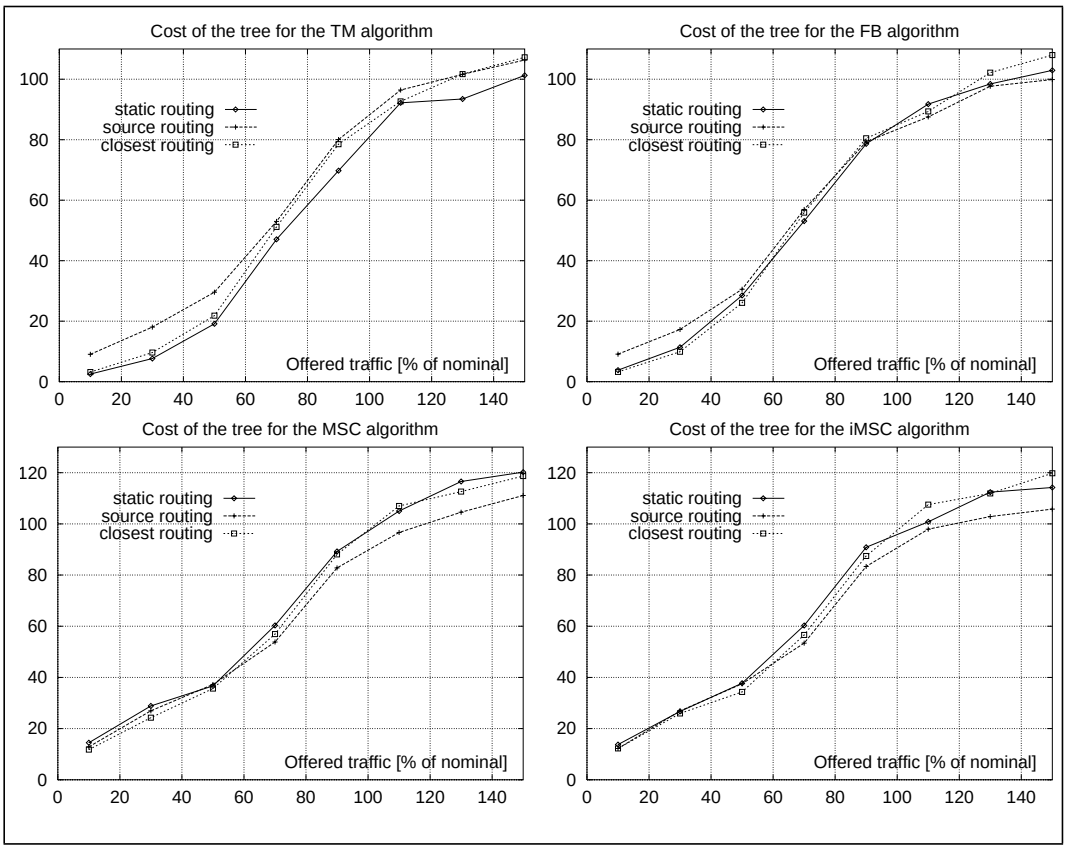

Figure 5.16: Cost of the dynamic tree (100% increase). *In reality, it may happen that the number of new nodes will approach the number of nodes already included in the static tree. The above graphs present the situation when the number of destination nodes was doubled after dynamic adjustment of the multicast tree. Comparing them with the Figure 5.14, one can see that the increased number of new destinations makes a difference only in case of cost-oriented algorithms (TM and FB), whereas the outcomes of the MSC and iMSC methods are relatively little changed.*

Increasing the number of new nodes from the 20% to the 100% of the size of the group does not seem to change the situation significantly. The graphs from Figure 5.16 are very similar to the graphs from Figure 5.14. The most visible difference can be observed for the TM algorithm, in which the outcomes of both dynamic techniques seem to merge at approximately 60% of nominal traffic. In both FB and TM algorithms, an increase of tree cost (compared with the previous simulations) can be noticed with low traffic load. The explanation is rather obvious - in this case, only half of the tree is constructed using the optimal low cost algorithm, the second half is routed with no concern about cost at all, which is especially visible when the background traffic is low.

The small difference in case of delay-conscious algorithms is probably caused by the fact that all these algorithms are based on the Dijkstra algorithm so the differences between the outcomes are not so dependent on the number of destinations, i.e. the shortest paths are basically the same in all cases. This is an additional argument to use the delay-conscious algorithms in multicasting.

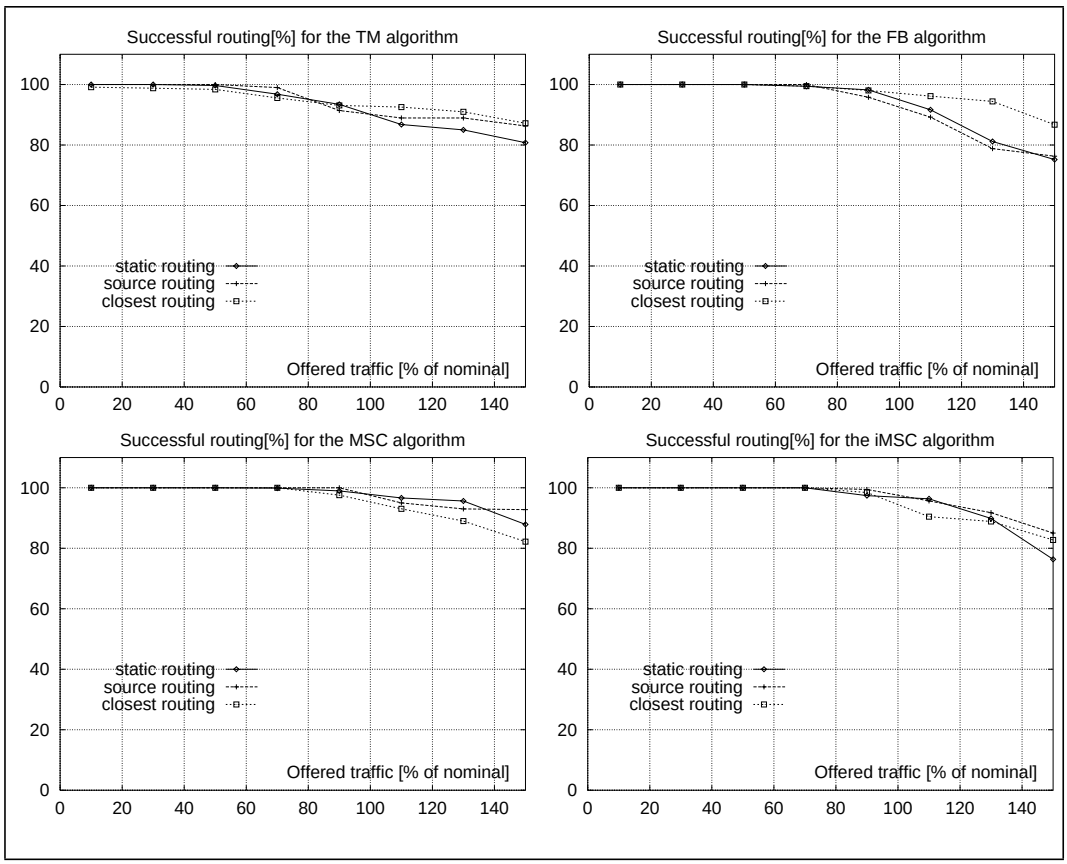

Figure 5.17: Efficiency of the dynamic algorithms (100% increase). *The percentage of the successful routing attempts for two dynamic multicast algorithms compared with the static routing efficiency. Similar to the 20% group increase, source routing is slightly better than closest node routing. In this case, however, the difference between both methods decreased in case of MSC algorithms and increased in case of cost-conscious ones (TM and FB).*

The doubled number of the new members does not change the outcomes of the algorithms significantly. It is still obvious that source routing performs best with the MSC algorithms, although the difference between this method and the closest node routing is no longer as visible as in the previous simulation. It is rather unclear why the FB algorithm seems to be working so well with the closest node routing - it may be just an accidental outcome of the network.

5.4.3 Conclusions

According to the simulations, the simplest approach to the dynamic multicast, i.e. when the new node finds the shortest path to the source, is also the most successful.The cost of such a tree is comparable to the original (static) tree and the efficiency of routing is sometimes higher. Moreover, it can be easily applied in practical applications - it does not require the complex protocols needed to distribute the list of all the members of a multicast group to all the nodes which could be potentially interested in joining the group. The delay constraint is also guaranteed to be fulfilled by the ‘naive’ technique if only such a path exists.

5.5 Experiment IV: multiple unicast vs multicast

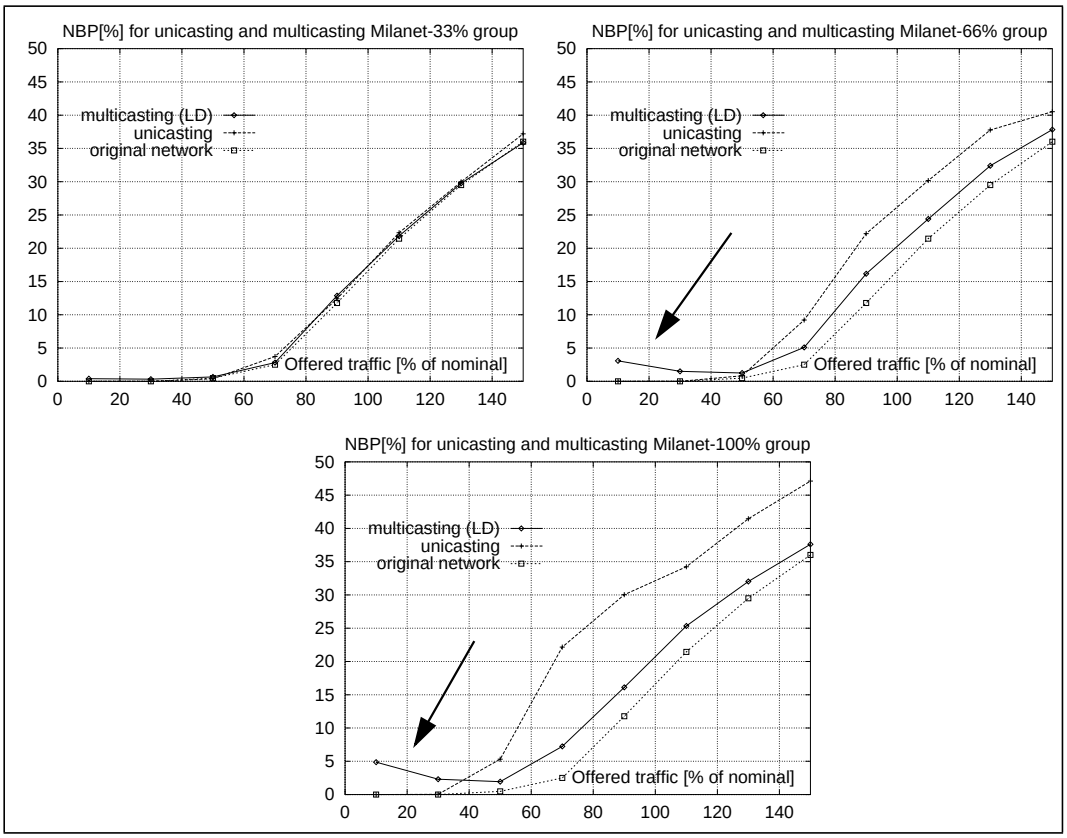

Figure 5.18: Network Blocking Probability for Milanet network. *Network is tested without any video conferences, with unicast video conference and with multicasting. As it can be seen, the more members of the group, the higher the gain the multicasting technique will bring.*

After simulating all the multicast techniques, it could be interesting to know how much we gain by employing multicast instead of multiple unicast. Figure 5.18 answers the question. The graphs present a comparison between the original network with only background traffic present, the network with video conference between 33, 66 and 100% of the nodes using traditional multiple unicast, and the network with the same connections using multicast (Least Delay algorithm with the overlaid technique). In the network with only 33% of all nodes taking part in the conference, the difference is almost invisible, but with increasing number of multicast group members, the gain with multicast is clear. For 66% group it performs 10% better than the multiple unicast method, and with the 100% group it can be as much as twice as effective as unicast. At the same time, it can be noticed that the increase of blocking probability caused by multicasting is relatively constant with respect to the size of the group (after reaching approximately 50% membership). This may be helpful in network design - multicast does not require high bandwidth margin for the largest possible group - the 66% group and 100% group increase blocking probability similarly. The relatively small differences between the blocking probabilities for the 33% group may lead to another issue. With small multicast groups, it may sometimes be reasonable to employ multiple unicast without any significant loss in the network performance. Indeed, using a complicated protocol, employing a dedicated server etc. to transmit low bandwidth data to 3-4 users seems to be a little exaggerated.

Increased blocking with low traffic may be caused by immediate blocking of the best links - with increased background traffic, the multicast algorithm will spread the load evenly.

5.6 Experiment V: bi- and unidirectional multicast

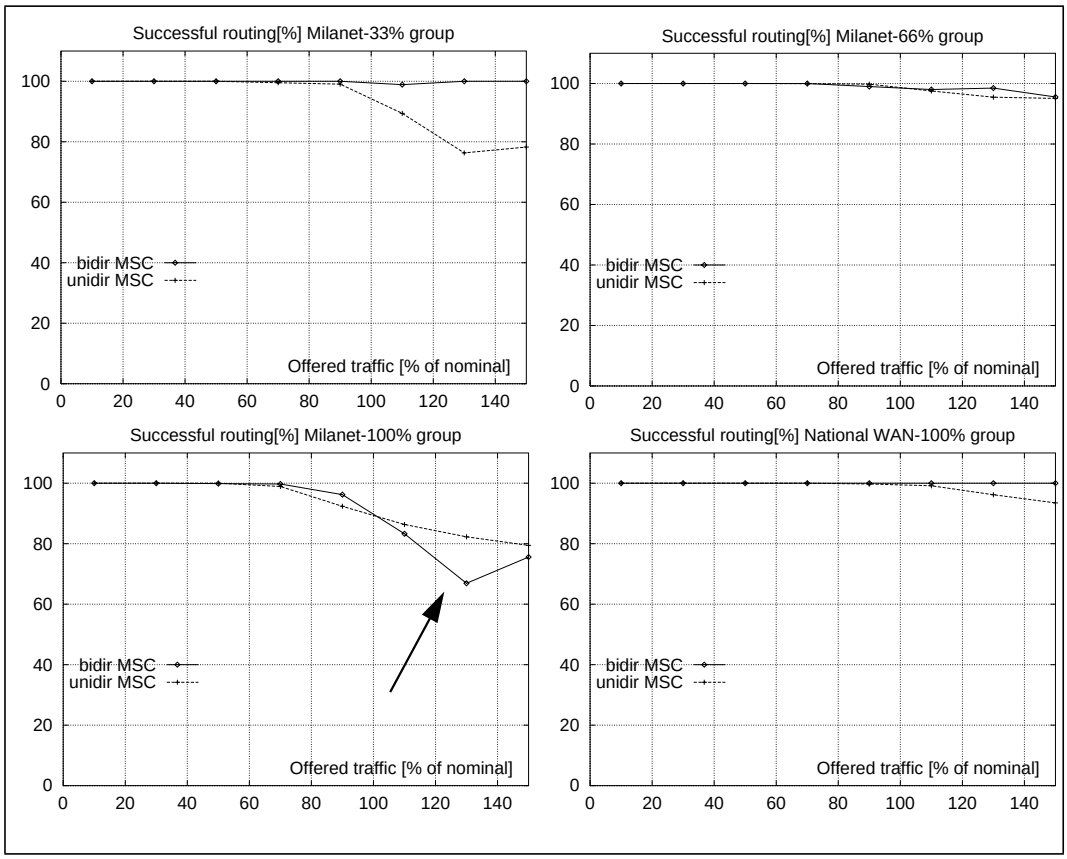

Figure 5.19: Comparison between bidirectional and unidirectional multicast. *The above graphs show the hypothetical situation of multicasting with bidirectional point-to-multipoint Virtual Channels using the MSC algorithm. As expected, the probability of finding the proper tree is increased due to the lower traffic caused by eliminating multiple unidirectional connections. In case of Milanet 100% group, however, unidirectional connections perform better with higher traffic intensities.*

It may be worthwhile to consider a situation when true bidirectional connections of ATM point-to-multipoint VC (p2mpt VC) are used in multicasting. Although it is impossible to achieve this with the present AAL standards, it may happen that a new protocol may be introduced or some form of hardware control applied (see Chapter 3). Clearly, this should improve the performance of the network since the overlaid mesh of many unidirectional p2mpt VCs would be reduced by half (this applies only to the overlaid model - not to the CBT model in which bidirectional connections are of lower importance). That does not mean that the traffic would be lowered - only the number of connections would be reduced resulting in easier management of a multicast group.

Almost all cases using the bidirectional pt2mpt VCs improved efficiency by a few percent. The only exception to the rule can be noticed in Milanet 100% group at higher traffic intensities. The unidirectional connections seem to perform better than bidirectional ones despite their quantity. It could be explained by the general difficulty of finding the free capacity in both directions for the link with high traffic congestion. The unidirectional connections have the possibility of ‘sneaking’ through different VPs giving improved performance. Moreover, it is easier for the unidirectional connections to choose the proper QoS links in asymmetrical networks. Employing bidirectional connections could, therefore, result in a new class of problems

6 TCP/IP over ATM

This chapter briefly presents methods of multicasting in TCP/IP networks and possibilities of its interaction with ATM multicast.

6.1 Multicasting in TCP/IP

TCP/IP, a backbone of the Internet, is currently the most used protocol in the wide area networks. The need for multicast in Internet was recognized long ago, but only quite recently the proper techniques allowing for efficient multicasting have been designed. The following sections describe three techniques that are used for IP multicasting. Only the techniques independent of the lower layer routing protocols are presented.

6.1.1 Resource Reservation Protocol

Resource Reservation Protocol (RSVP) is in principle the IP equivalent of QoS control in ATM. The IP end-systems specify parameters like bandwidth, burstiness, jitter etc. required from the network, which are controlled by intermediate systems (similar to ATM CAC). In the setup phase, a packet traverses the network storing the route information in the nodes - this way all subsequent packets will travel the same path. RSVP is fundamentally designed to support point-to-multipoint paths and treats unicast as a special case. Moreover, RSVP allows for the heterogeneity, which means that different receivers have different quality requirements.

6.1.2 Protocol Independent Multicasting

Protocol Independent Multicasting (PIM) is currently defined in two classes: dense mode (PIM-DM) and sparse mode (PIM-SM). The dense mode is used when the multicast group is large compared to the size of the network. This technique uses the “broadcast and prune” method - it forwards the data to all subnets away from the sender until a router indicates that it has no clients interested in receiving the multicast data and sends a “prune” message to the source. This method is not scalable and applies to rather small LAN environments with large available bandwidth. It is, however, very simple in the implementation.

The sparse mode is used for multicast groups where members are sparsely distributed over the network. It uses the so called rendez-vous point (RP), at which receivers can connect to the data streams from sources of the same multicast session. The method constructs a shared tree around the RP and all nodes connect to it via the shortest path. A receiver gets source parameters when it receives its message over the shared tree. It can decide then if it should keep the shared tree connections or ask for a source-specific tree. The method uses periodical refreshment (soft-states) of the group data which can create overhead in case of large groups.

6.1.3 Core Based Trees

In the Core Based Tree (CBT) technique, all sources use a single multicast tree to carry their traffic to all receivers. The tree can have one or more cores interconnected via a core backbone. This method, while being similar to PIM-SM, introduces much less overhead since it uses ‘hard-states’ i.e. there is no regular repetition of the join messages. Using ‘hard-states’ may, however, create problems when, due to the malfunction of part of the network, some data is sent to the nodes which already quit the session.

6.2 Current IP over ATM solutions

The interaction between IP and ATM is not a trivial question. IP is a connectionless protocol whereas ATM (as explained in Chapter 3) is usually a connection oriented technique. Two of the most popular approaches are presented below.

6.2.1 Classical IP over ATM

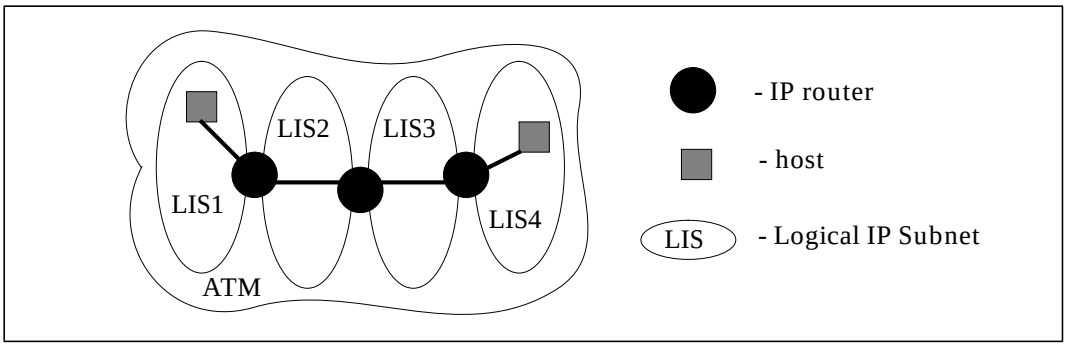

Figure 6.1: Classical IP over ATM. *The hosts belonging to different IP Subnets have to establish connections via respective routers even if there exists a direct ATM connection.*

The classical concept of interaction between IP and ATM assumes the creation of groups of IP hosts (Logical IP Subnet - LIS) attached to the same ATM network which is treated as a LAN network within respective LIS. Inside these subnets, the hosts communicate with each other using the Address Resolution Protocol (ARP) which assigns the ATM connection identifier (VPI/VCI) to the IP address. For connections to the outside of the group, IP hosts use standard forwarding using the IP router. The simplicity of this model is achieved at the expense of increased connection processing when the adjacent ATM nodes belong to different LIS groups. In the latter case, the connection has to pass via the IP router instead of using a direct ATM connection.

6.2.2 Next Hop Resolution Protocol

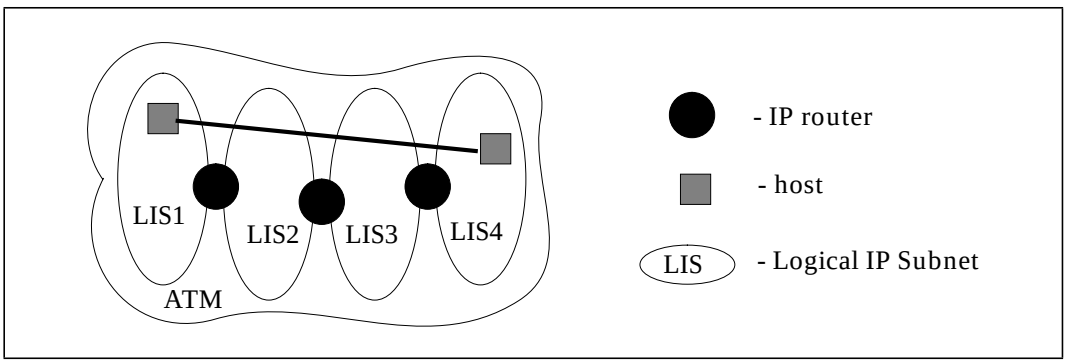

Figure 6.2: Next Hop Resolution Protocol (NHRP). *If there is a direct ATM level connection between hosts in non-adjacent IP Subnets, NHRP can bypass intermediate routers.*

The shortcomings of the classical IP over ATM model have been improved in the Next Hop Resolution Protocol. This technique uses dedicated NHRP servers (NHS) which provide ATM addresses of IP hosts outside of the local LIS domain. This protocol can, however, lead to some problems when a short-cut connection is set between two IP routers which are not logically adjacent and can result in a stable loop connection.

6.3 Multicast in IP over ATM

After evaluating multicast routing methods for IP and ATM, the natural question arises: how should both techniques interact? Is it better to map IP routing directly to ATM routing, or should ATM take care of the whole process?

A good protocol for IP multicast over ATM should address the following problems:

- how to multicast inside LIS groups?
- how to multicast between LIS groups?
- what happens if one of the nodes between communicating hosts does not provide RSVP protocol?
- how to map RSVP into QoS in ATM?
- how to provide different QoS requirements (heterogeneity) of different destination nodes (current ATM specification does not allow different branches of point-to-multipoint VCCs to have different QoS parameters)?
- how to solve the set-up delay problem inherent in all switched ATM connections?

6.3.1 Current solution - MARS

Some of the above questions were answered in the currently most popular concept of multicast in IP over ATM, known as Multicast Address Resolution Server (MARS). This technique groups ATM nodes (both hosts and routers) into clusters (usually mapped one-to-one with LIS groups). Every MARS server creates an extended table assigning multiple ATM addresses to a single IP multicast address which are sent to a requesting node via a permanent point-to-multipoint VC (ClusterControlVC) connected to all the nodes in the cluster. When multicast servers (MCS) are used, MARS also establishes a similar connection to the servers - this way it supports both source-specific VCC meshes and multicast servers. It is important to realize that MARS does not perform actual routing, it is only an administrative entity.

Unfortunately, MARS assumes that the multicast traffic between hosts in different clusters has to pass through the IP router. This is a serious drawback of the technique, since such a router is a potential bottleneck when there exist many multicast groups. Moreover, it may be impossible to provide proper QoS parameters for the indirect route.

6.3.2 Future solutions

There is an ongoing research (Internet drafts on architectures such as EARTH or VENUS) aimed at circumventing problems caused by MARS. It is mostly concentrated on supporting so called Multicast Logical IP Subnet (MLIS) spanning the whole physical ATM network. Grouping all ATM nodes into one cloud is a most natural way of providing global routing capabilities and quality of service parameters. However, maintaining a MLIS with many ATM nodes is not an easy problem, and most probably it will take some time until it will be deployed on a wider scale.

6.4 Where to route multicast connections?

It is not that clear if all multicasting should be done only at one level. There are indeed three different approaches to the problem:

- the connection oriented approach assumes that all classes of traffic should set up a separate multipoint VC. While this solution has certain advantages (for example providing proper QoS parameters for each IP flow) it is most likely that short-time multicast without QoS requirements would suffer from the intrinsic latency caused by the setup phase of ATM.
- the connectionless approach is the basic IP method - each flow is forwarded using pre-

viously set up VCCs. Such an approach overcomes the problem of the setup-delays, but makes it more difficult to provide QoS parameters. The demanding video-conferencing connections or similar services would almost always require direct routing at the ATM level.

- the middle ground approach recognizes advantages and disadvantages of the above methods and suggests classification of multicast applications, assigning proper routing layer to each of them. According to this approach, short duration and / or not demanding flows could be routed at the IP level, and for the high quality flows, RSVP protocol could be used to decide whether the new direct route at the ATM level is necessary. This way, the 'responsibility' could be placed on both levels.

It is clear that the pure IP layer of multicast routing may be unable to take advantage of the particular features of ATM. On the other hand, some types of connections are impossible or rather impractical to realize at the ATM level.

Type of algorithm	Layer of routing	Reasons
PIM-DM	IP	the number of branches (fanout) of the point-to-multipoint VC is limited. Simple techniques such as flooding can be used instead
PIM-SM / CBT	IP / ATM	the technique resembles closely the ATM multicast server. It may be possible to establish dedicated servers serving both at the IP and ATM layer. Upon receiving a request for joining the core could decide if it should respond to the IP level using current connections, or whether it should establish a new VCC with proper parameters.

Table 6.1: Possible division of routing among IP and ATM.

The above division may work without significant problems as long as each IP multicast server is doubled by an ATM multicast server. It was, however, shown in the previous chapter that there may be a need for establishing at least one multicast server in each ATM cluster and connecting them via backbone core connections. It may therefore happen, that there will be more ATM multicast servers than IP ones - the situation then gets rather complicated and may cause a need for significant changes both to the MARS and IP multicast protocols. The detailed discussion and / or simulation of the interaction between ATM and IP multicast could be, in fact, a subject for another thesis.

7 Conclusions

This report addressed questions concerning multicast routing in ATM networks. These questions included:

- which algorithm performs best?
- is it possible to use the same algorithm for one-to-many and many-to-many multicast?
- which techniques of many-to-many multicast should be employed?
- how to dynamically attach new destinations to the tree?
- how to combine IP and ATM multicast?

7.1 Results

After the simulations of the implemented networks, it became clear that only the algorithms concerned primarily with delay control can succeed in multicast routing with QoS requirements. The best tested algorithm (both in terms of performance and complexity) was the Multicast SemiConstrained (MSC), with time complexity almost identical to the classical Dijkstra method. It usually managed to outperform the other algorithms, especially purely cost-conscious Steiner Tree heuristic.

Unfortunately, the many-to-many problem seems to require another algorithm - for the multicast server technique, the best algorithm seemed to be the classical least delay Dijkstra (especially with lower traffic), while the VC mesh technique was usually working best with the MSC algorithm. Therefore it might be necessary to combine these two algorithms into one, choosing the appropriate method according to traffic conditions, especially for the core based trees. It is particularly important for the larger multicast groups where using the overlaid VC mesh technique is impossible due to the limited number of VCC branches in a single point-to-multipoint connection (see [1]).

The core based technique is more scalable and more likely to be employed on a wider scale. The most important problem of this method is, however, the inherent increase of delay. While this increase may not be so crucial in case of a single ATM cloud (LAN or MAN), connecting many clouds via long distance backbone connections may cause a necessity of establishing many multicast servers. This way, multicasting in a local ATM cloud could be performed using currently available techniques (MARS) and for connections outside the cloud, the dedicated links joining multicast servers in different ATM clouds could be used. This improvement would, unfortunately, require new extensions to the present protocols.

Dynamic reorganization of the multicast tree, although not thoroughly tested, seems to perform best with the simplest technique of connecting the joining node to the multicast tree over the shortest path to the source. This way the delay factor is controlled, signalling simplified and probability of successful connection increased. The inherent increase of the cost of a multicast tree created in this way does not seem to play a significant role.

As for the interaction between the IP and ATM routing, the decision on the routing layer should probably be based on the demanded quality of service. For the simple connections, without any stringent constraints (delay) it seems reasonable to route at the IP layer using already established ATM links within a LIS. For more demanding applications, where it may be impossible to use IP routers (due to the increased delay), the best way could be to create a dedicated multicast session on the ATM level. The possible interaction between both techniques is, however, a very complex problem. For example, the distribution of ATM multicast servers might not be identical with the distribution of IP multicast servers (cores). This, and other problems, would have to be taken into consideration when introducing new protocols.

7.2 Suggestions for further studies

The presented thesis is by no means a complete study of multicast routing in ATM. There are certain areas which, due to the limited time, were not addressed. Some of them could be subjects for future research:

- the networks used in simulations cannot be treated as representative for all the possible network configurations. Therefore it could be useful to develop an automatic generator of ATM networks giving much more universal testing field.
- the number of nodes in the tested networks is severely limited by the computing capabilities of used computers. A modification of PLASMA, or even a new, much simplified version allowing for simulation of 100-1000 nodes should be designed (this is a general remark - such a simulator is certainly useful not only for the multicast problem).
- the simulated networks were all symmetrical. It is not clear how the multicasting algorithms (especially for the many-to-many case) would work with asymmetrical links.
- all implemented algorithms were based on the classical Dijkstra algorithm. Simulation of the behaviour of a neural network (for example with Potts neurons) could be an interesting field of research.
- the dynamic behaviour of the multicast algorithms was not fully investigated. It could be useful to fully implement the signalling protocols and procedures used for both joining and leaving the multicast tree.
- the usage of multiple multicast servers should be further investigated

8 References

[1] ATM Forum - Private Network-Network Interface Specification ver. 1.0 March 1996

[2] ATM Forum - Traffic Management Specification ver. 4.0 April 1996

[3] ATM Forum - ATM User-Network Interface Signalling Specifiation ver. 4.0 June 1996

[4] Anthony Alles, “ATM Internetworking,” Cisco Sytems, Inc. , May 1995

[5] J.Abrahmsen, “Adaptive Routing in ATM-networks,” Ericsson Telecom AB, October 1996

[6] N.F.Maxemchuk, “Video Distribution on Multicast Networks,” AT&T Labs

[7] H.F. Salama, "Multicast Routing for Real-Time Communication on High-Speed Networks," Ph.D. dissertation, Department of Electrical and Computer Engineering, North Caroline State University, November 1996.

[8] Salama, Viniotis, Reeves, and Sheu, "Multicast Routing Algorithms for High-Speed Networks," IBM Technical Report, Sept. 1994

[9] Salama, Reeves, and Viniotis, “Evaluation of Multicast Routing Algorithms for Real-Time Communication on High-Speed Networks," IEEE Journal on Selected Areas in Communication , February 1997.

[10] Salama, Viniotis, Reeves, and Sheu, “Comparison of Multicast Routing Al gorithms for High-Speed Networks,"IBM Technical Report, September 1994

[11] Haberman, Brian K.; Rouskas, George N, “Cost, Delay, and Delay Varia tion Conscious Multicast Routing,” Department of Computer Science, North Caroline State University, March 3, 1997

[12] Rouskas, George N.; Baldine, Ilia, “Multicast Routing with End-to-End De lay and Delay Variation Constraints,” Department of Computer Science, North Caroline State University, July 11, 1995

[13] J.Häkkinnen, M.Lagerholm, C.Peterson, B.Söderberg, “A Potts Neuron Ap proach to Communication Routing,” Complex Systems Group, Department of Theoretical Physics, University of Lund, march 1997

[14] E.Gelenbe, A. Ghanwani, V.Srinivasan, “Steiner Trees and Improved Heuris tics for Multipoint Routing using the Random Neural Network,” Duke Uni-versity, Durham.

[15] F.Sestini, “Recursive Copy Generation for Multicast ATM Switching,” IEEE Transactions On Networking, vol.5, no.3, June 1997

[16] L.Wei, F. Liaw, D.Estrin, A.Romanow, T.Lyon, “Analysis of a Resequencer Model for Multicast over ATM Networks,” proceedings of the 3rd Interna tional Workshop on Network and OS Support for Digital Audio and Video, San Diego November 1992

[17] M.Grossglauser, K.K.Ramakrishnan, “SEAM: Scalable and Efficient ATM Multicast,”Proceeding of INFOCOM-97, April 1997

[18] Salama, Reeves, and Viniotis, "Shared Multicast Trees and The Center Selection Problem: A Survey," Technical Report TR-96/27, Center for Ad vanced Computing and Communication, North Carolina State University, June 1996.

[19] C.Huitema, “Routing in the Internet”, Prentice Hall PTR, New Jersey 1995

[20] H.Salama, D.S.Reeves, Y.Viniotis,"Delay-Constrained Shared Multicast Trees," ECE Department , North Caroline State University, October 1996.

[21] J.D. Cavanaugh, T.J.Salo, “Internetworking with ATM WANs,” Minnesota Supercomputer Center, Inc, December 1992

[22] M. Larsson, “Type-of-Service Adaptive Routing in ATM,” Ericsson Tele-com AB, October 1997

A Networks

This appendix presents three networks used in simulations. The purpose of employing different environments for testing algorithms is to maximize the probability of obtaining realistic data from simulations - one network can have specific characteristics which favour one algorithm over the other.

A.1 MAN type network - Milanet

The first network is based on an actual existing network in Milan. Its parameters (capacity of links, traffic distribution etc.) are calculated using real parameters.

The Milanet consists of 14 nodes and 34 uniformly bidirectional VPCs:

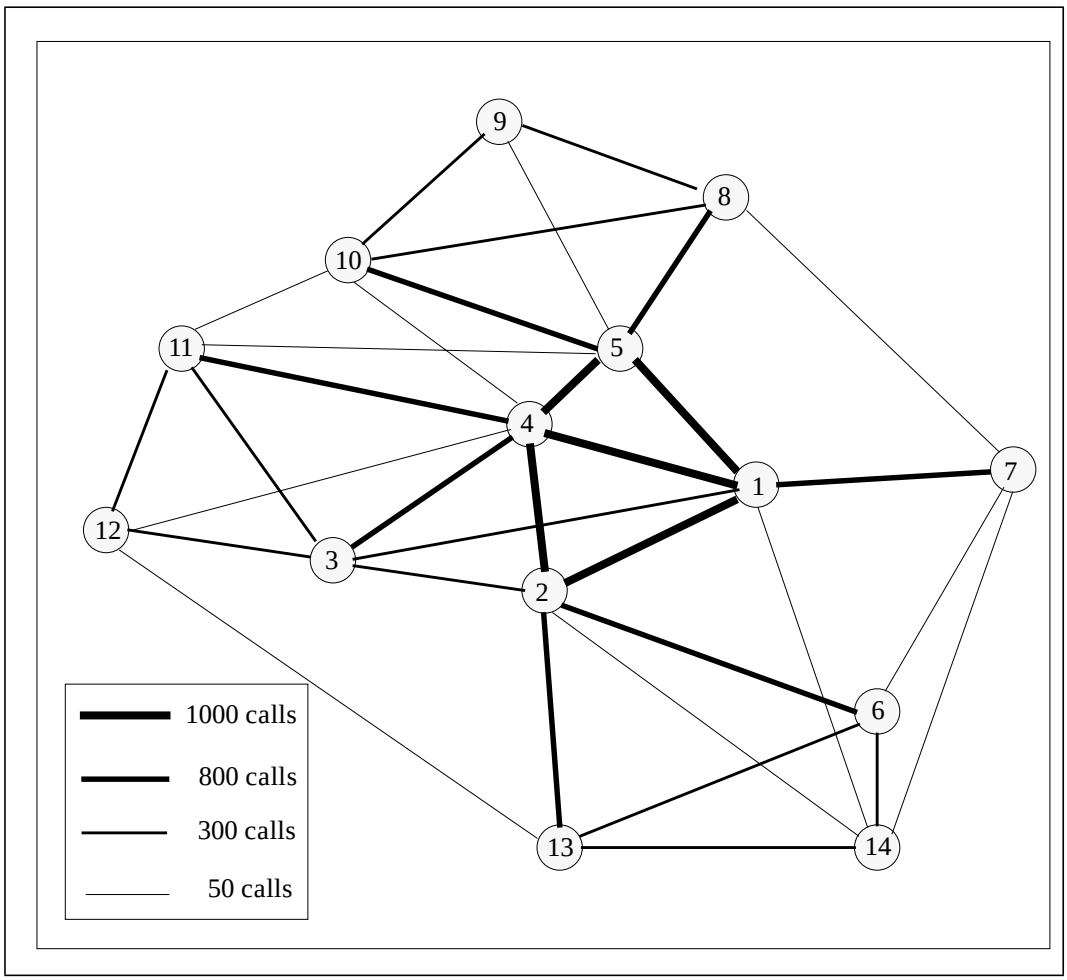

Figure A.1: The Milanet network

The characteristics of the network are the following:

- It can be divided into a backbone (nodes 1,2,4,5), three local networks (8-10; 3, 11, 12 and 6, 13, 14) and an international gateway (7).
- The CER, CLR, CMR and CDV values of links are equal to 0. All links have identical delay: MCTD=0.25ms.
- The traffic matrix is presented below - as it can be seen the traffic is not distributed

evenly - the difference can be as big as almost two orders.
- In this type of network the crucial role is played by the available bandwidth - the delay is usually in the requested range.

from:	to:													
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**
1		1	1	4	4	2	5	2	6	3	10	3	2	5
2	1		1	2	10	2	1	5	3	5	3	1	5	10
3	1	2		2	5	56	45	78	56	78	45	78	47	75
4	4	4	3		3	6	3	4	7	3	5	7	4	2
5	4	2	3	3		5	3	5	4	5	6	4	5	2
6	2	4	56	6	5		50	60	40	40	40	40	40	40
7	5	5	45	3	3	50		40	40	40	40	40	40	40
8	2	6	78	4	5	60	40		40	40	40	40	40	40
9	6	4	56	7	4	40	40	40		40	40	40	40	40
10	3	3	78	3	5	40	40	40	40		40	40	40	40
11	10	4	45	5	6	40	40	40	40	40		40	40	40
12	3	6	78	7	4	40	40	40	40	40	40		40	40
13	2	4	47	4	5	40	40	40	40	40	40	40		40
14	5	7	75	2	2	40	40	40	40	40	40	40	40	

Table A.1: Milanet network traffic matrix. *Nominal traffic=4388 Erlangs.*

A.2 WAN type network - national

This and the following network were created based on [6]. In this publication, a typical statistical distribution of nodes in Wide Area Networks was presented. Although the size of the networks in [6] was fixed at 1000 nodes, a serious reduction of this number was necessary due to the computing capabilities of available computers - the simulation of a network consisting of more than 20-30 nodes is extremely slow. The size of the network was therefore drastically reduced to 27 nodes, while keeping the statistical distribution of nodes and distances between them.

The national WAN consists of 27 nodes and 52 uniformly bidirectional VPCs:

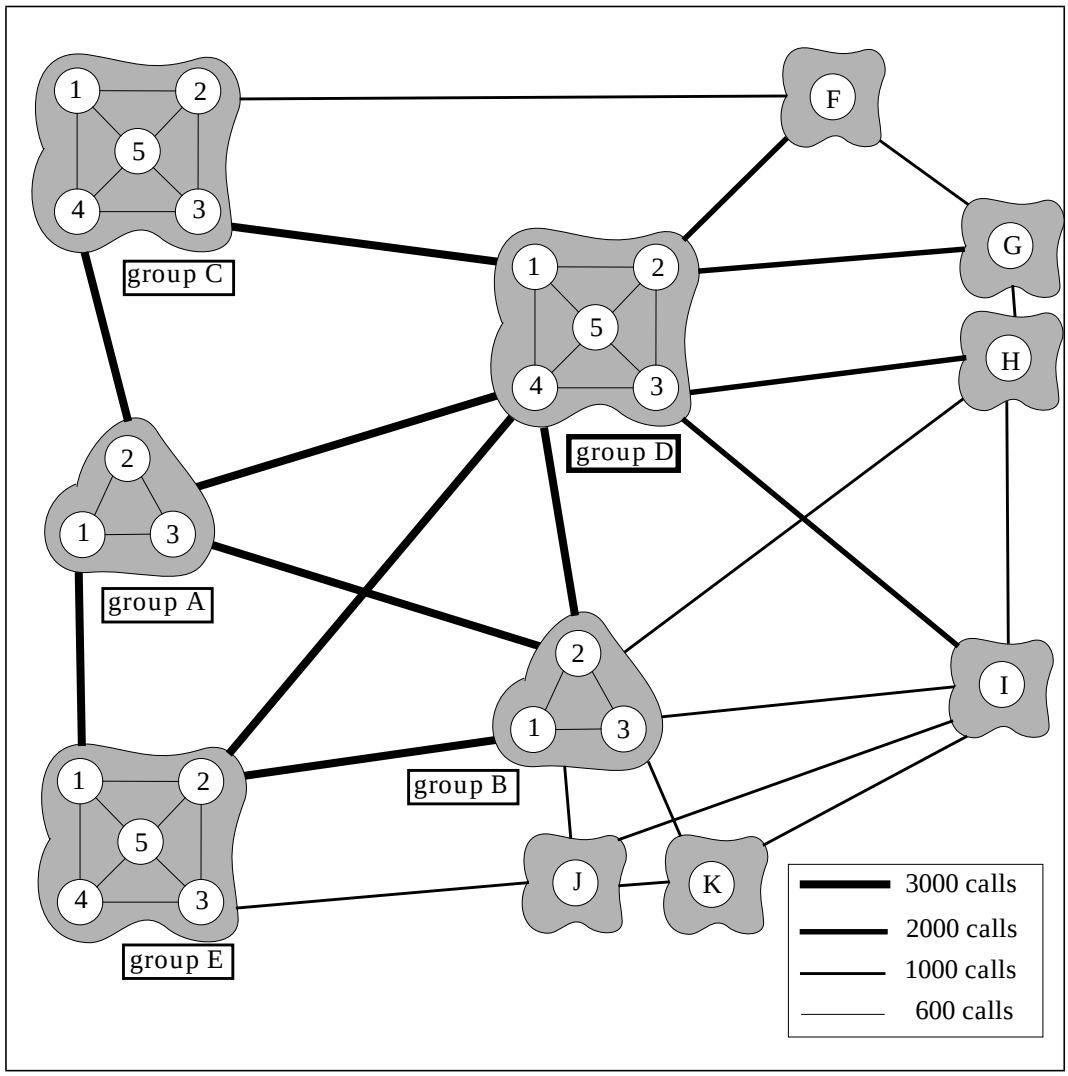

Figure A.2: National WAN network

The characteristics of the network are following:

- The network bears resemblance to a large country network with:

- 3 big cities: C,D and E - D is a main city with main international gateway

- 2 medium cities A and B,

- 6 towns: F,G,H,I,J and K

It can also be treated as many ATM clouds (LIS) connected by ATM high capacity backbone links - the most common situation in present networks.

- The CER, CLR, CMR and CDV values of links are equal to 0.
- All links have delays depending on the geographical distance - the difference between delays can reach an order of a magnitude (MCTD=0.5-7.5ms).
- The traffic is most intensive around group D (main international gateway).
- Links have higher capacity but are fewer in number than in the Milanet network.
- Only larger groups have many possibilities of redirecting traffic internally - small towns have practically only one switching node at this level of the hierarchy.
- In this type of network the delay and available bandwidth have comparable roles

from:	to:										
	A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**	**K**
A		180	300	**600**	300	30	30	30	30	30	30
B	180		300	**600**	300	30	30	30	30	30	30
C	300	300		**1000**	500	50	50	50	50	50	50
D	**600**	**600**	**1000**		**1000**	**150**	**150**	**150**	**150**	**150**	**150**
E	300	300	500	**1000**		50	50	50	50	50	50
F	30	30	50	**150**	50		10	10	10	10	10
G	30	30	50	**150**	50	10		10	10	10	10
H	30	30	50	**150**	50	10	10		10	10	10
I	30	30	50	**150**	50	10	10	10		10	10
J	30	30	50	**150**	50	10	10	10	10		10
K	30	30	50	**150**	50	10	10	10	10	10	

Table A.2: National WAN traffic matrix. *Nominal traffic=12380 Erlangs.*

As mentioned above, the links to group D (main international gateway) have the highest traffic concentration. The small town groups are much less burdened.
In addition to the inter-city traffic, there exists also internal traffic between each node inside every group:

from:	to:				
	x.1	**x.2**	**x.3**	**x.4**	**x.5**
x.1		5	5	5	5
x.2	5		5	5	5
x.3	5	5		5	5
x.4	5	5	5		5
x.5	5	5	5	5	

Table A.3: Internal traffic matrix of national 5-node groups. *Internal nominal traffic=100 Erlangs (in the thick lines 3-node group - internal nominal traffic=30 Erlangs).*

A.3 WAN type network - international

The international WAN consists of 26 nodes and 45 uniformly bidirectional VPCs:

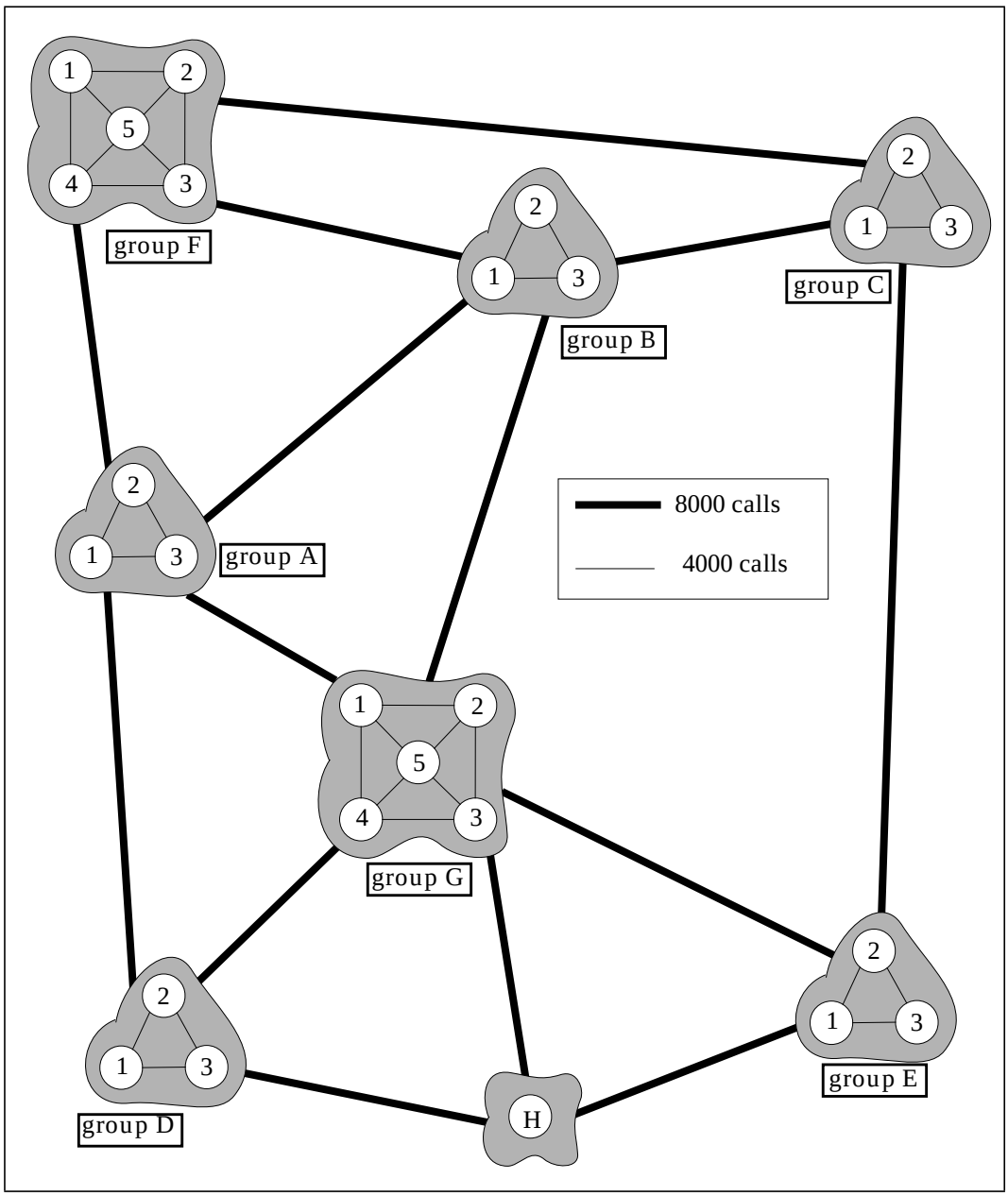

Figure A.3: International WAN network

The characteristics of the network are the following:

- The network bears resemblance to an international network with respective groups being national networks. They have similar switching capabilities and internal link capacity.
- The CER, CLR, CMR and CDV values of links are equal to 0.
- There are only two different delays: international MCTD=40ms and national MCTD=4ms. This way, only three international links may be included in the path since four of them will already violate the maximum MCTD of video connection.
- There are only two types of VPCs: international and national.

- The traffic matrix is presented below - as it can be seen, the traffic is distributed rather evenly.
- There are considerably fewer links than in the network of the National type.
- In this type of network the delay is the most important factor

from:	to:							
	A	**B**	**C**	**D**	**E**	**F**	**G**	**H**
A		810	810	810	810	1350	1350	270
B	810		810	810	810	1350	1350	270
C	810	810		810	810	1350	1350	270
D	810	810	810		810	1350	1350	270
E	810	810	810	810		1350	1350	270
F	1350	1350	1350	1350	1350		2250	450
G	1350	1350	1350	1350	1350	2250		450
H	270	270	270	270	270	450	450	

Table A.4: International WAN traffic matrix. *Nominal traffic=52200 Erlangs.*

The internal traffic of the groups is presented below:

from:	to:				
	x.1	**x.2**	**x.3**	**x.4**	**x.5**
x.1		20	20	20	20
x.2	20		20	20	20
x.3	20	20		20	20
x.4	20	20	20		20
x.5	20	20	20	20	

Table A.5: Internal traffic matrix of international 5-node groups. *Internal nominal traffic=400 Erlangs (in the thick lines 3-node group - internal nominal traffic=120 Erlangs).*

A.4 Network Blocking Probability

The distribution of the traffic was chosen so that the MAN network has the highest probability of blocking and the international WAN the lowest. It is also the case in reality, where the traffic congestion appears typically in the low capacity links of local networks.
Below the comparison of network blocking probability (for background traffic only) is presented.

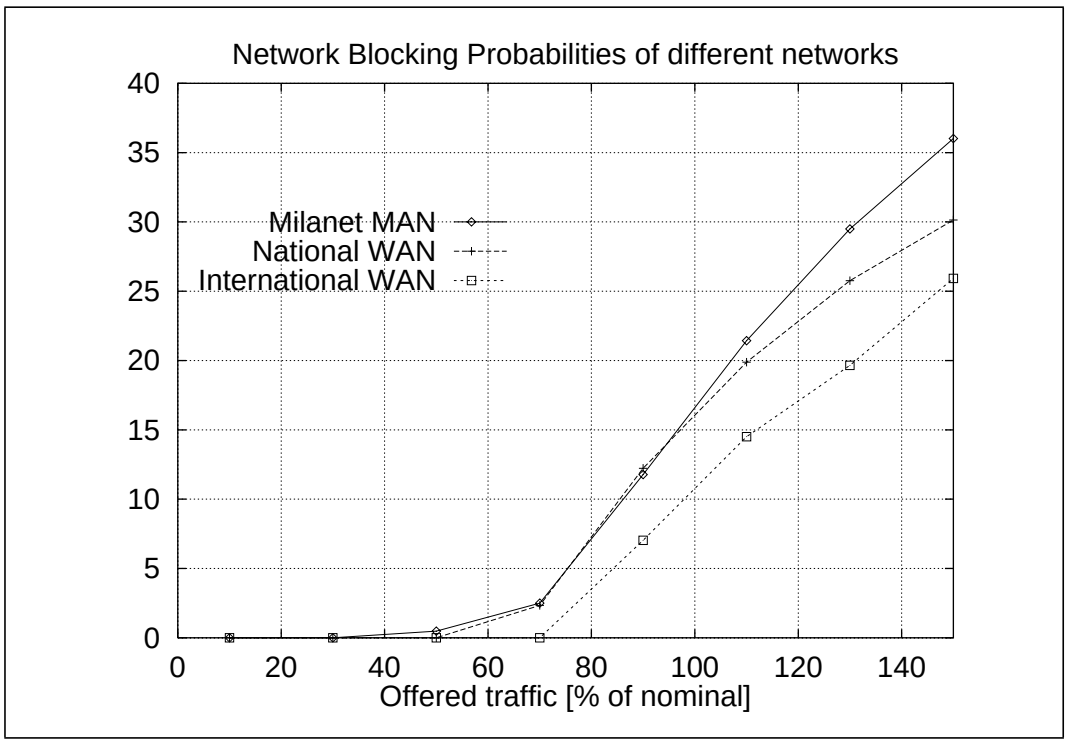

Figure A.4: Network blocking probabilities of simulated networks.

NOTE!
The national and international WAN networks are just an example of network profiles. The actual networks can differ significantly from the above simple models - indeed, good modelling of a network is a subject for another thesis.

B Simulator

All simulation were conducted in the PLASMA environment. PLASMA is an event driven simulation platform used for different types of networks: ATM, TCP/IP and many others. It was developed in cooperation between the Ericsson Telecommunication Systems Laboratories and the Technical University of Budapest.

The ATM version of PLASMA is a call-level simulator: it simulates only statistical behaviour of connections in a network. This property allows simulations of quite complex networks which are practically impossible to test using cell-oriented simulators (due to the memory and processing requirements needed for accommodating huge number of cells).

This method, however, has its limitations. It is difficult, sometimes impossible, to simulate processes like cell misalignment, realistic delay and delay variation of cells and, what is most important, dynamic changes of the connection parameters.

The user specifies the following properties of the network:

- Physical and virtual path capacities
- Virtual path (VPC) statistical properties in QoS standard - note that these values will be fixed, i.e. the load of the links will not influence them.
- Characteristics of traffic in terms of probability of connection and average holding time with different distributions (Poissonian, uniform, etc.)
- Policies for admitting calls (Connection Admission Controls and sinks).
- The properties of connections in terms of service class (bitrate and QoS parameters)

Once a network is loaded and the simulation started, the random generator begins producing calls based on declared parameters. Each call is routed in the typical ATM manner specified in [1] (data packets follow the setup path). Once Connection Admission Controls agree to accept a call, the simulator increases the load on all involved links - this load remains constant during the declared holding time. This method reduces computing complexity of simulations but makes more advanced measurements (like dynamic behaviour of the link) virtually impossible.

The above mentioned shortcomings do not, however, change the fact that this version of PLASMA simulator is a really useful and powerful tool. A user can get almost all the informations about a network behaviour and simulate a variety of events. Unfortunately, even despite the fact that this PLASMA is a connection-oriented simulator, any simulation of a large network (number of nodes bigger than 10) takes a lot of time. For example, a simulation of the National WAN network consisting of 27 nodes and supporting approximately 15000 simultaneous connections lasted 1 hour on an UltraSparc Sun computer with 256MB of memory. Therefore, in future research, it may be necessary to design a simplified simulator to support networks with 100-1000 nodes.

C Glossary

AAL	ATM Adaptation Layer
ALOHA	Technique of medium access control
asymmetric connection	Different requirements of the directions of the connection, e.g. different bandwidth. See symmetric connection.
ATM	Asynchronous Transfer Mode
AW	Administrative Weight. Set by the network operator to encourage or discourage the use of some resources.
broadcasting	The method of data transfer when the source transmits information to all possible destinations at a time.
CAC	Connection Admission Control. Checks that there are enough resources during the setup phase to carry the connection request. If not, the connection request is blocked.
CBP	Call Blocking Probability. The probability that the connection request is blocked, either by the router or by one of the CAC's in the path returned by the router.
CBR	Constant Bit Rate - parameter of a connection
CBT	Core Based Tree - type of multicasting tree
CDV	Peak-to-peak Cell Delay Variation (see QoS)
CER	Cell Error Ratio (see QoS)
CLR	Cell Loss Ratio (see QoS)
CMR	Cell Misinsertion Ratio (see QoS)
CRC	Cyclic Redundancy Code. Error correcting code.
DV	Delay Variation. Difference between slowest and fastest path in multicasting tree
DVMRP	Distance Vector Multicast Routing Protocol. First standard IP multicasting protocol.
ECR	Effective Cell Rate, also called Equivalent Cell Rate.
Erlang	Unitless measurement of traffic (in this report number of simultaneous connections).
flooding mechanism	An PNNI concept, that describes how to spread information about the state of the network within the network
GSDM	Geographic Spread Dynamic Multicast - type of dynamic multicast
LIS	Logical IP Subnet - group of ATM nodes simulating LAN network
MaxCR	Maximum Cell Rate. The capacity of a VPC.

MARS	Multicast Address Resolution Server. The proposed standard of CBT standard for ATM
MCS	MultiCast Server - ATM node responsible for redistribution of multicast data from many sources
MID	Multiplexing Identifier. A field in AAL 3/4 class cell.
MLIS	Multicast Logical IP Subnet - multicast capable LIS
multicasting	The way of data transferring when the source transmits to many destinations at a time.
PCR	Peak Cell Rate. The maximum cell rate of a connection.
PIM	Protocol Independent Multicasting. New multicasting protocol supporting DM - dense mode and SM -sparse mode.
PNNI	Private Network to Network Interface or Private Network Node Interface
QoS	Quality of Service. A set of parameters that describes the requirements of a connection request, or the resources of the network.
RIP	Routing Information Protocol - simple IP routing protocol
RPM	Reverse Path Multicasting - method of multicast routing
RP	Rendezvous Point - also called Core in CBT terminology
RSVP	Resource reSerVation Protocol. TCP/IP method of providing QoS-like parameters of a connection
SAR	Segmentation and Reassembly Sublayer - the layer of AAL
SCR	Sustained Cell Rate. The medium cell rate of a connection.
state dependent flooding	The flooding mechanism is triggered when some event occurs
symmetric connection	Equal requirements in both directions of the connection. See asymmetric connection.
SST	Source Specific Tree - method of multicasting.
ST	Shared Tree - method of multicasting.
time dependent flooding	The flooding mechanism is triggered on a constant time interval.
unicasting	The traditional way of data transferring when the source transmits to only one destination at a time.
VBR	Variable Bit Rate - parameter of a connection
VCC	Virtual Channel Connection - ATM concept
VPC	Virtual Path Connection - ATM concept